Seeing What's Wrong: A Trajectory-Guided Approach to Caption Error Detection

Gabriel Isaac Afriat
Massachusetts Institute of Technology
Cambridge, MA, USA
afriatg@mit.edu

Ryan Lucas
Massachusetts Institute of Technology
Cambridge, MA, USA
ryanlu@mit.edu

Xiang Meng
Massachusetts Institute of Technology
Cambridge, MA, USA
mengx@mit.edu

Yufang Hou*
IBM Research
Dublin, Ireland
yufang.hou1@ibm.com

Yada Zhu*
IBM Research
Yorktown Heights, NY, USA
yzhu@us.ibm.com

Rahul Mazumder*
Massachusetts Institute of Technology
Cambridge, MA, USA
rahulmaz@mit.edu

Abstract

Image-caption datasets are crucial for training vision models, but efforts to boost performance have often focused on making datasets larger rather than cleaner. This makes error detection especially important for improving dataset quality and, in turn, model performance. This paper defines a *caption trajectory*: an ordered sequence of captions obtained by iteratively editing a caption to maximize an image-text relevance score. Treating this trajectory as a signal for error detection reveals a clear pattern: correct captions stabilize after only minor tweaks, whereas incorrect ones can be substantially improved. Leveraging this observation, we present *TRACED*, a cost-efficient, model-agnostic and interpretable framework that turns trajectory statistics into a powerful model for caption error detection. *TRACED* is flexible and can be used on top of any state-of-the-art error detection method to enhance results. To better evaluate image-caption error detection, we introduce a fine-grained noise type that subtly alters caption meaning through minimal word changes, making it significantly harder to detect than standard caption swaps. We show that *TRACED* improves state-of-the-art methods, especially on challenging cases where they typically struggle.

CCS Concepts

• **Computing methodologies** → **Scene understanding**; **Natural language generation**; • **Information systems** → **Data cleaning**.

Keywords

Image-Caption Alignment, Error Detection, Caption Trajectory, Dataset Cleaning

*This work has been jointly supervised by Yufang Hou, Yada Zhu and Rahul Mazumder.

Agentic-GenAI-Eval@KDD '25, Toronto, ON, Canada

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Gabriel Isaac Afriat, Ryan Lucas, Xiang Meng, Yufang Hou, Yada Zhu, and Rahul Mazumder. 2025. Seeing What's Wrong: A Trajectory-Guided Approach to Caption Error Detection. In *Agentic & GenAI Evaluation KDD2025: KDD workshop on Evaluation and Trustworthiness of Agentic and Generative AI Models, August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 17 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Vision models have achieved remarkable success across diverse applications, including visual understanding [7], multimodal reasoning [1], and generative capabilities [8]. To reach their full potential, these models require extensive training on massive datasets, often containing millions of image-caption pairs [5, 6, 15, 20, 25, 30, 31]. Due to computational and data constraints, many models rely on pre-training with web-scraped [16, 19, 29] or even synthetic data [10, 14, 15]. However, these datasets often contain significant errors [18, 23, 35], which not only hampers model convergence during training but can also reinforce undesirable biases and reduce generalization capabilities.

Recent studies have demonstrated that removing incorrect image-caption pairs can substantially improve model performance [15, 35]. Therefore, detecting such errors is essential for boosting data quality and training better models. As manual annotation is infeasible at scale, many works have proposed automated error detection methods. These typically rely on assigning a quality or similarity score to each image-caption pair, using either model confidence [22, 27, 32], neighborhood consistency [4, 35, 36], or multimodal alignment [15, 29, 35].

While these existing methods are increasingly powerful, they typically rely on a single similarity score per image-caption pair. This poses a key limitation: *not all errors are equally detectable*. Some captions may mostly align with the image but include subtle mistakes—incorrect object labels, color description, or negation—that still yield high similarity scores (see Figure 3). Conversely, a correct caption might receive a low score if the image is difficult to describe or if the wording is imprecise (see Figure 4). In both cases, relying on a single similarity score can lead to unreliable error detection.

In this paper, we propose a novel approach that leverages caption improvement trajectories for more accurate error detection. Our key insight is that *the improvement potential of captions varies significantly between correct and incorrect captions*, a pattern we

observe consistently across the state-of-the-art alignment scoring functions we evaluated. Specifically, when starting with an accurate caption, iterative attempts to improve it yield minimal gains in similarity scores. In contrast, an incorrect caption presents substantial improvement potential through refinement.

We formalize this intuition by generating a sequence of increasingly refined captions for each image-caption pair and analyzing the resulting trajectory. Rather than making error detection decisions based on a single similarity score, our method examines the *pattern of improvement across the entire sequence*. This sequence-based approach offers a richer signal by capturing the magnitude of semantic changes between iterations and the rate at which image-caption alignment improves throughout the refinement process. Importantly, this trajectory-based method is model-agnostic and can be combined with existing state-of-the-art error detection baselines to enhance their performance.

Our contributions are as follows:

(1) We introduce a new error detection framework called *TRACED*[1], based on the novel idea of creating caption trajectories. By iteratively improving captions through token replacements and deletions, we generate a sequence of captions and analyze both their alignment with the corresponding image and the magnitude of changes between iterations. This trajectory-based approach provides richer signals and enables more accurate identification of mismatched image-caption pairs. *TRACED* is cost-efficient and interpretable. It is also flexible and can be applied on top of any existing error detection method to enhance its performance.

(2) We evaluate how *TRACED* improves the performance of several state-of-the-art error detection methods, including CLIP [29], LEMON [35], and BLIP [15]. Our experiments contain various types of label noise, including traditional random caption swaps and a more challenging type of synthetic noise we generated by prompting GPT-4o-mini [24]. This novel type of noise consists of plausible yet incorrect captions designed to better reflect real-world annotation errors. On average across all noise types, *TRACED* consistently improves detection AUC by up to 2.5% on MS COCO [20], 2.8% on Flickr30k [28], and 2.4% on MM-IMDb [3].

(3) We demonstrate that *TRACED* provides interpretable outputs by identifying specific misaligned tokens in erroneous captions. For captions involving only a few misleading words, *TRACED* is especially effective at pinpointing misaligned tokens and suggesting meaningful corrections through its trajectory-based analysis.

2 Related Work

Handling Noise in Vision Datasets. Vision datasets often contain substantial labeling errors, which can significantly degrade the performance of models trained on them [18, 23, 34, 35]. To address this, two main research directions have emerged. The first focuses on learning with noisy labels, either by modifying the loss function to account for noise [21] or by reducing the influence of likely corrupted image-text pairs during training [2, 11]. The second focuses on data cleaning, aiming to identify and remove mislabeled samples [9, 35]. Our work falls into the second category and aims at improving the filtering of noisy image-caption pairs.

Error Detection for Classification Datasets. Label noise can be detected through various approaches. Confident Learning [22] identifies label errors by analyzing the predicted confidence of a classifier under a class-conditional noise model. AUM [27] ranks training examples based on the average logit margin between the predicted label and the second most competitive label across training. Dataset Cartography [32] tracks the confidence and variability of predictions over epochs to identify label errors.

Another popular approach for noise detection utilizes the nearest neighbors to identify anomalies. Deep k-NN [4] detects label noise by checking agreement between each label and its neighbors in a DNN's embedding space. SimiFeat [36] extends this idea to a training-free setting, using k-NN voting and ranking in the features space.

With the emergence of foundation models, new stronger baselines for label error detection have appeared. Liang et al. [17] and Kang et al. [12] propose leveraging CLIP [29], pretrained on 400M image-text pairs, to score image-label consistency. Building on this, LEMoN [35] introduces a neighborhood-based method that aggregates relevance scores from multimodal nearest neighbors to improve label error detection in both classification and captioning tasks. LEMoN outperforms prior confidence-based and neighborhood-based approaches on several classification and image captioning benchmarks, making it one of the strongest available baselines. We therefore focus on this method in this work and investigate how its performance can be further enhanced through our proposed trajectory-based framework.

Error Detection for Image Captioning. We focus in this paper on error detection in image captioning, a more challenging task than image classification, as it requires deeper semantic understanding of both language and visual content. To improve caption quality, BLIP [15] builds on CLIP by learning a shared image-text embedding space but also by training a classifier to distinguish high-quality from noisy image-caption pairs. Although not originally intended for error detection, Zhang et al. [35] evaluates BLIP's filtering component and shows that it achieves strong performance in identifying mislabeled image-caption pairs on the downstream datasets it was fine-tuned on. We therefore additionally examine how our framework can enhance BLIP's performance on caption error detection.

Evaluation via Synthetic Noise Injection. To evaluate the effectiveness of error detection methods, synthetic label noise is often injected into the clean supervised datasets. For example, Pleiss et al. [27] and Kang et al. [12] use symmetric noise, where labels are randomly swapped across classes. Northcutt et al. [22] consider asymmetric noise, where labels are replaced with semantically similar ones according to a predefined noise transition matrix. Liang et al. [17] extend this setup by comparing three noise types: symmetric, asymmetric, and instance-dependent noise, where the incorrect label is selected based on the features of the instance itself. Zhu

[1] *TRACED* stands for **Tra**jectory **C**reation for **E**rror **D**etection

et al. [36] also explore instance-dependent label noise. However, these noise models are designed for classification tasks.

