Generating Multiple-Length Summaries via Reinforcement Learning for Unsupervised Sentence Summarization

Dongmin Hyun[♠*] **Xiting Wang**[♡] **Chanyoung Park**[♣]
Xing Xie[♡] **Hwanjo Yu**[♠†]

[♠]Pohang University of Science and Technology [♡]Microsoft Research Asia
[♣]Korea Advanced Institute of Science and Technology

{dm.hyun, hwanjoyu}@postech.ac.kr cy.park@kaist.ac.kr
{xitwan, xing.xie}@microsoft.com

Abstract

Sentence summarization shortens given texts while maintaining core contents of the texts. Unsupervised approaches have been studied to summarize texts without human-written summaries. However, recent unsupervised models are extractive, which remove words from texts and thus they are less flexible than abstractive summarization. In this work, we devise an abstractive model based on reinforcement learning without ground-truth summaries. We formulate the unsupervised summarization based on the Markov decision process with rewards representing the summary quality. To further enhance the summary quality, we develop a multi-summary learning mechanism that generates multiple summaries with varying lengths for a given text, while making the summaries mutually enhance each other. Experimental results show that the proposed model substantially outperforms both abstractive and extractive models, yet frequently generating new words not contained in input texts.

1 Introduction

The goal of sentence summarization is to enhance the readability of texts by reducing their lengths through word dropping, replacement, or paraphrasing. The applications of the task include subtitle generation (Luotolahti and Ginter, 2015) and email summarization (Zajic et al., 2008). An issue is that it is costly to have human editors write summaries for each text. Hence, it is critical to develop an unsupervised model that does not require any human-written summaries.

Early models focus on abstractive summarization that *generates* words from a vocabulary set rather than extractive summarization, which merely *selects* words from texts. Specifically, abstractive models have adopted autoencoder networks to summarize texts in an unsupervised manner (Wang and Lee, 2018; Févry and Phang, 2018; Baziotis et al., 2019). In contrast, extractive models summarize texts by finding word combinations from texts, aiming at maximizing predefined scores (e.g., fluency of summaries) (West et al., 2019). Despite their limited functionality, i.e., word selection, recent extractive models outperformed the abstractive models (Schumann et al., 2020; Liu et al., 2022).

Despite the success of the extractive models, we argue that they have an inherent downside. The extractive models only select words from texts, and thus they cannot generate new words that can be effective for sentence summarization. For example, extractive models are unable to generate acronyms (e.g., PM) for words (e.g., Prime Minister) if the acronyms do not appear in texts. In contrast, abstractive models can resolve the limitation of extractive models. However, the summary quality of existing abstractive models is sometimes worse than a simple baseline, which simply truncates input texts from the beginning (Schumann et al., 2020). This implies that existing abstractive models fall short of reducing text lengths while maintaining the summary quality. The aforesaid limitations of existing models motivate us to devise an abstractive model that produces high-quality summaries with generating new words not contained in input texts.

This work employs reinforcement learning (RL) for unsupervised abstractive summarization. RL enables a model to learn to summarize using rewards even though they are non-differentiable. Our model generates *high-quality* summaries with considering 1) the semantic similarity between the generated summary and its corresponding input text, and 2) fluency of the generated summaries. Notably, the semantic similarity is more robust to preserve core contents of input texts than the word-level reconstruction objective (Pagliardini et al., 2018), which is adopted by existing abstractive models.

Moreover, we argue that the difficulty of summarization depends on the summary lengths (e.g.,

[*]Work done during internship at Microsoft Research Asia.
[†]Corresponding author

the shorter the summary, the more difficult it is to summarize). In this respect, we develop a *multi-summary learning* mechanism that generates multiple summaries with varying lengths for a given text, while making the summaries mutually enhance each other. The main idea is to use a high-quality summary of a certain length, which is *easy* to generate, to enhance the quality of a low-quality summary of another length, which is *difficult* to generate, rather than independently generating summaries in each length. Specifically, we design the mechanism to make low-quality summaries semantically similar to high-quality ones.

We also devise a pretraining task to obtain well-initialized model parameters for the RL training. We first augment input texts by applying word-level perturbations and inserting length prompts, which indicate the lengths of the original texts. Then, we train the model to reconstruct the original text from the augmented one, which makes the model learn to summarize and control the output length. By pretraining the model in this manner, our goal is to equip the model with essential abilities for summarization, which results in an improved summary quality after the RL training with the pretrained model. We dub our model Multi-Summary based Reinforcement learning with Pretraining (MSRP).

Experiments show that MSRP outperforms the abstractive and extractive baseline models in both automatic and human evaluations. We also analyze summaries generated by MSRP to illuminate its benefits compared to the recent extractive models.

2 Related Work

2.1 Unsupervised Sentence Summarization

Supervised models depend on human-written summaries, which involve costly and time-consuming data creation (Rush et al., 2015; He et al., 2020; Song et al., 2021). In contrast, unsupervised models learn to summarize texts without any human-written summaries. Abstractive models mainly adopt autoencoders to build a summarization model. Févry and Phang (2018) adopt a denoising autoencoder to summarize texts by treating texts as noised data and summaries as clean data. Wang and Lee (2018); Baziotis et al. (2019) design autoencoders that generate word sequences as interim outputs of the autoencoders and use the word sequences as summaries. Zhou and Rush (2019) devise a model that selects the best next word based on a fluency score to generate summaries. In contrast, an extractive model (West et al., 2019) iteratively deletes words from texts to generate summaries while measuring the fluency of each intermediate summary. Schumann et al. (2020) select the best word combination that maximizes predefined scores based on a hill-climbing search algorithm, and it surpassed the abstractive models. However, the search requires exhaustive computation. In response, Liu et al. (2022) train an extractive model with summaries generated by Schumann et al. (2020) so that it can quickly generate summaries without the exhaustive search. Compared to extractive models, this work aims to design an abstractive model to enjoy its flexible operation, i.e., generating words not contained in texts.

2.2 Reinforced Summarization Models

RL has been used as a technique to solve summarization tasks. With referential summaries, Paulus et al. (2018); Bian et al. (2019) relieve the exposure bias of teacher forcing-based supervision. Without referential summaries, Böhm et al. (2019); Stiennon et al. (2020) devise RL-based models that maximize a reward representing the summary quality, which is annotated by human experts. Wang and Lee (2018) address the unsupervised sentence summarization where only input texts are available, which is our target scenario. They utilize RL to train an autoencoder with a word-level reconstruction loss to preserve contents of texts in summaries. In this work, we formulate a RL framework to achieve three aspects: 1) semantic similarity between input texts and summaries instead of word-level similarity, 2) controllability on summary length, and 3) model-agnostic RL framework.

