LM-Critic: Language Models for Unsupervised Grammatical Error Correction

Michihiro Yasunaga **Jure Leskovec** **Percy Liang**
Stanford University
{myasu,jure,pliang}@cs.stanford.edu

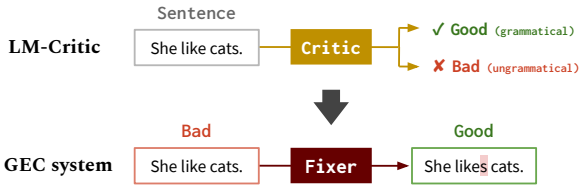
Abstract

Training a model for grammatical error correction (GEC) requires a set of labeled ungrammatical / grammatical sentence pairs, but manually annotating such pairs can be expensive. Recently, the Break-It-Fix-It (BIFI) framework has demonstrated strong results on learning to repair a broken program without any labeled examples, but this relies on a perfect critic (*e.g.*, a compiler) that returns whether an example is valid or not, which does not exist for the GEC task. In this work, we show how to leverage a pretrained language model (LM) in defining an LM-Critic, which judges a sentence to be grammatical if the LM assigns it a higher probability than its local perturbations. We apply this LM-Critic and BIFI along with a large set of unlabeled sentences to bootstrap realistic ungrammatical/grammatical pairs for training a corrector. We evaluate our approach on GEC datasets across multiple domains (CoNLL-2014, BEA-2019, GMEG-wiki and GMEG-yahoo) and show that it outperforms existing methods in both the unsupervised setting (+7.7 $F_{0.5}$) and the supervised setting (+0.5 $F_{0.5}$).

1 Introduction

Grammatical error correction (GEC) is the task of fixing grammatical errors in text, such as typos, tense and article mistakes. Recent works cast GEC as a translation problem, using encoder-decoder models to map bad (ungrammatical) sentences into good (grammatical) sentences (Yuan and Briscoe, 2016; Xie et al., 2016; Ji et al., 2017; Chollampatt and Ng, 2018; Junczys-Dowmunt et al., 2018). These methods rely on a combination of human-labeled data (*i.e.*, ⟨bad, good⟩ pairs) (Nicholls, 2003; Yannakoudakis et al., 2011; Bryant et al., 2019) and synthetic data, which are generated by corrupting good sentences into ⟨synthetic bad, good⟩ pairs (Awasthi et al., 2019; Kiyono et al., 2019). Human-labeled pairs are representative of real human errors but are expensive to obtain, while synthetic pairs are cheap but are unrealistic, deviating from the distribution of grammatical errors humans make (Grundkiewicz et al., 2019). How to obtain inexpensive yet realistic paired data to improve GEC remains a key challenge, especially in domains or languages with no labeled GEC data (Napoles et al., 2019; Náplava and Straka, 2019).

(a) Grammatical error correction (GEC) via LM-Critic



(b) Idea behind LM-Critic: Local optimum criterion

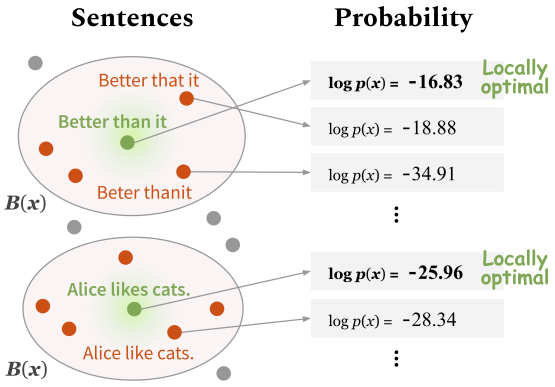


Figure 1: Illustration of LM-Critic. **(a)** In this work, we train a fixer for grammatical error correction (GEC) by leveraging LM-Critic that assesses the grammaticality. **(b)** LM-Critic deems a sentence to be grammatical if a pretrained language model (*e.g.*, GPT2) assigns it a higher probability than candidates in its local neighborhood (*e.g.*, edit distance 1).

Break-It-Fix-It (BIFI; Yasunaga and Liang (2021)) is a recent method to obtain realistic paired data from unlabeled data, which has shown promise in the task of source code repair. The idea of BIFI is that using an initial fixer (*e.g.*, trained on synthetic data) and a critic that tells if an input is bad or good (*e.g.*, compiler, which checks if code has an error), BIFI iteratively trains the fixer and a breaker to generate better paired data. Specifically, BIFI (1) applies the fixer to bad examples and keeps outputs *accepted by the critic*, (2) trains a breaker on the re-

sulting paired data and uses it to generate more pairs, and (3) trains the fixer on the pairs generated in Step (1) and (2). This way, BIFI adapts the fixer to more realistic distributions of ⟨bad, good⟩ pairs, only using unlabeled data. However, BIFI is not directly applicable to GEC because it requires an oracle critic (*e.g.*, compiler), which does not exist for GEC.

In this work, we propose *LM-Critic*, a simple approximate critic for assessing grammaticality (§3), and apply it with BIFI to learn GEC from unlabeled data (§4). Specifically, motivated by recent progress in large language models (LMs) (*e.g.*, GPT2, GPT3; Radford et al. (2019); Brown et al. (2020)) and an intuition that a good LM assigns a higher probability to grammatical sentences than ungrammatical counterparts, we use an LM's probability to define a critic for grammaticality. A naive approach is to deem a sentence as grammatical if its probability exceeds an absolute threshold, but this does not work in practice, *e.g.*, LMs may assign a high probability just because the sentence has more common words. We hence compare probabilities in *local* neighborhood of sentences. Concretely, LM-Critic is defined by two components, an LM (*e.g.*, GPT2) and a neighborhood function (*e.g.*, edit distance 1), and deems a sentence to be grammatical if the LM assigns it the highest probability in its local neighborhood (Figure 1; local optimum criterion). Using this LM-Critic, we apply BIFI to the GEC task. Notably, our approach, both the LM-Critic and GEC learning, does not require labeled data.

