Leveraging Prefix Transfer for Multi-Intent Text Revision

Ruining Chong[12*], Cunliang Kong[12*], Liu Wu[2], Zhenghao Liu[3], Ziye Jin[4], Liner Yang[12†], Yange Fan[5], Hanghang Fan[5], Erhong Yang[12]
[1]National Language Resources Monitoring and Research Center for Print Media, Beijing Language and Culture University, China
[2]School of Information Science, Beijing Language and Culture University, China
[3]Department of Computer Science and Technology, Northeastern University, China
[4]School of Arts and Sciences, New York University Shanghai, China
[5]Kika Tech, China
lineryang@gmail.com

Abstract

Text revision is a necessary process to improve text quality. During this process, writers constantly edit texts out of different edit intentions. Identifying edit intention for a raw text is always an ambiguous work, and most previous work on revision systems mainly focuses on editing texts according to one specific edit intention. In this work, we aim to build a multi-intent text revision system that could revise texts without explicit intent annotation. Our system is based on prefix-tuning, which first gets prefixes for every edit intent, and then trains a prefix transfer module, enabling the system to selectively leverage the knowledge from various prefixes according to the input text. We conduct experiments on the ITERATER dataset, and the results show that our system outperforms baselines. The system can significantly improve the SARI score with more than 3% improvements, which thrives on the learned editing intention prefixes.

1 Introduction

Revision is an essential process to improve the text quality (Vaughan and McDonald, 1986). During this process, writers perform various editing operations on the text with different editing intentions. As shown in Figure 1, the writer corrects misspelled words to improve text *fluency*, deletes redundant words to improve text *clarity*, adds connective words to improve text *coherence*, inserts adverbs to convey the writer's writing preferences (*style*) and modifies data to update text information (*meaning-changed*).

Lots of recent studies have focused on a text revision task corresponding to a specific edit intention, such as grammatical error correction (Yang et al., 2022; Liu et al., 2021; Omelianchuk et al., 2020; Kaneko et al., 2020), text simplification (Dong et al., 2019; Jiang et al., 2020; Omelianchuk et al., 2021; Martin et al., 2022), and text style transfer (Reid and Zhong, 2021; Malmi et al., 2020). The work divides text revision into several independent problems. While some methods with strong universality can be applied to multiple tasks (Malmi et al., 2019; Stahlberg and Kumar, 2020; Mallinson et al., 2020), they train different models on various data sets. Real-world scenarios require addressing multiple types of editing errors at the same time, such as grammatical errors, spelling errors, etc. But these methods failed to integrate knowledge from these tasks into a unified model.

To solve the problem, Du et al. (2022) attempted to train one model using data with multiple editing intentions and leveraged edit intent information by simply appending it to the input. However, when adding a new intent, the entire model must be re-

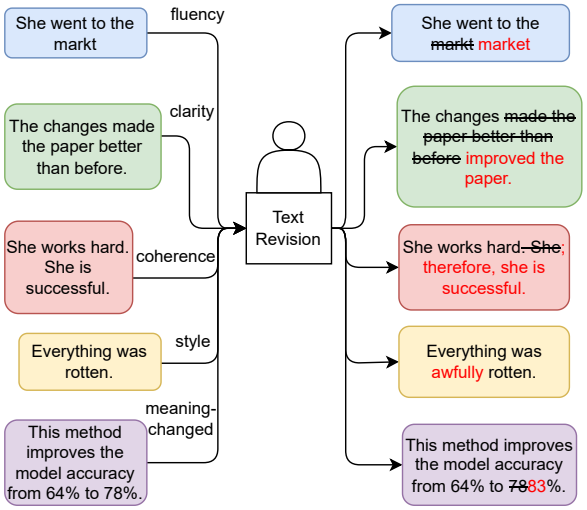

Figure 1: Writers edit texts out of different edit intentions through text revision.

*Equal contribution
†Corresponding author: Liner Yang

trained. A more lightweight and scalable approach to multi-intent text revision is still required.

Li and Liang (2021) proposed a new kind of prompt tuning method to quickly adapt a pre-trained model to new tasks, which is called prefix-tuning. Prompt tuning can help the pre-trained language model to locate the task learned in pretraining and enable the related knowledge to model text revision with different edit intentions (Reynolds and McDonell, 2021). This method enables a model to handle multiple edit intentions in a lightweight and scalable way.