2.3 Pretraining Task for Summarization

Pretraining tasks are crucial to obtain high accuracy on NLP tasks (Devlin et al., 2019; Lewis et al., 2020). Recent research invents pretraining tasks for long-document summarization (Zhang et al., 2020; Zhu et al., 2021). However, the approaches are not applicable to sentence summarization due to the absence of multiple sentences, and do not consider controlling the summary length. We thus propose an effective pretraining task to make models learn to summarize and control the summary length.

3 Method

3.1 Problem Formulation

The goal of sentence summarization is to shorten a text (i.e., a long sentence) $\mathbf{t} = [w_1, w_2, \cdots, w_{|\mathbf{t}|}]$ into a short summary $\mathbf{y} = [y_1, y_2, \cdots, y_{|\mathbf{y}|}]$ where w, y are words and $|\mathbf{y}| < |\mathbf{t}|$. It is important to note that the text-summary pairs are not available for training models. In other words, we focus on the unsupervised sentence summarization.

3.2 Reinforcement Learning Framework

Due to the absence of ground-truth summaries, we train a text generator based on the quality of generated summaries. However, the summary generation requires the word-sampling process, which is non-differentiable. We thus consider RL to address the non-differentiability (Figure 1), and describe the proposed Markov decision process as follows.

States describe the possible combinations of input texts $\mathbf{t}$ and generated summaries $\mathbf{y}_t = [y_1, y_2, \cdots, y_t]$ at time t. State at time t can be formulated as $s_t = [\mathbf{t}, \mathbf{y}_t]$. **Actions** are the candidate next words from a vocabulary set $\mathcal{V}$ at given states. A policy $\boldsymbol{\pi}_\theta$ selects an action $a_t \in \mathcal{V}$ as a next word y_{t+1} based on a given state s_t, resulting in next summary $\mathbf{y}_{t+1}$. **Transition function** determines next states based on a state s_t and action a_t, i.e., $s_{t+1} = \mathcal{T}(s_t, a_t) = [\mathbf{t}, \mathbf{y}_{t+1}]$.

Reward $\mathcal{R}(s_t, a_t)$ represents the summary quality when a target summary length l is given. We obtain the reward of the generated summaries such that:

$$\mathcal{R}(s_t, a_t) = \begin{cases} \mathcal{R}(\mathbf{y}, \mathbf{t}, l) & \text{if } a_t = [\texttt{EOS}] \vee t = M_g, \\ 0 & \text{otherwise,} \end{cases}$$

where $\mathbf{y}$ denotes the generated summary (i.e., $\mathbf{y} = \mathbf{y}_t$ for simplicity), [EOS] is the end-of-sentence token, M_g is the maximum length of generated summaries. We design pertinent aspects of summaries:

$$\mathcal{R}(\mathbf{y}, \mathbf{t}, l) = \mathcal{R}_C(\mathbf{y}, \mathbf{t}) + \mathcal{R}_F(\mathbf{y}) + \mathcal{R}_L(|\mathbf{y}|, l).$$

- **Content preservation** A requirement for high-quality summaries is to preserve the gist of the input texts. We consider the semantic similarity between summaries and the corresponding texts:

$$\mathcal{R}_C(\mathbf{y}, \mathbf{t}) = \texttt{sim}(f(\mathbf{y}), f(\mathbf{t})) \tag{1}$$

where $\mathcal{R}_C \in [0, 1]$, sim is a similarity function, and f is a function to embed texts (i.e., $\mathbf{y}$ and $\mathbf{t}$) such as BERT.[1] We use cosine similarity with normalization, i.e., $\texttt{sim}(\cdot, \cdot) = (\texttt{cos}(\cdot, \cdot) + 1)/2$. The semantic similarity enables the model to robustly capture the meaning of texts despite different words in two texts, e.g., *Who's the winner?* and *Who won the game?*.

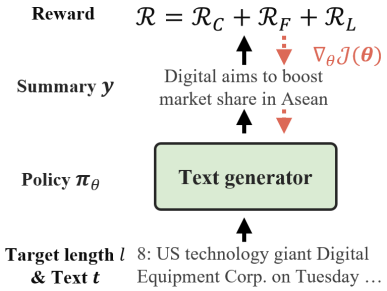

Figure 1: Reinforcement learning with a length prompt.

- **Fluency** Another requisite for summaries is fluency, representing how generated summaries are grammatically and semantically natural. We use the perplexity as fluency:

$$\texttt{PPL}(\mathbf{y}) = \exp\Big\{ -\frac{1}{|\mathbf{y}|} \sum_{t}^{|\mathbf{y}|} \log p_\psi(\mathrm{y}_t | \mathbf{y}_{t-1}) \Big\} \tag{2}$$

where PPL is the perplexity from a language model with its parameters ψ, and $\mathbf{y}_{t-1}$ is the generated summary before time t. Low PPL indicates high fluency. We define the fluency reward:

$$\mathcal{R}_F(\mathbf{y}) = \exp(-\texttt{PPL}(\mathbf{y})/\sigma_F) \tag{3}$$

where $\mathcal{R}_F \in (0, 1]$ and $\sigma_F \in \mathbb{R}_+$ is a tunable scaling factor to control the steepness of $\mathcal{R}_F$.[2]

- **Summary length** We design our model to summarize texts in a desired length. We first insert the desired length l (e.g., 8 words) at the beginning of an input text (e.g., '8:' Figure 1), and then optimize the following reward:

$$\mathcal{R}_L(|\mathbf{y}|, l) = \exp(-||\mathbf{y}| - l|/\sigma_L)$$

where $\mathcal{R}_L \in (0, 1]$ and $\sigma_L \in \mathbb{R}_+$ is a tunable scaling factor. After training, we can control the summary length by changing the desired length.

3.2.1 Policy Gradient

Policy gradient directly updates the policy parameters θ to minimize an objective function $\mathcal{J}$:

$$\mathcal{J}(\theta) = -\sum_{\mathbf{t} \in T} \mathbb{E}_{\mathbf{y} \sim \boldsymbol{\pi}_\theta(\cdot|l,\mathbf{t})} \mathcal{R}(\mathbf{y}, \mathbf{t}, l).$$

[1] The specific model is described in Section 4.1.4.

[2] The shape of the reward is provided in Appendix A.1.