We evaluate our proposed approach on GEC benchmarks across multiple domains, CoNLL-2014 (Ng et al., 2014), BEA-2019 (Bryant et al., 2019), GMEG-yahoo, and GMEG-wiki (Napoles et al., 2019). We achieve strong performance in the unsupervised setting (*i.e.*, no labeled data), outperforming the baseline fixer trained on synthetic data by 7.7 $F_{0.5}$ on average. We also evaluate in the supervised setting, where we take the state-of-the-art model GECToR (Omelianchuk et al., 2020) as the baseline fixer, and further fine-tune it by applying our approach using unlabeled data. We achieve 65.8 / 72.9 $F_{0.5}$ on CoNLL-2014 / BEA-2019, outperforming GECToR by 0.5 $F_{0.5}$. Our results also suggest that while existing BIFI assumed access to an oracle critic (*i.e.*, compiler), an approximate critic (*i.e.*, LM-Critic) can also help to improve model learning.

2 Problem setup

The task of grammatical error correction (GEC) is to map an ungrammatical sentence x_{bad} into a grammatical version of it, x_{good} (one that has the same intended meaning). A GEC model (*fixer*) f aims to learn this mapping, typically using a *paired* dataset $\mathcal{D}_{\text{pair}} = \{(x_{\text{bad}}^{(i)}, x_{\text{good}}^{(i)})\}$. In particular, we call it *labeled* if the pairs are human-annotated. In contrast, we call *unlabeled* data a set of raw sentences $\mathcal{D}_{\text{unlabel}} = \{x^{(i)}\}$. For simplicity, we use "good"/"bad" to mean grammatical/ungrammatical interchangeably. Unlike a fixer, which maps x_{bad} to x_{good}, a *critic* c merely assesses whether an input is good or bad: for a sentence x,

$$c(x) = \begin{cases} 1 & \text{if } x \text{ is good} \\ 0 & \text{if } x \text{ is bad.} \end{cases} \quad (1)$$

Given unlabeled data x's (some of which are good, some of which are bad), and a language model (LM), which returns a probability distribution $p(x)$ over sentences x, we aim to define the critic (§3; LM-Critic) and use that to obtain the fixer (§4; BIFI).

3 LM-Critic

The core of our approach to GEC is a critic, which returns whether a sentence is good (grammatical) or bad (ungrammatical). Motivated by recent progress in large pretrained LMs (*e.g.*, GPT2, GPT3; Radford et al. (2019); Brown et al. (2020)), we aim to use an LM's probability score to define a critic for grammaticality. Specifically, we propose a criterion that deems a sentence to be good if it has the highest probability within its local neighborhood (local optimum criterion; §3.1). We implement this criterion using a pretrained LM and a sentence perturbation function (LM-Critic; §3.2). We then do an intrinsic study on how well LM-Critic works in practice (§3.3).

3.1 Local optimum criterion of grammaticality

Our starting point is the idea that a good LM assigns a higher probability to grammatical sentences than ungrammatical ones. With this idea, a naive way to judge grammaticality might be to find a threshold (δ) for the absolute probability, and let the critic be:

$$\text{AbsThr-Critic}(x) = \begin{cases} 1 & \text{if } p(x) > \delta \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

However, this does not work in practice. In Figure 1, for instance, "*Alice likes cats*" (4th sentence) is grammatical but has a lower probability (according to GPT2) than "*Better that it*" (2nd sentence), which is ungrammatical. This is because the two sentences have different meanings and are not directly comparable. We also empirically find that this critic based on absolute threshold does not work well (§3.3.3).

This observation motivates us to compare sentences with the same intended meaning, and leads to the following two refined intuitions.

Intuition 1 (Correlation of grammaticality and probability). For a grammatical sentence, x_{good}, and an ungrammatical *version* of it (with the same intended meaning), x_{bad}, we have

$$p(x_{\text{bad}}) < p(x_{\text{good}}). \quad (3)$$

Intuition 2 (Local neighborhood of sentences). Assume for simplicity that every sentence has exactly one grammatical version of it (*i.e.*, if the sentence is grammatical, itself; if not, its corrected version).[1] For each sentence x, there is a set of sentences, $B(x)$ (*local neighborhood*), that consists of the grammatical version and all other ungrammatical versions of x.

Assuming the above two intuitions, we obtain the following criterion for judging grammaticality, where the idea is to compare sentences within the meaning-preserving local neighborhood.

Local optimum criterion of grammaticality. For each sentence x, we let $B(x)$ be its local neighborhood as defined in Intuition 2. We then have

$$x \text{ is grammatical iff } x = \underset{x' \in B(x)}{\operatorname{argmax}}\, p(x'). \quad (4)$$

The justification is as follows. If x is grammatical, then by Intuition 1, x has a higher probability than any other sentences in $B(x)$, as they are ungrammatical; hence, we have the RHS of iff. On the other hand, if x is ungrammatical, then by Intuition 1, the grammatical version of x has a higher probability than x, which contradicts with the RHS of iff.

The idea is to deem a sentence to be grammatical if it has the highest probability within its meaning-preserving local neighborhood (Figure 1). We will next describe how to implement this criterion in practice.

3.2 Implementation of LM-Critic

We implement LM-Critic by approximating the local optimum criterion. First, for the sentence probability $p(x)$, we use a pretrained LM's probability score. As obtaining the ground-truth local neighborhood $B(x)$ is difficult, we aim to get an approximate, $\hat{B}(x)$: we implement a sentence perturbation function b, and let $\hat{B}(x)$ be samples from $b(x)$. To check the grammaticality of a sentence, we apply the local optimum criterion (Eq 4) using $\hat{B}(x)$:

$$\text{LM-Critic}(x) = \begin{cases} 1 & \text{if } x = \underset{x' \in \hat{B}(x)}{\operatorname{argmax}}\, p(x') \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

There are three decisions for implementing LM-Critic: choice of a pretrained LM, perturbation function b, and sampling method of perturbations.

Pretrained LM. We experiment with various sizes of GPT2 models (Radford et al., 2019)—GPT2 (117M parameters), GPT2-medium (345M), GPT2-large (774M), GPT2-xl (1.6B). These LMs were trained on a large set of web text (40GB).

Perturbation function. We study three variants:

- **ED1.** Given a sentence, we generate edit-distance one (ED1) perturbations in the character space. Following prior works in typo generation (Pruthi et al., 2019; Jones et al., 2020), we randomly insert a lowercase letter, delete a character, replace a character, or swap two adjacent characters.
- **ED1 + Word-level heuristics (all).** ED1 can cover most of the character-level typos but may not cover word-level grammatical errors, such as missing an article. Besides ED1, here we include heuristics for word-level perturbations used in Awasthi et al. (2019), which randomly inserts, deletes, or replaces a word based on its dictionary. Please refer to Awasthi et al. for more details.
- **ED1 + Word-level heuristics.** We noticed that the above word-level heuristics include perturbations that may alter the meaning of the original sentence (*e.g.*, deleting/inserting "not"). Therefore, we remove such heuristics here.