In this paper, we present our method: a prefix-tuning-based model which adapts to text revision with multiple edit intentions. This method involves a two-step training process. In the first step, we initialize a pre-trained language model (PLM) and train multiple prefixes on it. Each edit intention corresponds to a prefix. In the second step, a prefix transfer module is trained at each attention layer of the PLM. The prefix transfer module is configured as two attention units that act respectively on this layer's key states and value states. It enables our model to learn a tailored prefix for the given input with the help of prefix embeddings from the predefined tasks.

We conduct experiments on ITERATER (Du et al., 2022), an iterative text revision dataset. It mainly contains parallel sentences with five edit intentions: *fluency*, *coherence*, *clarity*, *style*, and *meaning-changed*. The results show that our approach performs better than the fully fine-tuned BART (Lewis et al., 2020) and PEGASUS (Zhang et al., 2020) baselines reported in Du et al. (2022) with fewer training parameters.

2 Related Work

2.1 Iterative Text Revision

For the first time, Du et al. (2022) systematically studied the iterative revision phenomenon in human writing. They presented the ITERATER, an annotated dataset across multiple domains of formally human-written text, which includes Wikipedia, ArXiv, and Wikinews. And they trained several types of text revision models using ITERATER. Dwivedi-Yu et al. (2022) presented EDITEVAL, an instruction-based benchmark, to evaluate the editing capabilities of models and they also included the test set of ITERATER in it. Based on Du et al. (2022), our work further explores the method of text revision.

2.2 Transfer Learning of Prompt Tuning

Transfer learning is a common and powerful technique in NLP (Raffel et al., 2020). Some recent studies have tried to improve prompt tuning performance by leveraging the knowledge of multiple related or unrelated tasks. Asai et al. (2022) used an attention module to make use of the knowledge in exiting soft prompts (Lester et al., 2021) while learning a new task. Chen et al. (2022) improved the few-shot text summarization by multi-task pre-training and prefix-tuning. Specifically, they pre-trained a summarization model on a set of popular summarization datasets and then conducted prefix-tuning for it on an unseen summarization task. Different from their modeling of a new task through existing tasks, our work aims to achieve the mutual utilization of knowledge between different edit intents in text revision.

3 Method

The revision task can be defined as the following process: given a source sentence $\mathbf{x} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]$ and an optional edit intent $e \in E$ to generate a revised sentence $\mathbf{y} = [\mathbf{y}_1, \ldots, \mathbf{y}_n]$, where E is the set of all edit intentions. Note that e is optional because it can be inferred from the input $\mathbf{x}$.

Our method is depicted in Figure 2. It includes two stages: the multi-prefix tuning stage and the prefix transfer stage.

3.1 Multi-Prefix Tuning Stage

The prefix is a set of parameters on every attention layer of PLM. For an edit intention e, at each attention layer, the prefix can be described as $P_e = \{P_e^K, P_e^V\}$, where P_e^K and P_e^V are parameters added before the key states and value states in this attention layer. After adding these parameters, the calculation of the attention head in this layer becomes:

$$H = \text{Attention}(Q, [P_e^K; K], [P_e^V; V]) \quad (1)$$

where H is the output vector sequence; Q, K, V are query states, key states, and value states, respectively; Attention means scaled dot-product attention. Only P_e^K and P_e^V are updated during the training process. Note that we ignore the layer number information because the operation for each layer is the same.

As shown in the left part of Figure 2, for every edit intention e, we train a prefix P_e accordingly.

