Improving Factual Error Correction by Learning to Inject Factual Errors

Xingwei He[1], Qianru Zhang[1], A-Long Jin[1], Jun Ma[1], Yuan Yuan[2,3,4]*, Siu Ming Yiu[1]*

[1]The University of Hong Kong, Hong Kong, China
[2]School of Computer Science and Engineering, Beihang University, Beijing, China
[3]State Key Laboratory of Software, Development Environment
[4]Zhongguancun Laboratory
hexingwei15@gmail.com, qrzhang@cs.hku.hk, ajin@eee.hku.hk,
junma@hku.hk, yuan21@buaa.edu.cn, smyiu@cs.hku.hk

Abstract

Factual error correction (FEC) aims to revise factual errors in false claims with minimal editing, making them faithful to the provided evidence. This task is crucial for alleviating the hallucination problem encountered by large language models. Given the lack of paired data (i.e., false claims and their corresponding correct claims), existing methods typically adopt the '*mask-then-correct*' paradigm. This paradigm relies solely on unpaired false claims and correct claims, thus being referred to as distantly supervised methods. These methods require a masker to explicitly identify factual errors within false claims before revising with a corrector. However, the absence of paired data to train the masker makes accurately pinpointing factual errors within claims challenging. To mitigate this, we propose to improve FEC by Learning to Inject Factual Errors (LIFE), a three-step distantly supervised method: '*mask-corrupt-correct*'. Specifically, we first train a corruptor using the '*mask-then-corrupt*' procedure, allowing it to deliberately introduce factual errors into correct text. The corruptor is then applied to correct claims, generating a substantial amount of paired data. After that, we filter out low-quality data, and use the remaining data to train a corrector. Notably, our corrector does not require a masker, thus circumventing the bottleneck associated with explicit factual error identification. Our experiments on a public dataset verify the effectiveness of LIFE in two key aspects: Firstly, it outperforms the previous best-performing distantly supervised method by a notable margin of 10.59 points in SARI Final (19.3% improvement). Secondly, even compared to ChatGPT prompted with in-context examples, LIFE achieves a superiority of 7.16 points in SARI Final.

Introduction

As is well known, large language models (LLMs), such as GPT-3 (Brown et al. 2020), PaLM(Chowdhery et al. 2022), and LLaMA (Touvron et al. 2023), have revolutionized NLP, which possess an extensive number of parameters and undergo pre-training on vast amounts of data. Compared with small language models, LLMs present emergent abilities (Wei et al. 2022a), including in-context learning and reasoning (Wei et al. 2022b) capabilities. A representative of LLMs is the recently launched ChatGPT, which has attracted widespread attention due to its capacity to generate coherent and contextually appropriate responses across various conversational contexts. However, LLMs are prone to hallucinate unintended text, namely generating unfaithful, nonsensical, or factually incorrect text, a phenomenon referred to as '*hallucination*' (Maynez et al. 2020; Raunak, Menezes, and Junczys-Dowmunt 2021). *Factual error correction* (FEC) is meant to correct factual errors within the text to make the output factually consistent with the input. This task is crucial for alleviating the hallucination problem, as FEC models can be employed to post-edit text generated by LLMs, thereby enhancing their reliability and faithfulness.

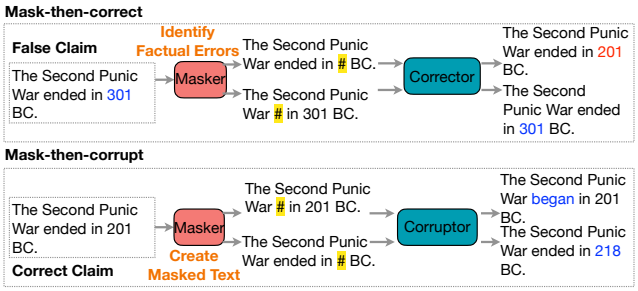


Figure 1: Comparison between the '*mask-then-correct*' and '*mask-then-corrupt*' pipelines during inference. Text in blue and red denotes the factual errors and correct revision, respectively. # refers to the mask token. We omit evidence retrieval for simplicity.

The most effective and straightforward approach is to develop fully supervised FEC models by fine-tuning pre-trained models, such as T5 (Raffel et al. 2020) on parallel data, which consists of false claims and their corresponding correct claims. However, it is very labor-intensive and time-consuming to create parallel data for FEC with human annotation, thus limiting the availability of such paired data. Therefore, previous methods (Shah, Schuster, and Barzilay 2020; Thorne and Vlachos 2021; Chen et al. 2023) have focused on developing FEC models in a distantly supervised manner, based on the assumption that unpaired false claims and correct claims are readily available. Distantly supervised models generally adhere to the '*mask-then-correct*' paradigm. To be concrete, during training, a masker is used

*Corresponding authors.

to mask a correct claim, following which a corrector is optimized to reconstruct the original correct claim. While during testing, the masker is tasked with identifying factual errors within a false claim, and then the corrector is expected to generate a correct claim based on the masked claim. For instance, as shown in the top of Figure 1, during testing, the masker model is expected to precisely identify the factual error (i.e., '*301*') in the false claim "*The Second Punic War ended in 301 BC*", and then the corrector will generate the correct claim based on the masked text "*The Second Punic War ended in # BC*". However, since there is a lack of paired data to train the masker, it is non-trivial to accurately identity the factual error in the false claim. If the masker misidentifies the factual error (e.g., '*ended*') in the mentioned false claim, it will lead to unreasonable masked text. As a result, the corrector is unlikely to generate the correct claim.