Sampling perturbations. As the output space of the perturbation function b is large, we obtain samples from $b(x)$ to be $\hat{B}(x)$. We experiment with random sampling with sizes of 100, 200 and 400, motivated by the finding that with the GPT2 models, a batch size of 100 sentences can fit into a single GPU of 11GB memory. Other (potentially more efficient) sampling methods include gradient-based sampling which picks perturbation sentences in a direction that increases the sentence probability (analogous to adversarial perturbations; Szegedy et al. (2013); Wallace et al. (2019)), but we focus on random sampling in this work.

The advantage of LM-Critic is that as LMs can be trained on a wide range of unlabeled corpora, it is unsupervised and usable in various domains of text.

[1] We acknowledge that this assumption may not hold in some cases, *e.g.*, an ungrammatical sentence may have no correction ("asdfghgfdsa"—just a random typo?) or multiple corrections ("The cat sleep."—change "sleep" to the present tense or past?). We accept this assumption considering that it is often sufficient in common GEC datasets, and leave the relaxation of the assumption for future work.

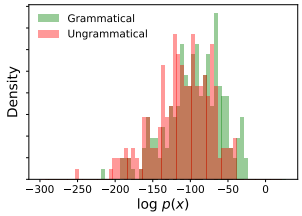


Figure 2: Probability of grammatical (green) and ungrammatical (red) sentences, computed by a pretrained LM (GPT2).

Pretrained LM	How often $p(x_{\text{bad}}) < p(x_{\text{good}})$?
GPT2	94.7%
GPT2-medium	95.0%
GPT2-large	95.9%
GPT2-xl	96.0%

Table 1: How well sentence probability returned by pretrained LMs correlates with grammaticality empirically.

3.3 Empirical analysis

We study how well our LM-Critic works in practice. We prepare an evaluation data for judging grammaticality in §3.3.1. We first perform a simple check to make sure that LMs' probability score correlates with grammaticality (§3.3.2). We then study the performance of LM-Critic judging grammaticality (§3.3.3). The analysis we conduct in this section is just an intrinsic evaluation of LM-Critic. Our main goal is to use LM-Critic with BIFI for learning GEC, which we describe and evaluate in §4.

3.3.1 Evaluation data

To gain insights into how well LM-Critic judges grammaticality, we prepare a simple evaluation data consisting of $(x_{\text{bad}}, x_{\text{good}})$ sentence pairs. As experimenting with multiple datasets is desired in GEC (Mita et al., 2019), we construct a combined evaluation set from the dev sets of multiple GEC benchmarks, GMEG-wiki (Napoles et al., 2019), GMEG-yahoo, and BEA-2019 (Bryant et al., 2019), which span the domains of Wikipedia, Yahoo! Answers, and essay/learner English. Specifically, we sampled ∼600 labeled pairs of $(x_{\text{bad}}, x_{\text{good}})$ in total from the three benchmarks. We filter out examples where $x_{\text{bad}} = x_{\text{good}}$ in this process. We acknowledge that while we use annotated $(x_{\text{bad}}, x_{\text{good}})$ pairs for the evaluation here, this does not fully match the way LM-Critic will be used in BIFI (§4), where the critic is run on unlabeled sentences; our study here is just to gain intrinsic insights into LM-Critic.

3.3.2 Analysis of LM probability

Using the evaluation data, we first make sure that pretrained LMs' probability correlates with grammaticality. Figure 2 shows a histogram for the probability $\log p(x)$ of grammatical (green) and ungrammatical (red) sentences computed by GPT2. In Table 1, we study how often pretrained LMs actually assign a higher probability to x_{good} than x_{bad} on the evaluation pairs $(x_{\text{bad}}, x_{\text{good}})$. We find that the LMs satisfy $p(x_{\text{bad}}) < p(x_{\text{good}})$ about 94% of the time, with a slight increase when using a larger model (from GPT2 to GPT2-xl). We find that the remaining pairs with $p(x_{\text{bad}}) > p(x_{\text{good}})$ consist mostly of cases where x_{good} adds commas or quotations to x_{bad} (see Table 3 top for examples).

Perturbation	Recognize "Good"			Recognize "Bad"		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$
ED1	58.7	90.1	63.1	78.8	36.8	64.2
ED1 + word(all)	69.7	10.2	32.2	51.5	95.5	56.7
ED1 + word	68.4	75.5	**69.7**	72.7	65.1	**71.1**

Sample size	Recognize "Good"			Recognize "Bad"		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$
100	68.4	75.5	69.7	72.7	65.1	71.1
200	71.3	71.5	71.4	71.4	71.3	**71.4**
400	72.6	68.7	**71.8**	70.3	74.0	71.0

Pretrained LM	Recognize "Good"	Recognize "Bad"
	$F_{0.5}$	$F_{0.5}$
GPT2	69.7	71.1
GPT2-medium	69.9	71.0
GPT2-large	70.3	71.3
GPT2-xl	69.9	71.0

Table 2: **Performance of LM-Critic**, when using different choices of a perturbation function, sample size, and pretrained LM described in §3.2. **(Top)** We set the LM to be GPT2 and the perturbation sample size to be 100, and vary the perturbation function b. "ED1 + word" achieves the best $F_{0.5}$. Henceforth, we use this perturbation function. **(Middle)** We set the LM to be GPT2 and vary the perturbation sample size. Increasing the sampling size improves the performance slightly. **(Bottom)** We vary the LM. Increasing the LM size makes slight or no improvement in $F_{0.5}$ on the dataset we used.

Examples of $p(x_{\text{bad}}) > p(x_{\text{good}})$
(Comma)
x_{bad}: The video was filmed on January 22 and is set to premiere on February 22.
x_{good}: The video was filmed on January 22, and is set to premiere on February 22.
(Quotation)
x_{bad}: Uprising is a 1980 roots reggae album by Bob Marley & The Wailers.
x_{good}: "Uprising" is a 1980 roots reggae album by Bob Marley & The Wailers.
(British spelling)
x_{bad}: The blast could be heard across the whole city centre.
x_{good}: The blast could be heard across the whole city center.
Examples of $p(x') > p(x_{\text{good}}), x' \in \hat{B}(x_{\text{good}})$
(Singular/plural)
x': They are affiliated to either the state boards or to national education boards.
x_{good}: They are affiliated to either the state board or to national education boards.
(Tense)
x': As well as touring Europe, they tour with such acts as Green Day.
x_{good}: As well as touring Europe, they toured with such acts as Green Day.