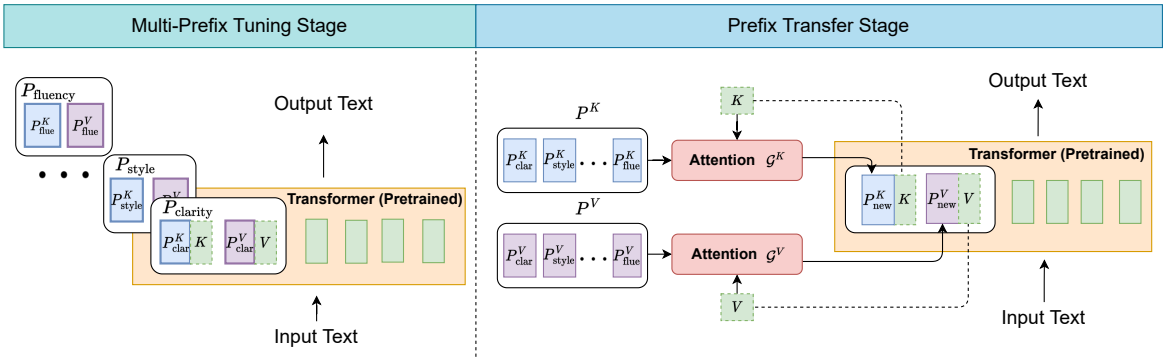

Figure 2: Overview of our method. The frozen weights of PLM are indicated with solid green rectangles. Two objects connected by a dotted line are the same.

In this way, the model could revise an intention-annotated text by activating the corresponding prefix at inference.

3.2 Prefix Transfer Stage

Identifying edit intention is always an ambiguous work. At the prefix transfer stage, we aim to build a new prefix for an unannotated input instance by transferring existing prefixes. The new prefix P_{new} is instance-specific.

The prefix transfer stage is described in the right part of Figure 2. At each layer, we rearrange the prefixes $\{P_e \mid e \in E\}$ obtained in the last stage as $P^K = \{P_e^K \mid e \in E\}$ and $P^V = \{P_e^V \mid e \in E\}$ according to whether they are configured before the key states or before the value states. Then a pair of attention units $\mathcal{G}^K$ and $\mathcal{G}^V$ are trained for P^K and P^V.

Take $\mathcal{G}^K$ as an example. It calculates the similarity between the key states K and every P_e^K in P^K to get attention scores.

The similarity can't be calculated directly, because K and P_e^K have different lengths. So we perform the max-pool operation for length dimension on K and P_e^K. After that, we obtain $\hat{K} \in \mathbb{R}^d$ and $\hat{P}_e^K \in \mathbb{R}^d$, where d is the dimension of the hidden states in the PLM.

To get attention scores, we train a fully connected layer to extract features from $\hat{K}$:

$$H = \text{NonLinear}(W^\top(\hat{K})) \tag{2}$$

where $W \in \mathbb{R}^{d \times d}$ is a transfer matrix updated during training. Following Asai et al. (2022), we use SiLU (Elfwing et al., 2018) for the non-linear layer and add a Layer Norm (Ba et al., 2016) layer:

$$H_{\text{norm}} = \text{LayerNorm}(H) \tag{3}$$

Then, we calculate the attention scores for intent e as follows:

$$a_e = \frac{\exp(\hat{P}_e^K \cdot H_{\text{norm}})/T}{\sum_{i \in E} \exp(\hat{P}_i^K \cdot H_{\text{norm}})/T} \tag{4}$$

where T is the softmax temperature (Radford et al., 2021) which could avoid making the attention unit over-confident.

Finally we use them to build P_{new}^K as follows:

$$P_{\text{new}}^K = \sum_{e \in E} a_e P_e^K \tag{5}$$

In the same way, we get P_{new}^V by $\mathcal{G}^V$. Using the new prefix $P_{\text{new}} = [P_{\text{new}}^K, P_{\text{new}}^V]$, our system could revise the unannotated input instance with the knowledge from existing prefixes.

4 Experimental Setup

We choose BART-large as the PLM for our system and use adapter-transformers (Pfeiffer et al., 2020) to implement prefix-tuning. More implementation details are in Appendix A.

4.1 Datasets

We conduct our experiments on the iterative text revision dataset: ITERATER (Du et al., 2022). We remove the *Other* class of the data as it essentially contains a variety of unrecognized edit intentions and accounts for a small proportion (1.44%). The entire dataset consists of two parts: ITERATER-HUMAN and ITERATER-FULL. The former is a smaller dataset with manual annotation of edit intentions, while the latter is a large dataset annotated by a classification model trained on ITERATER-HUMAN. We train our model on both of them. Following Du et al. (2022), we report the results on the test set of ITERATER-HUMAN in Section 5,