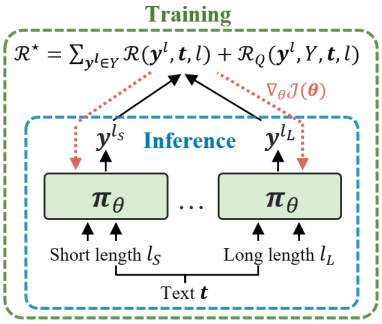

Figure 2: Multi-summary learning mechanism

where T is a set of input texts. Therefore, RL updates the policy parameters θ to maximize the expected rewards (i.e., $\mathcal{R}_L, \mathcal{R}_C$, and $\mathcal{R}_F$). We adopt a self-critical policy gradient (Rennie et al., 2017) to stabilize the training by reducing the variance. The gradient of the policy can be written as:

$$\nabla_\theta \mathcal{J}(\theta) \approx -\sum_{\mathbf{t} \in T} \left(\mathcal{R}(\mathbf{y}, \mathbf{t}, l) - \mathcal{R}(\bar{\mathbf{y}}, \mathbf{t}, l)\right) \nabla_\theta \sum_t \log \boldsymbol{\pi}_\theta(y_{t+1}|s_t)$$

where $\bar{\mathbf{y}}$ is a baseline summary, whose words are greedily selected, i.e., $y_{t+1} = \text{argmax}\, \boldsymbol{\pi}_\theta(\mathrm{y}_{t+1}|s_t)$, instead of sampling words, i.e., $y_{t+1} \sim \boldsymbol{\pi}_\theta(\mathrm{y}_{t+1}|s_t)$. The gradient has a direction to maximize the likelihood if $\mathcal{R}(\mathbf{y}, \mathbf{t}, l) > \mathcal{R}(\bar{\mathbf{y}}, \mathbf{t}, l)$.

3.3 Multi-Summary Learning Mechanism

We further improve the summary quality by making multiple summaries with varying lengths mutually enhance each other (Figure 3). The main idea is to use a high-quality summary of a certain length, which is *easy* to generate, to enhance the quality of a low-quality summary of another length, which is *difficult* to generate.We first generate multiple summaries in different lengths for each text:

$$Y = \{\mathbf{y}^l\}_{l \in L} \quad \text{where} \quad \mathbf{y}^l \sim \boldsymbol{\pi}_\theta(\cdot|l, \mathbf{t}),$$

where Y is the set of summaries generated for each length $l \in L$, L is a set of lengths, and $\mathbf{y}^l$ is a summary generated for the length l. For brevity, we denote a target summary as $\mathbf{y}$, while the other summaries as $\mathbf{y}'$ henceforth.

We then design the mechanism to make a summary semantically similar to the other summaries based on mutual relationship:

$$\mathcal{R}_Q(\mathbf{y}, Y, \mathbf{t}, l) = \lambda \sum_{\mathbf{y}' \in Y \setminus \mathbf{y}} u(\mathbf{y}, \mathbf{y}', \mathbf{t}, l) \cdot \mathcal{R}_C(\mathbf{y}, \mathbf{y}')$$

where $\mathcal{R}_Q \in [0, 1]$, $\lambda \in [0, 1]$ is a weight coefficient for this reward, $u \in [0, 1]$ is a function that measures the usefulness of a summary $\mathbf{y}'$ to a target summary $\mathbf{y}$ generated for a target length l given text t. Hence, the model makes a target summary $\mathbf{y}$ to be semantically similar to another summary $\mathbf{y}'$ based on its usefulness, i.e., refer to a summary $\mathbf{y}'$ if it is useful to a target summary $\mathbf{y}$.

We design the usefulness function u by considering the summary quality and length: 1) given the input text $\mathbf{t}$, a target summary $\mathbf{y}$ should refer to another summary $\mathbf{y}'$ with different length only if the quality of $\mathbf{y}'$ is higher than that of $\mathbf{y}$, i.e., $q(\mathbf{y}',\mathbf{t}) > q(\mathbf{y},\mathbf{t})$, where $q \in [0, 1]$ is a function that measures the summary quality. 2) A summary generated for a length l should refer to another summary with similar length to the target length l, i.e., the more similar the length, the higher the relevance. We define the usefulness function u:

$$u(\mathbf{y}, \mathbf{y}', \mathbf{t}, l) = [q(\mathbf{y}', \mathbf{t}) - q(\mathbf{y}, \mathbf{t})]_+^\alpha \cdot \mathcal{R}_L(|\mathbf{y}'|, l)$$

where $\alpha \in \mathbb{R}$ is a scaling factor, and $[\cdot]_+$ represents $\max(\cdot, 0)$. The first term produces a positive score if $q(\mathbf{y}',\mathbf{t}) > q(\mathbf{y},\mathbf{t})$. Similarly, second term produces a high score if the length of another summary $\mathbf{y}'$ is close to a target length l of a given summary $\mathbf{y}$. We consider the summary quality based on the content preservation and fluency:

$$q(\mathbf{y}, \mathbf{t}) = \mathcal{R}_C(\mathbf{y}, \mathbf{t}) \cdot \mathcal{R}_F(\mathbf{y}).$$

Finally, the total reward $\mathcal{R}^\star$ can be written with multiple summaries and the quality reward $\mathcal{R}_Q$:

$$\mathcal{R}^\star(Y, \mathbf{t}) = \sum_{\mathbf{y}^l \in Y} \mathcal{R}(\mathbf{y}^l, \mathbf{t}, l) + \mathcal{R}_Q(\mathbf{y}^l, Y, \mathbf{t}, l).$$

This mechanism makes summaries mutually enhance each other during training time, but generates summaries independently in inference time. Thus, the complexity of the inference does not increase.

3.4 Pretraining Task

We also devise *prompt-based text reconstruction* task (Table 1), and its main goal is to make our model learn to control the output length. We first apply perturbations to texts: shuffling, dropping, and adding words. We then insert the length of the original text at the beginning of the perturbed text, called *prompt*, e.g., '20:' in Table 1. Thus, by inserting the prompt to perturbed texts, the model can be explicitly informed about the target length for the original text. We train our model to reconstruct the original text from the perturbed text, which

Original text three researchers on monday won the nobel medicine prize for discovering how nitric oxide acts as a signal molecule
1. Shuffle **nitric three** researchers on monday won the nobel prize for discovering how signal **medicine** oxide acts as a molecule
2. Drop ~~**nitric**~~ three researchers on ~~**monday**~~ won the nobel ~~**prize**~~ for discovering ~~**how**~~ signal medicine oxide acts as a molecule
3. Add & Prompt Prompt **20:** three **crashing flight** researchers **town** on **103** won the **down** nobel medicine **on** prize for **tiny** how **this** nitric oxide as a signal molecule

Table 1: Example of text perturbation with a length prompt. Changes in each step are marked in red.

makes the model learn to control the output length and reorder, add, and remove words. After pretraining, we perform the RL training with the pretrained model. We provide the details in Appendix A.2.

4 Experiments

4.1 Experimental Settings

4.1.1 Datasets

We evaluate MSRP on benchmark datasets for sentence summarization. The Gigaword dataset contains a news headline per news article. The number of training and evaluation data are 3,803,957 and 1,951, respectively. We only use news articles to train MSRP so that our model does not draw on any article-headline pairs. We select 500 validation data only for tuning the hyperparameters as done in prior work (Schumann et al., 2020; Liu et al., 2022). We also use DUC2004 dataset, designed only for evaluation, consisting of four headlines per news article, and it contains 500 news articles.