Table 3: Failure cases of LM-Critic. (Top) GPT2 assigns a higher probability to bad sentences. (Bottom) our neighborhood function ("ED1 + word") includes sentences with a higher LM probability than the original good sentence.

3.3.3 Performance of LM-Critic

In §3.3.2 we simply made sure that pretrained LMs' probability correlates with grammaticality. Here we study LM-Critic's performance of *judging* bad/good

sentences, on the evaluation set $\{(x_{\text{bad}}^{(i)}, x_{\text{good}}^{(i)})\}$. We treat the label of x_{bad}'s and x_{good}'s to be "bad" and "good", respectively, and measure the precision (P), recall (R), $F_{0.5}$ of LM-Critic recognizing "bad" and "good". Denoting the critic as c, precision and recall for "bad" are defined as

$$P^{(\text{bad})} = \frac{|\{x: c(x)=0\}| \cap |\{x_{\text{bad}}\}|}{|\{x: c(x)=0\}|}, \quad (6)$$

$$R^{(\text{bad})} = \frac{|\{x: c(x)=0\}| \cap |\{x_{\text{bad}}\}|}{|\{x_{\text{bad}}\}|}. \quad (7)$$

$P^{(\text{good})}$ and $R^{(\text{good})}$ are defined similarly. $F_{0.5}$ score is a combined metric of P and R that is commonly used in grammatical error detection/correction literature.

Baseline critic. First, as a baseline, we evaluate the critic based on absolute threshold, described in Eq 2. We set the threshold δ as the average probability of all good and bad sentences in the evaluation data. This method achieves 54.3 $F_{0.5}^{(\text{bad})}$ and 56.0 $F_{0.5}^{(\text{good})}$, using GPT2.

Proposed LM-Critic. Table 2 shows the results of our proposed LM-Critic, using different choices of a perturbation function, sample size, and pretrained LM. Recall that LM-Critic predicts "bad" correctly if it finds a perturbed sentence with higher probability, and predicts "good" correctly if the input has the highest probability among the sampled perturbations.

- **Perturbation function *b* (top table).** We set the pretrained LM to be GPT2 and the perturbation sample size to be 100, and vary the perturbation function. We find that when the perturbation space is small ("ED1"), LM-Critic may make false predictions of "good", leading to low $P^{(\text{good})}$ and low $R^{(\text{bad})}$. When the perturbation space is large ("ED1 + word(all)"), LM-Critic may make false predictions of "bad", leading to low $R^{(\text{good})}$ and low $P^{(\text{bad})}$. "ED1 + word" is the most balanced and achieves the best $F_{0.5}$; henceforth, we use this perturbation method for all our experiments. Overall, our LM-Critic outperforms the baseline critic by substantial margins.
- **Sample size of perturbations (middle table).** We set the LM to be GPT2 and vary the perturbation sample size. Increasing the sample size tends to improve $P^{(\text{good})}$ and $R^{(\text{bad})}$, and improve the overall $F_{0.5}$ performance slightly.
- **Pretrained LM (bottom table).** We vary the LM. Increasing the LM size makes slight or no improvement in $F_{0.5}$ on the dataset we used.

We also analyze when LM-Critic fails. When LM-Critic predicts a false "good" (labeled "bad" but predicted "good"), it is commonly because of $p(x_{\text{bad}}) > p(x_{\text{good}})$ (as described in §3.3.2; Table 3 top), or perturbation sampling not hitting a better version of the input x_{bad}. When LM-Critic predicts a false "bad" (labeled "good" but predicted "bad"), it is because some perturbation $x' \in \hat{B}(x_{\text{good}})$ yields $p(x') > p(x_{\text{good}})$. Common examples are the change of tense or singular/plural (see Table 3 bottom for examples). This indicates that even if we use a conservative edit-distance like ED1, there may be unnecessary perturbations (tense, singular/plural) that pretrained LMs prefer, which is a limitation of our current LM-Critic.

The analysis done in this section is an intrinsic evaluation of LM-Critic. Our main goal is to use LM-Critic with BIFI for learning GEC, which we describe in §4. While LM-Critic is not perfect in itself as we have seen in this section (it is an *approximate* critic), we will show that it is helpful for obtaining realistic paired data to improve the downstream GEC performance. Henceforth, we use the "ED1 + word" perturbation, a sample size of 100, and GPT2 for our LM-Critic.

4 Learning GEC with LM-Critic

Break-It-Fix-It (BIFI; Yasunaga and Liang (2021)) is an existing method that uses a critic to obtain realistic paired data from unlabeled data. BIFI was originally studied in the source code repair task where an oracle critic (*e.g.*, compiler) exists, but there is no oracle critic in GEC. Here, we propose to apply BIFI to the GEC task by using LM-Critic as the critic (§4.1), and evaluate this approach on GEC benchmarks (§4.2). The difference from the original BIFI is that our task is GEC rather than code repair, and we use an approximate critic (*i.e.*, LM-Critic) instead of an oracle critic (*i.e.*, compiler).

4.1 Approach

Our goal is to learn a fixer f that maps an ungrammatical sentence x_{bad} into the grammatical version x_{good}. A common method to obtain paired data for GEC from unlabeled text is to heuristically corrupt good sentences (synthetic data) (Awasthi et al., 2019; Kiyono et al., 2019). However, such synthetic errors do not match the distributions of real grammatical errors humans make, which may result in accuracy drops (Daume III and Marcu, 2006). To mitigate this mismatch, BIFI aims to obtain more realistic paired data and train the fixer on it.

