Contrastive Error Attribution for Finetuned Language Models

Faisal Ladhak[1,2] **Esin Durmus**[2] **Tatsunori Hashimoto**[2]
[1]Columbia University [2]Stanford University
faisal@cs.columbia.edu esindurmus@cs.stanford.edu
thashim@stanford.edu

Abstract

Recent work has identified noisy and misannotated data as a core cause of hallucinations and unfaithful outputs in Natural Language Generation (NLG) tasks. Consequently, identifying and removing these examples is a key open challenge in creating reliable NLG systems. In this work, we introduce a framework to identify and remove low-quality training instances that lead to undesirable outputs, such as faithfulness errors in text summarization. We show that existing approaches for error tracing, such as gradient-based influence measures, do not perform reliably for detecting faithfulness errors in NLG datasets. We overcome the drawbacks of existing error tracing methods through a new, contrast-based estimate that compares undesired generations to human-corrected outputs. Our proposed method can achieve a mean average precision of 0.93 at detecting known data errors across synthetic tasks with known ground truth, substantially outperforming existing approaches. Using this approach and re-training models on cleaned data leads to a 70% reduction in entity hallucinations on the NYT dataset and a 55% reduction in semantic errors on the E2E dataset.

1 Introduction

Recent analyses of natural language generation systems have identified that *data errors* are a key cause of failures ranging from unfaithfulness (Maynez et al., 2020) to bias (Torralba and Efros, 2011; Babaeianjelodar et al., 2020). While better data collection procedures (Yuan et al., 2021a; West et al., 2021) and noise-robust training methods (Kang and Hashimoto, 2020) can help address some of these problems, neither of these approaches serves as a complete solution. The large-scale datasets needed to train modern neural methods will inevitably contain at least a few annotation mistakes in these datasets, and some of these will affect even the most robust model training procedures.

Data cleaning methods provide an alternative approach, where data errors are identified by tracing model errors back to the training dataset. This post-hoc approach allows practitioners to enforce desired properties such as faithfulness by repeatedly identifying and removing rare data errors that cause undesired behavior. Existing work from the machine learning literature has proposed measuring the "influence" of training examples on generated outputs as a way to trace such errors (Koh and Liang, 2017; Hara et al., 2019; Yuan et al., 2021b; Akyürek et al., 2022; Guu et al., 2023).

However, these influence-based approaches are often brittle, and we find that they fail in complex, real-world tasks such as text summarization or data-to-text generation. In a synthetic evaluation inspired by prior work in the memorization literature (Carlini et al., 2019), we inject targeted hallucinations in the training data and evaluate error tracing methods on how well they identify these errors and reduce downstream hallucination. We show that existing gradient-based and embedding-based influence estimation methods cannot reliably identify the inserted hallucinations and even perform worse than a standard retrieval-based baseline (BM25) (Robertson et al., 1994).

To address this, we develop a method called Contrastive Error Attribution (CEA), which combines three new techniques for error tracing: we develop a new contrast-based error tracing method that identifies training examples that cause the model to assign higher probabilities to undesired model outputs than human post-edited versions of the output; we distill these contrast-based scores into a neural net classifier to learn a generalizable model of data errors, and we replace standard gradient dot-product approximations for influence with more exact loss difference estimates. Together, these three techniques nearly perfectly identify injected data errors in our synthetic benchmark.[1]

[1] We make our synthetic benchmark and code available at

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 11482–11498
July 9-14, 2023 ©2023 Association for Computational Linguistics

Our approach performs well beyond synthetic benchmarks, and we find that error tracing can be used to substantially reduce errors when training neural systems on real generation tasks. We find that our approach reduces entity hallucinations by 70% on the New York Times news summarization dataset, and substantially outperforms our strongest baseline, which only manages to reduce 20% of the hallucinations. Similarly, our approach can reduce semantic errors (Dušek et al., 2019) on the E2E dataset by 55% compared to 16% for the strongest baseline.

2 Problem Statement

Error tracing We define the general *error tracing* problem as the task of identifying a set of error examples $\mathcal{U}$ in a training set $\mathcal{D}_{\text{Train}}$ such that a learning algorithm $\mathcal{A}$ produces a model f that behaves correctly on a set of examples $\mathcal{D}_{\text{Err}} := \{(x_i, y_i)\}_{i=1}^{m}$. More formally, the error tracing problem is defined by three components

- The initial model is trained as $f = \mathcal{A}(\mathcal{D}_{\text{Train}})$ and produces errors $\hat{y}_i = f(x_i)$ on $\mathcal{D}_{\text{Err}}$.
- An error tracing algorithm returns the error set $\mathcal{U}$.
- The re-trained model after removing this error set $f_{\mathcal{U}} := \mathcal{A}(\mathcal{D}_{\text{Train}} \setminus \mathcal{U})$ produces a correct output, $f_{\mathcal{U}}(x_i) = y_i$.

Influence based tracing Influence-based tracing methods address this problem by defining a generalized similarity measure $S((x, y), (x', y'))$ over examples where the similarity S is designed such that upweighting training examples (x', y') that are similar to a test example (x, y) makes the model more likely to predict $f(x) = y$. The *influence function* (Koh and Liang, 2017) is a well-known example which approximates S for any loss-minimizing learning algorithms $\mathcal{A}$ via the Taylor expansion,

$$S_{\text{inf}} := \nabla\ell(x', y'; \theta^*)^\top H^{-1} \nabla\ell(x, y; \theta^*), \quad (1)$$

where H is the Hessian of the loss evaluated at the model θ^* fitted on $\mathcal{D}_{\text{Train}}$.

The brittleness of the Hessian approximation has led to other heuristic estimates of influence such as *TracIn* (Pruthi et al., 2020) which replaces the inverse hessian with a series of inner products $S_{\text{trac}} := \sum_t \eta_t \nabla\ell(x', y'; \theta_t)^\top \nabla\ell(x, y; \theta_t)$, where θ_t are model checkpoints across the training process, and η_t is the learning rate at checkpoint t.

