

Learning to Summarize Without Human Feedback via Reference-Free Token-Level Reward Function

Matěj Kripner¹, Milan Straka¹, Jana Straková¹, Aleš Tamchyna², Ondřej Bojar¹

¹Charles University, Faculty of Mathematics and Physics, Prague, Czech Republic

²Memsources

Abstract

We propose an unsupervised reward function for reference-free reinforcement learning in summarization. Our reward function does not require human-supplied gold references, which are expensive to obtain and only provide one of many plausible summarization options. Instead, the function exploits the level of difficulty of reconstructing the source document when a portion of the summary has been masked, back-propagating the information densely on token level. The proposed method improves the summarization performance of BART in all reference-free automatic metrics of the SummEval package, as well as in manual metrics evaluated by human annotators.

1 Introduction

Summarization has seen dramatic improvements with the advent of deep learning and the Transformer architecture (Vaswani et al., 2017), and further substantial improvement has been achieved by aligning the models to human preferences using reinforcement learning from human feedback (RLHF; Stiennon et al. 2020).

However, obtaining human feedback is costly, and typically, only one gold reference variant is provided, although many plausible alternatives easily exist. Therefore, reinforcement learning has also been used to directly optimize automatic reference-free summarization metrics (Paulus et al., 2018; Pasunuru and Bansal, 2018; Gao et al., 2020) on a sentence level.

In this work, we design a novel, token-based, reference-free reward function for summarization. The proposed reward function directly leverages the difficulty of reconstructing the source document when a portion of the summary has been masked. To our knowledge, it is the first token-level unsupervised reward function for reinforcement learning in summarization.

We optimize the proposed reward function in a reinforcement learning reference-free setting for the information density and precision of summaries, and, as a result, our approach improves the summarization performance of BART (Lewis et al., 2020) in all evaluated reference-free automatic metrics of the SummEval package, as well as in manual metrics evaluated by human annotators.

2 Related Work

The idea of improving summarization capabilities using reinforcement learning is well established in the literature. Most notably, previous work has utilized the framework of Reinforcement Learning from Human Feedback (RLHF; Akrou et al., 2011; Christiano et al., 2017) to bypass proxy objectives and directly optimize for human feedback about the relative quality of summaries (Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022; OpenAI, 2022).

However, such human feedback is difficult to obtain and hard to reproduce. As a step towards a more automated way of guiding the learning process, Paulus et al. (2018) use ROUGE-L (Lin, 2004) as the reward function for REINFORCE (Williams, 1987, 1992), improving performance of their attention-based model on the CNN/Daily Mail dataset. Pasunuru and Bansal (2018) build on this, concurrently optimizing two new reward functions – *ROUGESal* and *Entail*.

The proposed approaches nevertheless still require a labeled dataset with reference summaries.

Alternatively, Gao et al. (2020) propose an unsupervised, reference-free metric called SUPERT, building pseudo-reference summaries by selecting salient sentences from the source document, and optimizing this metric improves summarization performance as measured by ROUGE.

More similarly to our approach, another line of work explores metrics based on language under-

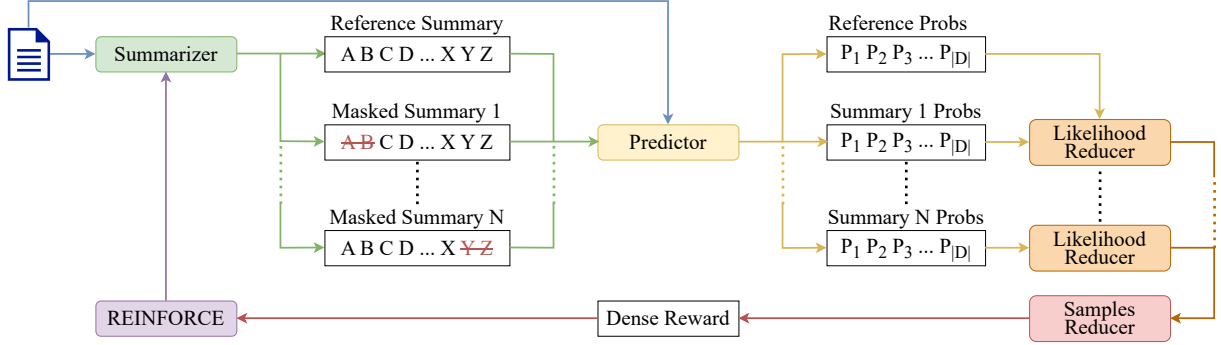


Figure 1: Overview of our summarization process with the proposed unsupervised dense reward function.

standing of the source document given a summary. This is based on the proposition that a quality summary should serve as a document’s substitute in satisfying the reader’s information needs. Arumae and Liu (2019) mask salient words and named entities in reference summaries to obtain Cloze-style questions about the source document and measure the helpfulness of an extractive summary in answering them. In this context, question answering performance serves as a proxy metric for summarization quality. Vasilyev et al. (2020) mask each 6-th word in a document sentence and measure the accuracy of BERT (Devlin et al., 2019) in unmasking them when the summary is provided as compared to when it is not. The difference of these accuracies constitutes their BLANC-help metric.