Zhang et al. [35] extend these noise types to the image captioning setting by introducing random caption swaps, swaps between captions sharing common nouns, and swaps within the same category when metadata is available. While these approaches move toward more realistic noise modeling, they still involve replacing the entire caption. In real-world settings, label noise can be subtler. Annotators may describe the correct image but misrepresent specific elements, resulting in partially incorrect captions that are mostly accurate but contain a few errors. In this paper, we introduce another type of noise for image captioning that aims at capturing this fine-grained form of caption noise and evaluate our framework in this more challenging setting.

3 *TRACED*: A Trajectory-Based Framework for Error Detection

To address the limitations of single-score image-caption alignment methods, we propose *TRACED*, a trajectory-based framework that leverages iterative caption refinement for error detection. Given an image and its caption, *TRACED* iteratively modifies the caption to increase its alignment with the image and tracks how alignment evolves across these edits. This produces a *caption trajectory*, i.e. a sequence of increasingly refined captions, which we use as a signal for error detection. Our core insight is as follows: (i) If the original caption is correct, alignment scores should improve only slightly, and edits will leave the meaning largely intact. (ii) If the caption is incorrect, alignment can typically be improved substantially—often requiring major semantic revisions. By capturing how easily and meaningfully a caption can be improved, *TRACED* provides a richer and more interpretable signal than any single similarity score. Moreover, it can be integrated with any existing scoring-based error detection method. Our methodology is detailed in the following subsections.

3.1 Trajectory Creation and Evaluation

Let $\mathcal{X}$ denote the set of captions and $\mathcal{Y}$ the set of images. We assume access to a relevance scoring function:

$$\begin{aligned} s\colon\ & \mathcal{X}\times\mathcal{Y}\longrightarrow\mathbb{R} \\ & (x,y)\mapsto s(x,y) \end{aligned}$$

This function assigns a real-valued relevance score to an image-caption pair, with higher values indicating stronger alignment. The choice of s is flexible: it may represent the matching probability in BLIP [15], the cosine similarity of CLIP image and text embeddings [12], or a multi-modal similarity metric like LEMoN [35].

To characterize how a caption evolves during the procedure, we introduce a trajectory evaluation function:

$$\begin{aligned} e\colon\ & \mathcal{X}^{T+1}\times\mathcal{Y}\longrightarrow\mathbb{R}^d \\ & (x_0,\dots,x_T,y)\mapsto e(x_0,\dots,x_T,y) \end{aligned}$$

where $T+1$ is the trajectory length and d is the dimensionality of the trajectory representation used for error detection. A simple choice of e is the concatenation of relevance scores: $e(x_0,\dots,x_T,y) = [s(x_0,y),\dots,s(x_T,y)]$.

Another interesting metric to keep track of is the semantic similarity between the caption at step t and the original (potentially noisy) caption x_0, denoted $c(x_t,x_0)$. This captures the degree of semantic change introduced at each step.

In this paper, we focus on these two key signals and construct the following evaluation function:

$$\begin{aligned} e(x_0,\dots,x_T,y) = [&s(x_0,y),\dots,s(x_T,y), \\ &c(x_1,x_0),\dots,c(x_T,x_0)] \end{aligned}$$

Given access to s and e, *TRACED* constructs and evaluates a caption trajectory as described in Algorithm 1:

Algorithm 1 Trajectory Creation and Evaluation

Input: initial caption x, image y, scoring function s, evaluation function e, trajectory length T, number of candidates N
Initialize $x_0 \leftarrow x$
for $t = 1$ **to** T **do**
  Generate candidate alternatives: $x_t^{(1)},\dots,x_t^{(N)}$
  Select best candidate: $j_t \leftarrow \arg\max_{j\in[N]} s(x_t^{(j)},y)$
  Set $x_t \leftarrow x_t^{(j_t)}$
end for
Output: $e(x_0,\dots,x_T,y)$

We apply Algorithm 1 to each image-caption pair in the dataset. From the resulting trajectory embeddings, we train a classifier to distinguish between correct and erroneous pairs. While the choice of e is flexible, we focus in this paper on two key signals: the extent to which image-caption relevance can be improved, and the magnitude of semantic change required in the caption to achieve this. Finally, the trajectory embeddings can be used as input features to train any classifier to distinguish between correct and incorrect image-caption pairs. The overall framework is described in Figure 1.

3.2 Caption Exploration

A critical component of Algorithm 1 is the generation of candidate captions at each step. We explore and evaluate several strategies for this purpose:

- **Elimination.** This simple and efficient method generates candidates by removing one token at a time from the current caption. Formally, for a caption $x = (w_1,\dots,w_L)$ with L tokens, we set $N = L$ in Algorithm 1 and produce L candidates:

  $$x^{(i)} = (w_1,\dots,w_{i-1},w_{i+1},\dots,w_L)$$

  This strategy is computationally cheap: it requires only L forward passes through the scoring function s in Algorithm 1 and no gradient computations.
- **Greedy Coordinate Descent (GCD).** Inspired by Zou et al. [37], this method aims to find improved captions by replacing individual tokens with alternatives that increase the relevance score s. For each token in a caption of length L, we consider the top-K gradient-guided replacements, leading to a candidate pool of size KL. Since this is often too large to evaluate exhaustively, we randomly sample N token replacements from this space. While

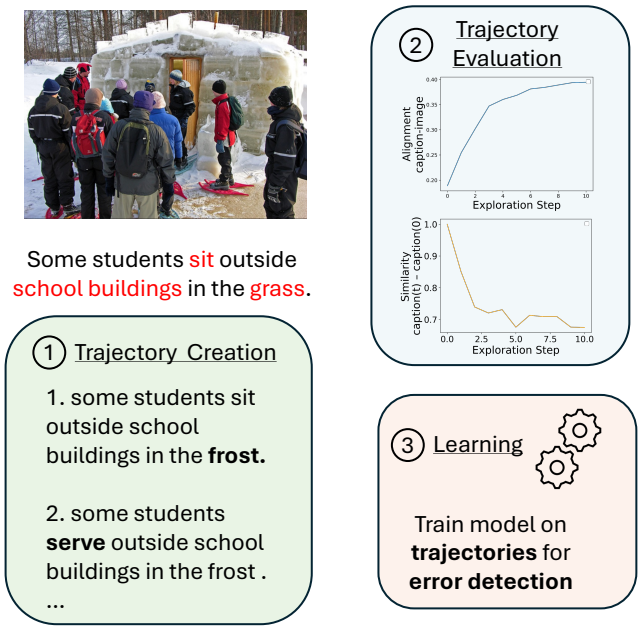


Figure 1: *TRACED* Pipeline. Given a noisy image-caption pair, a caption trajectory is generated by iteratively maximizing a relevance scoring function s. Each trajectory is then evaluated using various alignment metrics. These trajectories then serve as features to distinguish between correct and incorrect image-caption pairs. The example is from Flickr30k [28].

more powerful than elimination, this approach is more expensive due to the gradient computation and the larger candidate pool ($N > L$).

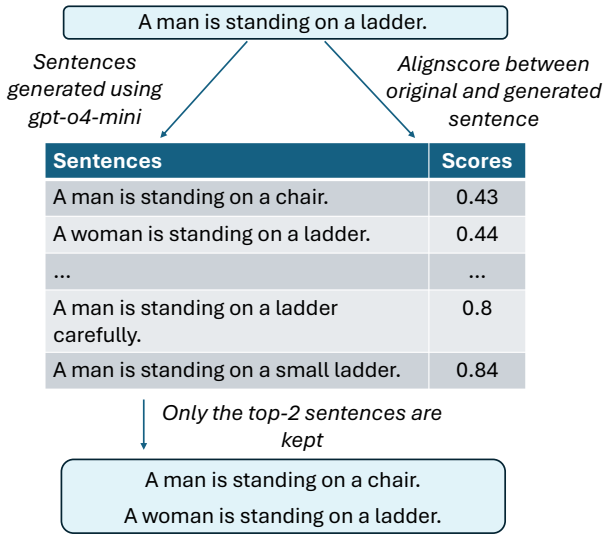
- **Fast GCD.** To balance the efficiency and quality of the caption trajectory, we introduce a hybrid strategy that combines Elimination with Greedy Coordinate Descent (GCD). We first apply the Elimination method to identify the token whose removal most improves the relevance score. Then, we explore only the top-K replacements for that specific token, reducing the search space to K candidates. This approach requires only one gradient computation and $K + L$ forward passes per iteration in Algorithm 1, a significant reduction compared to the KL evaluations needed for full exploration. Moreover, by focusing optimization on the most impactful token, we promote more effective substitutions than would be achieved by randomly sampling from a large candidate pool.

The full algorithm descriptions are provided in Appendix A.1.

3.3 New Benchmark Dataset Creation

Prior work on error detection in image captioning introduces noise via full caption swaps as a means of constructing evaluation benchmarks [35]. However, such swaps replace the entire caption, often resulting in text unrelated to the original. In contrast, real-world annotation errors can be more subtle, with annotators correctly describing an image but misrepresenting specific details. To better capture fine-grained noise, we propose a new approach for constructing a challenging benchmark by modifying only a few words within each caption. More specifically, for each original caption, we leverage a large language model (LLM) to generate K variants that maintain the same structure but introduce small semantic errors. The exact prompt is provided in Appendix A.2.