To circumvent the bottleneck of identifying factual errors before making corrections, we propose to improve factual error correction by **L**earning to **I**nject **F**actual **E**rrors into correct claims, referred to as **LIFE**. LIFE is a distantly supervised model, leveraging unpaired false claims and correct claims, yet it resorts to a three-step pipeline called '*mask-corrupt-correct*'. The main motivation behind LIFE is to train a corruptor to inject factual errors into correct text. Suppose we have an ideal corruptor, we first use a masker to create diverse masked text, and then the corruptor will generate false claims based on these masked claims. For example, in the bottom of Figure 1, the corruptor injects factual errors by replacing '*ended*' with '*began*' or substituting '*201*' with '*218*'. Consequently, we can create a sufficient collection of paired wrong claims and correct claims, which are utilized for training the corrector. It is worth mentioning that our proposed '*mask-then-corrupt*' pipeline eliminates the need for the masker to identify factual errors during testing, unlike the masker in the previous '*mask-then-correct*' pipeline, thus bypassing the aforementioned bottleneck.

So far, the critical challenge lies in training an effective corruptor capable of introducing factual errors into correct claims. To achieve this, the masker masks certain words within a false claim, prompting the corruptor to reconstruct the original false claim based on the masked claim. In this way, the corruptor progressively grasps the skill of fabricating false claims by intentionally injecting factual errors into correct claims.

To summarize, our contributions are threefold: (1) We propose **LIFE**[1], a distantly supervised model driven by the innovative three-step strategy: '*mask-corrupt-correct*'. (2) During testing, LIFE can revise false claims in an end-to-end manner, eliminating the need for the masker to identify factual errors before correction. Therefore, the proposed model nicely bypasses the bottleneck encountered by previous distantly supervised methods. (3) Our experimental results on a public dataset demonstrate the superior performance of LIFE compared to previous distantly supervised baselines and few-shot LLMs. LIFE achieves a remarkable state-of-the-art (SOTA) result on the test set, scoring 65.59 on SARI Final and 66.51 on ROUGE-2. These compelling outcomes validate the effectiveness of our proposed approach.

[1]Our code is available at: https://github.com/NLPCode/LIFE.

Approach

Problem Statement

Factual error correction aims to rectify factual errors within claim C using minimal revisions, making the revised claim C' align with the provided evidence E. Factual error correction follows three requirements: The revised claim C' should be grammatical, supported by the evidence, and rectify the factual errors present in C.

Overview

Following previous distantly supervised methods, we assume that unpaired false claims and correct claims are available. The set of false claims and correct claims are referred to as $\mathcal{S}^f = \{C_1^f, \ldots, C_n^f\}$ and $\mathcal{S}^t = \{C_1^t, \ldots, C_m^t\}$, where C_i^f and C_i^t denote the i-th false and correct claims within their respective sets. C_i^f and C_i^t are unpaired, namely C_i^t is not the correct claim for C_i^f. In this work, we propose LIFE to bypass the bottleneck of identifying factual errors before correction during testing. Our propose model consists of three key modules: a masker, a corruptor and a corrector. The masker and corruptor work in tandem through a '*mask-then-corrupt*' pipeline. Upon being trained on $\mathcal{S}^f$, the corruptor is developed, enabling us to subsequently introduce factual errors into each correct claim from $\mathcal{S}^t$. In this way, we can create a substantial amount of synthetic data, which will be used to train the corrector.

In the following subsections, we delve into the training process of the corruptor and its application in generating synthetic data during testing. Finally, we will introduce the utilization of filters to refine the synthetic data, followed by the training of the corrector using these refined data.

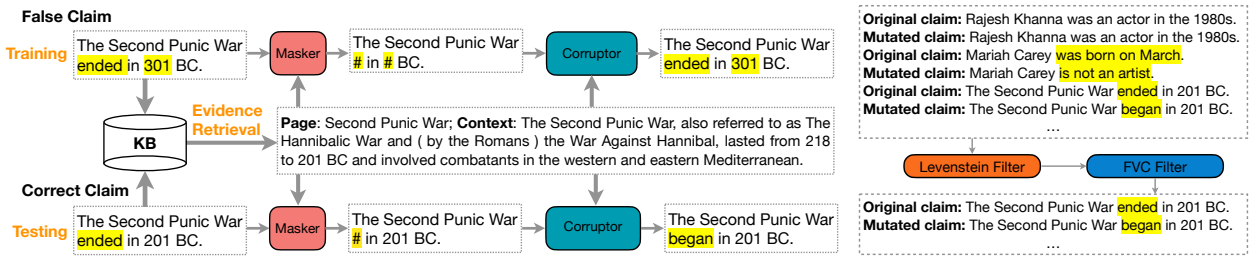
Corruptor Training and Inference

As depicted in Figure 2 (a), the '*mask-then-corrupt*' pipeline comprises two essential components: a masker and a corruptor. In the following, we will introduce two kinds of maskers, and how to effectively train and test the corruptor.

Masker. Following Thorne and Vlachos (2021), we resort to two simple maskers to mask the input claims, **random masking** and **heuristic masking**. Random masking randomly masks some words within the input claim. On the other hand, heuristic masking masks words that appear in the claim but do not appear in the evidence.

Corruptor. The corruptor is an encoder-decoder transformer (Vaswani et al. 2017), designed to reconstruct the claim based on the masked claim and the given evidence.

Training and Testing. During training, we input a false claim $C^f \in \mathcal{S}^f$ into the masker, and then optimize the corruptor to effectively recover C^f. When the masker successfully identifies the factual errors within the false claim, it implies that the corruptor must introduce factual errors into the masked claim to restore C^f accurately. In this scenario, the corruptor should deliberately introduce factual errors into the masked claim. However, if the masker fails to recognize



(a) The '*mask-then-corrupt*' pipeline (b) Illustration of synthetic data filtering.