Specifically, BIFI takes as inputs:

- **Critic** c, for which we use LM-Critic
- **Unlabeled data** $\mathcal{D}_{\text{unlabel}}$. Using the critic c, examples in $\mathcal{D}_{\text{unlabel}}$ can be split into bad ones

$\mathcal{D}_{\text{bad}} = \{x \mid x \in \mathcal{D}_{\text{unlabel}}, c(x) = 0\}$ and good ones $\mathcal{D}_{\text{good}} = \{y \mid y \in \mathcal{D}_{\text{unlabel}}, c(y) = 1\}$

- **Initial fixer** f_0, which could be trained on synthetic data (unsupervised setting; §4.2.2) or labeled data (supervised setting; §4.2.3)

and improves the fixer by performing a cycle of data generation and training: (1) we apply the fixer f to the bad examples $\mathcal{D}_{\text{bad}}$, which consists of real grammatical errors made by humans, and use the critic to assess if the fixer's output is good—if good, we keep the pair; (2) we train a *breaker* b on the resulting paired data—consequently, the breaker can generate more realistic errors than the initial synthetic data; (3) we apply the breaker to the good examples $\mathcal{D}_{\text{good}}$; (4) we finally train the fixer on the newly-generated paired data in (1) and (3). This cycle can be iterated to improve the fixer and the breaker simultaneously. Formally, BIFI does the following in each round k ($=1,2,\ldots,K$):

$$\mathcal{P}_k^{(f)} = \{(x, f_{k-1}(x)) \mid x \in \mathcal{D}_{\text{bad}}, c(f_{k-1}(x)) = 1\} \quad (8)$$

$$b_k = \textsc{Train}^{\text{good}\to\text{bad}}(\mathcal{P}_k^{(f)}) \quad (9)$$

$$\mathcal{P}_k^{(b)} = \{(b_k(y), y) \mid y \in \mathcal{D}_{\text{good}}, c(b_k(y)) = 0\} \quad (10)$$

$$f_k = \textsc{Train}^{\text{bad}\to\text{good}}(\mathcal{P}_k^{(f)} \cup \mathcal{P}_k^{(b)}), \quad (11)$$

where each equation corresponds to the steps (1)–(4) in the description above. $\textsc{Train}^{\text{good}\to\text{bad}}(\mathcal{P})$ trains an encoder-decoder model that maps "good"-side examples to "bad"-side examples in paired data $\mathcal{P}$, and $\textsc{Train}^{\text{bad}\to\text{good}}(\mathcal{P})$ does the reverse. **Red font** indicates the use of critic. The key intuition of BIFI is that thanks to the critic, (i) we can extract $\mathcal{D}_{\text{bad}}$ from the unlabeled data $\mathcal{D}_{\text{unlabel}}$ and incorporate realistic grammatical errors into our data (as opposed to the synthetic data), and (ii) we can verify if the "bad"-side and "good"-side of the generated pairs are actually "bad" and "good" (Eq 8, 10; **red font**), which improves the correctness of generated training data compared to vanilla backtranslation (Sennrich et al., 2016; Lample et al., 2018). We refer readers to Yasunaga and Liang (2021) for more details.

4.2 Experiments

We study our proposed approach (BIFI with LM-Critic) on GEC benchmarks, in both unsupervised and supervised settings.

4.2.1 Evaluation data

We evaluate on four GEC benchmarks, CoNLL-2014 test (Ng et al., 2014), BEA-2019 dev / test (Bryant et al., 2019), GMEG-yahoo and GMEG-wiki tests (Napoles et al., 2019), which span domains of essay/learner English, Wikipedia, and Yahoo! Answers. For CoNLL-2014, we use the official M^2 scorer (Dahlmeier and Ng, 2012), and for others we use the ERRANT metric (Bryant et al., 2017). We describe the training data separately for unsupervised (§4.2.2) and supervised (§4.2.3) settings.

4.2.2 Unsupervised setting

Setup and data. We consider the setup with no labeled training data. Existing GEC works (*e.g.*, Awasthi et al. (2019); Omelianchuk et al. (2020)) prepare synthetic paired data by heuristically corrupting sentences from the One-billion-word corpus (Chelba et al., 2013). We follow the same procedure, and train an encoder-decoder Transformer (Vaswani et al., 2017) on this synthetic data to be our **baseline fixer**. The size of the synthetic data is 9M pairs.

We then apply the BIFI training on top of the baseline fixer. As our **unlabeled data** to be used for BIFI, we want text that is likely to contain both ungrammatical and grammatical sentences. Hence, we take 10M sentences in total from the Yahoo! Answers corpus (Zhang et al., 2015) and the Wikipedia histories data (Grundkiewicz and Junczys-Dowmunt, 2014) for which we take sentences prior to revisions.[2] This unlabeled data is in the domains of two of our benchmarks (GMEG-wiki and GMEG-yahoo) but not of CoNLL-2014 and BEA-2019.

Implementation details. The encoder-decoder Transformer architecture has 12 layers, 16 attention heads and hidden state size of 768. The model parameters are initialized with the BART-base release (Lewis et al., 2020), and then optimized by Adam (Kingma and Ba, 2015), with batch size of 512 sequences, learning rate 0.0001, and gradient clipping 1.0 (Pascanu et al., 2013), on a single GTX Titan X GPU. For generation, we use beam search with beam size 10. We run the BIFI algorithm for $K = 1$ round. The total training time takes 2 days.

Results. Table 4 shows the results on the four GEC benchmarks. "Transformers" is our baseline fixer, trained on the synthetic paired data. Our proposed approach ("+BIFI") outperforms the baseline by substantial margins across the benchmarks, *e.g.*, +8 $F_{0.5}$ on GMEG-wiki and yahoo.

Since our method ("+BIFI") uses more (unlabeled) data than the baseline ("Transformer"), to be fully fair, we also conduct an experiment that controls the amount of training data seen by the model: Specifically, we apply BIFI to the baseline fixer without the critic, *i.e.*, the model sees the same amount of newly-generated paired data as

[2] This is not paired data, as we only take sentences pre revision, not post revision.

GEC system	CoNLL-2014 (test)			BEA-2019 (dev)			GMEG-wiki (test)			GMEG-yahoo (test)		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$	P	R	$F_{0.5}$	P	R	$F_{0.5}$
Transformer	59.2	29.2	49.1	44.2	17.9	34.1	52.1	26.5	43.7	44.4	36.9	42.7
+ BIFI with no critic	58.2	29.9	48.9	43.8	18.7	34.5	53.5	27.4	44.9	45.1	38.5	43.6
+ BIFI (ours)	**64.4**	**35.6**	**55.5**	**51.6**	**24.7**	**42.4**	**57.9**	**33.6**	**50.6**	**53.7**	**47.1**	**52.2**

Table 4: GEC results in the **unsupervised setting** (§4.2.2). "Transformers" is trained on synthetic paired data as in Awasthi et al. (2019). If we train it on more realistic paired data generated by BIFI (bottom row), it achieves improved results.