While many generated options are useful, some may be paraphrases or near-duplicates. To filter these, we apply Alignscore [33], a factual consistency metric based on a fine-tuned natural language inference model. Alignscore assigns low scores to captions that either omit key information from or contradict the original caption. The selected variants thus differ meaningfully in content while remaining structurally close, effectively modeling fine-grained semantic noise. Figure 2 illustrates this generation and filtering process.



Sentences	Scores
A man is standing on a chair.	0.43
A woman is standing on a ladder.	0.44
...	...
A man is standing on a ladder carefully.	0.8
A man is standing on a small ladder.	0.84

Figure 2: Illustration of the fine-grained noise generation pipeline. Given an original caption, a language model generates 20 variants. Alignscore is then used to evaluate the factual consistency of each variant with respect to the original. The top-2 least aligned (lowest-scoring) sentences are selected as fine-grained noisy captions.

3.4 Interpretability

Examining the caption trajectory can help identify the source of the error. As shown in Figure 3, the first tokens whose removal or replacement usually leads to the greatest improvement in alignment score often correspond to the source of the misalignment.

In this example, the initial alignment score from BLIP's classifier is 0.55, indicating a 55% probability that the image-caption pair is correct. However, the trajectory shows that a meaningful semantic change can increase the alignment score to approximately 99.4%, indicating that the original caption is likely erroneous.

On the contrary, Figure 4 shows that while *TRACED* improves the alignment score for this example, the changes involve only minor

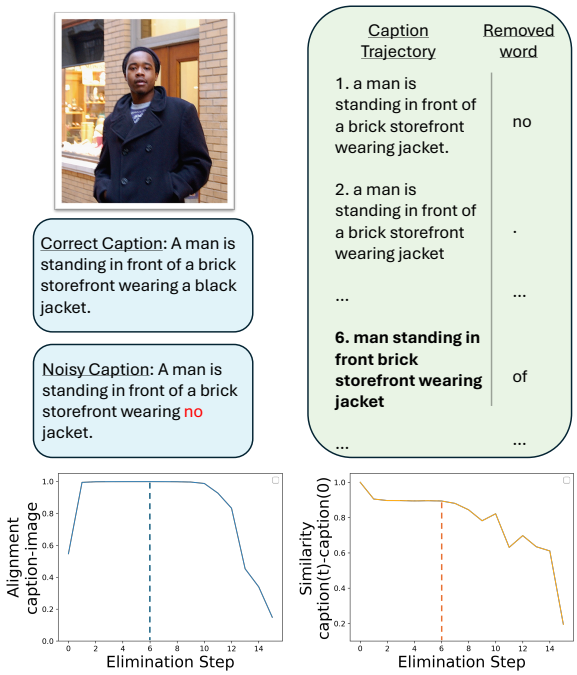


Figure 3: *TRACED* offers interpretability on a Flickr30k example [28], identifying "no" as the source of misalignment. The BLIP-based alignment score (ITM block) peaks at step 6, where the caption accurately matches the image. Removing "no" leads to a notable decline in semantic alignment across the caption trajectory.

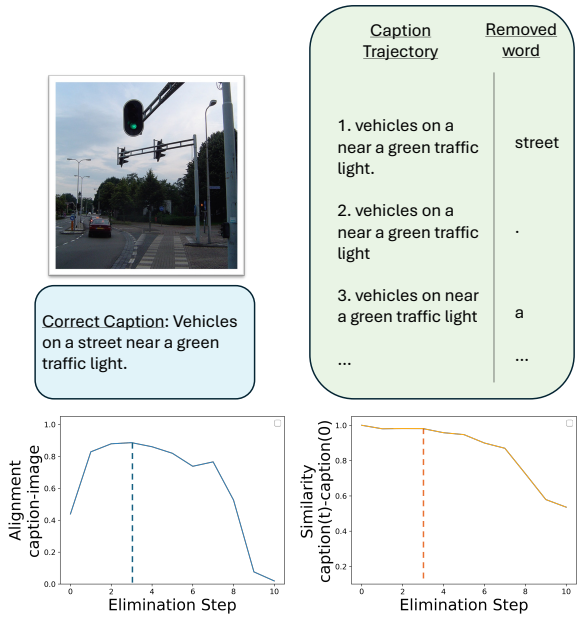


Figure 4: *TRACED* improves the BLIP-based image-caption alignment score (ITM) on an MS COCO example [20], with minimal semantic change in the revised captions, suggesting the original pair is likely accurate.

semantic edits. The initial BLIP (ITM) score is relatively low (0.44), but the trajectory reveals that these small revisions are sufficient to boost the score. This suggests that the original caption was likely correct, and the low initial score stems from the model's difficulty with the example rather than a true misalignment.

For incorrect captions, token replacements generated by GCD and Fast GCD often enhance the description by introducing details that are present in the image but missing from the noisy caption. In contrast, when the original caption is correct, the models tend to propose edits that refine the wording without introducing substantial new information, focusing instead on phrasing that better aligns with the visual content. This behavior is illustrated in the caption trajectories produced by GCD and Fast GCD for the examples in Figures 3 and 4. Full trajectories for these cases are provided in Figures 7 and 8 in Appendix A.3.

3.5 Parallelization benefits

TRACED applies the trajectory creation and evaluation from Algorithm 1 independently to each sentence in the training, validation, and test sets. This enables efficient large-scale parallelization. Our method benefits from both intra-GPU and multi-GPU parallelism: given access to n GPUs, the dataset can be split into n subsets processed in parallel, with each GPU handling one subset in batches.

4 Experiments

4.1 Setup

All experiments are conducted using 4 NVIDIA L40 GPUs, each with 40GB of memory. As described in Section 3.5, the datasets are split into 4 subsets, with each GPU processing one subset independently. Sentences are processed in batches of size 128 on each GPU. *TRACED* is implemented using PyTorch [26].

4.2 Baselines and Datasets

Baselines. We evaluate the performance of *TRACED* across several existing error detection baselines and datasets. Specifically, we apply *TRACED* to the following baselines:

- BLIP [15]
- LEMoN [35]
- CLIP [12, 29]

CLIP uses cosine similarity in a joint embedding space, LEMoN aggregates CLIP scores from nearest neighbors, and BLIP combines contrastive learning (ITC block) to learn a shared image-text embedding space with a classification head (ITM block) for alignment prediction.

LEMoN supports two versions: FIX (default hyperparameters) and OPT (hyperparameters tuned via validation).

We apply *TRACED* on top of each of these baselines by using their respective alignment scores as the scoring function s during trajectory construction. For BLIP, we evaluate *TRACED* using both the ITC and ITM modules. For LEMoN, we follow the protocol in

Zhang et al. [35], applying our method to both the FIX and OPT variants. For CLIP, we use the standard cosine similarity between image and text embeddings.

Datasets. We evaluate the impact of *TRACED* on LEMoN and CLIP using the following three datasets:

- Flickr30k [28]
- MS COCO [20]
- MM-IMDb [3]

For Flickr30k and MS COCO, we use the standard Karpathy split [13]. For MM-IMDb, we adopt the same random 80/10/10 train-validation-test split as described in [35].

For BLIP, finetuned models are publicly available only for Flickr30k and MS COCO. Therefore, we evaluate the improvements from *TRACED* on these two datasets only.

Noise Types. We evaluate *TRACED*'s improvements under three types of synthetic label noise, introducing 50% erroneous image-caption pairs for each seed:

- Random noise: A subset of captions is randomly replaced with others from the dataset.
- Noun noise: Captions are swapped with others that share at least one noun, introducing partial semantic overlap.
- Fine-grained noise: Captions are minimally perturbed using `gpt-o4-mini` to introduce subtle semantic inconsistencies, as described in Section 3.3.

Due to the higher cost of generating fine-grained noise using the ChatGPT API, we limit its use to Flickr30k and MS-COCO. For both random and noun noise, we follow the methodology introduced in Zhang et al. [35].

4.3 Trajectory Construction and Learning Framework

Trajectory Generation Hypeparameters. The trajectory generation hyperparameters for Elimination, GCD, and Fast GCD are detailed in Appendix A.4.

Trajectory Evaluation Metrics. For the alignment score $s(x_t, y)$, we use the scoring function of the baseline being evaluated—either CLIP, LEMoN, or BLIP.

For the semantic similarity $c(x_t, x_0)$, we compute the cosine similarity between the embeddings of x_t and x_0. When the baseline is BLIP, we use its ITC block to extract embeddings. For CLIP and LEMoN, we use CLIP embeddings.

We then construct the evaluation function e as the concatenation of these metrics:

$$e(x_0, \ldots, x_T, y) = [s(x_0, y), \ldots, s(x_T, y), \\ c(x_1, x_0), \ldots, c(x_T, x_0)]$$

Learning Procedure. Once the trajectory embeddings are constructed, they can be used as features to predict whether a given image-caption pair contains an error. While any standard classification model could be applied at this stage, we use XGBoost and CART due to their simplicity, as our primary goal is to demonstrate the effectiveness of our approach. More sophisticated models could be explored to further improve the performance gap between *TRACED* and the original baseline.

For datasets that use the Karpathy split, we combine the original training and validation sets. We then perform 3-fold grid-search cross-validation to select the best model and hyperparameters. The complete grid searches are provided in Appendix A.4.

The best-performing model (XGBoost or CART) and its corresponding hyperparameters are selected based on the highest cross-validation AUC score.

4.4 Main Results

The results are presented in Table 1, where AUC scores are averaged over all applicable noise types and random seeds.