Our proposed reward function is similar to the works mentioned above in being unsupervised and therefore without the need for gold references, but differs in being a purposefully designed reward function as opposed to an optimized metric, and most notably, by operating directly on the token level.

3 Methodology

The idea behind our reward function is to approximate the importance of a summary token T by how much harder it is to predict the source document when we mask out T (possibly together with other tokens). We hypothesize that masking out an important informative token T makes it on average harder to predict the source document. On the other hand, masking out a misleading token makes the prediction easier. Finally, masking out an irrelevant “filler” token does not have a major influence on the prediction.

To assess the ease of document prediction given its summary, we predict the likelihood of the document under a sequence-to-sequence model trained

to predict the source document based on its summary. One can obtain such a model, which we call the *predictor* from now on, by fine-tuning BART-base or BART-large (Lewis et al., 2020) on a summarization dataset, in which the labels and inputs are swapped.¹

More details about the predictor fine-tuning are given in Appendix C, and the details of the reward function are the subject of the rest of this section.

3.1 Reward Calculation Overview

For a generated summary $S = T_1 \dots T_n$, we create a set of binary masks $\{M_1, \dots, M_m\}$, each of length n . Let $I(M_j)$ be the indices where mask M_j is set (i.e., where masking occurs). The result of the application of a mask $M_j(S)$ is an altered summary S' which is equal to S except for the positions $k \in I(M_j)$ where S'_k is set to a masking token K .

The predictor is run individually with each of $M_1(S), \dots, M_m(S)$ as input to predict the source document.² This gives us representations of how much the predictor was confused by each of the individual masks. We call the individual representations the *confusion coefficients*: $c_1, \dots, c_m$. See Sec. 3.2 for more details on confusion coefficients computation.

The reward for a token T_i is then dependent on those c_j where $i \in I(M_j)$, i.e., on those confusion coefficients corresponding to masks where T_i is masked:

- If T_i is not masked by any mask, we set its reward to 0.
- Suppose that T_i is masked s times ($s > 0$), yielding the corresponding confusion coeffi-

¹BART-base has 140M parameters, and BART-large has 400M parameters.

²The runs are batched in our implementation to speed up computation.

cients $c_{\omega_1}, \dots, c_{\omega_s}$. To obtain a single scalar reward R_i , we employ a signed generalized mean $\mathcal{R}_p$ described in Sec. 3.3:

$$R_i = \mathcal{R}_{p_{\text{samples}}}(\{c_{\omega_k}\}_{k=1}^s) \quad (1)$$

We dub the Equation 1 the *samples reducer*.

3.2 Confusion Coefficients Computation

Confusion coefficients quantify the amount of “confusion” caused to the predictor by obscuring some summary tokens. A scalar confusion coefficient is calculated for every mask.

First, the predictor’s likelihoods on the source document tokens are calculated before and after some summary tokens are masked using a mask M .³ To reduce these to a scalar, we again employ the signed generalized mean $\mathcal{R}_p$ as follows:

$$c_j = \mathcal{R}_p \left(\left\{ -w_t \log \frac{\mathcal{P}(M(S))_{t,D_t}}{\mathcal{P}(S)_{t,D_t}} \right\}_{t=1}^{|D|} \right) \quad (2)$$

Here, D is the source document, w_t are arbitrary weights, $\mathcal{P}(S)_t$ is the predictor’s output distribution on the t -th token with $\mathcal{P}(S)_{t,D_t}$ being the probability of the correct token at the position t , and the p in $\mathcal{R}_p$ is a hyperparameter later referred to as $p_{\text{likelihood}}$. We dub the Equation 2 the *likelihood reducer*.

3.3 Signed Generalized Mean

We now define the reducer $\mathcal{R}_p$ used in Sec. 3.1. To this end, we extend the formula for generalized mean to allow for negative values.

First, we define a symmetric power function ψ :

$$\psi_e(b) = \text{sign}(b) \cdot |b|^e \quad (3)$$

Then:

$$\mathcal{R}_p(\{x_i\}) = \psi_{1/p} \left(\frac{1}{|\{x_i\}|} \sum_i \psi_p(x_i) \right) \quad (4)$$

This is a smooth interpolation between the mean and the maximum, as $\mathcal{R}_1(\{x_i\}) = \text{mean}(\{x_i\})$ and $\lim_{p \rightarrow \infty} \mathcal{R}_p(\{x_i\})$ is equal to the x_k with the maximum absolute value among $\{x_i\}$. We limit p to positive values only since any x_i can be negative.