Figure 2: The corruptor is trained to reconstruct the false claim, conditioned on the masked claim and retrieved evidence. At test time, the corruptor is able to incorporate relevant factual errors into the correct claim to generate a false claim. # denotes the mask token. The masked words and newly predicted words are highlighted.

any factual errors within the false claim, the corruptor simply needs to restore the masked words in the masked claim without requiring to inject additional factual errors.

During testing, we feed a correct claim $C_i^t \in \mathcal{S}^t$ into the masker and expect the corruptor to generate a false claim C_i^g based on the masked claim rather than reconstructing C_i^t. Accordingly, we acquire a set of synthetic paired data denoted as $\mathcal{D}' = \{(C_1^t, C_1^g), \ldots, (C_m^t, C_m^g)\}$, which is employed for training the corrector. Here, C_i^t and C_i^g represent the i-th correct claim and generated claim, respectively.

However, in reality, we cannot guarantee that the corruptor will always inject factual errors into the masked claim. Furthermore, there is no guarantee that the generated claim will have a high correlation with the original claim. It is possible for the corruptor to deviate from the masked input and create an entirely new claim. Therefore, it is necessary for us to apply additional filters to the generated data $\mathcal{D}'$.

Synthetic Data Filtering

In Figure 2 (b), we first filter out data instances, where C_i^t is completely the same with C_i^g, and then apply the Levenshtein filter and fact verification classifier-based filter to the synthetic data. The filtered data is referred to as $\mathcal{D}$.

Levenshtein Filter. The Levenshtein filter (LF) is based on the character-level Levenshtein edit distance (Levenshtein et al. 1966) between C_i^t and C_i^g:

$$LF(C_i^t, C_i^g) = \frac{Levenshtein(C_i^t, C_i^g)}{length(C_i^t)}. \quad (1)$$

If $LF(C_i^t, C_i^g)$ exceeds a certain threshold t_l, the corresponding data instance will be excluded. This filter relies on the assumption that a significant disparity in the Levenstein distance between C_i^t and C_i^g indicates a considerable semantic deviation of C_i^g from C_i^t.

Fact Verification Classifier-based Filter. The fact verification classifier-based filter (FVCF) relies upon the fact verification classifier, a 3-way classifier trained on FEVER. This filter is designed to classify a given claim as SUPPORTED, REFUTED, or NOTENOUGHINFO based on whether the given claim is supported, refuted, or unable to be verified by the provided evidence. If the predicted probability of NOTENOUGHINFO for the generated claim C_i^g surpasses a threshold value t_c, this indicates that C_i^g diverges significantly from the original claim C_i^t at the semantic level. This deviation is so substantial that C_i^g becomes unverifiable by the given evidence. Therefore, (C_i^g, C_i^t) will be discarded.

We fine-tune T5 on the synthetic data $\mathcal{D}$, which serves as the corrector. The corrector takes the mutated claim C_i^g and evidence as input. During training, our objective is to minimize the cross-entropy loss between the revised claim $\hat{C}_i^t$ and the original correct claim C_i^t.

Label / Split	Train	Valid	Test
SUPPORTED	37,961	1,477	1,593
REFUTED	20,075	2,091	2,289
Total	58,036	3,568	3,882

Table 1: The count of data instances for each split and label within FECDATA.

Experiment

Experimental Setups

Dataset. We assess the performance of our model using the evidence-based FEC dataset (FECDATA), a manually created data proposed by Thorne and Vlachos (2021). This dataset originates from the extensive fact verification dataset known as FEVER (Thorne et al. 2018). The claims present in the FEVER dataset are categorized into three classes: SUPPORTED, REFUTED, or NOTENOUGHINFO, depending on whether the claim is supported, contradicted, or cannot be verified by the provided evidence. FECDATA is curated by selecting instances belonging to the SUPPORTED and REFUTED categories. Notably, the REFUTED subset assesses models' ability to rectify factual errors within false claims. Conversely, the SUPPORTED subset evaluates their capacity to maintain correct claims. Please refer to Table 1 for the basic statistics of FECDATA.

Evaluation Metrics. For automatic evaluation, we use SARI (Xu et al. 2016) and ROUGE-2 (Lin 2004) metrics[2],

[2]The evaluation codes for SARI and ROUGE are available at: https://huggingface.co/spaces/evaluate-metric/sari and https://huggingface.co/spaces/evaluate-metric/rouge, respectively.

which exhibit a strong positive correlation with human evaluation, especially SARI, according to Thorne and Vlachos (2021)'s findings. We present the **Keep**, **Delete**, and **Add** scores of SARI, to assess the words in the revised claim (output) that are retained, removed, and introduced from the mutated claim (input), compared with the reference (ground truth). The SARI **Final** denotes the average of these three scores. ROUGE-2 (**RG-2**) measures the number of matching bigrams between the revised claim and the reference claim.

Baselines. We compare our proposed model with three kinds of baselines:

Fully Supervised Baselines are used to assess the ceiling performance of FEC models, assuming the availability of manually created paired data for training. Following Thorne and Vlachos (2021), we fine-tune **T5-base** (Raffel et al. 2020) on the FECDATA training set, where the encoder takes the false claim and its corresponding evidence as input, while the decoder generates the revised claim.

Distantly Supervised Baselines employ the '*mask-then-correct*' pipeline, which comprises a masker and a corrector. The masker can take various forms, such as the token-level explanations (Ribeiro, Singh, and Guestrin 2016; Chen et al. 2017) of a fact verification classifier (FVC), random masking, or heuristic masking. The FVC is typically initialized with BERT (Devlin et al. 2019) or RoBERTa (Liu et al. 2019) and is trained on FEVER. The last two maskers have been discussed in the approach section. The corrector is typically trained on the SUPPORTED data instances from FEVER. There are several specific approaches within this framework: (1) Dual encoder pointer network (**DEPN**) (Shah, Schuster, and Barzilay 2020) uses an FVC as the masker and employs the dual encoder pointer generator with the copy mechanism (See, Liu, and Manning 2017) as the corrector. (2) T5 Masker-Corrector (**T5MC**) (Thorne and Vlachos 2021) differs from DEPN in two aspects: (a) It utilizes random masking during training and heuristic masking during testing. (b) The corrector is based on T5-base. (3) Unlike T5MC, **T5MC-MLM** uses the masked language model BERT as the masker during inference. (4) **T5MC-V** is a variant of T5MC, using an FVC as the masker. (5) **VENCE** (Chen et al. 2023) iteratively runs the '*mask-then-correct*' pipeline over the claim until it becomes supported by evidence or the algorithm reaches the maximum steps.