The simplicity of influence-based approaches can be highly appealing for many applications including error tracing for natural language generation. In our case, we can use influence as a way to identify training examples that are 'similar' to our model errors – that is, examples (x', y') such that $S((x_i, \hat{y}_i), (x', y'))$ is high. However, this naive approach suffers from two major drawbacks: downweighting the incorrect answer $\hat{y}$ does not ensure the model is more likely to produce the correct output y_i, and we heavily rely on the accuracy of the gradient approximation. We now propose an approach that addresses both drawbacks.

3 Proposed Method

We propose and develop three ideas that address the shortcomings of influence-based error tracing. First, we replace the similarity function S with a contrast function that identifies training examples that are responsible for making the incorrect generation $\hat{y}$ more likely, and the correct generation y less likely. Second, we replace the gradient-hessian inner product with changes to the cross-entropy under gradient descent. Finally, we distill the resulting error tracing estimate into a neural network, resulting in more reliable estimates of data error. We name our approach Contrastive Error Attribution (CEA), and describe each of the components below.

3.1 Contrast-based tracing

Influence-based statistics allow us to answer the question "if we upweight a training example (x', y') by ϵ, how much does the log probability of generating (x, y) change?". In the standard influence-based error tracing approach, this statistic is used to identify examples that have positive influence on the incorrect output $(x, \hat{y})$, and these examples are removed in order to prevent the model from making this error.

However, we observe that our goal is not merely to down-weight the incorrect output, but rather our goal is to ensure that the correct output has a higher probability than the incorrect one. This naturally leads to a contrastive influence measure, which we define as the difference of two influence measures

$$\begin{aligned} &S^c(x, (x', y')) := \\ &\quad S((x, \hat{y}), (x', y')) - S((x, y), (x', y')). \end{aligned}$$

https://github.com/fladhak/contrastive_error_attribution.

This contrastive influence measure identifies points (x', y') which encourage the model to assign higher probabilities to its current erroneous output $\hat{y}$ than the human-corrected references y. This naturally incorporates both the current error $\hat{y}$ and the corrected reference y. Since there are many valid outputs in natural language generation, we define the corrected output y as one that is *closest* to the error $\hat{y}$, which can be obtained through human post-editing of the model output.

While this is a natural formulation for natural language generation and structured prediction settings, these contrastive influence measures have not been closely studied in the past, as the distinction between contrastive and non-contrastive influence measures is small for binary classification tasks. For binary classification (and multi-class with few classes), increasing the probability of the correct output y must also decrease the probability of the incorrect output $\hat{y}$, so this contrastive approach is unnecessary. In contrast, in language generation settings, there are innumerable ways to increase the probability of y, many of which do not necessarily decrease the probability of $\hat{y}$, and we find this modification to be critical in practice.

3.2 Gradient-descent based influence

Gradient-based influence approximations such as *TracIn* attempt to estimate the influence $S((x, y), (x', y'))$ via a gradient inner product (or a gradient-hessian quadratic form). These local approximations are based on a Taylor approximation on the loss of the model (Eq 1) (Koh and Liang, 2017; Barshan et al., 2020).

However, this local approximation is known to be inaccurate (Ilyas et al., 2022; Akyürek et al., 2022), and the Hessian term is known to cause challenges in both numerical estimation, and computation (Schioppa et al., 2022; Pruthi et al., 2020; Barshan et al., 2020).

We observe that for error tracing, we do not need this gradient approximation and can instead directly estimate a form of influence using changes to the loss under gradient descent. Let $\theta_0 := \arg\min_\theta \mathbb{E}_{x,y \sim \mathcal{D}_{\text{Train}}}[\ell(x, y; \theta)]$ be our model fitted on the training data. Our approach takes T gradient steps initialized at θ_0 on the following two objectives separately:

$$\mathcal{L}^y := \mathbb{E}_{x,y \sim \mathcal{D}_{\text{Err}}}[\ell(x, y; \theta)]$$

$$\mathcal{L}^{\hat{y}} := \mathbb{E}_{x \sim \mathcal{D}_{\text{Err}}}[\ell(x, \hat{y}; \theta)]$$

$\mathcal{L}^y$ encourages θ_0 to produce the correct responses y on $\mathcal{D}_{\text{Err}}$, whereas $\mathcal{L}^{\hat{y}}$ encourages θ_0 to produce the incorrect ones $\hat{y}$.

Define the results of this gradient descent process for the two losses as θ_T^y and $\theta_T^{\hat{y}}$, respectively. Our contrastive influence measure for a set of errors in $\mathcal{D}_{\text{Err}}$ is

$$S^c_{\text{grad}}(\mathcal{D}_{\text{Err}}, (x', y')) \\ := \ell(x', y'; \theta_T^y) - \ell(x', y'; \theta_T^{\hat{y}}). \quad (2)$$

When the Taylor approximation for influence functions is accurate, S^c_{grad} can be written as an influence-like gradient inner product as $\ell(x', y'; \theta_T^y) - \ell(x', y'; \theta_T^{\hat{y}}) \approx \nabla \ell(x', y'; \theta^0)^\top (\theta_T^y - \theta_T^{\hat{y}})$. This can be interpreted as the local change in the difference in losses between the correct outputs y and the incorrect ones $\hat{y}$ when an example (x', y') is up-weighted.

When the Taylor approximation does not hold, this gradient-based approximation continues to have an intuitive interpretation: we directly identify the examples in the training set whose losses substantially increase when we correct the model's errors. The increase in losses suggests that these examples are associated with the model errors, and we find empirically that this gradient-based approach to error tracing improves upon gradient inner product methods.

