Edit-Constrained Decoding for Sentence Simplification

Tatsuya Zetsu
LY Corporation
Japan
zetsu.t4@gmail.com

Yuki Arase
Tokyo Institute of Technology
Japan
arase@c.titech.ac.jp

Tomoyuki Kajiwara
Ehime University
Japan
kajiwara@cs.ehime-u.ac.jp

Abstract

We propose edit operation based lexically constrained decoding for sentence simplification. In sentence simplification, lexical paraphrasing is one of the primary procedures for rewriting complex sentences into simpler correspondences. While previous studies have confirmed the efficacy of lexically constrained decoding on this task, their constraints can be loose and may lead to sub-optimal generation. We address this problem by designing constraints that replicate the edit operations conducted in simplification and defining stricter satisfaction conditions. Our experiments indicate that the proposed method consistently outperforms the previous studies on three English simplification corpora commonly used in this task.

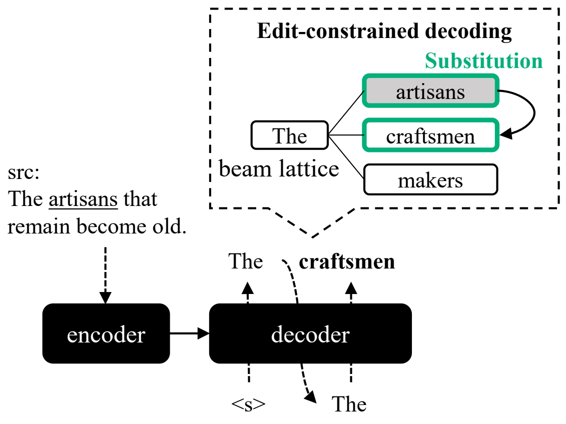

Figure 1: Our method constrains generation based on edit operations during beam search. Here, ‘artisans’ is replaced by ‘craftsmen’ by a substitution constraint.

1 Introduction

Lexically constrained decoding (Anderson et al., 2017; Post and Vilar, 2018; Hu et al., 2019) allows to explicitly apply human knowledge in language generation, which has been employed in various tasks. Applications include machine translation using a bilingual dictionary of technical terms as constraints (Chatterjee et al., 2017; Hokamp and Liu, 2017), data augmentation using references as constraints (Geng et al., 2023b), style transfer using style-specific vocabulary as constraints (Kajiwara, 2019), knowledge-grounded generation (Choi et al., 2023), and commonsense reasoning using common concepts as constraints (Lu et al., 2021). The notable advantage of lexically constrained decoding is its direct applicability to decoders without the need for model (re)training.

Lexically constrained decoding is particularly appealing for sentence simplification, where the transformation of complex tokens to simpler ones is crucial. Laufer (1989) points out that learners of a foreign language need to know 95% of tokens in the input text to comprehend the text message. A recent evaluation metric for sentence simplification (Heineman et al., 2023) also focuses on word-level edit quality. Previous studies employed negative and positive lexical constraints, i.e., to suppress the output of difficult tokens (negative constraints) (Dehghan et al., 2022) and to encourage the output of their corresponding simpler alternatives (positive constraints) (Zetsu et al., 2022). These studies showed that such simple constraints significantly improve the quality of simplification. However, their naive constraints are not tight enough, which leads to sub-optimal simplification satisfying easier constraints yet failing on difficult ones.

To address this problem, we design constraints more directed to simplification associated with stricter satisfaction conditions. Sentence simplification conducts three edit operations to rewrite sentences, i.e., *insertion*, *deletion*, and *substitution* of tokens. We expand NeuroLogic decoding (Lu et al., 2021) to perform these edit operations via constrained decoding. Figure 1 illustrates our method that constrains generation using a substitution operation during beam search.

We evaluate our method using standard sentence simplification corpora that are publicly available, namely, Turk (Xu et al., 2016), ASSET (Alva-Manchego et al., 2020a), and AutoMeTS (Van et al., 2020). The experiment results using oracle constraints extracted from references confirm that the proposed method largely outperforms previous methods utilising lexically constrained decoding. Furthermore, the results when constraints are predicted by a simple model indicate the consistent efficacy of the proposed method over these baselines. The implementation of our edit-constrained decoding, as well as model outputs, are available at `https://github.com/t-zetsu/EditConstrainedDecoding`.

2 Related Work

2.1 Sentence Simplification

Sentence simplification (Shardlow, 2014; Alva-Manchego et al., 2020b) paraphrases complex sentences into simpler forms. The primary approaches can be categorised into three types: (a) translation-based, (b) edit-based, and (c) hybrid approaches. The translation-based approach, e.g., (Nisioi et al., 2017; Zhang and Lapata, 2017; Kriz et al., 2019; Surya et al., 2019; Sheang and Saggion, 2021; Martin et al., 2022; Anschütz et al., 2023), formalises sentence simplification as monolingual machine translation from complex to simple sentences. This approach learns rewriting patterns from corpora and thus allows flexible rewriting; however, the infrequent nature of simplification operations hinders a model from learning necessary operations. As a result, it tends to maintain complex tokens intact and end up in too conservative rewriting (Zhao et al., 2018; Kajiwara, 2019).

In contrast, the edit-based approach (Alva-Manchego et al., 2017; Dong et al., 2019; Kumar et al., 2020; Mallinson et al., 2020; Omelianchuk et al., 2021) rewrites a source sentence by editing, i.e., deleting, inserting, and replacing, its tokens. This approach can address the conservativeness problem owing to explicit token-by-token edits. However, it lacks the flexibility to rewrite an entire sentence to change its structure (Zetsu et al., 2022).