GEC system	Ens.	CoNLL-2014 (test)			BEA-2019 (test)		
		P	R	$F_{0.5}$	P	R	$F_{0.5}$
GPT3 (175B) with prompting		62.4	25.0	48.0	50.8	38.2	47.6
Zhao et al. (2019)		67.7	40.6	59.8	-	-	-
Awasthi et al. (2019)		66.1	43.0	59.7	-	-	-
Kiyono et al. (2019)		67.9	**44.1**	61.3	65.5	**59.4**	64.2
Zhao et al. (2019)	✓	74.1	36.3	61.3	-	-	-
Awasthi et al. (2019)	✓	68.3	43.2	61.2	-	-	-
Grundkiewicz et al. (2019)	✓	-	-	64.2	72.3	60.1	69.5
Kiyono et al. (2019)	✓	72.4	46.1	65.0	74.7	56.7	70.2
Kantor et al. (2019)	✓	-	-	-	78.3	58.0	73.2
GECToR (Omelianchuk et al., 2020)		77.5	40.1	65.3	79.2	53.9	72.4
GECToR (our base)		77.5	40.1	65.3	79.2	53.9	72.4
+ BIFI (ours)		**78.0**	40.6	**65.8**	**79.4**	55.0	**72.9**

Table 5: GEC results in the **supervised setting** with labeled data available (§4.2.3). "Ens." indicates an ensemble system.

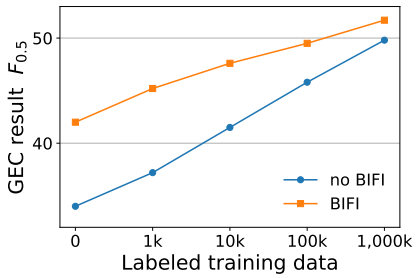


Figure 3: GEC results (y-axis) when varying the amount of labeled data available for training (x-axis). BIFI is particularly helpful in low-resource regimes.

"+BIFI" but they are not verified by LM-Critic. This system ("+BIFI with no critic") did not improve on the baseline much. These results indicate that the paired data generated by BIFI with LM-Critic is indeed more realistic and helpful than the initial synthetic data or pairs generated without LM-Critic.

The improved results in this unsupervised setting suggest that our approach is especially useful in domains with no labeled GEC data for training (*e.g.*, GMEG-wiki and yahoo; CoNLL-2014 and BEA-2019 have labeled data, which we use in §4.2.3).

Our results also suggest that while existing BIFI assumed access to an oracle critic (*i.e.*, compiler), an approximate critic (*i.e.*, LM-Critic) can also help to improve model learning. Our conjecture is that as long as the LM-Critic is better than random guessing (*e.g.*, 70 $F_{0.5}$ as shown in §3.3.3), it is useful for improving the quality of GEC training data generated in BIFI (Eq 8, 10), which in turns improves GEC performance. An interesting future direction is to use the breaker learned in BIFI (Eq 9 for the perturbation function in LM-Critic (§3.2) to further improve the critic, which may in turn help BIFI as well as GEC performance, creating a positive loop of learning.

4.2.3 Supervised setting

Setup and data. We also consider the common leaderboard setup that uses labeled training data and evaluates on CoNLL-2014 and BEA-2019. We take the state-of-the-art model, GECToR (Omelianchuk et al., 2020), as our **baseline fixer**. Following Omelianchuk et al. (2020), GECToR is first trained on the synthetic paired data described in §4.2.2, and is then trained on the labeled data available for the BEA-2019 task, which is the combination of:

- NUS Corpus of Learner English (NUCLE) (Dahlmeier et al., 2013)
- Lang-8 Corpus of Learner English (Lang-8) (Mizumoto et al., 2011; Tajiri et al., 2012)
- FCE dataset (Yannakoudakis et al., 2011)
- Write & Improve + LOCNESS Corpus (W&I + LOCNESS) (Bryant et al., 2019)

They are all in the domain of CoNLL-2014 and BEA-2019 (learner/essay English). The total size of the labeled data is 1M pairs.

We then apply the BIFI training on top of GECToR. As our **unlabeled data** to be used for BIFI, we use 10M sentences taken from Yahoo! Answers and Wikipedia histories (same as §4.2.2).

Implementation details. We use the same hyperparameters and training procedures for GECToR as in Omelianchuk et al. (2020). We run the BIFI algorithm for $K=1$ round. The total training time takes 4 days, on a single GTX Titan X GPU.

Results. Table 5 shows our results on CoNLL-2014 test and BEA-2019 test, along with existing systems on the leaderboard. Our approach ("+BIFI") provides an additional boost over our base model ("GECToR"). This suggests that BIFI with LM-Critic is helpful not only in the unsupervised setting but also when a substantial amount of labeled data (1M pairs) is available.

4.2.4 Analysis

Varying the amount of labeled data. We have studied GEC results when we have no labeled data (§4.2.2) and when we use all the labeled data (1M

(a) Pairs generated by synthetic corruption
x_{bad}: We look forward the to better treatments in the future. x_{good}: We look forward to better treatments in the future.
x_{bad}: The president-elect stayed away so as not to foregin matters until Bush. x_{good}: The president-elect stayed away so as not to complicate matters for Bush.
(b) Pairs generated by BIFI *without* LM-Critic
x_{bad}: If anyone is interested, here's the kink. x_{good}: If anyone is interested, here's the kinks.
x_{bad}: If you can't find a match yourself, horse trader will helps. x_{good}: If you can't find a match yourself, horse traders will help.
(c) Pairs generated by BIFI with LM-Critic (Ours)
x_{bad}: First Light is a award-winning novel by Sunil Gangopadhyay. x_{good}: First Light is an award-winning novel by Sunil Gangopadhyay.
x_{bad}: Except latter, the rivers are in underground tubes and not visible. x_{good}: Except for the latter, the rivers are in underground tubes and not visible.

Table 6: Examples of paired data generated by (a) synthetic corruption, (b) BIFI without critic, and (c) BIFI with LM-Critic. (a) tends to deviate from the type of grammatical errors humans make. (b) tends to have pairs where x_{good} is broken (*e.g.*, the first pair) or x_{bad} is already grammatical, as pairs are not verified by a critic. (c) is the most realistic.