Few-shot Baselines include two types of models: (1) **8-shot T5-base** directly fine-tunes T5-base using 8 data examples. (2) **8-shot LLMs** correct false claims via few-shot in-context learning, where we prompt three OpenAI LLMs: text-ada-001, text-babbage-001, and gpt-3.5-turbo-0301 (i.e., ChatGPT) with 8 in-context examples. Please refer to Table 1 in Appendix for the prompt applied to LLMs. For fair comparisons, all few-shot baselines use the same set of examples.

Implementation Details

Evidence Retrieval. As our work does not focus on evidence retrieval, we use the evidence retrieved by Thorne and Vlachos (2021) for all models. The retrieval process involves two steps: GENRE (Cao et al. 2021) is first used to predict the relevant Wikipedia pages for the input claim; DPR (Karpukhin et al. 2020) is then used to retrieve the top-k ($k = 2$) passages from the pages predicted by GENRE.

Masker and Corruptor. For the masker, we employ a heuristic masking strategy to mask the provided claim during training, while random masking with a mask ratio of 30% is utilized during inference. The corruptor is initialized with the T5-base model and optimized using the AdamW optimizer (Loshchilov and Hutter 2019) with a learning rate of $4e-5$, a batch size of 64, and a linear learning rate schedule with 10% warm-up steps for 4000 steps. The exploration of learning rates is conducted within the predefined set: $\{5e-6, 1e-5, 2e-5, 3e-5, 4e-5, 5e-5\}$. During inference, we use beam search decoding with a beam size of 5 to generate false claims for all correct claims (i.e., SUPPORTED data instances) in the training and validation sets. The corruptor takes the top-2 retrieved/gold evidence paired with the masked claim as input. We set the maximum source length to 512 and the maximum target length to 256.

Filters. We initialize the fact verification classifier-based filter with RoBERTa-base and then fine-tune it on FEVER for 4000 steps. We set the threshold $t_l = 0.3$ and $t_c = 0.2$ to filter the synthetic data produced by the corruptor.

Corrector. Following Thorne and Vlachos (2021), the corrector is based on the T5-base model, which takes the top-2 retrieved/gold evidence paired with the input claim as input. The corrector follows the corruptor's settings for training and inference, except it trains for only 1000 steps.

We employ the HuggingFace Transformers library (Wolf et al. 2019) to implement all models. Additionally, all experiments are carried out utilizing 2 NVIDIA Tesla V100 GPUs, each equipped with 32 GB of memory.

Overall Experimental Results

We present the main experimental results on the FECDATA test set in Table 2, which reveal the following key findings:

Distantly supervised baselines based on the mask-then-correct approach perform worst. This is mainly because these methods rely on the accurate identification of factual errors in the given claim by the masker. Excessive or incorrect masking of words in the claim by the masker will impair the performance of the corrector.

LLMs demonstrate strong performance with a few examples. Simply fine-tuning T5-base using 8 labeled instances (8-shot T5-base) cannot bring any enhancement compared to existing distantly supervised baselines, such as VENCE. By comparison, LLMs prompted with 8 in-context examples performs much better than 8-shot T5-base. For example, 8-shot ChatGPT achieves 58.43 points in SARI Final and surpasses VENCE by around 3 points, which underscores the remarkable few-shot ability of LLMs for FEC.

Our proposed LIFE achieves a new SOTA result. LIFE outperforms its best distantly supervised counterpart, VENCE, by a significant margin of 10.59 points in the SARI Final score. This notable achievement can be attributed to LIFE's effective circumvention of the bottleneck associated with mask-then-correct methods, which involves identifying factual errors in claims prior to the correction process. Notably, our approach even surpasses the performance of few-

Models	FVC	Retrieved Evidence					Gold Evidence				
		SARI Score				RG-2	SARI Score				RG-2
		Keep	Delete	Add	Final		Keep	Delete	Add	Final	
Fully Supervised Baselines											
Fully Supervised T5-base*	-	85.40	88.92	48.40	74.24	73.50	88.56	91.40	58.38	79.45	78.04
Distantly Supervised Baselines											
DEPN‡	BERT-base	34.5	48.1	1.7	28.1	34.8	45.2	56.9	3.9	35.3	-
T5MC (Thorne and Vlachos 2021)†	-	65.2	62.7	15.5	47.8	50.3	66.7	62.2	16.1	48.3	-
+ Enumerate†	BERT-base	66.2	64.3	17.1	49.2	51.2	-	-	-	-	-
T5MC-MLM‡	-	56.1	52.9	7.8	38.9	42.7	-	-	-	-	-
T5MC-V (Thorne and Vlachos 2021)†	BERT-base	61.1	54.3	19.4	44.9	42.0	61.8	62.2	10.2	44.7	-
+ Enumerate†	BERT-base	63.0	55.7	24.1	47.6	45.5	-	-	-	-	-
VENCE (Chen et al. 2023)†	BERT-base	66.0	60.1	34.8	53.6	57.7	67.5	61.5	34.6	54.5	-
	RoBERTa-large	67.1	61.9	36.0	55.0	59.1	-	-	-	-	-
Few-shot Baselines											
8-shot T5-base*	-	61.75	85.23	8.70	51.89	49.83	63.50	82.44	13.56	53.17	51.38
8-shot text-ada-001*	-	61.43	75.21	9.86	48.83	42.95	-	-	-	-	-
8-shot text-babbage-001*	-	69.69	76.39	18.07	54.72	52.57	-	-	-	-	-
8-shot ChatGPT*	-	72.09	75.92	27.29	58.43	49.43	79.98	81.61	38.81	66.80	60.72
Distantly Supervised (Our Method)											
LIFE*	RoBERTa-base	75.23	91.88	29.67	**65.59**	**66.51**	79.33	93.01	39.81	**70.72**	**70.45**