We observe some variability in performance across random seeds, complicating direct comparisons based solely on raw Test AUC. However, when controlling for seed and noise type, *TRACED* consistently outperforms the baselines. To capture this pattern, we report the mean percent improvement in Test AUC relative to the baselines, averaged across all seeds and noise types. *TRACED* indeed yields consistent and significant gains over each baseline, highlighting its effectiveness for error detection.

Table 1 also suggests that the Elimination algorithm generates more informative trajectories for the error detection task compared to GCD and Fast GCD. We attribute this to two main factors:

- The Elimination algorithm progressively removes words from the caption, producing a trajectory in which the similarity score typically increases before decreasing. Unlike GCD and Fast GCD, which only replace tokens to increase alignment, Elimination trajectories reflect both the positive and negative contributions of individual words, revealing which original tokens align well with the image and which do not.
- The Elimination algorithm operates in a much more constrained search space, which introduces a form of regularization. In contrast, GCD and Fast GCD allow broader substitutions, sometimes leading to non-meaningful token replacements that nonetheless increase the alignment score, reflecting the tendency of CLIP and BLIP to assign high scores to tokens that lack semantic relevance.

A detailed breakdown by noise type is provided in Table 6 (Appendix A.5). The largest gains from *TRACED* occur under the fine-grained noise setting, where subtle word-level changes make detection especially challenging and baseline methods struggle most. In contrast, random and noun-based noise often result in clearly mismatched captions, making them easier for baselines to detect. These results highlight *TRACED* 's strength in handling more realistic and semantically nuanced errors.

4.5 Computation Overhead

Table 2 reports the computation time required by *TRACED* and the original baselines to process 1,000 image-caption pairs on a single L40 GPU. Baseline models such as BLIP, LEMoN, and CLIP are very fast as they require only a single forward pass per pair. Despite performing multiple model evaluations to construct trajectories, all variants of *TRACED*, including Elimination, Fast GCD, and GCD,

Table 1: Comparison of *TRACED* with baselines. "Elim" and "FGCD" denote Elimination and Fast GCD, respectively. Results are averaged over 3 seeds and the applicable noise types: noun and random for MM-IMDB, and noun, random, and fine-grained for Flickr30k and MS-COCO (50% noise). We report mean AUC and mean AUC improvement compared to the baseline, with standard errors.

Dataset	Method	Algorithm	AUC (%)	Improvement (%)
Flickr-30k	*TRACED*-BLIP (itm)	Elim	**89.5 ± 0.3**	**1.3 ± 0.6**
		FGCD	89.2 ± 0.4	0.8 ± 0.5
		GCD	88.8 ± 0.4	0.3 ± 0.5
	BLIP (itm)	-	88.5 ± 0.6	0.0 ± 0.0
	TRACED-BLIP (itc)	Elim	88.1 ± 0.3	0.9 ± 0.4
		FGCD	**88.1 ± 0.3**	**0.9 ± 0.6**
		GCD	88.0 ± 0.4	0.7 ± 0.4
	BLIP (itc)	-	87.4 ± 0.5	0.0 ± 0.0
	TRACED-LEMoN$_{OPT}$	Elim	85.6 ± 0.7	1.8 ± 0.6
		FGCD	85.5 ± 0.6	1.9 ± 0.6
		GCD	**85.7 ± 0.8**	**2.1 ± 0.9**
	LEMoN$_{OPT}$	-	84.3 ± 0.5	0.0 ± 0.0
	TRACED-LEMoN$_{FIX}$	Elim	85.0 ± 0.5	1.7 ± 1.0
		FGCD	85.0 ± 0.5	1.7 ± 0.3
		GCD	**85.6 ± 0.6**	**2.6 ± 0.9**
	LEMoN$_{FIX}$	-	83.9 ± 0.5	0.0 ± 0.0
	TRACED-CLIP	Elim	**85.7 ± 0.3**	**2.8 ± 0.9**
		FGCD	85.5 ± 0.2	2.6 ± 0.5
		GCD	85.5 ± 0.4	2.5 ± 1.0
	CLIP	-	83.8 ± 0.4	0.0 ± 0.0
MM-IMDb	*TRACED*-LEMoN$_{OPT}$	Elim	**79.0 ± 0.4**	**1.4 ± 0.5**
		FGCD	78.0 ± 0.4	0.2 ± 0.3
		GCD	78.3 ± 0.5	0.5 ± 0.4
	LEMoN$_{OPT}$	-	77.9 ± 0.3	0.0 ± 0.0
	TRACED-LEMoN$_{FIX}$	Elim	**78.3 ± 0.2**	**2.4 ± 0.4**
		FGCD	77.2 ± 0.1	0.9 ± 0.5
		GCD	77.6 ± 0.3	1.4 ± 0.4
	LEMoN$_{FIX}$	-	76.5 ± 0.4	0.0 ± 0.0
	TRACED-CLIP	Elim	**78.5 ± 0.3**	**1.8 ± 0.2**
		FGCD	77.5 ± 0.2	0.4 ± 0.3
		GCD	77.8 ± 0.2	0.9 ± 0.3
	CLIP	-	77.2 ± 0.4	0.0 ± 0.0
MS-COCO	*TRACED*-BLIP (itm)	Elim	**90.5 ± 0.3**	**1.7 ± 0.1**
		FGCD	89.8 ± 0.3	0.9 ± 0.2
		GCD	89.7 ± 0.3	0.8 ± 0.1
	BLIP (itm)	-	89.1 ± 0.2	0.0 ± 0.0
	TRACED-BLIP (itc)	Elim	**88.7 ± 0.3**	**1.8 ± 0.1**
		FGCD	88.4 ± 0.2	1.4 ± 0.3
		GCD	88.1 ± 0.3	1.0 ± 0.2
	BLIP (itc)	-	87.4 ± 0.3	0.0 ± 0.0
	TRACED-LEMoN$_{OPT}$	Elim	**85.0 ± 0.4**	**1.6 ± 0.2**
		FGCD	84.6 ± 0.5	1.0 ± 0.2
		GCD	84.5 ± 0.5	1.0 ± 0.1
	LEMoN$_{OPT}$	-	83.8 ± 0.5	0.0 ± 0.0
	TRACED-LEMoN$_{FIX}$	Elim	**84.3 ± 0.3**	**2.3 ± 0.4**
		FGCD	83.9 ± 0.6	1.8 ± 0.5
		GCD	83.9 ± 0.5	1.8 ± 0.4
	LEMoN$_{FIX}$	-	82.6 ± 0.5	0.0 ± 0.0
	TRACED-CLIP	Elim	**84.5 ± 0.4**	**2.5 ± 0.2**
		FGCD	83.7 ± 0.4	1.5 ± 0.4
		GCD	83.8 ± 0.3	1.6 ± 0.2
	CLIP	-	82.7 ± 0.3	0.0 ± 0.0

remain practical and scalable. Among the proposed methods, Elimination is the most efficient, offering substantial speed advantages while maintaining among the best performance. Fast GCD achieves a strong balance between speed and trajectory quality. For example, when scaled to 1,000,000 image-caption pairs using BLIP (ITM), the most expensive baseline, and 4 L40 GPUs, Elimination completes in approximately 6.5 hours and Fast GCD takes about 2.6 days.

As described in Section 3.5, thanks to the high degree of parallelism in our method, leveraging more GPUs can substantially further reduce total processing time.

We want to emphasize that *TRACED* needs to be applied only once to identify and filter out incorrect image-caption pairs. This one-time computational cost is reasonable for generating a cleaner dataset that can be reused across various downstream tasks, including pre-training, fine-tuning, and evaluation.

Table 2: Computation time comparison across algorithms. Reported times (in seconds) corresponds to the duration required to process 1,000 sentences with a single L40 GPU, including both trajectory exploration and alignment score evaluation.

Method	Algorithm	Computation Time (s)
BLIP (ITM)	-	3.82 ± 0.11
TRACED-BLIP (ITM)	Elimination	92.53 ± 1.16
	Fast GCD	905.22 ± 0.99
	GCD	1617.06 ± 0.13
BLIP (ITM)	-	3.56 ± 0.24
TRACED-BLIP (ITM)	Elimination	49.05 ± 0.28
	Fast GCD	389.19 ± 0.51
	GCD	688.90 ± 0.55
LEMoN$_{OPT}$	-	3.13 ± 0.08
TRACED-LEMoN$_{OPT}$	Elimination	43.77 ± 0.59
	Fast GCD	451.28 ± 0.43
	GCD	799.03 ± 1.79
LEMoN$_{FIX}$	-	3.19 ± 0.40
TRACED-LEMoN$_{FIX}$	Elimination	43.44 ± 0.44
	Fast GCD	452.48 ± 1.07
	GCD	802.13 ± 1.76
CLIP	-	2.40 ± 0.07
TRACED-CLIP	Elimination	43.10 ± 0.57
	Fast GCD	444.80 ± 0.55
	GCD	788.97 ± 0.29

5 Ablations

5.1 Contribution of Image-Caption Alignment and Caption-Caption Similarity Metrics

To isolate the contribution of each trajectory evaluation metric, we conduct ablation studies using *TRACED* with either the alignment score s or the semantic similarity score c alone. Table 3 reports the mean percent change in Test AUC when using one of the two metrics alone, relative to using both jointly.