Finally, the hybrid approach gets the best of both worlds by applying lexical constraints to translation-based models. Agrawal et al. (2021) biases a non-autoregressive simplification model by setting an initial state of decoding, considering the lexical complexity of a source sentence. Kajiwara (2019) and Dehghan et al. (2022) apply a negative lexical constraint using lexically constrained decoding to avoid outputting complex tokens while Zetsu et al. (2022) employ both positive and negative lexical constraints. In line with these previous studies, we further enhance the hybrid approach by defining constraints based on edit operations.

2.2 Enhanced Constrained Decoding

Earlier studies on lexically constrained decoding have focused on computational efficiency (Post and Vilar, 2018; Hu et al., 2019; Miao et al., 2019; Sha, 2020). Since its prevalence in text generation tasks, recent studies have expanded the types of constraints considered. Bastan et al. (2023) enhanced NeuroLogic decoding to consider syntactic constraints, namely, dependency types up to two entities. Similarly, Geng et al. (2023a) proposed constrained decoding by grammar and McCarthy et al. (2023) proposed finite-state constraints. Transformation of syntactic structure is another primary factor in simplification (Niklaus et al., 2019). In our method, structure transformation is implicitly handled by the base translation-based model. Our constraints are lexical but expand the simple concepts of positive and negative constraints to replicate edit operations conducted in sentence simplification.

3 Edit-Constrained Decoding

We expand NeuroLogic Decoding (Lu et al., 2021) to realise edit-based constraints, for its efficacy and computational efficiency when applied to sentence simplification (Zetsu et al., 2022).

3.1 Preliminary: NeuroLogic Decoding

We briefly review NeuroLogic Decoding. Conceptually, it seeks for a hypothesis with the highest generation likelihood and constraint satisfaction:

$$\arg\max_{Y \in \mathcal{Y}} P_\theta(Y|X) \text{ s.t. } \sum_{i=1}^{L} C_i = L, \tag{1}$$

where $P_\theta(X|Y)$ is the likelihood to generate a sequence $Y = \{y_0, y_1, \cdots, y_N\}$ given an input sequence $X = \{x_0, x_1, \cdots, x_M\}$, $\mathcal{Y}$ is the possible generation space, and C_i is i-th positive or negative constraint (in total L constraints). Specifically, NeuroLogic Decoding achieves this by expanding the beam search to conduct **pruning**, **grouping**, and **selecting** processes at every timestep.

NeuroLogic Decoding first **prunes** candidates by discarding hypotheses confirmed as non-satisfying constraints and then filtering out less likely hypotheses with fewer constraint satisfaction. Namely, it drops any candidates not in the top-α in terms of likelihood and not in the top-β in terms of number of satisfied conditions. These $\alpha \in \mathbb{N}^+$ and $\beta \in \mathbb{N}^+$ are hyper-parameters. Next, it **groups** the remaining candidates based on the states of constraint satisfaction, and then **selects** a hypothesis from the candidates in each group that preserves high generation likelihood and the larger number of constraint satisfaction.

These grouping and selecting steps aim to diversify candidates in a beam and broaden the search space to avoid biasing toward hypotheses being highly likely yet satisfying only easy constraints at early timesteps. However, as we discuss in §3.2, further consideration is needed for properly handling negative constraints (i.e., deletion operations). For more details on NeuroLogic Decoding, please refer to Lu et al. (2021).

3.2 Edit-based Constraints

Inspired by the edit operations conducted in sentence simplification, we define edit constraints of *insertion*, denoted as C^I, *deletion*, denoted as C^D, and *substitution*, denoted as a tuple $C^R = (C^{R_i}, C^{R_o})$ where C^{R_i} is replaced by C^{R_o}. These edit constraints are associated with satisfaction and dissatisfaction conditions. Furthermore, instead of a binary weight, we use a numerical score s to allow flexibility in appreciating different types of constraints. Therefore, the second term in Equation (1) changes to maximise the total edit score:

$$\sum_{i=0}^{t} y_i(s), \tag{2}$$

where $y_i(s)$ denotes the score that the i-th output token receives and t is the current timestep. Note that y_i corresponds to a node n_j in the lattice of beam search; we use y_i and n_j interchangeably in the following for notation simplicity.

Insertion operation is equivalent to the positive constraint. If the node n_i is equivalent to the j-th insertion constraint C_j^I,[1] the n_i receives the score s as follows.

$$s = \begin{cases} \lambda^I & \text{if } n_i = C_j^I, \\ 0 & \text{otherwise}, \end{cases} \tag{3}$$

where $\lambda^I \in \mathbb{R}^+$ is a weight of insertion operation.

[1]We use exact matching of node and constraint tokens.

Deletion operation conceptually corresponds to the negative constraint. Previous studies (Kajiwara, 2019; Dehghan et al., 2022; Zetsu et al., 2022) naively defined negative constraints, i.e., they regard a negative constraint as satisfied if a certain word is simply avoided, without further conditions. However, we argue that the deletion operation, or negative constraint, requires more careful consideration to be properly handled. Otherwise, we may end up with a sub-optimal generation that satisfies only negative constraints C^D at early timesteps. This is because the size of the vocabulary is typically much larger than that of deletion constraints; their satisfaction is far easier than the positive counterpart. As a result, hypotheses that have high likelihoods and exclude negative constraints at early timesteps remain in a beam, which struggles to satisfy harder positive constraints later on. For example, in Figure 2, the second beam candidate 'are old .' can be the best hypothesis because it satisfies all positive ('old') and negative (exclusion of 'remain' and 'aged') constraints and has high likelihood due to its shorter length.[2]

To address this problem, we design a stricter condition for the satisfaction of deletion constraint. Intuitively, the deletion operation is regarded as satisfied by selecting sibling nodes other than the one equivalent to the constraint. Specifically, the node n_i is regarded as satisfying the deletion constraint C_j^D and avoids a penalty (negative score) if and only if its sibling node $n_k \in \pi$ is equivalent to C_j^D, where π denotes the set of sibling nodes of n_i and n_k in the lattice. These nodes receive the score s:

$$s = \begin{cases} -\lambda^D & n_k = C_j^D, \\ 0 & \forall n_i \in \{\pi \setminus n_k = C_j^D\}, \end{cases} \tag{4}$$

where $\lambda^D \in \mathbb{R}^+$ is a weight of deletion operation. Note that no penalty is given when no node in siblings is equivalent to C_j^D, i.e., the score $s = 0$.