		IteraTeR-human				IteraTeR-full			
Model	Intent	SARI	BLEU	R-L	Avg.	SARI	BLEU	R-L	Avg.
BART-FineTune	✗	33.20	78.56	85.20	65.66	33.88	78.55	86.05	66.16
Pegasus-FineTune	✗	33.09	79.09	86.77	66.32	34.67	78.21	87.06	66.65
BART-SinglePrefix	✗	30.97	81.82	**87.57**	66.79	36.81	79.65	86.37	67.61
BART-FineTune	✓	34.77	74.43	84.45	64.55	**37.28**	77.50	86.14	66.97
Pegasus-FineTune	✓	34.43	78.85	86.84	66.71	37.11	77.60	86.84	67.18
BART-SinglePrefix	✓	31.23	81.66	87.39	66.76	36.54	77.46	85.80	66.60
Multi-Prefix	✓	33.12	**82.00**	**87.57**	67.56	37.25	78.25	86.18	67.23
PrefixTransfer	✗	**36.01**	80.53	87.18	**67.91**	37.12	**80.34**	**87.61**	**68.36**

Table 1: Model performances on the test set of IteraTeR-human. **Avg.** is the average of **SARI**, **BLEU** and **R-L**. **R-L** refers to Rouge-L. **Intent** indicates whether an intent label is needed during inference. Multi-Prefix is the model that only trains the multi-prefix tuning stage. PrefixTransfer is the model that trains both the multi-prefix tuning stage and the prefix transfer stage.

which is completely a human-created dataset and is reliable for evaluation. We show more details of the datasets in Appendix B.

4.2 Evaluation Metrics

Following previous work, we report three metrics: SARI (Xu et al., 2016), Rouge-L (Lin, 2004), and BLEU (Papineni et al., 2002). Among them, SARI is considered an important metric in situations where input text and output text have a large overlap in words. It also indicates the positive impact of revisions on document quality.

The setting of evaluation metrics is the same as Du et al. (2022). We use the metrics package from Huggingface transformers (Wolf et al., 2020) to calculate the SARI, BLEU, and Rouge-L scores.

4.3 Models Setup and Baselines

Using our method, we train the models in two ways: the model that only trains the multi-prefix tuning stage and that trains both the multi-prefix tuning stage and the prefix transfer stage.

We compare our method with three baselines: full fine-tuning BART (BART-FineTune), full fine-tuning Pegasus (Pegasus-FineTune), and prefix-tuning of BART with a single prefix (BART-SinglePrefix). Both BART and Pegasus are generative models based on the transformer architecture. Compared to the edit-based model FELIX, they perform better. We use the results reported by Du et al. (2022) for these two models. Furthermore, we compare BART-SinglePrefix as a possible technical solution as we choose BART as our backbone model. BART-SinglePrefix trains only one prefix on the entire dataset.

All three baselines are trained with two configurations. The first configuration is using the pure sentence pairs without edit intention annotations to train the model. The second configuration is appending an edit intent token at the beginning of the input text during the training process, which is the same as the approach of Du et al. (2022).

5 Results and Analysis

5.1 Main Results

The main results are shown in Table 1. Compared to training with a single prefix, the setting of multiple prefixes can improve the results, especially training on IteraTeR-human. Meanwhile, with fewer training parameters, the multi-prefix setting could achieve a comparable SARI score and better average score than the fully fine-tuned BART and Pegasus baselines.

Moreover, prefix transfer could further improve the model's performance. Training on IteraTeR-human, prefix transfer significantly improves the SARI score from 33.12 to 36.01 and gets the highest average score of 67.91. Training on IteraTeR-full, prefix transfer can also improve the average score from 67.23 to 68.36.

An interesting phenomenon is that training on different datasets results in different gains for prefix transfer in evaluation metrics. On IteraTeR-human, prefix transfer improves the SARI score significantly. While on IteraTeR-full, prefix transfer mainly improves the BLEU score and Rouge-L score. One possible explanation is that in situations when the training data is small, prefix transfer tends to learn more editing operations to improve text quality. In this way, the SARI score related to editing operations will be improved significantly. When the training data is sufficient, pre-

Stage 1	Stage 2	SARI	BLEU	R-L	Avg.
HUMAN	HUMAN	36.01	**80.53**	87.18	67.91
FULL	FULL	37.12	80.34	**87.61**	68.36
FULL	HUMAN	**38.44**	80.24	86.90	**68.53**

Table 2: Results on the test set of ITERATER-HUMAN. Stage 1 indicates the training data used in the multi-prefix tuning stage. Stage 2 indicates the training data used in the prefix transfer stage.

fix transfer will model the gold reference in more detail. So the BLEU score and the Rouge-L score will be improved.