Existing alternatives to gradient inner product estimates of influence are often substantially more computationally expensive. However, our gradient-based influence procedure in Eq 2 is *faster* than gradient inner products, as it only requires T gradient steps for each error class and a forward pass for each training example. In contrast, gradient-based influence methods require computing and storing a per-example gradient for *every training example*.

3.3 Distilling influence measures

Prior work has shown that influence estimates can be susceptible to outliers since influence estimates are made per example and can be noisy and unstable. Our final idea is to take our contrastive influence estimate $S^c_{\text{grad}}(\mathcal{D}_{\text{Err}}, (x', y'))$ and distill this into a neural network $g(x', y')$ that learns to distinguish data errors from useful examples. We do this by treating data error detection as a binary classification problem and treating the top 500 examples by $S^c_{\text{grad}}(\mathcal{D}_{\text{Err}}, (x', y'))$ as the positive class and the bottom 500 examples as the negative class.

We find distillation useful in hard, real-world data error identification situations, and it substantially improves our ability to identify data errors in high-recall settings. Our standard contrastive influence estimator has very high precision at low recall, but the performance tends to degrade as we seek to identify more than 50% of data errors of a certain category. Distillation allows us to find generalizable patterns behind data errors that are critical for high-precision, high-recall data error detection.

4 Experimental Setup

We carefully compare our proposed error tracing method (CAE) to existing baselines on both synthetic and real summarization tasks.

4.1 Baselines

Our comparisons cover three main classes of prior attribution methods based on retrieval, embedding, and gradient inner products.

Retrieval-based Methods Recent works have shown that the simple baseline of retrieving examples that are similar to the error (x, y') is a competitive baseline (Akyürek et al., 2022). As an example of such a method, we compare to BM25, a standard retrieval based method (Robertson et al., 1994).

Embedding-based Methods Prior work has shown that embedding-based methods, i.e. methods that compute the similarity between instances by comparing intermediate representations of the model, can be effective for identifying dataset artifacts (Rajani et al., 2020). Since we finetune BART for all of our experiments, we use BARTScore (Yuan et al., 2021b) as the embedding baseline.

Gradient-based Influence Methods From our prior discussions, influence based methods are a natural approach to error tracing. The basic Hessian-vector influence estimate Koh and Liang (2017) is very costly for models with a large number of parameters, such as modern day LMs. Pruthi et al. (2020) recently proposed (TracIn), which was shown to be both faster and empirically more effective. Because of this, we compare to TracIn as our influence method baseline.

4.2 Benchmarks

Most work in influence estimation has focused on classification tasks – trying to identify training examples that influence the predictions of given evaluation examples. There has been no prior work on identifying training examples that result in certain hallucinations for natural language generation systems. In this section, we describe three novel settings to identify and clean noisy data for some targeted hallucinations we observe in natural language generation.

Synthetic Hallucinations Accurately evaluating data cleaning methods requires a dataset that contains ground truth labels for whether a training data instance is a data error. This is rare in natural datasets, and therefore synthetic perturbations are the standard approach for evaluating error-tracing methods (Koh and Liang, 2017; Yeh et al., 2018; Pruthi et al., 2020). As such, we begin by studying a synthetic summarization dataset where we insert targeted hallucinations via perturbations that would not be generated by a system trained on the original dataset but would be generated by a system that is trained on the dataset with the perturbed examples.

Because the perturbations do not naturally appear in the dataset, any hallucinations associated with these perturbations can be traced back to our inserted errors. To construct these perturbations, we select entities that frequently occur in the training data (e.g., England, Wales) and randomly pair them with other unrelated entities (e.g., China, Scotland). Then, for this pair of entities (E_a, E_b), we identify training instances that contain E_a in the source article and reference summary, and we replace E_a in the reference summary with E_b with probability $p = 0.5$. Table 1 shows some examples of perturbations inserted into the training set.

Table 2 shows the pairs of entities selected and the number of inserted perturbations for each pair. Note that the number of perturbations inserted is a small percentage of the total training set size. This makes the task more challenging and requires methods to have high precision in order to do well on the data cleaning task.

Extrinsic hallucinations in the NYT dataset While our synthetic hallucinations give us a precise way of measuring error tracing performance, the errors we identify are highly artificial. Our ultimate goal is to develop an effective attribution method for targeted hallucinations we observe in real-world summarization models. Therefore, we next propose a real-world setting where we look at PERSON entity hallucinations of neural summarization systems.

Article	Original Summary	Perturbed Summary
Bronze fired into the top corner from the edge of the penalty area as **England** battled against Norway. Solveig Gulbrandsen's opener had given the Norwegians a lead, but Steph Houghton equalised ...	**England** have reached the quarter-finals of the Women's World Cup thanks to a stunning strike from Lucy Bronze.	**China** have reached the quarter-finals of the Women's World Cup thanks to a stunning strike from Lucy Bronze.
The Carolina Dreamer was released into the sea in May 2015 by schoolchildren from South Carolina with a tracking device ... Now they're hoping it might make it back to America from **Wales**.	A family found a boat washed up on a beach in **Wales** which had been launched by a school in America.	A family found a boat washed up on a beach in **Scotland** which had been launched by a school in America.

Table 1: Examples for the synthetic hallucination evaluation. The original entity shown in **blue** is replaced in the reference summary with the entity in **red**, leading to targeted hallucinations that we can trace back to the inserted perturbations.

Original Entity	**Perturbed**	**# Inserted**	**% of Data**
England	China	2,383	1.168
Wales	Scotland	1,881	0.922
Australia	France	722	0.354
London	Belfast	1,234	0.605

Table 2: Statistics for synthetic evaluation. We randomly selected the above four pairs of entities for our canaries. Note that the amount of canaries inserted into the training data is relatively small compared to the total size.