Substitution operation replaces a token C^{R_i} to C^{R_o}. You may think that this operation can be realised by a combination of the insertion and deletion operations (or positive and negative constraints); however, it would lead to sub-optimal generation due to the lack of a mechanism to ensure the simultaneous satisfaction of both constraints like in (Zetsu et al., 2022). Therefore, similar to the

[2]Length normalisation in beam search should alleviate this problem in general; however, we empirically confirmed that adjustment of length normalisation parameter has little effect on our problem.

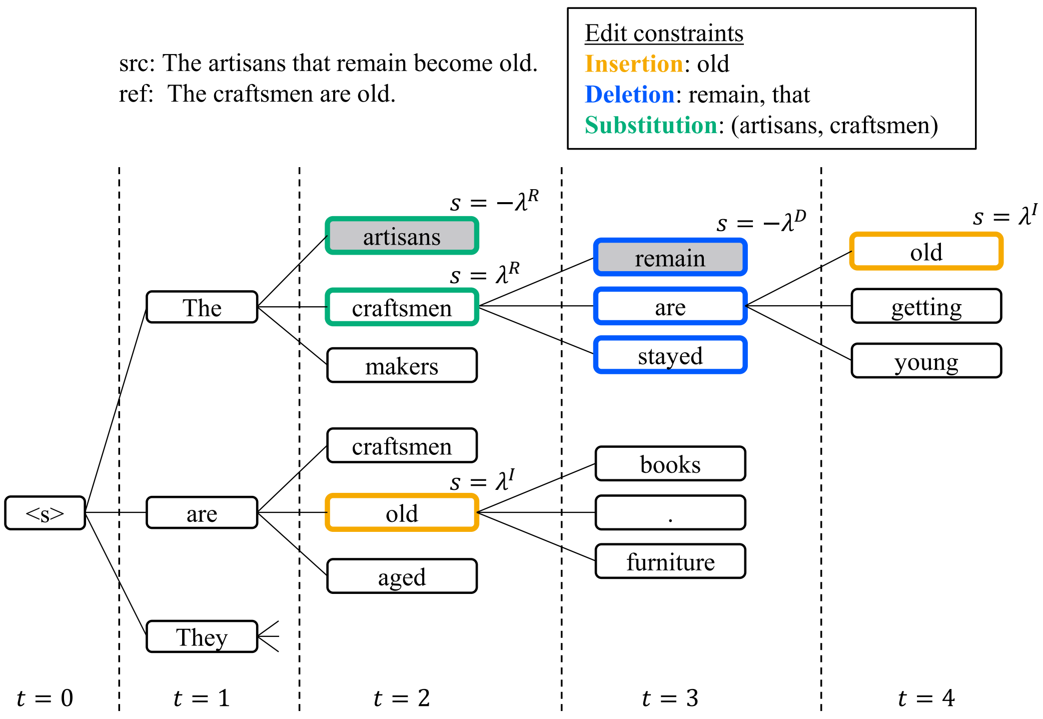

Figure 2: Edit-based constraint decoding example (beam size is 3)

deletion operation, we design a strict satisfaction condition for substitution. The constraint is regarded as satisfied when selecting a node n_i equivalent to C^{R_o} if and only if n_i has a sibling node $n_k \in \pi$ equivalent to C^{R_i}. These sibling nodes receive the score s:

$$s = \begin{cases} -\lambda^R & n_k = C^{R_i}, \\ \lambda^R & n_i = C^{R_o}, \end{cases} \tag{5}$$

where $\lambda^R \in \mathbb{R}^+$ is a weight of substitution operation. When there is no pair of nodes equivalent to C^{R_i} and C^{R_o} in π, these nodes receive no reward nor penalty, i.e., $s = 0$.

3.3 Algorithm

Our method conducts the same pruning, grouping, and selecting processes as NeuroLogic Decoding at each timestep of beam search. The only difference is the pruning conditions. Namely, our method drops candidates whose edit scores (Equation (2)) are less than $\delta \in \mathbb{R}^+$ from the top score.[3]

Figure 2 illustrates an example of edit-constrained decoding with a beam size of 3. At timestep 2, in the first beam, the substitution constraint is satisfied as the token 'artisans' is replaced by its sibling of 'craftsmen.' These nodes receive the penalty and reward scores, respectively. In contrast, in the second beam, the substitution constraint is unsatisfied because the token 'craftsmen' appears in siblings but 'artisans' does not. At timestep 3, the deletion constraint is satisfied in the first beam as 'remain' appears among siblings. Finally, at timestep 4, the insertion constraint of 'old' is satisfied.

4 Experiment Settings

We conducted two kinds of evaluations. The first evaluation (§5) aims to compare the performance of lexically constrained decoding with different constraint designs. To make the effects of methodological differences clearer, this evaluation used oracle constraints extracted from references. The second evaluation (§6) follows to observe the performance under a practical setting with predicted constraints. Preceding these evaluations, this section describes the common experiment settings.

4.1 Datasets

We measured the performance on three sentence simplification corpora, namely, **Turk** (Xu et al., 2016), **ASSET** (Alva-Manchego et al., 2020a), and **AutoMeTS** (Van et al., 2020). Turk and ASSET are commonly used public evaluation corpora in simplification. In addition, we also included AutoMeTS, aiming to investigate the effects of our method in

[3]Remind that NuroLogic Decoding filters out candidates other than top-β ones.