4.1.2 Metrics and Evaluation Protocol

We use ROUGE, a word-overlapping ratio between generated and human-written summaries: ROUGE-*n* for *n*-gram matching and ROUGE-L for longest common subsequence matching. We use ROUGE F-1 (RF) on the Gigaword dataset, but use ROUGE recall (RR) on the DUC2004 dataset by following its evaluation protocol. In addition, we measure the *fidelity* (i.e., content preservation) of generated summaries to input texts using SentenceBERT[3] (Reimers and Gurevych, 2019) and the *fluency* of generated summaries with a language model, i.e., GPT-2 (Radford et al., 2019), based on Equation 3.

[3]We use cosine similarity between input texts and generated summaries, which are embedded by SentenceBERT.

Besides, since ROUGE gets higher as the summary gets longer, we group models based on the average length of the generated summaries for fair comparisons by following Schumann et al. (2020); Liu et al. (2022). We consider both settings of summarizing with a condition of a length (i.e., 8, 10, 13 words) and compression ratio (i.e., 50% of the length of input texts) as done in the prior work. We also note that the evaluation protocol of DUC2004 truncates summaries that exceed 75 characters for fair comparisons in terms of the summary length.

4.1.3 Models Compared

Abstractive models Zajic et al. (2004) summarize texts using a syntax tree trimming. Wang and Lee (2018) train a model with an adversarial and cycle consistency loss. Févry and Phang (2018) utilize a denoising autoencoder. Zhou and Rush (2019) model fidelity and fluency of summaries via contextual matching. Baziotis et al. (2019) stack autoencoders to impose the cycle consistency loss.
Extractive models Lead baseline truncates texts from the beginning to the target lengths. West et al. (2019) iteratively delete words from a text to generate a summary based on a fluency score. Schumann et al. (2020) search for the best word combination from texts based on a hill-climbing algorithm. Liu et al. (2022) train a non-autoregressive transformer using summaries generated by Schumann et al. (2020) with corresponding input texts in a supervised manner. We also report another non-autoregressive model (Su et al., 2021) that is trained similarly to Liu et al. (2022).

4.1.4 Implementation Details

We use sent2vec (Pagliardini et al., 2018) as a word embedding-based projection function (i.e., f in Equation 1) that is trained on the text corpus (i.e., news articles) by following the prior work (Schumann et al., 2020; Liu et al., 2022). We also report the results of MSRP with SentenceBERT as a BERT-based projection function (Section 4.5). As a language model, we use pretrained GPT-2 to obtain the fluency reward (ψ in Equation 2). We fine-tuned the language model on a target corpus (i.e., news headlines) as done in prior work (Zhou and Rush, 2019; Schumann et al., 2020). As a policy π_θ, we use pretrained T5 (Raffel et al., 2020). For the multi-summary learning mechanism, we train MSRP with a set of lengths $L = \{8, 10, 13\}$ for the length-based evaluation and with a set of compression ratios $L = \{30\%, 40\%, 50\%\}$ for the

Group	Type	Model	RF-1	RF-2	RF-L	ΔR	Fidelity	Fluency	Len.
A (desired length 8)	Ext.	Lead (8 words)	21.40	7.43	20.04	18.48	0.856	0.723	7.9
	Ext.	Schumann et al. (2020)	26.01	9.64	23.94	7.76	0.836	**0.914**	7.9
	Ext.	Su et al. (2021)	26.88	9.37	24.54	6.56	0.817	0.883	7.7
	Ext.	Liu et al. (2022)	27.94	9.24	25.51	4.66	0.857	0.760	7.8
	Ext.	Liu et al. (2022)†	26.94	9.97	24.93	5.51	0.847	0.878	7.9
	Abst.	MSRP	**29.09**	**11.46**	**26.80**	0.0	**0.875**	0.899	7.9
	Abst.	MSRP w/o RL	23.54	8.36	21.93	13.52	0.855	0.766	7.8
B (desired length 10)	Ext.	Lead (10 words)	23.04	7.96	21.30	16.62	0.884	0.729	9.8
	Abst.	Wang and Lee (2018)	27.29	10.01	24.59	7.03	–	–	10.8
	Abst.	Zhou and Rush (2019)	26.51	10.04	24.45	7.92	0.850	0.900	9.3
	Ext.	Schumann et al. (2020)	27.03	10.13	24.61	7.15	0.856	**0.914**	9.8
	Ext.	Su et al. (2021)	27.86	9.88	25.51	5.64	0.832	0.889	9.4
	Ext.	Liu et al. (2022)	28.55	9.97	25.78	4.62	0.873	0.798	9.8
	Ext.	Liu et al. (2022)†	27.61	10.23	25.04	6.04	0.865	0.848	9.8
	Abst.	MSRP	**29.94**	**11.86**	**27.12**	0.0	**0.897**	0.886	9.9
	Abst.	MSRP w/o RL	25.01	8.86	22.95	12.10	0.885	0.751	9.9
C (desired length 50% of the input)	Ext.	Lead (50% words)	24.97	8.65	22.43	8.72	0.917	0.739	14.6
	Abst.	Févry and Phang (2018)	23.16	5.93	20.11	15.57	–	–	14.8
	Abst.	Baziotis et al. (2019)	25.49	8.27	22.76	8.25	0.919	0.680	14.9
	Ext.	Schumann et al. (2020)	27.05	9.75	23.89	4.08	–	–	14.9
	Ext.	Liu et al. (2022)	28.53	9.88	25.10	1.26	0.901	0.789	14.9
	Abst.	MSRP	**28.60**	**11.00**	**25.17**	0.0	**0.924**	**0.795**	14.8
	Abst.	MSRP w/o RL	26.40	9.37	23.76	5.24	0.921	0.715	14.4

Table 2: Automatic evaluation on Gigaword dataset. Δ_R: the improvement of total ROUGE of MSRP over each model, Len: averaged length of summaries, †: Liu et al. (2022) with the same pretrained model used for MSRP.