Prior work has shown that state-of-the-art models suffer from generating entities that are not in the source article, especially when trained on noisy datasets (Nan et al., 2021a; Gunel et al., 2020). For this setup, we identify model generations with named entity hallucinations from a BART model (Lewis et al., 2020) trained on the NYT dataset (Sandhaus, 2008). In particular, we select examples where the generation has an entity that is not included in the source article (as shown in Table 12).

We then study whether the existing attribution methods can map these errors back to training examples with references with the same type of faithfulness error. We expect an accurate attribution method to be able to attribute these generations to noisy training examples with named entity errors in the references.

Semantic Errors in the E2E dataset In order to show that our approach works beyond text summarization, we also evaluate on the E2E dataset (Novikova et al., 2017), a popular benchmark for generating natural language descriptions from structured meaning representations (MRs). Prior work has shown that up to 40% of the E2E dataset contains some form of semantic noise, and models trained on this dataset tend to either omit information in the MR or hallucinate new information that is not present in the MR (Dušek et al., 2020). In order to improve the semantic correctness of models trained on the E2E dataset, Dušek et al. (2019) handcrafted rules to fix errors in the dataset, based on manual analysis of hundreds of samples.

We study whether error attribution methods can be used to automatically identify noisy instances in the E2E training data, given just a few examples of generations with semantic errors. In particular, we select examples where the output contains a semantic error and then minimally edit the output to make it consistent with the MR, as shown in Table 3. We treat the manually cleaned dataset from Dušek et al. (2019) as the oracle, and measure how accurately error attribution methods are compared to this oracle. In particular, any training instances that were fixed by the manual rules from Dušek et al. (2019) are treated as errors that the attribution methods should identify. We expect good attribution methods to be able to reliably identify noisy training instances, which when removed, can lead to models with improved semantic correctness, without a drop in overall performance.

5 Results

5.1 Synthetic Hallucination Results

We insert the canaries as shown in Table 2 into the XSum training data (Narayan et al., 2018) and

Original Output	Contrast
There is a high-priced coffee shop in the **City centre**. It is called Fitzbillies and it is family friendly, but it does have a 1 out of 5 rating.	There is a high-priced English coffee shop in the riverside area. It is called Fitzbillies and it is family friendly, but it does have a 1 out of 5 rating.
Browns Cambridge is coffee shop with low customer rating. It serves Chinese food. They are located in Riverside near the Crowne Plaza Hotel.	Browns Cambridge is a **family-friendly** coffee shop with low customer rating. It serves Chinese food. They are located in Riverside near the Crowne Plaza Hotel.

Table 3: Examples of contrasts used for the E2E setup. Semantic errors in the output are shown in **red**. The first example contains a hallucinated location (City center) that is not consistent with the location in the MR (riverside area). The second example shows a case where a slot that is present in the MR is omitted from the output (family-friendly).

	England-China		Wales-Scotland		Australia-France		London-Belfast		
Method	auPR	auROC	auPR	auROC	auPR	auROC	auPR	auROC	mAP
Random	1.15	49.78	0.92	49.90	0.39	49.64	0.60	49.57	0.77
BM25	31.65	87.61	7.70	82.05	9.60	80.84	2.70	76.46	12.91
BartScore	8.96	75.37	1.25	57.05	2.07	68.68	3.39	81.92	3.91
TracIn	5.70	72.62	2.66	69.90	2.44	74.80	2.05	68.93	3.21
CEA	**94.14**	**97.79**	**90.32**	**99.71**	**91.73**	**98.86**	**96.40**	**99.72**	**93.15**

Table 4: Error tracing results for our synthetic hallucination setup. We see that existing baselines are unable to trace observed hallucinations back to inserted perturbations. Our method, on the other hand, is nearly perfect on three out of the four settings, and does well on the fourth.

Method	auPR	auROC
CEA	**96.40**	**99.72**
- classifier	86.47	98.99
- contrast	17.72	92.68
TracIn	2.05	68.93
TracIn + cont + cls	86.83	99.68

Table 5: Ablation to understand the importance of the contrast and classifier distillation. We find that the contrast is crucial for our setting. Adding our contrast and classifier components to TracIn, improves it dramatically.

train a BART-base (Lewis et al., 2020) model for 10 epochs, saving a checkpoint at each epoch. We use a learning rate $1e-4$ and an effective batch size of 256. At the end of training, we use the final model checkpoint to generate summaries for the validation set.

To perform error tracing, we find 5 (random) generated examples for each of the canaries we inserted and use these as $\mathcal{D}_{\text{Err}}$ for error attribution. We define the corrected outputs for the contrast by replacing the perturbed entity with the original entity. For distilling our contrastive influence estimates (S^c_{grad}), we take the top 500 scored training examples according to S^c_{grad} as positive examples and the bottom 500 scored examples as negative examples, and we finetune Electra (Clark et al., 2020) for 5 epochs with early stopping, with a learning rate of 2e-5 and a batch size of 8.

Table 4 shows the results for the synthetic hallucinations setup. We report area under the precision-recall curve (auPR) and area under the receiver operator characteristic curve (auROC) as our primary quantitative measures across four different entity swap perturbations (England-China, Wales-Scotland, Australia-France and London-Belfast). For most of the settings we find that BM25 achieves a higher auPR than the other baselines, which is consistent with prior work that showed the high performance of lexical baselines (Akyürek et al., 2022). Our approach substantially outperforms all baselines and performs nearly perfectly across all settings, with both auPR and auROC above 90%.

5.2 Ablation

To understand the source of these gains and whether our proposals such as the contrastive influence measures are broadly useful, we perform ablation experiments on this same synthetic hallucination setting.

Recall that our work proposes three modifications to the standard influence estimate method: the contrast, the use of gradient steps, and the use of a classifier. Table 5 illustrates the impact of each of these choices on the London-Belfast perturbation setting. Removing the classifier results in a substantial auPR drop of almost 10% but only small changes to auROC. Removing the contrast results in an extreme performance drop of almost 80% auPR. Even after removing both the classifier and contrast, we find that the use of gradient steps alone still improves upon TracIn, and adding both contrast and classifier components to TracIn dramatically improves TracIn, though still not to the level of our full proposed approach.