	Train	Validation	Test
Turk	488,332 (Wiki-Auto)	2,000	359
ASSET			
AutoMeTS	1,967	411	418

Table 1: Numbers of sentence pairs in train, validation, and test sets used in our experiments

a domain where technical terms are prevalent, and thus, lexical paraphrasing is crucial.

Turk focuses on lexical paraphrasing, i.e., replacing difficult tokens with simpler synonyms. Turk was created by crowd-sourcing, where annotators were instructed to rewrite sentences by reducing the number of difficult tokens or idioms.

ASSET expanded the Turk corpus to encompass a variety of rewriting patterns, not only lexical paraphrasing but also sentence splitting and deleting unimportant information. ASSET uses the same source sentences with Turk and crowdsourced these dynamic rewrites.

AutoMeTS is a simplification dataset in the medical domain. It extracted pairs of expert-oriented and simpler sentences from the sentence-aligned English Wikipedia corpus (Kauchak, 2013) using the medical term dictionary selected from the Unified Medical Language System (Bodenreider, 2004). This dataset contains technical terms frequently that are expected to be properly rephrased. We discarded pairs whose source and reference were identical, which reduced the number of pairs from what was reported in the original paper.

As Turk and ASSET miss a training set, we employed Wiki-Auto (Jiang et al., 2020) following the convention. All of these corpora originate from English Wikipedia. Table 1 lists the numbers of training, validation, and test sets used.

4.2 Evaluation Metrics

As the primary evaluation metric to measure the quality of sentence simplification, we used **SARI** (Xu et al., 2016). SARI evaluates the success rates of addition, keeping, and deletion of tokens by comparing model-generated simplifications with the source sentence and references. The final score is computed by averaging F1 scores of addition, keep, and deletion operations. For fine-grained analysis of the effects of our edit-based constraints, we also report the individual scores of these operations.

In addition, we also employed metrics commonly used in simplification, namely, **BLEU** (Papineni et al., 2002), Flesch-Kincaid Grade Level (**FKGL**) (Kincaid et al., 1975), and **BERTScore** (Zhang et al., 2020). SARI, BLEU, and FKGL were computed by a standard evaluation package of EASSE (Alva-Manchego et al., 2019)[4] while BERTScore used its official implementation.[5]

4.3 Baselines

We compared our method to previous studies also using lexically constrained decoding, namely, Kajiwara (2019) and Zetsu et al. (2022). While lexically constrained decoding theoretically applies to any base models using decoders, we limited ourselves to a basic pretrained sequence-to-sequence model, namely BART (Lewis et al., 2020), expecting to make the analysis simpler and easier. Exploration of the best possible performance by applying to superior base models, such as large language models (LLMs) (Kew et al., 2023) or models trained with large corpus (Martin et al., 2022) constitutes our future work. We also compare our method to Agrawal et al. (2021) as the strong previous study that takes the same hybrid approach to simplification with us (see §2). In addition, the performance of naive beam search on the fine-tuned BART-base was also examined as the bottom line.

4.4 Implementation

The baseline methods were replicated using the codes released by the authors. For methods using lexically constrained decoding, we fine-tuned BART-base[6] with an AdamW (Loshchilov and Hutter, 2019) optimiser using the corresponding training set of each corpus. The fine-tuning was conducted for 5 epochs at maximum with early stopping based on validation SARI score. For inference, we set the beam size to 20.

The proposed method was implemented utilising the released codes of NeuroLogic Decoding[7]. Its hyper-parameters, namely, λ^I, λ^D, λ^R, and δ, were searched using Optuna (Akiba et al., 2019) to maximise validation SARI score. The λ values were searched in the range of $[0, 1]$, while the δ value

[4]The default 4-gram BLEU and SARI were measured.
[5]https://github.com/Tiiiger/bert_score
[6]https://huggingface.co/facebook/bart-base
[7]https://github.com/GXimingLu/neurologic_decoding

	SARI (↑)	add	keep	del	BLEU (↑)	FKGL (↓)	BS (↑)	Len
				Turk				
Beam search	39.7	4.6	58.7	55.8	35.2	6.5	91.7	13.6
Agrawal et al. (2021)	42.0	4.0	63.9	58.0	27.9	7.4	90.4	27.1
Kajiwara (2019)	38.1	1.7	64.9	47.7	22.6	**5.3**	89.5	36.9
Zetsu et al. (2022)	50.7	10.8	74.9	66.4	53.4	7.2	94.8	16.9
Proposed method	**55.4**	**21.0**	**76.5**	**68.7**	**56.9**	7.6	**94.9**	18.4
				ASSET				
Beam search	38.4	5.7	53.0	56.4	30.7	6.5	**93.9**	13.6
Agrawal et al. (2021)	43.1	3.6	64.4	61.4	41.1	7.3	92.7	17.4
Kajiwara (2019)	37.8	2.6	62.2	48.5	21.5	**5.8**	89.5	36.2
Zetsu et al. (2022)	48.3	10.4	65.8	**68.6**	40.2	6.1	93.6	14.8
Proposed method	**50.8**	**17.0**	**66.8**	**68.6**	**43.7**	6.8	93.8	16.7
				AutoMeTS				
Beam search	35.9	0.8	57.5	49.4	43.9	11.5	91.5	27.1
Agrawal et al. (2021)	38.9	0.6	45.3	70.8	20.1	**6.4**	86.7	20.5
Kajiwara (2019)	42.9	3.2	59.2	66.3	29.3	10.7	88.9	35.8
Zetsu et al. (2022)	49.3	3.7	69.4	74.8	44.2	9.2	92.0	22.3
Proposed method	**52.3**	**8.0**	**72.9**	**75.9**	**50.0**	9.1	**92.3**	25.2

Table 2: Evaluation results with oracle constraints; the scores were measured on the single references from which the constraints were extracted ('BS' and 'Len' represent BERTScore and average output length, respectively).

was searched in $[0, 2]$[8] with 100 attempts. For α, we used the default value due to its limited impact on the performance. §A.1 describes more details of implementation and shows hyper-parameter values.