Table 2: Automatic evaluation results (%) of different models with retrieved/gold evidence on the FECDATA test set. Results marked with †, ‡, and ∗ are from VENCE, T5MC-V and our reproduction, respectively. Enumerate means using the FVC model to rank 20 generated claims and select the best one. Underline indicates the best model and bold indicates the second best.

shot LLMs. These results offer compelling evidence for the effectiveness of LIFE in boosting the performance for FEC.

While LIFE outperforms distantly supervised and few-shot models, it still falls short of the fully supervised baseline (scoring 65.59 compared to 74.24 on SARI Final), highlighting the potential for further enhancement. In addition, we found that using gold evidence instead of retrieved evidence to modify the given claim has led to significant improvements across all models. This indicates that retrieved evidence may be inadequate, namely fails to provide useful information for rectifying factual errors within claims. Given that this work does not focus on enhancing retrieval models, this discovery unveils a promising direction for future advancements in FEC models.

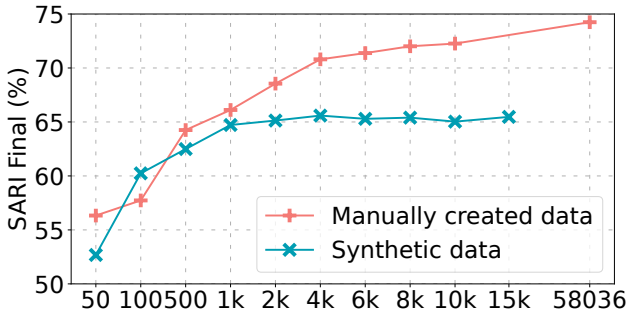


Figure 3: Performance of correctors initialized with T5-base fine-tuned on different numbers of synthetic data or manually created data, i.e. FECDATA training data.

More Analysis and Discussion

In this subsection, unless explicitly stated otherwise, all experiments are conducted with the following settings: When training the corruptor, heuristic masking is used as the masker, while during testing, random masking is employed to generate synthetic data. Subsequently, the Levenshtein filter (LF) and the fact verification classifier-based filter (FVCF) are utilized to filter the generated data. The minimum masking granularity for the masker is at the word level, and adjacent masked words are not merged. Both the corrector and corruptor utilize the top-2 retrieved evidence.

How does the number of synthetic data affect the performance of LIFE? To answer the first question, we train LIFE's corrector using varying number of the synthetic data. For the sake of comparison, we also train a supervised corrector on the manually created FECDATA. As illustrated in Figure 3, when the number of data used to training correctors is less than $1k$, both correctors exhibit a linear increase in performance as the dataset size grows. Notably, the corrector trained on synthetic data achieves a performance level that is comparable to the one trained on FECDATA. However, when the synthetic data reaches $4k$ samples, LIFE's performance reaches its peak. In contrast, the performance of the supervised corrector trained on gold data continues to improve, even when using the entire FECDATA training set. The significant performance gap between LIFE and the supervised corrector may stem from noise in synthetic data.

How do different masking strategies affect the performance of LIFE? In this work, we employ two masking strategies to mask the given claim. We exhaustively enumerate all potential combinations of masking strategies utilized during both the training and testing phases of the LIFE corruptor. From Table 3, we draw two crucial findings: (1) The

Mask Strategy		SARI Score				RG-2
# Train	Test	Keep	Delete	Add	Final	
1 Random	Random	70.20	93.61	16.31	60.04	63.15
2 Heuristic	Heuristic	75.25	88.85	19.79	61.30	61.81
3 Random	Heuristic	72.09	86.68	16.85	58.54	60.03
4 Heuristic	Random	75.23	91.88	29.67	**65.59**	**66.51**

Table 3: Impact of masking strategies on LIFE.

Masked Claim	Injected Factual Error
The Second Punic War # in 201 BC.	began
The Second Punic War ended in # BC.	218
The Second Punic War ended in 201 #.	AD
The # Punic War ended in 201 BC.	Third

Figure 4: The first masked claim is created by heuristic masking, the others are produced by random masking. Please refer to Figure 2 for the original correct claim and retrieved evidence. Mutated claims can be obtained by replacing # in masked claims with injected factual errors. (#=masked token, blue text=injected factual errors).

heuristic strategy, when employed during training, enhances the performance of LIFE more effectively than the random strategy (row 1 vs. row 4, and row 2 vs. row 3). (2) However, during testing, the random strategy proves to be more useful (row 1 vs. row 3, and row 2 vs. row 4).

During training, a false claim is fed into the masker. The heuristic strategy is more likely to find erroneous spans. For example, in Figure 2, factual errors related words (*'ended'* and *'301'*) are masked due to their absence in the supporting evidence. Consequently, the corruptor is compelled to introduce factual errors to recover the original false claim. By comparison, the random strategy is less likely to be able to accurately mask the problematic words. In such instances, the masked claim still contains factual errors, thereby the corruptor only needs to complete the masked claim without requiring to inject factual errors into it. At this point, the corruptor is not optimized in the expected direction. Hence, the heuristic strategy manifests greater efficacy during training.