Group D (desired length 13)							
Type	Model	RR-1	RR-2	RR-L	ΔR	FD	FL
Ext.	Lead (75 char.)	22.54	6.52	19.76	12.90	0.88	0.73
Abst.	Zajic et al. (2004)	25.12	6.46	20.12	10.02	–	–
Abst.	Baziotis et al. (2019)	22.13	6.18	19.30	14.11	0.88	0.71
Ext.	West et al. (2019)	22.85	5.71	19.87	13.29	–	–
Ext.	Schumann et al. (2020)	26.13	7.98	22.88	4.73	0.86	**0.94**
Ext.	Su et al. (2021)	26.26	7.66	22.83	4.97	0.84	0.90
Ext.	Liu et al. (2022)	26.71	7.68	23.06	4.24	0.54	0.82
Ext.	Liu et al. (2022)†	26.28	8.11	22.93	4.41	0.86	0.91
Abst.	MSRP	**27.88**	**9.35**	**24.49**	0.0	**0.90**	0.89
Abst.	MSRP w/o RL	24.66	7.69	21.90	7.48	0.88	0.79

Table 3: Automatic evaluation on DUC2004 dataset. FD and FL stand for the fidelity and fluency, respectively.

compression ratio-based evaluation. During beam search, we select a summary that maximizes the rewards (i.e., $\mathcal{R}_C, \mathcal{R}_F, \mathcal{R}_L$) and does not include predefined patterns. We provide more details in Appendix A.3.

4.2 Automatic Evaluation

We compare the summary quality of the models in Table 2 and 3, and make the following observations. MSRP consistently shows the best ROUGE scores compared to both abstractive and extractive models over different groups of summary length. In terms of the *fidelity*, MSRP consistently achieves the best score compared to the baseline models, although Schumann et al. (2020); Liu et al. (2022) also consider the fidelity score ($\mathcal{R}_C$ in MSRP) during training time. MSRP achieves competitive *fluency* scores, while MSRP is generally better than the best baseline model (Liu et al., 2022).

Moreover, as Liu et al. (2022) do not use a pretrained model, we include another baseline denoted by Liu et al. (2022)† that uses the same initial model (i.e., pretrained T5) as MSRP for fair comparisons. We observe that MSRP still outperforms Liu et al. (2022)† with the pretrained model.

To investigate the effect of our RL framework, we consider MSRP that is not trained under the RL framework (denoted by MSRP w/o RL). The model is substantially inferior compared to MSRP and the baseline models, indicating that our RL framework is vital to surpassing the recent extractive models.

In a nutshell, MSRP achieves the best ROUGE, fidelity, and competitive fluency thanks to our RL framework. We also observe that the inference time of MSRP is competitively short compared to the state-of-the-art baseline models (Appendix A.4).

4.3 Human Evaluation

We perform human evaluations to compare the summary quality between MSRP and the baseline models, i.e., Schumann et al. (2020) and Liu et al. (2022), on Gigaword data with 10 words as the summary length (Table 4). We provide the summaries generated by MSRP and each baseline model along with the corresponding input texts to annotators, who are asked to choose a better summary in

Criteria	Majority			κ	Unanimity		
	Win	Tie	Lose		Win	Tie	Lose
Comparison to Schumann et al. (2020)							
Fidelity	52	31	17	0.33	26	10	3
Fluency	32	58	10	0.22	13	9	5
Comparison to Liu et al. (2022)							
Fidelity	69	4	27	0.59	53	3	16
Fluency	69	10	21	0.51	50	2	10

Table 4: Human evaluation results. κ denotes Fleiss' kappa representing inter-annotator agreements.

Dataset	Gigaword ($l = 8$)		Gigaword ($l = 10$)		DUC2004 ($l = 13$)	
Ratio	43.1%		51.4%		60.4%	
Avg. # words	1.29		1.35		1.43	
Top-3 POS	IN	33%	IN	38%	IN	46%
	NNS	17%	NNS	17%	NNS	18%
	TO	14%	NN	12%	NN	10%

Tag	Meaning	Tag	Meaning
IN	Preposition or conjunction	NNS	Noun, plural
NN	Noun, singular or mass	TO	"*to*"

Table 5: Statistics of new words in summaries generated by MSRP (top) and the meaning of POS tags (bottom). l denotes the target summary length.

terms of fidelity and fluency. We ask a global annotation corporation to have three native speakers annotate 100 summaries. We use majority voting and unanimity to consolidate the annotators' responses. We analyze the inter-annotator agreement based on Fleiss' kappa κ,[4] indicating *fair* agreement for the comparison between MSRP and Schumann et al. (2020) and *moderate* agreement for the comparison between MSRP and Liu et al. (2022).

In Table 4, MSRP substantially outperforms the baseline models in both criteria. Particularly, the annotators indicate that MSRP generates more fluent summaries than Schumann et al. (2020), despite their highest fluency score in Table 2. Such a discrepancy between automatic and human evaluation results has been also observed in recent work (Kuribayashi et al., 2021), and thus we argue that human evaluations are crucial for accurately evaluating the fluency of generated summaries. From this experiment, we conclude that MSRP is indeed superior to the baseline models based on both automatic and human evaluations.

4.4 Frequency Analysis of New Words

We demonstrate the benefit of MSRP as an abstractive model in Table 5. In this experiment, we examine the generated summaries that contain new words, i.e., words that do not appear in input texts. We observe that the ratio of summaries that contain new words is around 50%, and roughly 1.3 new words appear per summary. This result indicates that MSRP frequently performs the abstractive operation (i.e., generating new words) so that MSRP achieves higher summary quality than the extractive baselines, which merely select words from the input texts. We also report POS tags of new words, and observe that MSRP mainly generates prepositions and nouns as new words. We illustrate the generated summaries with new words in the following section 4.8.

[4] We follow Landis and Koch (1977) to interpret kappa κ.

Dataset	Gigaword ($l = 8$)		Gigaword ($l = 10$)		DUC2004 ($l = 13$)	
Metric	RF-1	RF-L	RF-1	RF-L	RR-1	RR-L
MSRP	29.09	26.80	29.94	27.12	27.88	24.49
− MSL	28.49	26.33	29.79	27.03	27.57	24.20
− PTR	28.02	25.91	29.20	26.55	28.02	24.32
$-\mathcal{R}_F$	27.28	25.23	27.99	25.34	26.30	23.22
$-\mathcal{R}_C$	26.31	24.49	27.82	25.52	25.97	22.80
$-\mathcal{R}_C + \mathcal{R}_{AE}$	26.26	24.41	27.79	25.44	26.28	22.91
SBERT as f	**29.99**	**27.56**	**30.76**	**27.93**	**28.92**	**25.25**

Table 6: Ablation study.

4.5 Ablation Study

This section provides an ablation study to inspect the effect of each component in MSRP (Table 6). We first train MSRP without the multi-summary learning mechanism (− MSL) and the prompt-based text reconstruction task (− PTR), and observe that the performance generally degrades. Thus, both components are necessary to enhance the summary quality. In the following section 4.6 and 4.7, we provide in-depth analyses on each component. We then train MSRP without the fluency ($-\mathcal{R}_F$) and content preservation ($-\mathcal{R}_C$) reward, and observe that both rewards are essential to generating high-quality summaries.