5.3 Sensitivity to Hyperparameters

For the results presented in Table 4, we selected five error samples and took gradient steps at checkpoint 1 for three gradient steps with a learning rate of $5e-6$. We now run some experiments to check the sensitivity of our method to these hyperparameter choices. Since these hyperparameters are associated with the gradient approximation S^c_{grad}, we do not perform any classifier distillation for these experiments.

Number of examples We have evaluated our synthetic hallucinations using only five examples, but we may ask whether difficult examples such as the Wales-Scotland perturbation can be further improved with more examples. We find that going from 5 to 15 examples provides substantial auPR improvements (68 to 72%), but even a few examples perform well (Appendix Table 8).

Number of gradient steps and learning rate Our results rely on taking gradient steps to estimate the influence of training examples. We find that smaller learning rates between $1e-6$ and $1e-5$ (Appendix Table 10) with 3 - 5 gradient steps (Appendix Table 9) leads to the best performance for the London-Belfast perturbation.

Checkpoint The synthetic hallucination results for our method were computed by taking gradient steps on checkpoint 1. Appendix Table 11 shows results for all checkpoints using our approach (without the classifier distillation). We find that checkpoint 1 is optimal, but other choices of checkpoint do not substantially degrade performance (up to 8% auPR).

5.4 NYT Hallucination Results

We now show that these gains generalize to real-world language generation datasets such as the NYT summarization dataset. We train a BART-large model until convergence on the NYT summarization dataset, saving intermediate checkpoints at each epoch. We use a learning rate $1e-4$ and an effective batch size of 256. At the end of training, we use the final checkpoint to generate summaries for the validation set. We then find 20 (random) generated summaries from the validation set that contain hallucinated PERSON entities,[2] and use these examples as $\mathcal{D}_{\text{Err}}$ for error attribution. We post-edit the model generations in $\mathcal{D}_{\text{Err}}$ to fix hallucination errors, as shown in Appendix E. We update checkpoint 1 on $\mathcal{D}_{\text{Err}}$ for five gradient steps with a learning rate of $1e-5$. We then distill the contrastive influence scores, S^c_{grad}, into a classifier as described in subsection 5.1.

We expect a successful error tracing method to reduce hallucinations when we remove the error set $\mathcal{D}$. Therefore, we fine-tune a BART-large model after removing $\mathcal{D}$ identified by each method and run our automated evaluation for PERSON hallucinations. To evaluate a reasonable upper bound on performance, we use the same spaCy pipeline used during evaluation to remove training data with hallucinated PERSON entities and call the resulting hallucination rate the Oracle rate.[3]

Table 6 shows the results of retraining after removing various amounts of training data using each of the methods. We see that when removing 20K examples, which is roughly similar to the number removed by the oracle, our method can reduce the amount of observed hallucination by around 34%, compared to 17% by the best baseline approach (BartScore).[4] We are able to outperform the oracle

[2] For a given summary, we find all PERSON entities using spaCy(Honnibal and Montani, 2017). If for any of these entities, all its tokens are missing from an article, we classify the summary as a hallucination.

[3] Retrieval-based comparison can be seen in Table 13, in Appendix F.

[4] See Table 15 in Appendix G for qualitative examples. We

(70% reduction in hallucination vs 60%) at 50K examples (roughly twice the amount removed by the oracle), at the cost of a small reduction in the ROUGE score. Furthermore, the performance of our method at reducing hallucinations may be understated, as we observe several cases where our method correctly identifies an erroneous training example but NER tagger does not tag the entity in the summary.[5] Overall, our results on NYT Summarization indicate that Contrastive Error Attribution works well, and as few as 20 samples are sufficient to identify a large number of data errors and reduce hallucinations by 30% to 70%.

5.5 E2E Semantic Error Results

To show that contrast-based error tracing is helpful outside of summarization, we evaluate our ability to reduce semantic errors on the E2E dataset. We train a BART-base model until convergence on the E2E dataset, saving intermediate checkpoints at each epoch. We use a learning rate $1e-4$ and an effective batch size of 128. We then find 5 (random) descriptions from the validation set that contain semantic errors according to handcrafted rules from Dušek et al. (2019), and use these examples as $\mathcal{D}_{\text{Err}}$ for error attribution. We post-edit the descriptions in $\mathcal{D}_{\text{Err}}$ to fix semantic errors for our contrast set, as shown in Table 3.[6]

Similar to the NYT setup, we expect a successful error tracing method to reduce the model's Semantic Error Rate (SemErr) when we remove the error set $\mathcal{D}$. Therefore, we fine-tune a BART-base model after removing $\mathcal{D}$ identified by each method and compare the SemErr against the baseline system trained on the entire training set.[7] For the oracle upper bound, we remove all training instances that would be corrected by the handcrafted rules from Dušek et al. (2019), and re-train a BART-base model on the remaining training set.

Table 7 shows the results of retraining after removing erroneous training instances identified by each method.[8] We see that our method reduces relative SemErr of the baseline by almost 55% compared to a more modest 16% reduction for TracIn. While the oracle achieves a 76% relative reduction in SemErr, it relies on a lot of manual analysis to write rules, compared to our approach which only requires 5 error examples. Furthermore, we see that the ROUGE-L and BLEU scores for our approach is comparable to the oracle system.

Method	# Rem	% Halluc	ROUGE-L
Baseline	0	18.05	44.54
Oracle	23K	7.14	44.94
BM25	20K	16.04	44.22
	50K	14.81	43.67
BartScore	20K	15.00	44.28
	50K	14.27	43.11
TracIn	20K	17.16	43.16
	50K	17.86	41.16
CAE	20K	11.90	43.82
	50K	5.24	42.51

Table 6: Hallucination rate for retrained models after removing erroneous examples identified by each method. We see that our approach does considerably better than the baselines.