Across all baselines, using either metric in isolation results in a consistent and significant drop in performance. The alignment

Table 3: Mean percent improvement in Test AUC when using either s or c alone in *TRACED*, compared to using both jointly. Experiments are conducted on MS-COCO using the Elimination algorithm. Results are averaged over 3 seeds and all 3 noise types (50% noise), with standard errors reported.

Method	Alignment image-caption $s(x, y)$	Similarity caption-caption $c(x_t, x_0)$
BLIP (itm)	-0.41 ± 0.20	-6.03 ± 0.41
BLIP (itc)	-0.59 ± 0.20	-10.18 ± 0.41
LEMoN$_{\text{OPT}}$	-0.55 ± 0.26	-7.22 ± 0.49
LEMoN$_{\text{FIX}}$	-0.41 ± 0.38	-6.44 ± 0.56
CLIP	-0.60 ± 0.12	-7.00 ± 0.29

score s alone is much more informative, likely because a notable increase in alignment often signals an error in the original caption. In contrast, the semantic similarity score c is less useful on its own, as captions along the trajectory may differ substantially from the original, reducing its standalone discriminative power. However, combining s and c consistently yields the best performance: s captures the degree of alignment improvement, while c indicates whether that improvement involves a substantial semantic change or only a minor rephrasing.

5.2 Importance of the Trajectory

To assess whether the full caption trajectory is necessary for effective error detection, we compare the full *TRACED* trajectory to three simplified variants: (i) using only the first step ($s(x_0, y)$; note that $c(x_0, x_0) = 1$ provides no additional signal, (ii) using only the last step ($s(x_T, y)$ and $c(x_T, x_0)$, and (iii) using the mean of all alignment and similarity values across the trajectory ($\frac{1}{T+1}\sum_{t=0}^{T} s(x_t, y)$ and $\frac{1}{T}\sum_{t=1}^{T} c(x_t, x_0)$). Table 4 reports the mean percent change in Test AUC for each variant, relative to using the complete trajectory.

Table 4: Mean percent improvement in Test AUC when using only the first step, last step or mean trajectory alone in *TRACED*, compared to using the whole trajectory. Experiments are conducted on Flickr30k using the Elimination algorithm. Results are averaged over 3 seeds and all 3 noise types (50% noise), with standard errors reported.

Method	First Step	Last Step	Mean Trajectory
BLIP (itm)	-1.25 ± 0.56	-43.07 ± 0.94	-4.90 ± 0.39
BLIP (itc)	-0.85 ± 0.34	-40.54 ± 1.35	-5.48 ± 0.95
LEMoN$_{\text{OPT}}$	-1.75 ± 0.57	-38.67 ± 1.46	-5.72 ± 0.53
LEMoN$_{\text{FIX}}$	-1.62 ± 0.93	-38.18 ± 0.83	-5.67 ± 1.15
CLIP	-2.62 ± 0.82	-38.49 ± 1.18	-5.37 ± 0.26

Across all baselines, removing the full trajectory leads to consistent and substantial drops in performance. Retaining only the first step causes the mildest decline. Relying only on the last step leads to the sharpest drop (over 38% in all cases) while averaging over the trajectory performs slightly better but remains far behind the full sequence. These results highlight the importance of modeling the entire trajectory, which captures how the alignment evolves and provides richer information than a single point summary.

5.3 Maximizing or Minimizing the Scoring Function?

In *TRACED*, we proposed to generate the trajectories by maximizing the image-caption alignment score s at each step. To test whether the opposite strategy is also effective, we compare against a variant that minimizes s instead. Table 5 reports the mean percent improvement in Test AUC when using the minimization approach, relative to maximization.

Table 5: Mean percent improvement in Test AUC when generating the trajectory in *TRACED* by minimizing s rather than maximizing it. Experiments are conducted on Flickr30k using the Elimination algorithm. Results are averaged over 3 seeds and all 3 noise types (50% noise), with standard errors reported.

BLIP (itm)	BLIP (itc)	LEMoN$_{\text{FIX}}$	LEMoN$_{\text{OPT}}$	CLIP
-0.76 ± 0.26	-0.46 ± 0.33	-0.70 ± 0.48	-0.58 ± 0.63	-0.24 ± 0.43

Across all baselines, maximizing the alignment score yields modest but consistent improvements over minimization. This suggests that constructing trajectories toward higher-scoring captions—rather than worse ones—provides a more reliable signal for detecting semantic inconsistencies.

6 Conclusion

We presented *TRACED*, a flexible and interpretable framework for image-caption error detection. By iteratively improving captions and analyzing alignment and semantic similarity over time, *TRACED* extracts rich signals that help distinguish between correct and erroneous image-caption pairs. Our framework can be applied on top of existing error detection methods such as BLIP, CLIP, and LEMoN, consistently boosting their performance across multiple datasets and noise types.

We also introduced a new fine-grained noise generation process that reflects real-world annotation errors and provides a more challenging benchmark for evaluation. In addition to improved performance, *TRACED* offers interpretability by revealing which parts of a caption contribute most to the misalignment, and even suggests possible corrections.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. 2022. Flamingo: a Visual Language Model for Few-Shot Learning. In *Advances in Neural Information Processing Systems*, Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (Eds.). https://openreview.net/forum?id=EbMuimAbPbs

[2] Eric Arazo, Diego Ortego, Paul Albert, Noel O'Connor, and Kevin Mcguinness. 2019. Unsupervised Label Noise Modeling and Loss Correction. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 312–321. https://proceedings.mlr.press/v97/arazo19a.html

[3] John Arevalo, Thamar Solorio, Manuel Montes y Gómez, and Fabio A. González. 2017. Gated Multimodal Units for Information Fusion. arXiv:1702.01992 [stat.ML] https://arxiv.org/abs/1702.01992

[4] Dara Bahri, Heinrich Jiang, and Maya Gupta. 2020. Deep k-NN for Noisy Labels. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 540–550. https://proceedings.mlr.press/v119/bahri20a.html

[5] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. 2021. Frozen in Time: A Joint Video and Image Encoder for End-to-End Retrieval. In *ICCV*. 1708–1718. https://doi.org/10.1109/ICCV48922.2021.00175

[6] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. 2021. Conceptual 12M: Pushing Web-Scale Image-Text Pre-Training To Recognize Long-Tail Visual Concepts. In *CVPR*. 3558–3568. https://openaccess.thecvf.com/content/CVPR2021/html/Changpinyo_Conceptual_12M_Pushing_Web-Scale_Image-Text_Pre-Training_To_Recognize_Long-Tail_Visual_CVPR_2021_paper.html

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*. https://openreview.net/forum?id=YicbFdNTTy

[8] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. 2024. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis. In *Forty-first International Conference on Machine Learning*. https://openreview.net/forum?id=FPnUhsQJ5B

[9] Andreas Grivas, Beatrice Alex, Claire Grover, Richard Tobin, and William Whiteley. 2020. Not a cute stroke: Analysis of Rule- and Neural Network-based Information Extraction Systems for Brain Radiology Reports. In *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis*, Eben Holderness, Antonio Jimeno Yepes, Alberto Lavelli, Anne-Lyse Minard, James Pustejovsky, and Fabio Rinaldi (Eds.). Association for Computational Linguistics, Online, 24–37. doi:10.18653/v1/2020.louhi-1.4

[10] Hasan Abed Al Kader Hammoud, Hani Itani, Fabio Pizzati, Adel Bibi, and Bernard Ghanem. 2024. SynthCLIP: Are We Ready for a Fully Synthetic CLIP Training?. In *Synthetic Data for Computer Vision Workshop @ CVPR 2024*. https://openreview.net/forum?id=oKwYycMSrf

[11] Runhui Huang, Yanxin Long, Jianhua Han, Hang Xu, Xiwen Liang, Chunjing Xu, and Xiaodan Liang. 2023. NLIP: Noise-Robust Language-Image Pre-training. In *AAAI*. 926–934. https://doi.org/10.1609/aaai.v37i1.25172

[12] Woo-Young Kang, Jonghwan Mun, Sungjun Lee, and Byungseok Roh. 2022. Noise-aware Learning from Web-crawled Image-Text Data for Image Captioning. *CoRR* abs/2212.13563 (2022). https://doi.org/10.48550/arXiv.2212.13563

[13] Andrej Karpathy and Fei-Fei Li. 2015. Deep visual-semantic alignments for generating image descriptions. In *CVPR*. 3128–3137. https://doi.org/10.1109/CVPR.2015.7298932

[14] Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. 2023. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. *CoRR* abs/2301.12597 (2023). https://doi.org/10.48550/arXiv.2301.12597

[15] Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. 2022. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. In *ICML*. 12888–12900. https://proceedings.mlr.press/v162/li22n.html

[16] Junnan Li, Ramprasaath R. Selvaraju, Akhilesh Deepak Gotmare, Shafiq Joty, Caiming Xiong, and Steven Hoi. 2021. Align before Fuse: Vision and Language Representation Learning with Momentum Distillation. In *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (Eds.). https://openreview.net/forum?id=OJLaKwiXSbx

[17] Chao Liang, Linchao Zhu, Humphrey Shi, and Yi Yang. 2024. Combating Label Noise with a General Surrogate Model for Sample Selection. *International Journal of Computer Vision* 133 (12 2024), 3166–3179. doi:10.1007/s11263-024-02324-z

[18] Thomas Liao, Rohan Taori, Inioluwa Deborah Raji, and Ludwig Schmidt. 2021. Are We Learning Yet? A Meta Review of Evaluation Failures Across Machine Learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*. https://openreview.net/forum?id=mPducS1MsEK

[19] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. 2024. Video-LLaVA: Learning United Visual Representation by Alignment Before Projection. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (Eds.). Association for Computational Linguistics, Miami, Florida, USA, 5971–5984. doi:10.18653/v1/2024.emnlp-main.342

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014*, David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (Eds.). Springer International Publishing, Cham, 740–755.