We further compare the semantic similarity and a word-level similarity adopted by prior abstractive models (Wang and Lee, 2018; Baziotis et al., 2019). By following their approaches, we build an autoencoder with an additional seq2seq model (i.e., pretrained T5). We then design a reward to minimize the reconstruction loss L_{AE} with a scaling factor, i.e., $\mathcal{R}_{AE} = \exp(-L_{AE}/\sigma_{AE}) \in (0, 1]$. MSRP with the word-level reward, i.e., $-\mathcal{R}_C + \mathcal{R}_{AE}$, substantially decreases ROUGE scores, indicating that the semantic similarity is more effective in captur-

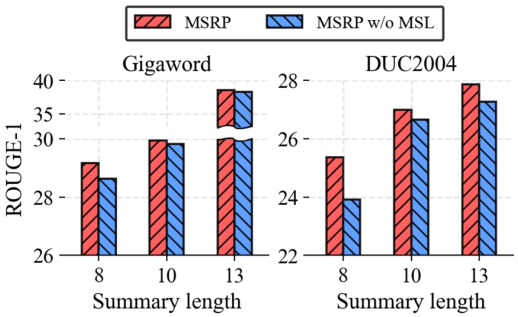

Figure 3: Effect of multi-summary learning mechanism.

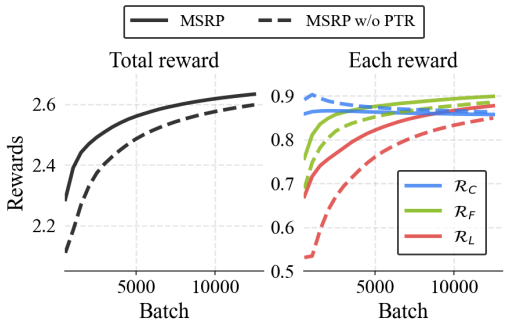

Figure 4: Learning curve of rewards.

ing the core contents than the word-level similarity. From the result, we show that the semantic similarity is a reason for the superior performance of MSRP compared to prior abstractive models.

Lastly, we replace the projection function f from sent2vec with SentenceBERT (SBERT as f) and observe further improvements in ROUGE. This result implies that an accurate projection function f can enhance the summary quality of MSRP.

4.6 Effect of Multi-Summary Learning

We investigate which summary length benefits from the MSL mechanism in Figure 3. MSRP tends to generate higher-quality summaries in the short length, i.e., 8 words, than our model not trained under the MSL mechanism (MSRP w/o MSL). This result connotes that MSRP can better learn to generate short summaries by referring to corresponding long summaries than independently generating short summaries.

4.7 Effect of Pretraining Task

In Figure 4, we inspect the effect of the PTR task. MSRP more quickly optimizes the rewards (particularly the length reward $\mathcal{R}_L$) than the model not pretrained (MSRP w/o PTR). This result implies that PTR task enables the model to learn how to control the summary length and summarize before RL training. We thus posit that PTR task improves the summary quality as RL training takes advantage of the well-initialized model parameters.

Input: israeli prime minister shimon peres said monday he was confident the ceasefire in lebanon would hold because it was in the best interests of both countries as well as syria .
Reference: peres confident ceasefire will hold
MSRP: israeli **pm** confident ceasefire in lebanon **will** hold
NAUS: israeli **minister** shimon peres confident ceasefire in Lebanon
HC: israeli **prime minister** shimon peres confident in syria
Input: president bill clinton announced reforms of the central intelligence agency aimed at restoring credibility in an espionage agency tarnished by the discovery of a russian mole in its midst .
Reference: clinton announces us intelligence reforms
MSRP: president bill clinton **announces** reforms of intelligence agency
NAUS: bill **reforms** intelligence agency aimed at restoring credibility
HC: clinton **reforms** intelligence agency aimed at restoring credibility

Table 7: Case study with generated summaries. NAUS: Liu et al. (2022), HC: Schumann et al. (2020).

4.8 Case Study

We study the generated summaries to deeply understand the behavior and benefits of MSRP compared to the best-performing baseline models (Table 7). In the top example, MSRP generates an acronym *pm* to replace *prime minister*, a new word that does not appear in the input text. Similarly, MSRP generates another new word, *will*, resulting in a similar summary to the human-written summary that cannot be generated only by the extractive operation.

In the bottom example, MSRP changes the past tense of the word *announced* to the present tense *announces*, which is more appropriate for news headlines than the past tense (Chovanec, 2003). In contrast, the baseline models use *reforms* as a verb, which can be reasonable. However, MSRP preserves both important words *announces* and *reforms* so that the summary of MSRP is more similar to the referential summary. We thus affirm that MSRP surpasses the state-of-the-art extractive models by performing the abstractive operations for summarization.

5 Conclusion

This work employs the RL for unsupervised abstractive sentence summarization with the rewards representing the summary quality and length. We invent the multi-summary learning mechanism to make the summaries with varying lengths mutually enhance each other. In addition, we design the prompt-based text reconstruction task to further improve the RL training. Experimental results show

that MSRP achieves the state-of-the-art summary quality on both automatic and human evaluation.

Limitations

RL enables summarization models to learn how to summarize with rewards representing the summary quality even though the rewards are non-differentiable. However, RL requires the word-sampling process to generate summaries in the training time. Thus, the computation time per input text is inherently longer than the sequence-to-sequence training with the cross-entropy loss, which the best baseline adopt (Liu et al., 2022).

As a remedy, we expect non-autoregressive models can enhance the training efficiency of the RL framework by generating words in parallel instead of sequentially generating words, i.e., autoregressive generation. An issue of non-autoregressive models is the inferior quality of generated texts compared to the autoregressive models (Su et al., 2021), as non-autoregressive models are limited to consider the previously-generated words. Thus, future work can study non-autoregressive models in the RL framework to enhance training efficiency while maintaining the summary quality.

It is worth noting that the total training time of MSRP is shorter than the one of the best baseline (Liu et al., 2022) despite the RL training. The best baseline depends on the summaries generated by Schumann et al. (2020), while their inference time is excessively long due to the search operation. Based on the inference time in Appendix A.4, 27 hours are required to generate summaries for 3M texts that are used by Liu et al. (2022), while the training time of MSRP with the pretraining task is about 8 hours. Thus, MSRP is more efficient in terms of the total training time than the best baseline if we consider its data-generation time.

References

Christos Baziotis, Ion Androutsopoulos, Ioannis Konstas, and Alexandros Potamianos. 2019. Seqˆ 3: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression. *arXiv preprint arXiv:1904.03651*.

Junyi Bian, Baojun Lin, Ke Zhang, Zhaohui Yan, Hong Tang, and Yonghe Zhang. 2019. Controllable length control neural encoder-decoder via reinforcement learning. *arXiv preprint arXiv:1909.09492*.