Conversely, during testing, a correct claim is fed into the masker. At this stage, we do not expect the masker to identify erroneous parts of the input claim. As shown in Figure 4, the random strategy can produce much more diverse masked claims than the heuristic strategy, thus generating more diverse false claims. That is why the random strategy is more beneficial during testing.

Will different levels of masking granularity affect the performance of LIFE? When using the T5 tokenizer, the input claim will be transformed into a subword sequence. Consequently, a complete word might be split into multiple subwords. For instance, if the text to be masked is 'pleasingly large', after tokenization it will be split into '_pleasing', 'ly', '_large'. If the masker's minimum masking granularity is at the subword level, we will use '# # #' to represent the masked subwords. On the other hand, if the minimum masking granularity is at the word level, we will use

Mask Span		SARI Score				RG-2
Granularity	Merge	Keep	Delete	Add	Final	
Subword		74.37	93.10	28.07	65.18	**66.75**
Subword	✓	73.59	92.62	27.90	64.70	66.27
Word		75.23	91.88	29.67	**65.59**	66.51
Word	✓	73.94	93.05	27.48	64.82	66.30

Table 4: Impact of masking granularities on LIFE.

# Variants	SARI Score				RG-2
	Keep	Delete	Add	Final	
1 LIFE	75.23	91.88	29.67	**65.59**	**66.51**
2 – LF	73.65	93.98	26.65	64.76	66.38
3 – FVCF	74.80	91.63	28.78	65.07	66.10
4 – LF & FVCF	72.90	93.99	25.47	64.12	66.05

Table 5: Ablation study of LIFE on the test set.

'# #' to represent the masked words. Furthermore, we experiment with merging consecutive mask tokens. This means that each span is replaced with a single # token, thereby 'pleasingly large' will be represented as '#'. Table 4 shows that LIFE consistently performs well across different masking granularity, highlighting the robustness of our approach. In other experiments, we use the word level masking granularity and do not merge adjacent masked words.

Impact of Filters. We perform an ablation study to demonstrate the importance of filters. We first train the full model on the synthetic data that has been processed by the filters: LF and FVCF. The results, as shown in Table 5, highlight two key points when compared to the full model (row 1): (1) Eliminating LF (row 2) or FVCF (row 3) leads to performance drop. (2) The removal of both filters results in the poorest performance among all variants. These observations verify the effectiveness of the proposed filters.

Human Evaluation

Apart from automatic evaluation, we also conduct a human evaluation to compare LIFE with the fully supervised T5-base, 8-shot T5-base, and 8-shot ChatGPT models. The fully supervised T5-base model employs gold evidence to rectify false claims, while the others use retrieved evidence. We randomly sample 50 samples from the test set and shuffle them to avoid bias. Following Chen et al. (2023), three annotators[3] are asked to assess the revised claims using the following Boolean criteria: (1) Is it *grammatically* correct? (2) Is it *supported* by evidence? (3) Are the factual errors *corrected*? The final question, measuring the correction of factual errors, is the primary metric in our human evaluation. Table 6 shows that our proposed model outperforms the few-shot baselines on the corrected metric. However, there still exists a gap between our model and the fully supervised baseline. Inter-annotator agreement measured by Fleiss' *kappa* (Fleiss 1971) is 0.859, implying almost perfect agreement (> 0.8) (Landis and Koch 1977).

[3]All annotators are Ph.D. holders unrelated to our research.

Models	Grammar	Support	Correct
Fully supervised models with gold evidence			
T5-base	100	89.3*	86.7*
Models with retrieved evidence			
8-shot T5-base	83.3*	22.0*	5.3*
8-shot ChatGPT	92.0*	90.0*	42.0*
LIFE	99.3	68.7	**58.7**

Table 6: Human evaluation results (%) on the test set. p-values between results of our proposed model LIFE and the results marked with * are less than 0.01, indicating the differences between them are statistically significant.

Gold Claim: The IPhone 4 is a smartphone.
False Claim: The IPhone 4 is only a computer.
Retrieved Evidence: Page: IPhone 4; Context: The iPhone 4 is a smartphone that was designed and marketed by Apple Inc. . . .
Supervised T5-base: The IPhone 4 is a smartphone. ✓
8-shot T5-base: The IPhone 4 is only a computer. ✗
8-shot ChatGPT: The iPhone 4 is a smartphone designed and marketed by Apple Inc. ✓
LIFE: The IPhone 4 is a smartphone. ✓

Gold Claim: Lucy Hale was in American Juniors.
False Claim: Lucy Hale was not in American Juniors.
Retrieved Evidence: Page: Lucy Hale; Context: Karen Lucille Hale (born June 14, 1989) is an American actress and singer. Earlier in her career, she was sometimes credited as Lucy Kate Hale. Hale first came to prominence as one of the five winners of the reality show American Juniors . . .
Supervised T5-base: Lucy Hale was in American Juniors. ✓
8-shot T5-base: Lucy Hale was not in American Juniors. ✗
8-shot ChatGPT: Lucy Hale was one of the winners of the reality show American Juniors. ✗
LIFE: Lucy Hale was in American Juniors. ✓

Figure 5: Revised claims generated by our model and baselines for false claims from the test set. The supervised T5-base method corrects with gold evidence, while others use retrieved evidence. We omit the gold evidence for simplicity (blue text=factual errors; red text=correct modifications; orange text=content copied from the evidence).

Case Study

In this subsection, we analyze the claims revised by our approach and baselines. As shown in Figure 5, the supervised T5-base can correctly revise the factual errors in false claims, which represents the ceiling performance. However, the 8-shot T5-base method does not make any modifications, thus performing worst. While 8-shot ChatGPT can occasionally rectify factual errors within false claims, it typically involves directly copying content from the evidence into revised claims rather than making direct corrections to the erroneous parts. This tendency results in revised claims containing extra information. For instance, in the first example, "*designed and marketed by Apple Inc*" is irrelevant to the correction. This phenomenon also explains why this method achieves a significant supported score in human evaluation, yet a lower corrected score. In contrast, our approach can pinpoint factual errors and make precise corrections.