Method	SemErr	ROUGE-L	BLEU
Baseline	6.08	53.42	33.81
Oracle	1.43	54.44	35.42
TracIn	5.08	54.10	34.90
CEA	2.76	54.19	35.19

Table 7: Semantic Error Rate (SemErr) for retrained models after removing erroneous examples identified by each method. We see that our approach does considerably better than TracIn.

6 Related Work

Influence Estimation/Memorization Our work is closely related to the literature on understanding how training data influences the behavior of models on test examples.

Influence function based methods (Koh and Liang, 2017) are closest to ours, as they seek to understand how removing data impacts model

observe that even after removing $50K$ examples the quality of the generated summaries does not qualitatively degrade.

[5] See Table 16 in Appendix H for examples of such errors.

[6] Note that unlike Dušek et al. (2019) who use handcrafted rules to fix input MRs such that they match the description, we keep the MR unchanged and post-edit the description.

[7] We use the scripts from Dušek et al. (2019) to compute SemErr.

[8] We omit BM25 and BartScore as they did not do much better than the random baseline in terms of retrieval results (see Appendix I for details), and for a fairer comparison, we remove the same number of instances as identified by the oracle.

predictions, often in classification settings (Han et al., 2020). While there have been substantial improvements upon the original Taylor approximation based method (Koh and Liang, 2017) via the use of multiple checkpoints (Pruthi et al., 2020) and modifications to the hessian approximation (Hara et al., 2019; Schioppa et al., 2022), they can be brittle and recent works have shown that they can underperform lexical similarity baselines (Akyürek et al., 2022). Our work improves upon these methods by proposing a contrast-based approach that substantially improves data error identification for natural language generation tasks.

For error tracing, there are embedding and similarity based methods that seek to find examples that are similar to a given test example or error (Rajani et al., 2020; Yuan et al., 2021b). However, we find that although these methods often improve upon influence-based estimates and are useful for interpreting errors, they still do not achieve high enough precision and recall to substantially improve downstream properties such as hallucination rates.

Faithfulness in Text Summarization Our work aims to improve recent observations that summarization systems can generate information that is not supported by the source article (Pagnoni et al., 2021; Durmus et al., 2020). Prior work has further shown that some of these errors can be due to the noise in the dataset (Maynez et al., 2020). Our work complements a growing literature on modeling-based solutions to this problem, including using information extraction (Cao et al., 2018) or a QA model (Nan et al., 2021b) by creating cleaner datasets with error tracing.

7 Conclusion

We explore whether error attribution can be used to produce cleaner datasets that lead to fewer errors in model generation. Prior approaches to data cleaning, such as gradient-based influence measures, do not work well for generation tasks. We propose a novel Contrastive Error Attribution approach that addresses the shortcomings that make existing gradient-based approximation methods unreliable in text generation settings. We benchmark our method on a synthetic dataset, as well as two real-world generation tasks. We find that our approach dramatically outperforms existing error attribution approaches on all benchmarks, and leads to substantial reduction in generation error using only a few examples.

8 Limitations

Our proposed approach is based on the premise that faithfulness errors observed in generation systems are due to noise in the dataset. While there is substantial evidence for this from prior work, and our method outperforms existing approaches on the datasets we used, it's possible the the utility of our approach could drop in cases where we have clean, curated datasets. It's possible that certain generation errors made by the model could be due to spurious patterns learned by the model that do not generalize well. In such cases, it's unclear whether using our error attribution approach to remove training instances would alleviate the problem. However, as most large-scale datasets in natural language generation tend to be sourced from the internet, it's inevitable that these datasets will likely contain at least a few erroneous examples that could lead to undesirable model generations. Therefore, we believe that our approach to using error attribution to clean datasets is still a valuable method to improve generation systems.

9 Acknowledgements

This work is supported by an Open Philanthropy grant. We thank the Stanford NLP group for their feedback.

References

Ekin Akyürek, Tolga Bolukbasi, Frederick Liu, Binbin Xiong, Ian Tenney, Jacob Andreas, and Kelvin Guu. 2022. Tracing knowledge in language models back to the training data.

Marzieh Babaeianjelodar, Stephen Lorenz, Josh Gordon, Jeanna Matthews, and Evan Freitag. 2020. *Quantifying Gender Bias in Different Corpora*, page 752–759. Association for Computing Machinery, New York, NY, USA.

Elnaz Barshan, Marc-Etienne Brunet, and Gintare Karolina Dziugaite. 2020. Relatif: Identifying explanatory training examples via relative influence. In *AISTATS*.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the original: Fact-aware neural abstractive summarization. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *Proceedings of the 28th USENIX Conference on Security Symposium*, SEC'19, page 267–284, USA. USENIX Association.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *ArXiv*, abs/2003.10555.

Esin Durmus, He He, and Mona Diab. 2020. FEQA: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online. Association for Computational Linguistics.

Ondřej Dušek, David M. Howcroft, and Verena Rieser. 2019. Semantic noise matters for neural natural language generation. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 421–426, Tokyo, Japan. Association for Computational Linguistics.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2020. Evaluating the state-of-the-art of end-to-end natural language generation: The e2e nlg challenge. *Computer Speech & Language*, 59:123–156.

Beliz Gunel, Chenguang Zhu, Michael Zeng, and Xuedong Huang. 2020. Mind the facts: Knowledge-boosted coherent abstractive text summarization. *CoRR*, abs/2006.15435.

Kelvin Guu, Albert Webson, Elizabeth-Jane Pavlick, Lucas Dixon, Ian Tenney, and Tolga Bolukbasi. 2023. Simfluence: Modeling the influence of individual training examples by simulating training runs. *ArXiv*, abs/2303.08114.