5 Evaluation with Oracle Constraints

We first measure the performance of the proposed method under a setting where highly reliable constraints are given. This setting eliminates possible performance variations depending on the quality of constraints and clarifies how different lexically constrained decoding mechanisms contribute to sentence simplification.

5.1 Oracle Constraint Extraction

We extracted highly reliable constraints from references. Specifically, we conducted word alignment between the source and reference sentences in a corpus using the state-of-the-art word aligner (Lan et al., 2021). The alignment results were converted to constraints using simple heuristics below, which are referred to as 'oracle' constraints hereafter[9].

- Source tokens of null alignment were set as deletion operations (the proposed method) or negative constraints (baselines).
- Source tokens aligned to the identical tokens were set as insertion operations (the proposed method) or positive constraints (baselines).[10]
- Source tokens aligned to ones with different surfaces were regarded as substitution operations (the proposed method) or a set of negative and positive constraints (baselines).

While Turk and ASSET provide multiple references, we used the every first reference of each source sentence for extracting the oracle constraints to keep the constraints and reference consistent. Therefore, the evaluation metrics were computed with these first references to meet the condition of oracle constraint extraction.

5.2 Results

Table 2 shows results on the test sets under the oracle setting. The proposed method largely outperforms the previous studies. Breakdown scores of

[8]This is because when λ ranges in $[0, 1]$, the maximum gap between edit scores is at most 2 for each timestep.

[9]To be precise, these constraints are imperfect due to alignment errors; we assume their effects are minimal in this study.

[10]We discarded reference tokens of null alignment from insertion constraints assuming that it is challenging in practice to predict new content to add based on a given sentence.

	Turk		ASSET		AutoMeTS	
	Comp.	Ours	Comp.	Ours	Comp.	Ours
Insertion	**92.2**	91.9	96.6	**97.1**	95.4	**96.3**
Deletion	82.0	**89.9**	89.0	**94.0**	86.3	**93.1**
Substitution	17.9	**55.3**	20.9	**53.1**	10.1	**44.3**

Table 3: Percentage of satisfied constraints; 'Comp.' represents (Zetsu et al., 2022).

	Comp.	Ours
Failed to generate named entities	6	3
Dropped information when a sentence is complex and has multiple clauses	15	4
Altered meaning or focus of the sentence	4	3

Table 4: Results of manual error analysis; 'Comp.' represents (Zetsu et al., 2022)

src	He left a detachment of 11,000 troops to garrison the newly conquered region .
reference	He left 11,000 troops there to defend the region .
Agrawal et al. (2021)	He left a of 11 000 troops to the conquered region .
Zetsu et al. (2022)	He left 11,000 troops .
Proposed method	He left 11,000 troops to defend the region .
edit-constraints	[insert] **left**, **troops** [delete] **detachment**, **newly** [substitute] **(garrison, defend)**
src	by most accounts , the instrument was nearly impossible to control .
reference	most people said that the device was very hard to control .
Agrawal et al. (2021)	the instrument was to control .
Zetsu et al. (2022)	It was almost impossible to control .
Proposed method	Most people did not know how to control it , and most of them thought it was very hard .
edit-constraints	[insert] **most**, was, **control** [substitute] (accounts, said), (instrument, device), **(nearly, very)**, **(impossible, hard)**

Table 5: Example outputs of simplification models (**bold** constraints are satisfied by the proposed method.)

SARI indicate that our method improved all operations. We conjecture that this is because the stricter satisfaction conditions allowed proper handling of edit-based constraints.

Among the methods employing lexically constrained decoding, Kajiwara (2019) uses only negative constraints. Table 2 indicates that this method generated significantly longer sentences than others. As we observed their outputs, sentences were often disfluent with repeated tokens. We conjecture that this is an adverse effect using multiple negative constraints, which may have corrupted the decoder.

In contrast, Zetsu et al. (2022) consider both positive and negative constraints, which may enable it to avoid this corruption problem. As we discussed in §3.2, their conditions of constraint satisfaction are looser than ours. Table 3 shows the percentages of constraints satisfied by Zetsu et al. (2022) and the proposed method, where our method achieved significantly higher constraint satisfaction rates. We further investigated the differences between Zetsu et al. (2022) and our method by error analysis of their simplification outputs. We sampled 100 source sentences from the ASSET corpus and manually annotated errors to simplification of each method. Table 4 shows the three major error types and their numbers of occurrences. The proposed method consistently decreased errors of all types, in particular, avoided overly dropping information when a source sentence is syntactically complex. These results indicate that our strict conditions for edit-constraint satisfaction effectively avoid sub-optimal generation.

Table 5 shows example outputs.[11] For both cases, Agrawal et al. (2021) rewrote less, which has been known as the conservativeness problem of the simplification models (see §2). In contrast, Zetsu et al. (2022) and our method addressed this problem. Indeed, Agrawal et al. (2021) generated exactly the same sentences as their sources for 10.0% cases on ASSET, while the proposed method decreased it down to 4.5%. In addition, the proposed method satisfied more constraints compared to Zetsu et al. (2022) by avoiding sub-optimal generation.

[11]We omitted outputs of Kajiwara (2019) due to their disfluent generation.