³The *before* calculation is done only once for each summary, since it does not depend on the mask.

	BOS	Machi	##nes	take	me	by	sur	##pri	##se	.	EOS
binomial											
individual tokens											
random bursts											
all bursts											
words											

Figure 2: An example of a mask for each mask-generation strategy. The tokenization is only illustrative, unrelated to the BART tokenizer. In all examples, `burst_length = 3`.

3.4 Mask Generation Strategies

We experiment with the following strategies for mask generation:

- **binomial** – Each token is masked with probability `mask_prob` independently randomly. A total of `mask_count` masks are generated.
- **individual_tokens** – In each mask, exactly one token is masked. The total number of masks is equal to the summary length.
- **random_bursts** – Generate a pre-defined number of burst positions at random. For each such position i , mask out the summary sub-string $S[T_i : T_{i+\text{burst_length}}]$.
- **all_bursts** – Similar to `random_bursts`, but each possible burst position i is used. Furthermore, shorter masks are created at both ends of the summary so that each token is masked exactly `burst_length`-times. This means that for reasonably long summaries, the mask count is equal to $|S| + 2(\text{burst_length} - 1)$.
- **words** – Define *words* by splitting the de-tokenized summary along spaces (with no additional preprocessing). Each mask is then constructed by masking a burst of `burst_length` consecutive words. Similarly to the `all_bursts` strategy, shorter masks are created at both ends of the summary so that each word is masked exactly `burst_length`-times.

Importantly, the beginning-of-sequence (BOS) and end-of-sequence (EOS) tokens are never masked since their absence causes BART to have radically higher cross-entropy and obscures any signal.

As for the masking token K , we experiment with the `<unk>` and `<mask>` special token of the BART tokenizer, as well as a random strategy which selects from all tokens at random for each occurrence. We also experimented with the `<pad>` token but dis-

carded it early in the process because of its inferior performance.

3.5 IDF Weights

We weight the relative log-likelihoods in Equation 2 based on the IDF of the word that the corresponding token is part of. This is similar to the weighting done by BERTScore (Zhang* et al., 2020). Specifically, assume that the t -th document token is part of a word with IDF I . We first set $w'_t = I^\omega$, where ω is a hyperparameter. Then, we normalize w'_t across all document tokens so that their average is 1, obtaining weights w_t .

4 Quantitative Analysis of Reward Functions for Summarization

We design two automatic metrics of the reward function’s performance.

Distractor-Document Test: Consider a source document D and its summary S . The reward function gives per-token rewards $p_1, \dots, p_n$ on S (positive samples). We then swap D for another *distractor document* D' and again compute per-token rewards $n_1, \dots, n_n$ on S (negative samples).⁴

With this, we obtain a set of positive measurements P and a set of negative measurements N on the same population, i.e., the same summary tokens. To evaluate to what extent P is stochastically greater than N we employ the one-sided Wilcoxon signed-rank statistic T . If one reward function yields substantially larger statistic than another one, we use that as a signal on their relative quality.⁵

This metric evaluates a reward function’s ability to assign greater rewards to summaries that contain some relevant information about the document, allowing for tuning of the predictor architecture and pre-training, the masking token, and $p_{\text{likelihood}}$.

However, this metric does not take into account the distribution of rewards among individual summary tokens, which renders it unusable for tuning `masking_strategy`, `burst_length`, and p_{samples} .

For example, increasing `burst_length` and p_{samples} will typically increase P more than N , thus increasing the metric even though this makes the reward function lose its ability to correctly allocate reward inside a summary. In the limit, a

uniform function assigning the same reward to all summary tokens might seem to perform well under this metric while being useless in practice.

Interlaced-Summary Test: To address the shortcomings of the distractor-document test, the interlaced-summary test places more importance on how the reward function distributes reward among individual summary tokens. For a source document D , its summary S , and another summary S' chosen at random, the interlaced summary $S_{\text{interlaced}}$ is created by taking individual words from either S or S' independently randomly.⁶ This again yields positive samples $\{p_i\}$ and negative samples $\{n_i\}$ based on whether the corresponding token came from S or S' . These samples are not paired. We therefore compute the one-sided Wilcoxon rank-sum U -statistic (also known as Mann–Whitney U -statistic).

Notably, since the artificial summary T is created by concatenating *words* from S and S' , we expect it to favor the words masking strategy.

5 Reinforcement Learning

We optimize our reward in the setting of reinforcement learning. Formally, we cast the summary generation as a Markov Decision Process, treating each output token as a separate action and assigning it a scalar reward. Note that while the reward function is *dense* in the sense that it rewards tokens individually, it can only be evaluated after the whole output is generated (i.e., after the episode end). In our implementation, we use the REINFORCE algorithm.