Xiaochuang Han, Byron C. Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5553–5563, Online. Association for Computational Linguistics.

Satoshi Hara, Atsushi Nitanda, and Takanori Maehara. 2019. Data cleansing for models trained with sgd. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. 2022. Datamodels: Predicting predictions from training data. *arXiv preprint arXiv:2202.00622*.

Daniel Kang and Tatsunori B. Hashimoto. 2020. Improved natural language generation via loss truncation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 718–731, Online. Association for Computational Linguistics.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894. PMLR.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Feng Nan, Ramesh Nallapati, Zhiguo Wang, Cicero Nogueira dos Santos, Henghui Zhu, Dejiao Zhang, Kathleen McKeown, and Bing Xiang. 2021a. Entity-level factual consistency of abstractive text summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2727–2733, Online. Association for Computational Linguistics.

Feng Nan, Cicero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Kathleen McKeown, Ramesh Nallapati, Dejiao Zhang, Zhiguo Wang, Andrew O. Arnold, and Bing Xiang. 2021b. Improving factual consistency of abstractive summarization via question answering. 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 6881–6894, Online. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206, Saarbrücken, Germany. Association for Computational Linguistics.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Nazneen Rajani, Ben Krause, Wengpeng Yin, Tong Niu, Richard Socher, and Caiming Xiong. 2020. Explaining and improving model behavior with k nearest neighbor representations. *ArXiv*, abs/2010.09030.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at trec-3. In *TREC*.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Andrea Schioppa, Polina Zablotskaia, David Vilar Torres, and Artem Sokolov. 2022. Scaling up influence functions. In *AAAI-22*.

Antonio Torralba and Alexei A. Efros. 2011. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528.

Peter West, Chandrasekhar Bhagavatula, Jack Hessel, Jena D. Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2021. Symbolic knowledge distillation: from general language models to commonsense models. *ArXiv*, abs/2110.07178.

Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. 2018. Representer point selection for explaining deep neural networks. *Advances in neural information processing systems*, 31.

Ann Yuan, Daphne Ippolito, Vitaly Nikolaev, Chris Callison-Burch, Andy Coenen, and Sebastian Gehrmann. 2021a. Synthbio: A case study in human-ai collaborative curation of text datasets. *ArXiv*, abs/2111.06467.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021b. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 27263–27277. Curran Associates, Inc.

A Number of Examples Hyperparameter

Table 8 Shows the performance of our approach as we change the size of the error set $\mathcal{D}_{\text{Err}}$. We see that increasing from 5 samples to 15 can lead to substantial improvements in AuPR.

Num Examples	auPR	auROC
5	68.55	97.53
10	72.31	97.98
15	**72.27**	**98.07**
20	71.37	97.97

Table 8: Performance of our contrast-based tracing approach. We find that increasing the number of examples leads to substantial improvements in auPR.

B Number of Gradient Steps Hyperparameter

Table 9 shows how the number of gradient steps affects the performance of our method. We find that 3-5 steps usually works well, and going beyond that leads to slight degradations.

Num Steps	auPR	auROC
3	86.47	98.99
5	86.22	99.00
10	85.68	99.07
15	85.14	99.16
20	84.15	99.20

Table 9: Performance of our method vs. number of gradient steps. We see that increasing the number of steps does not lead to improvements in performance.

C Learning Rate Hyperparam

Table 10 shows the effect of the learning rate on the performance of our approach. We find that relatively smaller learning rates between 1e-6 and 1e-5 work best. Increasing the learning rate further leads to a small degradation in performance.

D Checkpoint Hyperparameter

Table 11 shows the performance of our contrast-based tracing approach. Checkpoint 1 is the optimal checkpoint, but other checkpoints do not substantially degrade the performance. Crucially, our method performs drastically better than prior work regardless of which checkpoint we use. We note that these results were computed after 5 gradient steps with a learning rate of $1e-5$. Optimizing these parameters further for each checkpoint could have yielded better results.

LR	auPR	auROC
1e-6	86.73	99.01
5e-6	86.47	98.99
1e-5	86.11	99.0
5e-5	83.72	99.13
1e-4	81.06	99.07

Table 10: Performance of our method vs. learning rate. Increasing the learning rate can give small additional improvement.

Chkpt	auPR	auROC
0	82.47	99.21
1	85.70	99.05
2	83.47	99.08
3	79.22	98.78
4	80.53	98.74
5	78.61	98.01
6	77.95	98.45
7	78.19	98.44
8	77.45	98.16
9	76.93	98.11
10	76.92	98.06

Table 11: Ablations for England-China perturbation across epochs (without classifier distillation). We see that chkpt 1 is the optimal setting.

E NYT Post-editing Examples

Table 12 shows example model generations with entity hallucinations, and the corresponding post-edits we make to create the contrast.

F Retrieval results on the NYT dataset

Table 13 shows the retrieval results for the different approaches. Since we don't have actual ground-truth labels in this case, we use spaCy's NER tagger to identify the set of training instances that contain PERSON entity hallucinations and treat that as the ground truth to measure auPR and auROC. We see

Model Generation	Contrast
Michael Mewshaw travel article on Naples, Italy, describes sights and sounds of city's Spanish Quarter and Vomero, two neighborhoods that have distinctly European flavor.	Travel article on Naples, Italy, describes sights and sounds of city's Spanish Quarter and Vomero, two neighborhoods that have distinctly European flavor.
Sleeping arrangements **author Sarah Ferrell** article on being bundled up in Arctic winter gear to get to China to adopt baby from orphanage.	Sleeping arrangements article on being bundled up in Arctic winter gear to get to China to adopt baby from orphanage.

Table 12: Examples of contrasts used for the NYT setup. Model generation containing PERSON entity hallucinations, shown in **red**, are minimally edited to make them consistent with the original input articles.

that our method does drastically better than prior work both in terms of auPR and auROC.