[21] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. 2013. Learning with Noisy Labels. In *Advances in Neural Information Processing Systems*, C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2013/file/3871bd64012152bfb53fdf04b401193f-Paper.pdf

[22] Curtis Northcutt, Lu Jiang, and Isaac Chuang. 2021. Confident Learning: Estimating Uncertainty in Dataset Labels. *J. Artif. Int. Res.* 70 (May 2021), 1373–1411. doi:10.1613/jair.1.12125

[23] Curtis G Northcutt, Anish Athalye, and Jonas Mueller. 2021. Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*. https://openreview.net/forum?id=XccDXrDNLek

[24] OpenAI. 2024. GPT-4o-mini. Available at https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/.

[25] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. 2011. Im2Text: Describing Images Using 1 Million Captioned Photographs. In *Advances in Neural Information Processing Systems*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger (Eds.), Vol. 24. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2011/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf

[26] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf

[27] Geoff Pleiss, Tianyi Zhang, Ethan Elenberg, and Kilian Q Weinberger. 2020. Identifying Mislabeled Data using the Area Under the Margin Ranking. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 17044–17056. https://proceedings.neurips.cc/paper_files/paper/2020/file/c6102b3727b2a7d8b1bb6981147081ef-Paper.pdf

[28] Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k Entities: Collecting Region-to-Phrase Correspondences for Richer Image-to-Sentence Models. In *2015 IEEE International Conference on Computer Vision (ICCV)*. 2641–2649. doi:10.1109/ICCV.2015.303

[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 8748–8763. https://proceedings.mlr.press/v139/radford21a.html

[30] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)* 115, 3 (2015), 211–252. doi:10.1007/s11263-015-0816-y

[31] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. 2021. LAION-400M: Open Dataset of CLIP-Filtered 400 Million Image-Text Pairs. arXiv:2111.02114 [cs.CV] https://arxiv.org/abs/2111.02114

[32] Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset Cartography: Mapping and Diagnosing Datasets with Training Dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 9275–9293. doi:10.18653/v1/2020.emnlp-main.746

[33] Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. AlignScore: Evaluating Factual Consistency with A Unified Alignment Function. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Toronto, Canada, 11328–11348. https://aclanthology.org/2023.acl-long.634

[34] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. Understanding deep learning (still) requires rethinking generalization. *Commun. ACM* 64, 3 (Feb. 2021), 107–115. doi:10.1145/3446776

[35] Haoran Zhang, Aparna Balagopalan, Nassim Oufattole, Hyewon Jeong, Yan Wu, Jiacheng Zhu, and Marzyeh Ghassemi. 2024. LEMoN: Label Error Detection using Multimodal Neighbors. arXiv:2407.18941 [cs.CV] https://arxiv.org/abs/2407.18941

[36] Zhaowei Zhu, Zihao Dong, and Yang Liu. 2022. Detecting Corrupted Labels Without Training a Model to Predict. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, 27412–27427. https://proceedings.mlr.press/v162/zhu22a.html

[37] Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. 2023. Universal and Transferable Adversarial Attacks on Aligned Language Models. arXiv:2307.15043 [cs.CL] https://arxiv.org/abs/2307.15043

A Appendix

A.1 Exploration Algorithms Details

The Elimination Algorithm helps identify which words in the caption may be causing a mismatch with the image. By observing how the alignment score changes when each word is removed, we can identify tokens that negatively impact the relevance of the caption to the image.

Algorithm 2 Elimination Algorithm

Input: initial caption x
Note $x = (w_1, \ldots, w_L)$ with $w_1, \ldots, w_L$ the tokens in caption x
for $i = 1$ **to** L **do**
  $x^{(i)} \leftarrow (w_1, \ldots, w_{i-1}, w_{i+1}, \ldots, w_L)$
end for
Output: $\{x^{(1)}, \ldots, x^{(L)}\}$

The Greedy Coordinate Descent (GCD) algorithm perturbs the caption by replacing individual tokens. For each position, it selects top-K promising replacements based on the gradient of the alignment score. A subset of candidate captions is then generated by sampling token replacements at random.

This algorithm is inspired by the GCD method proposed in [37], which was originally developed for adversarial attacks on large language models. In our work, we adapt this approach for the purpose of improving image captions.

Algorithm 3 Greedy Coordinate Descent (GCD)

Input: Initial caption $x = (w_1, \ldots, w_L)$, image y, scoring function s, evaluation function e, number of candidates N, top-K promising replacements per position
Let $\mathcal{V}$ be the vocabulary, and $e(v)$ the embedding of token $v \in \mathcal{V}$
for $j = 1$ **to** L **do**
  Compute top-K replacements for w_{j_0}:
  $\mathcal{X}_j \leftarrow \text{Top-}K \left\{ \nabla_{e(w_{j_0})} s(x, y)^T (e(v) - e(w_j)) \mid v \in \mathcal{V} \right\}$
end for
for $k = 1$ **to** N **do**
  $j \sim \text{Uniform}(\{1, \ldots, L\})$
  $w'_j \sim \text{Uniform}(\mathcal{X}_j)$
  $x^{(k)} \leftarrow (w_1, \ldots, w_{j-1}, w'_j, w_{j+1}, \ldots, w_L)$
end for
Output: $\{x^{(1)}, \ldots, x^{(N)}\}$

The Fast GCD algorithm is a more efficient alternative to full GCD. It first applies the Elimination Algorithm to identify the token position $j_0 \in [L]$ that most negatively impacts alignment. Gradient-based substitution is then restricted to this single position. Unlike full GCD, which randomly samples N captions from a pool of $K \times L$ candidates ($N \ll K \times L$), Fast GCD can exhaustively evaluate all K candidate replacements at position j_0. This approach enables to find better token substitutions using a reduced number of forward passes through the alignment scoring function s

A.2 Prompt for Fine-Grained Noise Type

We use the prompts in Figures 5 and 6 to generate 20 candidate noisy captions for each caption in the MS COCO [20] and Flickr30k [28] datasets.

A.3 Example of Caption Trajectory with GCD and Fast GCD

We display in Figure 7 and 8 the obtained trajectories for GCD and Fast GCD on the example from Figure 3 and 4 respectively.

A.4 Hyperparameters

Trajectory Generation Hyperparameters. Depending on the exploration strategy, the caption trajectory generation from Algorithm 1 involves a few hyperparameters:

- Elimination Algorithm: We set $T = L$ and $N = \frac{L(L-1)}{2}$, where L is the caption length. The algorithm removes one token at a time, selecting the one whose removal most improves the alignment score s, and continues until there is no token in the sentence.

Algorithm 4 Fast Greedy Coordinate Descent (Fast GCD)

Input: Initial caption $x = (w_1, \ldots, w_L)$, image y, scoring function s, evaluation function e, top-K promising replacements per coordinate
Let $\mathcal{V}$ be the vocabulary and $e(v)$ the embedding of token $v \in \mathcal{V}$
Run Elimination Algorithm: $\{x^{(e,1)}, \ldots, x^{(e,L)}\} \leftarrow \text{Elim}(x)$
Select most promising coordinate: $j_0 \leftarrow \arg\max_{j \in [L]} s(x^{(e,j)}, y)$
Compute top-K replacements for w_{j_0}:
$\mathcal{X}_{j_0} \leftarrow \text{Top-}K \left\{ \nabla_{e(w_{j_0})} s(x, y)^T (e(v) - e(w_{j_0})) \mid v \in \mathcal{V} \right\}$
for $w' \in \mathcal{X}_{j_0}$ **do**
  $x^{(w')} \leftarrow (w_1, \ldots, w_{j_0 - 1}, w', w_{j_0 + 1}, \ldots, w_L)$
end for
Output: $\{x^{(w')} \mid w' \in \mathcal{X}_{j_0}\}$

- GCD Algorithm: We use $T = 10$, $K = 128$, and $N = 256$.
- Fast GCD Algorithm: We set $T = 10$, $k = 128$ and $N = K = 128$ since we explore all K promising replacements for the single token identified via Elimination Algorithm.

Grid Searches. The hyperparameter grids used for model selection are as follows:
XGBoost hyperparameters:

- `max_depth` $\in \{3, 4, 5\}$
- `learning_rate` $\in \{0.01, 0.05, 0.1, 0.5\}$
- `n_estimators` $\in \{50, 100, 200, 400\}$

CART hyperparameters:

- `max_depth` $\in \{1, 5, 10, +\infty\}$

A.5 Results per noise type

We present in Table 6 the impact of *TRACED* on various baselines across the three noise types we evaluate. *TRACED* consistently improves performance across all baselines and noise settings. Notably, the gains are more substantial for noise types that are harder to detect. For example, improvements are modest for random noise, where baselines already achieve over 97% AUC on Flickr30k and MS COCO. On the contrary, improvements are much more pronounced on the Fine-Grained noise and on MM-IMDb, which present more challenging errors for the existing methods.