(Input) The system is designed to use amplitude comparision for height finding. **(Baseline)** The system is designed to use amplitude comparison for height find. **(BIFI)** The system is designed to use amplitude comparison for height finding.
(Input) Lugu Lake, set in the subalpine zone in Hengduan is a landscape of pine-covered ecoregion. **(Baseline)** Lugu Lake, set in the subalpine zone in Hengduan, is their landscape of pine-covered ecoregion. **(BIFI)** Lugu Lake, set in the subalpine zone in Hengduan, is a landscape of pine-covered ecoregion.

Table 7: Examples where the baseline fixer trained with synthetic data fails but BIFI succeeds. The baseline tends to make unnecessary edits (*e.g.*, changing verb inflection or articles, due to heuristics used when generating synthetic data).

pairs) (§4.2.3). Here we analyze the interpolation. In Figure 3, we show the GEC performance ($F_{0.5}$) on the BEA-2019 dev set, when varying the amount of labeled data available for training from 0 to 1M. The blue line indicates a Transformer model first trained on the synthetic data and then trained on the available labeled data, which is our baseline. The orange line indicates that this baseline model is further trained with BIFI. We observe that BIFI outperforms the baseline consistently and is particularly helpful in low-resource regimes.

Pairs generated by BIFI. We quantitatively saw in §4.2.2 that the paired data generated by BIFI is helpful for learning GEC. Here we provide qualitative examples to compare the paired data generated by (a) synthetic corruption, (b) BIFI without critic, and (c) BIFI with LM-Critic (Table 6). We observe that (a) tends to deviate from the type of grammatical errors humans make (*e.g.*, inserting /replacing words arbitrarily); (b) tends to have pairs where x_{good} is broken (*e.g.*, the first pair in Table 6(b)) or x_{bad} is actually grammatical, as pairs are not verified by a critic; and (c) is the most realistic.

GEC model outputs. In Table 7, we analyze examples where the baseline fixer trained on synthetic data ("Transformer") fails but our model ("+BIFI") succeeds. We find that the baseline tends to make unnecessary edits (e.g., changing verb inflection or articles), due to the heuristics used when generating synthetic data. In contrast, BIFI achieves higher precision.

5 Related work and discussion

Grammatical error correction (GEC). GEC models are commonly trained from human-labeled data (Nicholls, 2003; Dahlmeier et al., 2013; Yannakoudakis et al., 2011; Bryant et al., 2019), or synthetic data generated by heuristically corrupting unlabeled sentences (Awasthi et al., 2019; Zhao et al., 2019; Grundkiewicz et al., 2019; Katsumata and Komachi, 2019; Omelianchuk et al., 2020). Several works aim to improve the methods for generating paired data, such as learning a breaker from existing labeled data (Lichtarge et al., 2019), applying backtranslation (Sennrich et al., 2016) to GEC (Xie et al., 2018; Kiyono et al., 2019), and synthesizing extra paired data by comparing model predictions and references (Ge et al., 2018). Different from the above works, our method (i) does not require labeled data (works for both unsupervised and supervised settings), and (ii) uses LM-Critic to filter the "bad"-side and "good"-side of generated pairs.

Automatic text evaluation. Popular metrics used to assess the quality of text in GEC include GLEU (Napoles et al., 2015, 2017), M^2 (Dahlmeier and Ng, 2012), ERRANT (Bryant et al., 2017) and I-measure (Felice and Briscoe, 2015). While these methods require reference text to compare to, LM-Critic does not. Several prior works also study reference-less methods to assess grammaticality of text: Wan et al. (2005); Mutton et al. (2007); Vadlapudi and Katragadda (2010) use part-of-speech (POS) tagger or parser predictions to score grammaticality; Napoles et al. (2016); Warstadt et al. (2018); Katinskaia et al. (2019); Niu and Penn (2020) train grammatical error detection (GED) or acceptability judgement systems. However, these works require POS taggers, parsers or GED systems trained on labeled data, which may not scale or generalize well beyond the domain of training data. In contrast, LM-Critic only requires an LM, which is unsupervised and can be pretrained on various domains of unlabeled corpora.

Pretrained LM for text evaluation. Several works use pretrained LMs for text evaluation. For reference-based metrics, Zhang et al. (2020) use an LM's embeddings to measure the similarity between input text and reference text. For reference-less

metrics, several works (Kann et al., 2018; Stahlberg et al., 2019) use an LM's probability as a fluency score of text. While this provides a continuous score for fluency, it in itself cannot classify grammatical / ungrammatical sentences. Our LM-Critic goes a step further to consider the local optimum criterion for classifying grammaticality. The reason we want a classifier (critic) is that we work on unsupervised learning of GEC. In the unsupervised setting, there is a distributional shift problem—the synthetically-generated paired data does not match the distribution of grammatical errors humans make. BIFI is a solution for obtaining realistic paired data in an unsupervised way, but it requires a critic. This led us to design a critic for GEC in this work. We note that LM-Critic is not meant to replace existing evaluation metrics for GEC, but rather is an approximate critic to assess grammaticality and help the learning of GEC.

Separately, several works (Tenney et al., 2019; Hewitt and Manning, 2019; Yasunaga and Lafferty, 2019; Cao et al., 2020) induce grammar or syntactic structures from LMs, suggesting that LMs can learn about grammaticality in an unsupervised way. As this capacity is likely to grow with the size of LMs (Radford et al., 2019; Brown et al., 2020; Kaplan et al., 2020), we think that how to leverage pretrained LMs for GEC will become an increasingly important research problem.

6 Conclusion

We presented LM-Critic, a method that uses a pretrained language model (LM) as a critic for assessing sentence grammaticality. Using LM-Critic and the BIFI algorithm, we learn grammatical error correction (GEC) by generating realistic training data from unlabeled text. Notably, our approach does not require labeled data, and can also be viewed as an unsupervised method to turn a (GPT2-scale) pretrained LM into an actual GEC system. Using multiple GEC datasets, we showed that our approach achieves strong performance on unsupervised GEC, suggesting the promise of our method for domains and languages with no labeled GEC data. We hope this work opens up research avenues in LM-based critics and unsupervised GEC.

Acknowledgments

We thank Pang Wei Koh, Tianyi Zhang, Rodrigo Castellon, members of the Stanford P-Lambda, SNAP and NLP groups, as well as our anonymous reviewers for valuable feedback. This work was supported in part by a Funai Foundation Scholarship and NSF CAREER Award IIS-1552635.