Florian Böhm, Yang Gao, Christian M. Meyer, Ori Shapira, Ido Dagan, and Iryna Gurevych. 2019. Better rewards yield better summaries: Learning to summarise without references. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3110–3120, Hong Kong, China. Association for Computational Linguistics.

Jan Chovanec. 2003. The uses of the present tense in headlines. *Theory and Practice in English Studies*, 1:83–92.

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 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Thibault Févry and Jason Phang. 2018. Unsupervised sentence compression using denoising auto-encoders. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 413–422.

Junxian He, Wojciech Kryściński, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2020. Ctrl-sum: Towards generic controllable text summarization. *arXiv preprint arXiv:2012.04281*.

Tatsuki Kuribayashi, Yohei Oseki, Takumi Ito, Ryo Yoshida, Masayuki Asahara, and Kentaro Inui. 2021. Lower perplexity is not always human-like. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5203–5217, Online. Association for Computational Linguistics.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

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

Puyuan Liu, Chenyang Huang, and Lili Mou. 2022. Learning non-autoregressive models from search for unsupervised sentence summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7916–7929, Dublin, Ireland. Association for Computational Linguistics.

Juhani Luotolahti and Filip Ginter. 2015. Sentence compression for automatic subtitling. In *Proceedings of the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015)*, pages 135–143.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In *NAACL 2018 - Conference of the North American Chapter of the Association for Computational Linguistics*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7008–7024.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Raphael Schumann, Lili Mou, Yao Lu, Olga Vechtomova, and Katja Markert. 2020. Discrete optimization for unsupervised sentence summarization with word-level extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5032–5042.

Kaiqiang Song, Bingqing Wang, Zhe Feng, and Fei Liu. 2021. A new approach to overgenerating and scoring abstractive summaries. *arXiv preprint arXiv:2104.01726*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Yixuan Su, Deng Cai, Yan Wang, David Vandyke, Simon Baker, Piji Li, and Nigel Collier. 2021. Non-autoregressive text generation with pre-trained language models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 234–243.

Yaushian Wang and Hung-Yi Lee. 2018. Learning to encode text as human-readable summaries using generative adversarial networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4187–4195.

Peter West, Ari Holtzman, Jan Buys, and Yejin Choi. 2019. Bottlesum: Unsupervised and self-supervised sentence summarization using the information bottleneck principle. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3752–3761.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, 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.

David Zajic, Bonnie Dorr, and Richard Schwartz. 2004. Bbn/umd at duc-2004: Topiary. In *Proceedings of the HLT-NAACL 2004 Document Understanding Workshop, Boston*, pages 112–119.

David M Zajic, Bonnie J Dorr, and Jimmy Lin. 2008. Single-document and multi-document summarization techniques for email threads using sentence compression. *Information Processing & Management*, 44(4):1600–1610.

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. PMLR.

Jiawei Zhou and Alexander M Rush. 2019. Simple unsupervised summarization by contextual matching. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5101–5106.

Chenguang Zhu, Ziyi Yang, Robert Gmyr, Michael Zeng, and Xuedong Huang. 2021. Leveraging lead bias for zero-shot abstractive news summarization. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1462–1471.

A Appendix

This appendix provides the details of our work, which could not be included in the submission due to the space limit. We hope this appendix will help you further understand our work.

A.1 Shape of scaling factors

In Figure 5, we provide the shape of the fluency and length reward functions ($\mathcal{R}_F, \mathcal{R}_L$) over different values of the scaling factors. By tuning the scaling factors (Section A.3.1), we observe that $\sigma_F = 1000$, $\sigma_L = 10$ produce the best result. This result indicates that smoother functions are desirable to train MSRP than the original and steep functions, i.e., $\sigma_F = 1, \sigma_L = 1$. We note that the mean of the perplexity of GPT-2 on summaries is around 3,000. Low perplexity means high fluency. In addition, we do not apply the exponential function to the reward of content preservation ($\mathcal{R}_C$) as its range is bounded into $[0, 1]$ by cosine similarity with normalization, i.e., $(\texttt{cos}(\cdot, \cdot) + 1)/2$.

A.2 Details of Pretraining Task

We provide the details of *prompt-based text reconstruction* task (Table 1). First, we shuffle a portion of words in a given text $\mathbf{t}$ to make the model learn to *reorder* the shuffled words into the original order. Second, we drop a small number of words to give the model the ability of *adding* words. Lastly, we add words from another text $\mathbf{t}'$ into the target text $\mathbf{t}$, which enables the model to learn to shorten a given text by *removing* words in it. The resulting text after the perturbations, i.e., $\tilde{\mathbf{t}}$, and its clean text $\mathbf{t}$ act as a text-summary pair. We set the ratio of shuffling, dropping, and adding words to 10%, 10%, and 100% of the number of words in input text $\mathbf{t}$ after tuning them on the validation data.

To control output lengths, we specify the target length $|\mathbf{t}|$ at the beginning of the perturbed text $\tilde{\mathbf{t}}$:

$$\tilde{\mathbf{t}} = [p(|\mathbf{t}|), \tilde{w}_1, \tilde{w}_2, \cdots, \tilde{w}_n]$$

where $p(|\mathbf{t}|)$ is the prompt in the form of '$|\mathbf{t}|$:' (e.g., '20:' in Table 1), $\tilde{w}$ is a word in the perturbed text $\tilde{\mathbf{t}}$, and n is the length of the perturbed text $\tilde{\mathbf{t}}$. Thus, by inserting the prompt to texts, the model can be explicitly informed about the target length for the original text. We train our model under this task based on the cross-entropy loss.

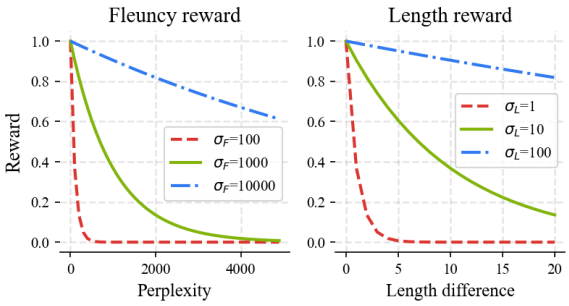

Figure 5: Reward over different scaling factors.