	SARI (↑)	add	keep	del	BLEU (↑)	FKGL (↓)	BS (↑)	Len
				Turk				
Beam search	39.7	4.6	58.7	55.8	35.2	6.5	91.7	13.6
Agrawal et al. (2021)	38.6	2.9	58.9	53.8	37.0	6.6	90.7	16.7
Kajiwara (2019)	33.4	2.9	**64.8**	32.5	**49.5**	8.7	**92.9**	19.7
Zetsu et al. (2022)	42.2	**8.5**	59.2	**58.9**	36.0	**6.0**	91.9	14.3
Proposed method	**42.6**	7.9	61.5	58.5	38.9	6.4	92.1	15.3
				ASSET				
Beam search	38.4	5.7	53.0	56.4	30.7	6.5	**93.9**	13.6
Agrawal et al. (2021)	36.7	2.2	53.3	54.6	31.5	6.6	91.5	16.7
Kajiwara (2019)	32.8	2.3	**60.8**	35.2	**44.0**	8.7	**93.9**	19.7
Zetsu et al. (2022)	**40.9**	**7.5**	54.8	**60.4**	31.7	**6.0**	92.7	14.3
Proposed method	**40.9**	7.4	56.2	59.1	33.9	6.4	92.9	15.3
				AutoMeTS				
Beam search	35.9	0.8	57.5	49.4	43.9	11.5	91.5	27.1
Agrawal et al. (2021)	36.0	0.4	38.9	**68.7**	16.1	**6.2**	86.0	19.4
Kajiwara (2019)	42.2	**6.5**	**63.4**	56.8	**44.7**	11.6	**92.4**	31.1
Zetsu et al. (2022)	41.6	2.0	56.8	66.0	34.3	8.6	90.7	21.3
Proposed method	**42.9**	2.4	60.2	66.2	38.9	9.0	91.2	23.7

Table 6: Evaluation results with predicted constraints; the scores were measured on the same single references with Table 2 ('BS' and 'Len' represent BERTScore and average output length, respectively).

6 Evaluation with Predicted Constraints

In this section, we evaluate the proposed method under a practical setting, where constraints are predicted by a simple neural model.

6.1 Constraint Prediction

We employed the simple constraint prediction model developed by Zetsu et al. (2022). It first predicts which tokens in a source sentence should be inserted, deleted, and substituted as a token classification problem. Specifically, we fine-tuned BERT-base (Devlin et al., 2019)[12] using the oracle constraints extracted in §5.1 as described in the original paper.

For tokens predicted as substitution, replacing tokens were determined using a lexical translation table, which was assembled from each training set using the Moses toolkit (Koehn et al., 2007) .[13] The pairs of tokens were set as substitution operations in the proposed method (or a set of negative and positive constraints in baselines). When a token had multiple substitution candidates, these pairs were handled as 'OR' constraints.

6.2 Results

Table 6 shows results on test sets where constraints were predicted.[14] To ensure side-by-side comparison with Table 2, the evaluation metrics were computed using the same single references on Turk and ASSET (i.e., every first reference). §A.2 provides the results computed on multiple references, which show the same trends with Table 6. While the gains became smaller, the proposed method consistently outperformed the baselines on SARI. The results of Agrawal et al. (2021) and Kajiwara (2019) revealed their instability; they can be comparable to or even worse than the naive beam search, depending on the corpus. In contrast, our method achieved consistent improvements over the baselines.

Obviously, the quality of simplification by these

[12]`https://huggingface.co/google-bert/bert-base-uncased`

[13]While Zetsu et al. (2022) fine-tuned a pretrained model for predicting replacing tokens, our preliminary experiment confirmed our simpler method utilising the translation table is superior and computationally faster. Note that we reused the same word alignment results computed for creating the oracle constraints for assembling the translation table. Furthermore, we postprocessed the translation table to discard target tokens with probabilities smaller than 0.002.

[14]The simplification outputs on Turk and ASSET become equivalent because their source sentences and thus corresponding predicted constraints are the same, as can be seen in the equal values in FKGL and Len.

	Turk	ASSET	AutoMeTS
Insertion: C^I	80.5	74.6	76.6
Deletion: C^D	50.7	54.9	66.1
Substitution: C^R	23.2	23.8	43.3
from: C^{R_i}	29.2	28.1	43.3
to: C^{R_o}	56.8	62.2	73.3

Table 7: Precision of predicted constraints (%)

methods depends on the quality of constraint prediction. Table 7 shows the precision of each edit constraint compared to the oracle constraints. As expected from the naive design of the prediction model, the results indicate that there is a large room for improvement in constraint prediction, particularly in deletion and substitution operations. The development of a sophisticated prediction model constitutes our future work. Nonetheless, it is remarkable that the proposed method outperforms the previous studies even with imperfect constraints.

7 Summary and Future Work

In this study, we proposed the edit-constrained decoding for sentence simplification. Our constraints directly replicate the edit operations involved in simplification and are associated with strict satisfaction conditions. These features allow us to avoid the sub-optimal generation observed in previous studies. The evaluation with oracle constraints confirmed that our method largely outperforms the previous methods using lexically constrained decoding. Furthermore, the evaluation with a simple constraint prediction model confirmed consistent improvement by our method over the previous studies even with imperfect constraints.

In future work, we will apply our edit-constrained decoding to LLMs to further enhance the sentence simplification technology. As Valentini et al. (2023) revealed, LLMs still lack the ability to control lexical complexities. The proposed method is promising as complementation on this issue. We will also improve the constraint prediction model, which should boost the quality of simplification by a large margin.

Limitations

To derive the full potential of our method for sentence simplification, we should improve the quality of constraint prediction as discussed in §6.2. Employment of LLMs is promising given their high performances in simplification (Kew et al., 2023).