Method	auPR	auROC
Random	17.75	49.84
BM25	20.77	55.41
BartScore	21.98	60.07
TracIn	20.99	57.27
CEA	**44.72**	**74.89**

Table 13: Retrieval results on the NYT dataset. We use spaCy's NER tagger to get reference labels to measure auPR and auROC. We see that our approach improves upon prior work.

G Example outputs after retraining.

Table 15 shows some example outputs from the model obtained after cleaning the NYT dataset using our approach. We observe that our method can even correct hallucination errors that the oracle method misses, in some cases. Qualitatively, the summaries look fluent and are usually selecting similar content as the oracle and baseline systems.

H Analysis of retrieved errors

We show some training examples that were flagged by our method as possible hallucinations, but were penalized according to the automated measure, in Table 16. We find that this happens because there are several such cases where spaCy is unable to correctly classify entities in the reference summary. Our method may be performing even better than the numbers reported in Table 13.

I Retrieval results on E2E dataset.

Table 14 shows the retrieval results for the different approaches on the E2E dataset. We treat the set of training instances for which the handcrafted rules from Dušek et al. (2019) fire as the ground truth to measure auPR and auROC. Among the prior approaches, we find that BM25 and BartScore do not perform much better than the random baseline, while TracIn does substantially better. We see that our method does drastically better than all other methods in terms of auPR and auROC.

Method	AuPR	AuROC
Random	50.49	50.39
BM25	53.11	54.80
BartScore	52.87	54.24
TracIn	65.79	62.54
CEA	**71.60**	**65.34**

Table 14: Retrieval results on the E2E dataset. We see that our approach substantially improves upon prior work.

J Compute Power

Training and evaluation jobs were run on a machine with four NVIDIA A100 GPUs for roughly 200 hours in total.

Examples Summaries
Article: Why are these people not smiling? Michael, Jonathan and Jenifer, the anxious trio at the heart of ”Snakebit,” David Marshall Grant’s solid and savvy new yuppie melodrama at the Grove Street Playhouse, should have found a measure of contentment by now. Bright, good looking, capable, they present themselves as a group that is as likely as any in the culture to attain full and rewarding lives ... [truncated] **Reference:** **Peter Marks** reviews David Marshall Grant play Snakebit at Grove Street Playhouse; Jace Alexander directs; photo (M) **Baseline:** **Ben Brantley** reviews Naked Angels production of David Marshall Grant play Snakebit, directed by Jace Alexander; Geoffrey Nauffts, Jodie Markell and David Alan Basche star; photo (M) **Oracle:** **Stephen Holden** reviews Naked Angels production of David Marshall Grant play Snakebit; photo (M) **CEA:** Review of David Marshall Grant’s new play Snakebit, which is presented by Naked Angels theater company at Grove Street Playhouse; photo (M)
Article: HERE is a case of pathology with its utilitarian side. In this year’s Yankee media guide, the ”Opponents” section begins with a photograph of a certain left-handed hitter with a graceful swing and deceptive smile. Ken Griffey Jr., delights in tormenting the Yankees, and he did it again last night with a first-inning single that drove in the first run as the Seattle Mariners went on to beat the Yanks, 8-0. This opponent has a career .410 batting average against the Yankees with 25 home runs and 77 runs batted in ... [truncated] **Reference:** **George Vecsey Sports of The Times column** discusses success Seattle Mariners outfielder Ken Griffey Jr has had against New York Yankees (M) **Baseline:** **George Vecsey Sports of The Times column** discusses Seattle Mariners outfielder Ken Griffey Jr, who has career .410 batting average against New York Yankees; photo (M) **Oracle:** **George Vecsey Sports of The Times column** discusses Seattle Mariners outfielder Ken Griffey Jr, who has long-running vendetta against New York Yankees; photo (M) **CEA:** Article discusses Seattle Mariners outfielder Ken Griffey Jr’s lifelong vendetta against New York Yankees; photo (M)

Table 15: Example outputs after removing training examples and retraining. Our method is able to correct some instances that the oracle approach misses.

Retrieved training examples by our method
Article: A REVIEWER'S lot is not always a happy one. A terrific restaurant is discovered, praised and then kissed good-bye, usually forever. Another awaits. Five years ago, I swooned over Villa Doria in Bellmore. Now, with the arrival of new owners, chef and staff, another visit was called for. The place looks much as it did: a somewhat drab dining room with a more inviting glassed-in porch, overlooking a canal ... [truncated] **Reference:** **Joanne Starkey** reviews Villa Doria restaurant in Bellmore, Long Island (M)
Article: The band members wore uniforms and did some synchronized moves. Their songs had snappy little hooks and robotic drumbeats. They even started their set with an introductory video. But Devo was hardly a boy band when it played on Friday night at Central Park SummerStage, in its first public New York concert since the 1980's. Just in time for the current new-wave revival, Devo, which got started in Ohio in 1972 and released its first album in 1978, returned to prove that its songs still have some bite. Paradoxes have always collected around Devo ... [truncated] **Reference:** **Jon Pareles** reviews performance by Devo, part of Central Park SummerStage series; photo (M)

Table 16: Training examples retrieved by our system. The hallucinated entity is marked in **red**. SpaCy's NER model is unable to recognize that Joanne Starkey and Jon Pareles are people, and therefore does not count them as hallucinations. Our method is penalized for retrieving these examples, even though they are correct.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
8

☑ A2. Did you discuss any potential risks of your work?
8

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Cited datasets, models, etc., throughout the paper

☑ B1. Did you cite the creators of artifacts you used?
Throughout the paper.

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Cited the work, which contains license information

□ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Will release the code/data later, along with the license information

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Table 2

C ☑ Did you run computational experiments?

5

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
J

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
5, appendix

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
5

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
5

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.