A.3 Implementation Details

A.3.1 Hyperparameters and Models

We tune the hyperparameters of MSRP based on RF-1 on the validation data. We tune the learning rate in $\{0.0001, \underline{0.00005}, 0.00001\}$[5] with AdamW optimizer, the batch size in $\{16, \underline{24}, 30\}$, the number of training data in $\{100\text{K}, \underline{500\text{K}}, 1\text{M}, 3.8\text{M}\}$. We set the weight for $L2$ regularization to 0.01. For rewards, we tune the scaling factors such as σ_F in $\{100, \underline{1000}, 10000\}$ and σ_L in $\{1, \underline{10}, 100\}$. We note that the balancing of each reward does not significantly improve summary quality in our experiments, except λ for balancing the quality reward in Section 3.3. We tune λ in $\{0.001, 0.005, \underline{0.01}, 0.05, 0.1\}$ and α in $\{0.0, \underline{0.3}, 0.5, 1\}$. We use early stopping if the validation performance does not improve in 10 consecutive evaluations, while we evaluate our models for each 500 batches. We observe that the best validation RF-1 is 42 for length-based evaluation and 38 for compression ratio-based evaluation. We use GeForce RTX 3090 to accelerate the training.

For transformer models, we use HuggingFace library (Wolf et al., 2020). As a language model, we use pretrained GPT-2 (Radford et al., 2019), which consists of 6 layers, to obtain the fluency reward (ψ in Equation 2).[6] As a policy $\boldsymbol{\pi}_\theta$, we use pretrained T5 (Raffel et al., 2020), which consists of 6 layers for each encoder and decoder.[7] We select the small architectures to save GPU memory. We also use SentenceBERT in the public repository.[8]

A.3.2 Beam Search

In the inference time, we perform beam search to generate summaries where the beam size is 20. Among the generated summaries, we select the best

[5] The best value for each hyperparamter is underlined.
[6] Model ID in HuggingFace library: `distilgpt2`
[7] Model ID in HuggingFace library: `t5-small`
[8] https://github.com/UKPLab/sentence-transformers

Patterns	Words
Ending with	in, at, to, on, the, 's, of, a, for, with, is, into, by, his, her, when, and, but
Including	sunday, monday, tuesday, wednesday, thursday, friday, saturday

Table 8: Undesirable patterns with the detailed words

summary that maximizes the following scores:

$$s(\mathbf{y}) = \mathcal{R}_C(\mathbf{y}, \mathbf{t}) + \mathcal{R}_F(\mathbf{y}) + \mathcal{R}_C(|\mathbf{y}|, l)$$

where s is the score for the generated summaries, the first term is the content preservation reward, the second term is the fluency reward, and the third term is the length rewards with a target length l. Before computing the score s, we remove undesirable patterns from generated summaries. We use two types of patterns: 1) a preposition, interrogative pronoun, or conjunction such as *to* or *when* at the end of summaries, i.e., ungrammatical texts, and 2) a day of week such as *monday*, i.e., less essential information. Refer to Table 8 for the patterns. In table 9, we provide the effect of beam sizes, which implies MSRP shows consistently higher ROUGE scores than the baseline models over different beam sizes, while MSRP reaches the similar summary length to the baselines with around 20 beams.

A.4 Analysis on Inference Time

Table 9 tabulates the inference time of MSRP and the baseline models. We note that the number of beams used by Liu et al. (2022) is 6. Schumann et al. (2020) require the excessively-long generation time due to the exhaustive search in the inference time. Liu et al. (2022) reduce the generation time by training a model based on the outputs of Schumann et al. (2020) and using a non-autoregressive model. Similarly, we train MSRP based on the rewards, and thus it generates summaries in short times while producing higher ROUGE scores than the baseline models. It is worth noting that the generation time of MSRP is competitive to Liu et al. (2022) when we consider that 1) the model size of MSRP is double of Liu et al. (2022) and 2) MSRP is an autoregressive model while Liu et al. (2022) use a non-autoregressive model, which is faster than autoregressive models.

A.5 Pseudocode of MSRP

Algorithm 1 describes the pseudocode of MSRP during the training time. We note that MSRP generates summaries without referring to other summaries with varying lengths in the inference time.

Model	RF-1	RF-2	RF-L	Len.	Inf. Time
Schumann et al. (2020)	27.03	10.13	24.61	9.8	33.214
Liu et al. (2022)	28.55	9.97	25.78	9.8	0.043
MSRP ($\|B\| = 1$)	29.63	11.83	26.88	11.0	0.004
MSRP ($\|B\| = 2$)	30.29	12.28	27.56	10.2	0.018
MSRP ($\|B\| = 5$)	30.08	12.08	27.35	10.1	0.031
MSRP ($\|B\| = 10$)	30.03	12.00	27.25	10.0	0.060
MSRP ($\|B\| = 20$)	29.94	11.86	27.12	9.9	0.095
MSRP ($\|B\| = 25$)	29.80	11.83	26.99	9.8	0.121

Table 9: Metrics for summary quality and inference time per text in second. The dataset is Gigaword and a desired length is 10. $|B|$: beam size.

A.6 Reproducibility

Type	MSRP trained with Sent2Vec as f
Length	anonsubms/msrp_length
Ratio	anonsubms/msrp_ratio
Type	**MSRP trained with SentenceBERT as f**
Length	anonsubms/msrp_length_sb
Ratio	anonsubms/msrp_ratio_sb

Table 10: Model ID of MSRP in HuggingFace library.

We provide our source codes and data in ***msrp.zip*** for reproducing the experimental results. We also upload our models to HuggingFace library, enabling anyone to use MSRP with a few lines of code. Refer to the uploaded model in Table 10. The type *Length* indicates MSRP trained for length-based evaluation (Group A, B, and D in Table 2 and 3) and type *Ratio* is MSRP trained for compression ratio-based evaluation (Group C in Table 2).

Algorithm 1: Pseudocode of MSRP

Input : A policy $\boldsymbol{\pi}_\theta$, a set of texts T, a set of lengths L, a learning rate η, a training type $Type$
Output : A trained policy $\boldsymbol{\pi}_\theta$

while *Convergence* **do**
  foreach $\boldsymbol{t} \in T$ **do**
    if $Type =$ *multi-summary learning* **then**
      $Y = \{\emptyset\}$
      foreach $l \in L$ **do**
        $\mathbf{y}^l \sim \boldsymbol{\pi}_\theta(\cdot|l, \mathbf{t})$
        $Y = Y \cup \{\mathbf{y}^l\}$
      end
      $\mathcal{J}(\theta) = -\mathbb{E}_{Y \sim \boldsymbol{\pi}_\theta(\cdot|L,\mathbf{t})} \mathcal{R}^\star(Y, \mathbf{t})$.
    end
    else
      $l \sim L$ ▷ Sample a length
      $\mathbf{y}^l \sim \boldsymbol{\pi}_\theta(\cdot|l, \mathbf{t})$
      $\mathcal{J}(\theta) = -\mathbb{E}_{\mathbf{y}^l \sim \boldsymbol{\pi}_\theta(\cdot|l,\mathbf{t})} \mathcal{R}(\mathbf{y}^l, \mathbf{t}, l)$.
    end
    $\theta \leftarrow$ `optimizer`$(\theta, \bigtriangledown_\theta \mathcal{J}(\theta), \eta)$
  end
end