We also observed that there is room for improvement in the mechanisms of lexically constrained decoding itself. While existing methods use constant weights for constraints and attempt to apply them since the beginning of generation, this often excessively affects generation and ends up in degenerate solutions. Considering the generation likelihood changes over timesteps, the constraint weights should be adjusted depending on the generation states. This direction also constitutes our future work.

Acknowledgments

This work was supported by Cross-ministerial Strategic Innovation Promotion Program (SIP) on "Integrated Health Care System" Grant Number JPJ012425 and JSPS KAKENHI Grant Number JP21H03564.

References

Sweta Agrawal, Weijia Xu, and Marine Carpuat. 2021. A non-autoregressive edit-based approach to controllable text simplification. In *Findings of the Association for Computational Linguistics: ACL*, pages 3757–3769.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, page 2623–2631.

Fernando Alva-Manchego, Joachim Bingel, Gustavo Paetzold, Carolina Scarton, and Lucia Specia. 2017. Learning how to simplify from explicit labeling of complex-simplified text pairs. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, pages 295–305.

Fernando Alva-Manchego, Louis Martin, Antoine Bordes, Carolina Scarton, Benoît Sagot, and Lucia Specia. 2020a. ASSET: A dataset for tuning and evaluation of sentence simplification models with multiple rewriting transformations. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4668–4679.

Fernando Alva-Manchego, Louis Martin, Carolina Scarton, and Lucia Specia. 2019. EASSE: Easier automatic sentence simplification evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 49–54.

Fernando Alva-Manchego, Carolina Scarton, and Lucia Specia. 2020b. Data-driven sentence simplification: Survey and benchmark. *Computational Linguistics*, 46(1):135–187.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2017. Guided open vocabulary image captioning with constrained beam search. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 936–945.

Miriam Anschütz, Joshua Oehms, Thomas Wimmer, Bartłomiej Jezierski, and Georg Groh. 2023. Language models for German text simplification: Overcoming parallel data scarcity through style-specific pre-training. In *Findings of the Association for Computational Linguistics: ACL*, pages 1147–1158.

Mohaddeseh Bastan, Mihai Surdeanu, and Niranjan Balasubramanian. 2023. NEUROSTRUCTURAL DECODING: Neural text generation with structural constraints. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 9496–9510.

Olivier Bodenreider. 2004. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32(Database-Issue):267–270.

Rajen Chatterjee, Matteo Negri, Marco Turchi, Marcello Federico, Lucia Specia, and Frédéric Blain. 2017. Guiding neural machine translation decoding with external knowledge. In *Proceedings of the Workshop on Statistical Machine Translation (WMT)*, pages 157–168.

Sehyun Choi, Tianqing Fang, Zhaowei Wang, and Yangqiu Song. 2023. KCTS: Knowledge-constrained tree search decoding with token-level hallucination detection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 14035–14053.

Mohammad Dehghan, Dhruv Kumar, and Lukasz Golab. 2022. GRS: Combining generation and revision in unsupervised sentence simplification. In *Findings of the Association for Computational Linguistics: ACL*, pages 949–960.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186.

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 Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3393–3402.

Saibo Geng, Martin Josifoski, Maxime Peyrard, and Robert West. 2023a. Grammar-constrained decoding for structured NLP tasks without finetuning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 10932–10952.

Xiang Geng, Yu Zhang, Zhejian Lai, Shuaijie She, Wei Zou, Shimin Tao, Hao Yang, Jiajun Chen, and Shujian Huang. 2023b. Improved pseudo data for machine translation quality estimation with constrained beam search. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 12434–12447.

David Heineman, Yao Dou, Mounica Maddela, and Wei Xu. 2023. Dancing between success and failure: Edit-level simplification evaluation using SALSA. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3466–3495, Singapore.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1535–1546.

J. Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In *NAACL-HLT*, pages 839–850.

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 Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 7943–7960.

Tomoyuki Kajiwara. 2019. Negative lexically constrained decoding for paraphrase generation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6047–6052.

David Kauchak. 2013. Improving text simplification language modeling using unsimplified text data. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1537–1546.

Tannon Kew, Alison Chi, Laura Vásquez-Rodríguez, Sweta Agrawal, Dennis Aumiller, Fernando Alva-Manchego, and Matthew Shardlow. 2023. BLESS: Benchmarking large language models on sentence simplification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 13291–13309.

J. Peter Kincaid, Robert P. Fishburne Jr., Richard L. Rogers, and Brad S. Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Institute for Simulation and Training.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180.

Reno Kriz, João Sedoc, Marianna Apidianaki, Carolina Zheng, Gaurav Kumar, Eleni Miltsakaki, and Chris Callison-Burch. 2019. Complexity-weighted loss and diverse reranking for sentence simplification. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 3137–3147.

Dhruv Kumar, Lili Mou, Lukasz Golab, and Olga Vechtomova. 2020. Iterative edit-based unsupervised sentence simplification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 7918–7928.

Wuwei Lan, Chao Jiang, and Wei Xu. 2021. Neural semi-Markov CRF for monolingual word alignment. In *Proceedings of the Joint Conference of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 6815–6828.

Batia Laufer. 1989. What Percentage of Text-Lexis is Essential for Comprehension? In Christer Laurén and Marianne Nordman, editors, *Special language: From humans thinking to thinking machines*, chapter 25, pages 316–323. Multilingual Maters.

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 Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 7871–7880.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. NeuroLogic decoding: (un)supervised neural text generation with predicate logic constraints. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4288–4299.

Jonathan Mallinson, Aliaksei Severyn, Eric Malmi, and Guillermo Garrido. 2020. FELIX: Flexible Text Editing Through Tagging and Insertion. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1244–1255.