Reproducibility

Code and data are available at
https://github.com/michiyasunaga/LM-Critic.
Experiments are available at
https://worksheets.codalab.org/worksheets/0x94456a63e1ee4ccfaabdc7f6a356cc82.

References

Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. 2019. Parallel iterative edit models for local sequence transduction. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Christopher Bryant, Mariano Felice, Øistein E Andersen, and Ted Briscoe. 2019. The bea-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*.

Christopher Bryant, Mariano Felice, and Edward Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Association for Computational Linguistics (ACL)*.

Steven Cao, Nikita Kitaev, and Dan Klein. 2020. Unsupervised parsing via constituency tests. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

Shamil Chollampatt and Hwee Tou Ng. 2018. A multilayer convolutional encoder-decoder neural network for grammatical error correction. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner english: The nus corpus of learner english. In *Proceedings of the eighth workshop on innovative use of NLP for building educational applications*.

Hal Daume III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of artificial Intelligence research*.

Mariano Felice and Ted Briscoe. 2015. Towards a standard evaluation method for grammatical error detection and correction. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Tao Ge, Furu Wei, and Ming Zhou. 2018. Fluency boost learning and inference for neural grammatical error correction. In *Association for Computational Linguistics (ACL)*.

Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2014. The wiked error corpus: A corpus of corrective wikipedia edits and its application to grammatical error correction. In *International Conference on Natural Language Processing*.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Jianshu Ji, Qinlong Wang, Kristina Toutanova, Yongen Gong, Steven Truong, and Jianfeng Gao. 2017. A nested attention neural hybrid model for grammatical error correction. In *Association for Computational Linguistics (ACL)*.

Erik Jones, Robin Jia, Aditi Raghunathan, and Percy Liang. 2020. Robust encodings: A framework for combating adversarial typos. In *Association for Computational Linguistics (ACL)*.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching neural grammatical error correction as a low-resource machine translation task. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Katharina Kann, Sascha Rothe, and Katja Filippova. 2018. Sentence-level fluency evaluation: References help, but can be spared! In *Conference on Computational Natural Language Learning (CoNLL)*.

Yoav Kantor, Yoav Katz, Leshem Choshen, Edo Cohen-Karlik, Naftali Liberman, Assaf Toledo, Amir Menczel, and Noam Slonim. 2019. Learning to combine grammatical error corrections. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Anisia Katinskaia, Sardana Ivanova, Roman Yangarber, et al. 2019. Multiple admissibility in language learning: Judging grammaticality using unlabeled data. In *The 7th Workshop on Balto-Slavic Natural Language Processing Proceedings of the Workshop*.

Satoru Katsumata and Mamoru Komachi. 2019. (almost) unsupervised grammatical error correction using synthetic comparable corpus. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2019. An empirical study of incorporating pseudo data into grammatical error correction. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations (ICLR)*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Association for Computational Linguistics (ACL)*.

Jared Lichtarge, Chris Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar, and Simon Tong. 2019. Corpora generation for grammatical error correction. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Masato Mita, Tomoya Mizumoto, Masahiro Kaneko, Ryo Nagata, and Kentaro Inui. 2019. Cross-corpora evaluation and analysis of grammatical error correction models—is single-corpus evaluation enough? In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning sns for automated japanese error correction of second language learners. In *International Joint Conference on Natural Language Processing (IJCNLP)*.

Andrew Mutton, Mark Dras, Stephen Wan, and Robert Dale. 2007. Gleu: Automatic evaluation of sentence-level fluency. In *Association of Computational Linguistics (ACL)*.

Jakub Náplava and Milan Straka. 2019. Grammatical error correction in low-resource scenarios. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*.

Courtney Napoles, Maria Nădejde, and Joel Tetreault. 2019. Enabling robust grammatical error correction in new domains: Data sets, metrics, and analyses. *Transactions of the Association for Computational Linguistics*.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *Association of Computational Linguistics (ACL)*.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2016. There's no comparison: Reference-less evaluation metrics in grammatical error correction. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. In *European Chapter of the Association for Computational Linguistics (EACL)*.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The conll-2014 shared task on grammatical error correction. In *Conference on Computational Natural Language Learning (CoNLL)*.

Diane Nicholls. 2003. The cambridge learner corpus: Error coding and analysis for lexicography and elt. In *Proceedings of the Corpus Linguistics 2003 conference*.

Jingcheng Niu and Gerald Penn. 2020. Grammaticality and language modelling. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. Gector–grammatical error correction: Tag, not rewrite. In *Proceedings of the 15th Workshop on Innovative Use of NLP for Building Educational Applications*.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning (ICML)*, pages 1310–1318.

Danish Pruthi, Bhuwan Dhingra, and Zachary C Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *Association for Computational Linguistics (ACL)*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Association for Computational Linguistics (ACL)*.

Felix Stahlberg, Christopher Bryant, and Bill Byrne. 2019. Neural grammatical error correction with finite state transducers. In *North American Association for Computational Linguistics (NAACL)*.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for esl learners using global context. In *Association for Computational Linguistics (ACL)*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. In *Association for Computational Linguistics (ACL)*.

Ravikiran Vadlapudi and Rahul Katragadda. 2010. On automated evaluation of readability of summaries: Capturing grammaticality, focus, structure and coherence. In *Proceedings of the NAACL HLT 2010 student research workshop*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Stephen Wan, Robert Dale, and Mark Dras. 2005. Searching for grammaticality: Propagating dependencies in the viterbi algorithm. In *Proceedings of the Tenth European Workshop on Natural Language Generation (ENLG-05)*.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Ziang Xie, Anand Avati, Naveen Arivazhagan, Dan Jurafsky, and Andrew Y Ng. 2016. Neural language correction with character-based attention. *arXiv preprint arXiv:1603.09727*.

Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Y Ng, and Dan Jurafsky. 2018. Noising and denoising natural language: Diverse backtranslation for grammar correction. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading esol texts. In *Association for Computational Linguistics (ACL)*.

Michihiro Yasunaga and John D Lafferty. 2019. Topiceq: A joint topic and mathematical equation model for scientific texts. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Michihiro Yasunaga and Percy Liang. 2021. Break-It-Fix-It: Unsupervised Learning for Program Repair. In *International Conference on Machine Learning (ICML)*.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations (ICLR)*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *North American Association for Computational Linguistics (NAACL)*.