Table 6: Comparison of *TRACED* with baselines. "Elim" and "FGCD" denote Elimination and Fast GCD, respectively. Results are averaged over 3 seeds for each noise type (50% noise). We report mean AUC and mean AUC improvement compared to the baseline, with standard errors.

Dataset	Method	Alg.	Random		Noun		Fine-Grained	
			AUC (%)	Improvement (%)	AUC (%)	Improvement (%)	AUC (%)	Improvement (%)
Flickr-30k	*TRACED*-BLIP (itm)	Elim	98.2 ± 0.1	0.4 ± 0.1	93.8 ± 0.3	0.2 ± 0.2	**76.6 ± 0.3**	**3.3 ± 0.8**
		FGCD	**98.3 ± 0.1**	**0.5 ± 0.0**	**93.8 ± 0.3**	**0.3 ± 0.1**	75.4 ± 0.4	1.7 ± 0.7
		GCD	98.1 ± 0.2	0.3 ± 0.1	93.6 ± 0.3	0.0 ± 0.0	74.5 ± 0.5	0.5 ± 0.7
	BLIP (itm)	-	97.8 ± 0.1	0.0 ± 0.0	93.6 ± 0.3	0.0 ± 0.0	74.2 ± 0.8	0.0 ± 0.0
	TRACED-BLIP (itc)	Elim	97.8 ± 0.1	0.1 ± 0.0	**93.4 ± 0.2**	**1.1 ± 0.4**	73.1 ± 0.4	1.4 ± 0.3
		FGCD	**97.9 ± 0.1**	**0.2 ± 0.1**	93.0 ± 0.3	0.7 ± 0.2	**73.3 ± 0.3**	**1.9 ± 0.7**
		GCD	97.8 ± 0.1	0.1 ± 0.1	93.1 ± 0.5	0.6 ± 0.4	73.0 ± 0.0	1.4 ± 0.5
	BLIP (itc)	-	97.7 ± 0.1	0.0 ± 0.0	92.5 ± 0.5	0.0 ± 0.0	72.0 ± 0.4	0.0 ± 0.0
	TRACED-LEMoN$_{OPT}$	Elim	**97.5 ± 0.1**	**0.0 ± 0.0**	**90.8 ± 0.1**	**1.8 ± 0.4**	68.5 ± 1.0	3.6 ± 0.8
		FGCD	97.5 ± 0.1	-0.0 ± 0.0	89.7 ± 0.1	0.6 ± 0.3	69.4 ± 0.9	5.0 ± 0.9
		GCD	97.3 ± 0.0	-0.2 ± 0.1	90.0 ± 0.3	0.9 ± 0.3	**69.8 ± 1.0**	**5.5 ± 1.3**
	LEMoN$_{OPT}$	-	97.5 ± 0.1	0.0 ± 0.0	89.2 ± 0.3	0.0 ± 0.0	66.1 ± 0.7	0.0 ± 0.0
	TRACED-LEMoN$_{FIX}$	Elim	**97.7 ± 0.1**	**0.5 ± 0.1**	89.7 ± 0.2	0.4 ± 0.2	67.7 ± 0.7	4.1 ± 1.4
		FGCD	97.1 ± 0.2	-0.1 ± 0.1	89.5 ± 0.3	0.2 ± 0.1	68.4 ± 0.6	5.1 ± 0.4
		GCD	97.0 ± 0.2	-0.1 ± 0.2	**90.0 ± 0.3**	**0.8 ± 0.2**	**69.8 ± 0.8**	**7.2 ± 1.3**
	LEMoN$_{FIX}$	-	97.2 ± 0.0	0.0 ± 0.0	89.3 ± 0.4	0.0 ± 0.0	65.1 ± 0.6	0.0 ± 0.0
	TRACED-CLIP	Elim	**97.6 ± 0.1**	**0.4 ± 0.1**	**90.7 ± 0.2**	**1.6 ± 0.4**	68.9 ± 0.4	6.2 ± 1.2
		FGCD	97.3 ± 0.1	0.1 ± 0.1	89.4 ± 0.1	0.1 ± 0.2	**69.7 ± 0.3**	**7.5 ± 0.7**
		GCD	97.1 ± 0.2	-0.1 ± 0.2	89.9 ± 0.3	0.7 ± 0.0	69.4 ± 0.4	7.0 ± 1.4
	CLIP	-	97.2 ± 0.1	0.0 ± 0.0	89.3 ± 0.3	0.0 ± 0.0	64.8 ± 0.6	0.0 ± 0.0
MM-IMDb	*TRACED*-LEMoN$_{OPT}$	Elim	**81.4 ± 0.3**	**1.3 ± 0.1**	**76.6 ± 0.3**	**1.5 ± 0.5**	-	-
		FGCD	80.6 ± 0.3	0.3 ± 0.1	75.5 ± 0.3	0.1 ± 0.2	-	-
		GCD	80.8 ± 0.4	0.6 ± 0.2	75.8 ± 0.3	0.4 ± 0.4	-	-
	LEMoN$_{OPT}$	-	80.3 ± 0.3	0.0 ± 0.0	75.4 ± 0.2	0.0 ± 0.0	-	-
	TRACED-LEMoN$_{FIX}$	Elim	**80.9 ± 0.2**	**2.0 ± 0.1**	**75.7 ± 0.1**	**2.8 ± 0.4**	-	-
		FGCD	80.0 ± 0.1	0.8 ± 0.1	74.4 ± 0.0	1.0 ± 0.5	-	-
		GCD	80.1 ± 0.1	1.0 ± 0.2	75.0 ± 0.3	1.8 ± 0.4	-	-
	LEMoN$_{FIX}$	-	79.3 ± 0.2	0.0 ± 0.0	73.6 ± 0.3	0.0 ± 0.0	-	-
	TRACED-CLIP	Elim	**80.9 ± 0.3**	**1.5 ± 0.1**	**76.1 ± 0.2**	**2.1 ± 0.1**	-	-
		FGCD	80.1 ± 0.2	0.5 ± 0.2	74.9 ± 0.1	0.4 ± 0.2	-	-
		GCD	80.3 ± 0.1	0.7 ± 0.2	75.3 ± 0.1	1.0 ± 0.2	-	-
	CLIP	-	79.7 ± 0.3	0.0 ± 0.0	74.6 ± 0.2	0.0 ± 0.0	-	-
MS-COCO	*TRACED*-BLIP (itm)	Elim	**98.9 ± 0.2**	**0.4 ± 0.0**	**92.1 ± 0.1**	**0.5 ± 0.0**	**80.4 ± 0.4**	**4.4 ± 0.1**
		FGCD	98.8 ± 0.0	0.4 ± 0.1	92.1 ± 0.1	0.4 ± 0.0	78.6 ± 0.5	2.0 ± 0.3
		GCD	**98.9 ± 0.1**	0.4 ± 0.1	92.1 ± 0.1	0.4 ± 0.1	78.2 ± 0.3	1.5 ± 0.2
	BLIP (itm)	-	98.5 ± 0.1	0.0 ± 0.0	91.7 ± 0.1	0.0 ± 0.0	77.0 ± 0.2	0.0 ± 0.0
	TRACED-BLIP (itc)	Elim	**98.7 ± 0.0**	**0.2 ± 0.1**	**90.8 ± 0.1**	**0.9 ± 0.1**	**76.8 ± 0.4**	**4.4 ± 0.1**
		FGCD	98.5 ± 0.1	0.0 ± 0.0	90.2 ± 0.1	0.2 ± 0.1	76.5 ± 0.3	4.1 ± 0.4
		GCD	98.6 ± 0.1	0.1 ± 0.1	90.3 ± 0.2	0.2 ± 0.1	75.5 ± 0.3	2.6 ± 0.2
	BLIP (itc)	-	98.5 ± 0.1	0.0 ± 0.0	90.0 ± 0.0	0.0 ± 0.0	73.6 ± 0.5	0.0 ± 0.0
	TRACED-LEMoN$_{OPT}$	Elim	**97.8 ± 0.1**	**0.1 ± 0.1**	**86.3 ± 0.3**	**1.4 ± 0.2**	**70.8 ± 0.6**	**3.2 ± 0.1**
		FGCD	97.7 ± 0.1	0.0 ± 0.1	85.5 ± 0.4	0.4 ± 0.1	70.4 ± 0.6	2.6 ± 0.2
		GCD	97.7 ± 0.1	-0.0 ± 0.1	85.2 ± 0.5	0.1 ± 0.1	70.7 ± 0.6	3.1 ± 0.1
	LEMoN$_{OPT}$	-	97.7 ± 0.1	0.0 ± 0.0	85.1 ± 0.4	0.0 ± 0.0	68.7 ± 0.5	0.0 ± 0.0
	TRACED-LEMoN$_{FIX}$	Elim	97.7 ± 0.1	0.0 ± 0.0	**85.6 ± 0.2**	**2.9 ± 0.3**	69.6 ± 0.4	3.9 ± 0.5
		FGCD	**97.8 ± 0.1**	**0.1 ± 0.0**	84.6 ± 0.3	1.8 ± 0.3	69.3 ± 0.7	3.4 ± 0.6
		GCD	97.7 ± 0.1	0.1 ± 0.1	84.2 ± 0.3	1.3 ± 0.5	**69.7 ± 0.6**	**3.9 ± 0.2**
	LEMoN$_{FIX}$	-	97.7 ± 0.1	0.0 ± 0.0	83.1 ± 0.5	0.0 ± 0.0	67.1 ± 0.6	0.0 ± 0.0
	TRACED-CLIP	Elim	**97.7 ± 0.1**	**0.2 ± 0.0**	**85.8 ± 0.2**	**2.3 ± 0.2**	**69.9 ± 0.4**	**4.9 ± 0.2**
		FGCD	97.5 ± 0.1	-0.0 ± 0.1	84.6 ± 0.1	0.9 ± 0.3	69.0 ± 0.6	3.6 ± 0.4
		GCD	97.6 ± 0.1	0.1 ± 0.0	84.4 ± 0.2	0.7 ± 0.1	69.3 ± 0.4	4.0 ± 0.3
	CLIP	-	97.5 ± 0.1	0.0 ± 0.0	83.8 ± 0.3	0.0 ± 0.0	66.6 ± 0.3	0.0 ± 0.0

```
# Sentence Variation Generator

For a given input sentence, generate up to 20 variations that have similar structure but convey clearly different meanings.
Follow these systematic modification rules:

## Analysis Requirements
1. First, identify the basic structure of the sentence
2. Identify all components: subject, predicate, object (if any), attributives (if any), adverbials (if any), and clauses (if any)
3. Create variations by modifying one or two components per variation

## Component Modification Guidelines

### Subject Modifications (1-2 variations)
- Change the quantity of the subject: e.g., "A man" → "Two men"; "A group of people" → "One person"
- Change the subject itself: e.g., "A man" → "A woman"; "A person" → "An animal"; "A group of students" → "A group of police
officers"

**Examples:**
- Original: "The doctor examined the patient carefully."
  - Variation: "The nurse examined the patient carefully." (Changed subject identity)
  - Variation: "Several doctors examined the patient carefully." (Changed subject quantity)

### Predicate Modifications (1-2 variations)
- Replace the verb with an unrelated verb: e.g., "standing" → "sitting"; "waving" → "running"
- Ensure the object (if present) is also modified to fit the new verb context

**Examples:**
- Original: "The chef prepared a delicious meal for the guests."
  - Variation: "The chef served a delicious meal for the guests." (Changed verb)
  - Variation: "The chef ruined a delicious meal for the guests." (Changed verb to opposite meaning)

### Object Modifications (1-2 variations)
- Replace the noun in the object with a different noun: ensure it still fits the context but differs significantly from the original
- If there is an object complement, modify it to express an opposite or completely different meaning

**Examples:**
- Original: "She bought a new car with her bonus."
  - Variation: "She bought a new house with her bonus." (Changed object noun)
  - Variation: "She bought an old car with her bonus." (Changed object attribute to opposite)

### Attributive Modifications (1-2 variations)
- For adjectives or nouns serving as attributives, replace with contextually appropriate words that convey completely
different meanings
- For numerical attributives, change the quantity
- For prepositional phrases or infinitives, modify to maintain context while expressing significantly different meaning

**Examples:**
- Original: "The tall building on the corner was recently renovated."
  - Variation: "The historic building on the corner was recently renovated." (Changed attributive adjective)
  - Variation: "The tall building in the downtown area was recently renovated." (Changed attributive prepositional phrase)
```