Related Work

Grammatical Error Correction (Ng et al. 2014; Yuan and Briscoe 2016; Bryant, Felice, and Briscoe 2017; Awasthi et al. 2019; Liu et al. 2021) is meant to identify and rectify grammatical errors in written text, which is critical for aiding non-native speakers in enhancing their writing skills. Compared with grammatical error correction, factual error correction aims to correct factual errors, such as incorrect dates, names, or historical events, instead of grammatical errors. At the same time, He et al. (2023a) proposed using LLMs, such as ChatGPT, as annotators (He et al. 2023b) to inject factual errors into correct text to create synthetic data. In contrast, we suggest utilizing a trained corruptor to inject factual errors into the correct text, without relying on LLMs.
Retrieval-augmented Generation (RAG) (Lewis et al. 2020) integrates information retrieval and language generation techniques to enhance the quality of generated content. He et al. (2022) use dense retrievers to fetch relevant sentences from an external corpus using specific keywords, improving lexically constrained text generation (He and Li 2021; He 2021). By incorporating external knowledge, RAG effectively mitigates the risk of generating inaccurate or nonsensical content. Factual error correction revises factual errors based on the retrieved evidence, thereby falling under the category of RAG.
Fact Verification, also referred to as fact-checking, seeks to check whether a claim is supported or refuted by the given evidence, which has undergone extensive research in recent years. Researchers evaluate claims by analyzing both unstructured sources (Vlachos and Riedel 2014; Wang 2017; Thorne et al. 2018; Wadden et al. 2020) and structured sources (Chen et al. 2020; Iso, Qiao, and Li 2020). In contrast, factual error correction is more challenging than fact verification, since it necessitates models not only to evaluate whether input claims contain factual errors, but also to pinpoint and rectify them.

Conclusion and Future Work

In this paper, we present LIFE, which improves FEC by learning to inject factual errors into correct claims. LIFE is a distantly supervised model comprising three components: masker, corruptor and corrector. The core of our approach involves training a corruptor to inject factual errors into correct claims using the '*mask-then-corrupt*' strategy. This approach enables us to generate a substantial amount of synthetic data, which can be used to train the corrector. Our approach circumvents the bottleneck of identifying factual errors before making modifications encountered by previous distantly supervised models. Consequently, our proposed model, LIFE, significantly outperforms previous distantly supervised baselines and the few-shot LLMs, achieving a new SOTA result on FECDATA.

However, the training of the corruptor depends on whether the masker can correctly identify factual errors. To be concrete, during training, if the masker fails to recognize any factual errors within the false claim, the corruptor simply needs to restore the masked words in the masked claim without requiring to inject additional factual errors. This is why the corruptor may fail to introduce factual errors into correct claims during testing. Therefore, researching how to accurately identify factual errors in false claims can be considered as a future research direction.

Ethics Statement

We take ethical considerations very seriously and strictly adhere to AAAI's ethics policy. This work concentrates on enhancing factual error correction, analyzed through publicly available datasets and evaluation methods. We guarantee the authenticity of our experimental results and the objectivity of our empirical conclusions.

Acknowledgments

This work is supported by National Natural Science Foundation of China (Grant No. 62202023), HKU-SCF FinTech Academy, Shenzhen-Hong Kong-Macao Science and Technology Plan Project (Category C Project: SGDX20210823103537030), and Theme-based Research Scheme of RGC, Hong Kong (T35-710/20-R). We would like to thank the anonymous reviewers for their constructive and informative feedback on this work.

References

Awasthi, A.; Sarawagi, S.; Goyal, R.; Ghosh, S.; and Piratla, V. 2019. Parallel Iterative Edit Models for Local Sequence Transduction. In *Proceedings of EMNLP*, 4260–4270. Hong Kong, China: Association for Computational Linguistics.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *NIPS*, volume 33, 1877–1901. Curran Associates, Inc.

Bryant, C.; Felice, M.; and Briscoe, T. 2017. Automatic Annotation and Evaluation of Error Types for Grammatical Error Correction. In *Proceedings of ACL*, 793–805. Vancouver, Canada: Association for Computational Linguistics.

Cao, N. D.; Izacard, G.; Riedel, S.; and Petroni, F. 2021. Autoregressive Entity Retrieval. In *ICLR*.

Chen, J.; Xu, R.; Zeng, W.; Sun, C.; Li, L.; and Xiao, Y. 2023. Converge to the Truth: Factual Error Correction via Iterative Constrained Editing. In *Proceedings of AAAI*.

Chen, Q.; Zhu, X.; Ling, Z.-H.; Wei, S.; Jiang, H.; and Inkpen, D. 2017. Enhanced LSTM for Natural Language Inference. In *Proceedings of ACL*, 1657–1668. Vancouver, Canada: Association for Computational Linguistics.

Chen, W.; Wang, H.; Chen, J.; Zhang, Y.; Wang, H.; Li, S.; Zhou, X.; and Wang, W. Y. 2020. TabFact: A Large-scale Dataset for Table-based Fact Verification. In *ICLR*.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; et al. 2022. PaLM: Scaling Language Modeling with Pathways. *arXiv preprint arXiv:2204.02311*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL*, 4171–4186. Minneapolis, Minnesota: Association for Computational Linguistics.

Fleiss, J. L. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5): 378–382.

He, X. 2021. Parallel Refinements for Lexically Constrained Text Generation with BART. In *Proceedings of EMNLP*, 8653–8666. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics.