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

Arya McCarthy, Hao Zhang, Shankar Kumar, Felix Stahlberg, and Ke Wu. 2023. Long-form speech translation through segmentation with finite-state decoding constraints on large language models. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 247–257.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. CGMH: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, pages 6834–6842.

Christina Niklaus, Matthias Cetto, André Freitas, and Siegfried Handschuh. 2019. Transforming complex sentences into a semantic hierarchy. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3415–3427.

Sergiu Nisioi, Sanja Štajner, Simone Paolo Ponzetto, and Liviu P. Dinu. 2017. Exploring neural text simplification models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 85–91.

Kostiantyn Omelianchuk, Vipul Raheja, and Oleksandr Skurzhanskyi. 2021. Text Simplification by Tagging. In *Proceedings of the 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 Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *NAACL-HLT)*, pages 1314–1324.

Lei Sha. 2020. Gradient-guided unsupervised lexically constrained text generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8692–8703.

Matthew Shardlow. 2014. A survey of automated text simplification. *International Journal of Advanced Computer Science and Applications (IJACSA), Special Issue on Natural Language Processing*, 4(1).

Kim Cheng Sheang and Horacio Saggion. 2021. Controllable sentence simplification with a unified text-to-text transfer transformer. In *Proceedings of the International Conference on Natural Language Generation (INLG)*, pages 341–352.

Sai Surya, Abhijit Mishra, Anirban Laha, Parag Jain, and Karthik Sankaranarayanan. 2019. Unsupervised neural text simplification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2058–2068.

Maria Valentini, Jennifer Weber, Jesus Salcido, Téa Wright, Eliana Colunga, and Katharina von der Wense. 2023. On the automatic generation and simplification of children's stories. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3588–3598.

Hoang Van, David Kauchak, and Gondy Leroy. 2020. AutoMeTS: The autocomplete for medical text simplification. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 1424–1434.

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

Tatsuya Zetsu, Tomoyuki Kajiwara, and Yuki Arase. 2022. Lexically constrained decoding with edit operation prediction for controllable text simplification. In *Proceedings of the Workshop on Text Simplification, Accessibility, and Readability (TSAR)*, pages 147–153.

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

Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 584–594.

Sanqiang Zhao, Rui Meng, Daqing He, Andi Saptono, and Bambang Parmanto. 2018. Integrating transformer and paraphrase rules for sentence simplification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3164–3173.

A Additional Experiment Details

A.1 Implementation Details

All the experiments were conducted on NVIDIA RTX A6000 (48GB memory) GPUs installed on a Ubuntu server. The main memory size was 1TB, and the central processor was AMD EPYC CPU.

We used a fine-tuned BART-base as the sequence-to-sequence generation model. For experiments targeting ASSET, we reused the model fine-tuned employing Turk's validation set (for early stopping) to save computational costs. Given that the source sentences of Turk and ASSET are common, we assume this setting does not significantly deteriorate the model's performance on ASSET.

	Turk		ASSET		AutoMeTS	
	oracle	predicted	oracle	predicted	oracle	predicted
λ^I	0.11	0.01	0.05	0.01	0.03	0.05
λ^D	0.66	0.94	0.67	0.94	0.81	0.94
λ^R	0.23	0.05	0.28	0.05	0.16	0.01
δ	0.12	0.18	0.14	0.18	0.10	0.10

Table 8: Hyper-parameter settings

Table 8 shows the hyper-parameter values on the proposed method. To speed up the hyper-parameter search, we used 300 random samples from the validation sets on Optuna. When evaluating the performance on Turk and ASSET with predicted constraints, we used the common hyper-parameter values tuned on Turk to save computational costs.

The hyper-parameter values in Table 8 are considerably different between the oracle and predicted constraints. We conjecture the discrepancy is due to the combination of the generation model performance and constraint prediction quality. The plain beam search on fine-tuned BART ('Beam search' rows in Table 2 and Table 6) has a much higher deletion score than the addition score (see 'del' and 'add' columns), which indicates that the generation model is originally weaker on insertion and substitution operations. As oracle constraints are completely reliable, our method can put more emphasis on them to promote these edit operations. In contrast, predicted constraints are imperfect as shown in Table 7, our method cannot increase weights on these operations, which may deteriorate their performance.

A.2 More Experiment Results

Table 9 shows results when constraints were predicted, where the scores were measured on test sets of Turk, ASSET, and AutoMeTS using all of the multi-references. The trend is the same with Table 6; the proposed method consistently outperforms the baselines.

	SARI (↑)	add	keep	del	BLEU (↑)	FKGL (↓)	BS (↑)	Len
Turk								
Beam search	36.3	2.8	60.6	45.6	71.3	6.5	93.9	13.6
Agrawal et al. (2021)	37.3	1.7	62.8	47.4	68.2	6.6	92.7	16.7
Kajiwara (2019)	36.6	2.6	**72.9**	34.3	**89.9**	8.7	**95.8**	19.7
Zetsu et al. (2022)	39.2	4.9	61.4	**51.2**	69.1	**6.0**	93.8	14.3
Proposed method	**40.0**	**5.2**	63.8	50.9	72.0	6.4	94.1	15.3
ASSET								
Beam search	38.9	3.2	54.7	58.6	84.3	6.5	96.0	13.6
Agrawal et al. (2021)	38.5	1.6	55.7	58.1	71.7	6.6	94.4	16.7
Kajiwara (2019)	33.4	2.2	**60.1**	38.0	**87.8**	8.7	**97.5**	19.7
Zetsu et al. (2022)	42.1	5.3	57.2	**63.9**	80.6	**6.0**	96.1	14.3
Proposed method	**42.2**	**5.6**	58.4	62.5	82.4	6.4	96.3	15.3

Table 9: Evaluation results with predicted constraints; the scores were measured using all of the multi-references ('BS' and 'Len' represent BERTScore and average output length, respectively).