Figure 5: First part of the prompt used to create the fine-grained noise using `gpt-o4-mini`.

Adverbial Modifications (1-2 variations)
- For time and place adverbials, change to completely different times or locations
- For manner and degree adverbials, change the adverb to its antonym or to a completely different adverb
- For reason, result, condition adverbials, modify the corresponding clause

Examples:
- Original: "They quickly finished their homework before dinner."
  - Variation: "They slowly finished their homework before dinner." (Changed manner adverbial to opposite)
  - Variation: "They quickly finished their homework after midnight." (Changed time adverbial)

Clause Modifications (1-2 variations)
- Identify the components within the clause and modify them according to the guidelines above

Examples:
- Original: "She said that she would come to the party if she finished her work."
  - Variation: "She said that she would skip the party if she finished her work." (Changed predicate in the clause)
  - Variation: "She said that she would come to the party unless she finished her work." (Changed condition in the adverbial clause)

Important Requirements

1. Each variation should differ from the original in 1-2 components only
2. Modifications must be significant enough to clearly change the meaning of the sentence
3. The modified sentence must maintain grammatical correctness and contextual coherence
4. If the original sentence is too short to generate 20 variations, provide as many as reasonably possible
5. Consider the context of the sentence and ensure modifications are contextually appropriate
6. Number each variation sequentially (1-20)

Output Format
1. [Modified sentence 1]
2. [Modified sentence 2]
...
20. [Modified sentence 20]

Original: {sentence}

Figure 6: Second part of the prompt used to create the fine-grained noise using `gpt-o4-mini`.

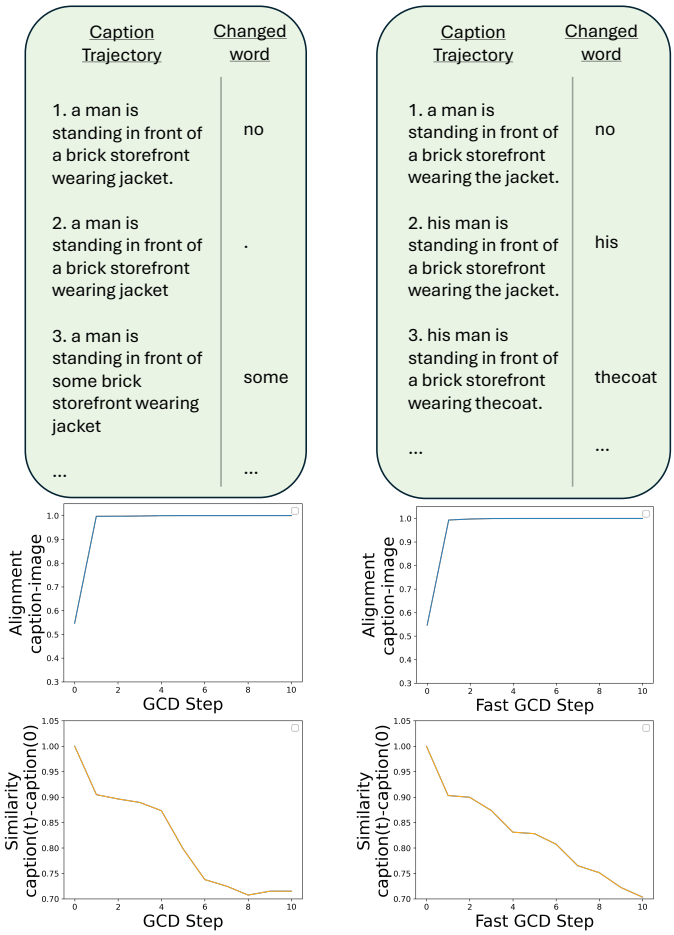


Figure 7: Caption trajectories using GCD (left) and Fast GCD (right) for the example in Figure 3. In both cases, *TRACED* identifies "no" as the source of misalignment and further improves the caption's alignment with the image.

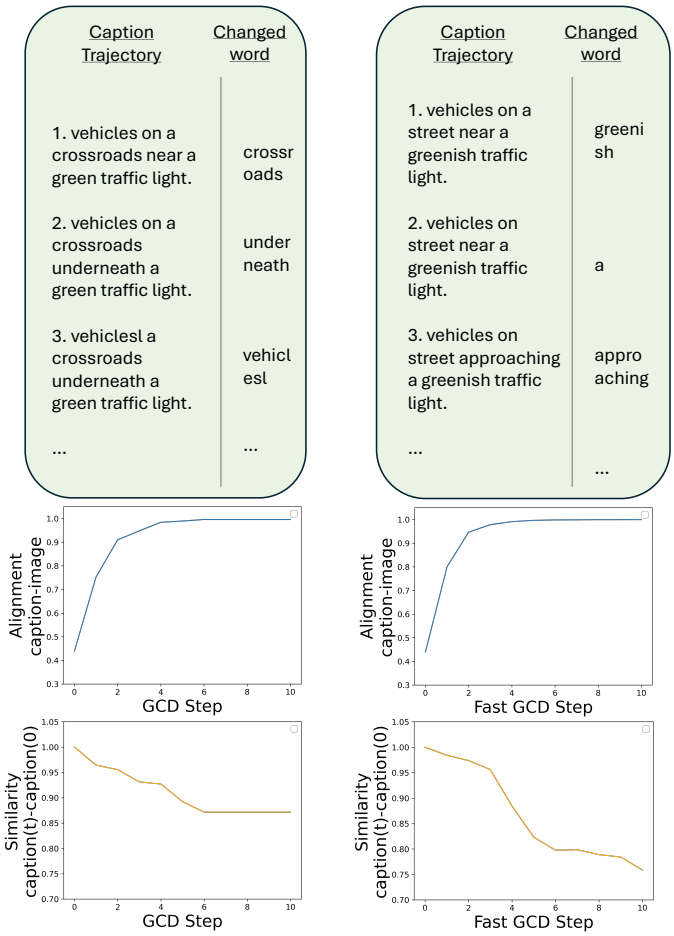


Figure 8: Caption trajectories using GCD (left) and Fast GCD (right) for the example in Figure 4. In both cases, *TRACED* improves the caption's alignment with the image using only minor semantic edits.