He, X.; Gong, Y.; Jin, A.-L.; Qi, W.; Zhang, H.; Jiao, J.; Zhou, B.; Cheng, B.; Yiu, S.; and Duan, N. 2022. Metric-guided Distillation: Distilling Knowledge from the Metric to Ranker and Retriever for Generative Commonsense Reasoning. In *Proceedings of EMNLP*, 839–852. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics.

He, X.; Jin, A.-L.; Ma, J.; Yuan, Y.; and Yiu, S. M. 2023a. PivotFEC: Enhancing Few-shot Factual Error Correction with a Pivot Task Approach using Large Language Models. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Findings of EMNLP*, 9960–9976. Singapore: Association for Computational Linguistics.

He, X.; and Li, V. O. 2021. Show Me How To Revise: Improving Lexically Constrained Sentence Generation with XLNet. In *Proceedings of AAAI*, volume 35, 12989–12997.

He, X.; Lin, Z.; Gong, Y.; Jin, A.-L.; Zhang, H.; Lin, C.; Jiao, J.; Yiu, S. M.; Duan, N.; and Chen, W. 2023b. AnnoLLM: Making Large Language Models to Be Better Crowdsourced Annotators. *arXiv preprint arXiv:2303.16854*.

Iso, H.; Qiao, C.; and Li, H. 2020. Fact-based Text Editing. In *Proceedings of ACL*, 171–182. Online: Association for Computational Linguistics.

Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W.-t. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of EMNLP*, 6769–6781. Online: Association for Computational Linguistics.

Landis, J. R.; and Koch, G. G. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33(1): 159–174.

Levenshtein, V. I.; et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, 707–710. Soviet Union.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; Riedel, S.; and Kiela, D. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Proceedings of NIPS*. Red Hook, NY, USA: Curran Associates Inc. ISBN 9781713829546.

Lin, C.-Y. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, 74–81. Barcelona, Spain: Association for Computational Linguistics.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Liu, Z.; Yi, X.; Sun, M.; Yang, L.; and Chua, T.-S. 2021. Neural Quality Estimation with Multiple Hypotheses for Grammatical Error Correction. In Toutanova, K.; Rumshisky, A.; Zettlemoyer, L.; Hakkani-Tur, D.; Beltagy, I.; Bethard, S.; Cotterell, R.; Chakraborty, T.; and Zhou, Y., eds., *Proceedings of NAACL*, 5441–5452. Online: Association for Computational Linguistics.

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. In *ICLR*.

Maynez, J.; Narayan, S.; Bohnet, B.; and McDonald, R. 2020. On Faithfulness and Factuality in Abstractive Summarization. In *Proceedings of ACL*, 1906–1919. Online: Association for Computational Linguistics.

Ng, H. T.; Wu, S. M.; Briscoe, T.; Hadiwinoto, C.; Susanto, R. H.; and Bryant, C. 2014. The CoNLL-2014 Shared Task on Grammatical Error Correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, 1–14. Baltimore, Maryland: Association for Computational Linguistics.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *JMLR*, 21(1).

Raunak, V.; Menezes, A.; and Junczys-Dowmunt, M. 2021. The Curious Case of Hallucinations in Neural Machine Translation. In *Proceedings of NAACL*, 1172–1183. Online: Association for Computational Linguistics.

Ribeiro, M.; Singh, S.; and Guestrin, C. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of NAACL*, 97–101. San Diego, California: Association for Computational Linguistics.

See, A.; Liu, P. J.; and Manning, C. D. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of ACL*, 1073–1083. Vancouver, Canada: Association for Computational Linguistics.

Shah, D.; Schuster, T.; and Barzilay, R. 2020. Automatic Fact-guided Sentence Modification. In *Proceedings of AAAI*, volume 34, 8791–8798.

Thorne, J.; and Vlachos, A. 2021. Evidence-based Factual Error Correction. In *Proceedings of ACL*, 3298–3309. Online: Association for Computational Linguistics.

Thorne, J.; Vlachos, A.; Christodoulopoulos, C.; and Mittal, A. 2018. FEVER: a Large-scale Dataset for Fact Extraction and VERification. In *Proceedings of NAACL*, 809–819. New Orleans, Louisiana: Association for Computational Linguistics.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. u.; and Polosukhin, I. 2017. Attention is All you Need. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *NIPS*, volume 30. Curran Associates, Inc.

Vlachos, A.; and Riedel, S. 2014. Fact Checking: Task definition and dataset construction. In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, 18–22. Baltimore, MD, USA: Association for Computational Linguistics.

Wadden, D.; Lin, S.; Lo, K.; Wang, L. L.; van Zuylen, M.; Cohan, A.; and Hajishirzi, H. 2020. Fact or Fiction: Verifying Scientific Claims. In *Proceedings of EMNLP*, 7534–7550. Online: Association for Computational Linguistics.

Wang, W. Y. 2017. "Liar, Liar Pants on Fire": A New Benchmark Dataset for Fake News Detection. In *Proceedings of ACL*, 422–426. Vancouver, Canada: Association for Computational Linguistics.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; et al. 2022a. Emergent Abilities of Large Language Models. *arXiv preprint arXiv:2206.07682*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Chi, E.; Le, Q.; and Zhou, D. 2022b. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *NIPS*.

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; and Brew, J. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *arXiv preprint arXiv:1910.03771*.

Xu, W.; Napoles, C.; Pavlick, E.; Chen, Q.; and Callison-Burch, C. 2016. Optimizing Statistical Machine Translation for Text Simplification. *Transactions of the Association for Computational Linguistics*, 4: 401–415.

Yuan, Z.; and Briscoe, T. 2016. Grammatical error correction using neural machine translation. In *Proceedings of NAACL*, 380–386. San Diego, California: Association for Computational Linguistics.