5.2 Analysis

We further tried to use different training data at different stages of training to conduct experiments. The results are shown in Table 2.

We find that the best practice is to train the model on ITERATER-FULL in the multi-prefix tuning stage and on ITERATER-HUMAN in the prefix transfer stage, which gets the highest SARI score and average score. This may be because of the different distributions of manually annotated edit intent and automatically annotated edit intent. The auto-annotated dataset ITERATER-FULL contains many incorrectly classified sentences, which may cause mismatched knowledge in prefixes. In the prefix transfer stage, due to the existence of mismatched knowledge and incorrectly classified sentences, the continued use of the same training data may finally cause a certain degree of negative transfer. However, if we use ITERATER-HUMAN in the prefix transfer stage, the impact of negative transfer will be mitigated, because ITERATER-HUMAN only contains correctly classified sentences.

In Appendix C, we separately provide the performance results on different edit intentions of the best-performing model.

6 Conclusion

In this paper, we introduce a new method for multi-intent text revision. The system is based on prefix-tuning, which first obtains a prefix for every edit intention and then learns to transfer the knowledge in prefixes for every input instance by training a prefix transfer module. This prefix transfer module is configured as two attention units that act respectively on the key states and the value states at each attention layer of the PLM. In this way, our method can make full use of the knowledge of various edit intentions and does not need to annotate the intentions of the input. The experimental results show that our method significantly outperforms baselines, and both multi-prefix and prefix transfer settings could improve the performance.

Limitations

Due to the lack of multi-intent text revision datasets, we only conduct experiments on ITERATER. Although it is a multi-domain dataset, we only use its sentence-level data, and each sentence pair only contains one editing operation. The robustness of our method is still to be verified by evaluating it on more types of datasets in future work.

Another limitation of our work is that we only made improvements at the model level. We have noticed that Kim et al. (2022) recently improved text revision by leveraging extra data from other text editing tasks and performing editable span detection before revising. Similar methods can also be applied to our model and will be tried in our future work.

Ethics Statement

The PrefixTransfer method mainly aims at fusing multiple prefixes to obtain a unified model that can perform multi-intent text revision. The experiments are based on the ITERATER dataset, which is unlikely to include harmful content.

Acknowledgments

This work was supported by the funds of the Research Project of the National Language Commission (No. ZDI145-24), Natural Science Foundation of China (No. 62206042), Fundamental Research Funds for the Central Universities (No. 21PT04, No. N2216013) and China Postdoctoral Science Foundation (No. 2022M710022). We would like to thank all anonymous reviewers for their valuable comments and suggestions on this work.

References

Akari Asai, Mohammadreza Salehi, Matthew E Peters, and Hannaneh Hajishirzi. 2022. ATTEMPT: Parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6655–6672.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Yulong Chen, Yang Liu, Ruochen Xu, Ziyi Yang, Chenguang Zhu, Michael Zeng, and Yue Zhang. 2022. Unisumm: Unified few-shot summarization with multi-task pre-training and prefix-tuning. *arXiv preprint arXiv:2211.09783*.

Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. 2019. EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3393–3402.

Wanyu Du, Vipul Raheja, Dhruv Kumar, Zae Myung Kim, Melissa Lopez, and Dongyeop Kang. 2022. Understanding iterative revision from human-written text. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 3573–3590.

Jane Dwivedi-Yu, Timo Schick, Zhengbao Jiang, Maria Lomeli, Patrick Lewis, Gautier Izacard, Edouard Grave, Sebastian Riedel, and Fabio Petroni. 2022. EditEval: An instruction-based benchmark for text improvements. *arXiv preprint arXiv:2209.13331*.

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. 2018. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, and Wei Xu. 2020. Neural CRF model for sentence alignment in text simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7943–7960.

Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4248–4254.

Zae Myung Kim, Wanyu Du, Vipul Raheja, Dhruv Kumar, and Dongyeop Kang. 2022. Improving iterative text revision by learning where to edit from other revision tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9986–9999.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059.

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

Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 4582–4597.

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

Zhenghao Liu, Xiaoyuan Yi, Maosong Sun, Liner Yang, and Tat-Seng Chua. 2021. Neural quality estimation with multiple hypotheses for grammatical error correction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5441–5452.

Jonathan Mallinson, Aliaksei Severyn, Eric Malmi, and Guillermo Garrido. 2020. FELIX: Flexible text editing through tagging and insertion. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1244–1255.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5054–5065.

Eric Malmi, Aliaksei Severyn, and Sascha Rothe. 2020. Unsupervised text style transfer with padded masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 8671–8680.

Louis Martin, Angela Fan, Éric Villemonte De La Clergerie, Antoine Bordes, and Benoît Sagot. 2022. MUSS: Multilingual unsupervised sentence simplification by mining paraphrases. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1651–1664.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170.

Kostiantyn Omelianchuk, Vipul Raheja, and Oleksandr Skurzhanskyi. 2021. Text simplification by tagging. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 11–25.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Machel Reid and Victor Zhong. 2021. LEWIS: Levenshtein editing for unsupervised text style transfer. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3932–3944.

Laria Reynolds and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–7.

Felix Stahlberg and Shankar Kumar. 2020. Seq2Edits: Sequence transduction using span-level edit operations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 5147–5159.

Marie M. Vaughan and David D. McDonald. 1986. A model of revision in natural language generation. In *Proceedings of the 24th Annual Meeting on Association for Computational Linguistics*, pages 90–96, USA. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Liner Yang, Chengcheng Wang, Yun Chen, Yongping Du, and Erhong Yang. 2022. Controllable data synthesis method for grammatical error correction. *Frontiers of Computer Science*, pages 1–10.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339.

A Details on Computational Experiments

Our system is established based on BART-large (with 400 million parameters). We use the AdamW optimizer with weight decay and adopt Noam Optimizer. We initial the learning rate of $5e^{-5}$ in the multi-prefix tuning stage and $1e^{-5}$ in the prefix transfer stage. We set the warm-up steps to 4000. Regarding batch size, we use max-token configuration and set the maximum of tokens to 1024. The maximum epoch is set to 100. And we set an early stop strategy in the patience of 20 epochs.

The length of the prefix (with 40 million parameters) is 12 and the prefix vectors are not optimized directly but reparameterized via a bottleneck MLP which has a middle dimension of 512. We set the prefix dropout to 0.2.

We do validation every epoch while training the model on ITERATER-HUMAN and every 200 steps while training the model on ITERATER-FULL.

We report descriptive statistics with a single run. We deploy all our experiments on a slurm cluster. We train the prefixes on 4 Tesla V100-SXM2 (16GB) GPUs and train prefix transfer modules on an NVIDIA TITAN RTX(24GB) GPU.

Dataset	Train	Dev	Test
ITERATER-HUMAN	3,215	385	360
ITERATER-FULL	157,579	19,705	19,703

Table 3: Data split of ITERATER after removing the **Other** class.

B Details of Dataset

B.1 Taxonomy

The taxonomy of edit intentions in ITERATER after removing **Other**:

- **Fluency** Fix grammatical errors in the text.
- **Coherence** Make the text more cohesive, logically linked, and consistent as a whole.
- **Clarity** Make the text more formal, concise, readable, and understandable.
- **Style** Convey the writer's writing preferences, including emotions, tone, voice, etc.
- **Meaning-changed** Update or add new information to the text.

B.2 Data split

ITERATER dataset is splited as in Table 3 after romoving the **Other** class.

B.3 License

The ITERATER dataset uses Apache License, and it allows the data for academic usage.

C Model Performance of Different Edit Intentions

Edit Intention	SARI	BLEU	R-L	Avg.
CLARITY	34.01	78.18	84.62	65.60
FLUENCY	48.91	90.81	97.30	79.01
COHERENCE	38.66	84.83	90.39	71.29
STYLE	32.12	76.34	87.49	65.32
MEANING-CHANGED	37.65	51.46	68.98	52.70

Table 4: The performance results on different edit intentions of the best-performing model