To obtain a baseline as well as a starting checkpoint for the reinforcement learning, we first fine-tune BART on a summarization dataset in a supervised fashion, marking the resulting checkpoints SFT. We start with the fine-tuning hyperparameters listed in the official Fairseq tutorial⁷ and tune them on the XSum dataset (Narayan et al., 2018). We employ 16-bit mixed precision training and the Adam optimizer with 8-bit quantized state. In the case of CNN/Daily Mail dataset (Nallapati et al., 2016), we extend the trained absolute position embedding matrix of BART from length 1024 to length 2048, initializing it with two copies of the original matrix.

⁴Specifically, D' is taken from the following sample in the dataset. Choosing D' at random from the whole dataset incurs unacceptable overhead.

⁵We also experimented with Student’s t -statistic and obtained the same ordering in approximately 90% of cases.

⁶Importantly, all the random choices have a fixed seed and are therefore identical across evaluation runs on the same dataset.

⁷<https://github.com/facebookresearch/fairseq/blob/main/examples/bart/README.summarization.md>

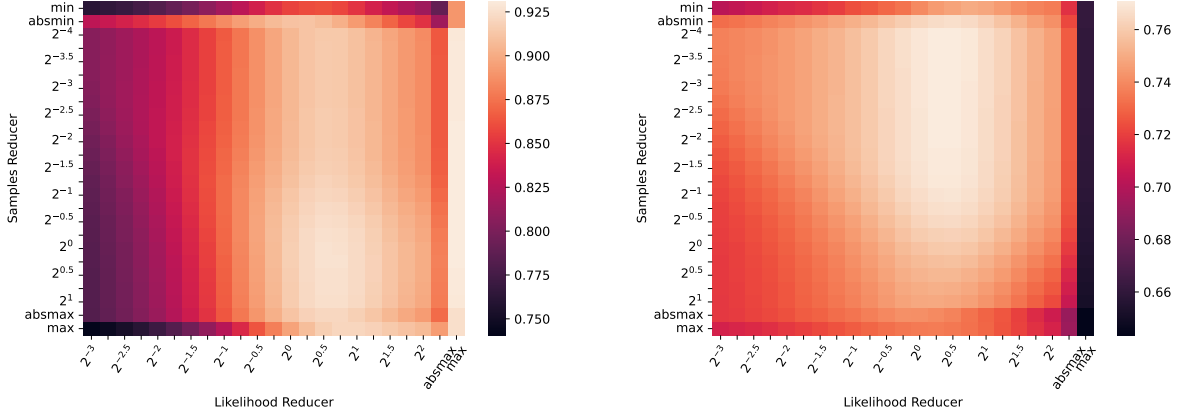


Figure 3: Reward function performance for various loss reducers and samples reducers. Evaluated with `mask_token = random`, `mask_strategy = words`, `burst_length = 2`, and IDF weighting turned off. **Left:** Distractor-Document Test, **Right:** Interlaced Summary Test.

Although the reinforcement learning stage does not necessitate reference summaries and can thus be applied with any unlabeled documents, in our experiments, we opt for the in-domain setting where we use the first half of a summarization dataset for supervised fine-tuning and the second half (with labels removed) for the reinforcement learning. The first half of the dataset is also used for the predictor fine-tuning.

KL-Divergence Anchoring: During reinforcement learning, we use the standard approach of incentivizing the model to not deviate too far from the SFT checkpoint, as measured by the KL-divergence between their respective output distributions (Jaques et al., 2019; Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022; Bai et al., 2022; Glaese et al., 2022). If we denote our reward function as r , this means optimizing the following:

$$\max_{\pi_{\phi}} \{ \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_{\phi}(y|x)} [r(x, y)] - \beta D_{\text{KL}}[\pi_{\phi}(y|x) \parallel \pi_{\text{SFT}}(y|x)] \} \quad (5)$$

Traditionally, this optimization is realized through reinforcement learning, maximizing the following compound reward:

$$r'(x, y) = \{ r(x, y) - \beta [\log \pi_{\phi}(y|x) - \log \pi_{\text{SFT}}(y|x)] \} \quad (6)$$

However, in this approach, the training signal from the KL-divergence term is unnecessarily sparse. This is because $D_{\text{KL}}[\pi_{\phi} \parallel \pi_{\text{SFT}}]$ is differentiable with respect to the parameter vector ϕ and can thus be optimized directly with back-propagation. We use this alternative approach to improve convergence of our reinforcement learning experiments.

KL-Divergence Coefficient: To obtain checkpoints with predictable $D_{\text{KL}}[\pi_{\phi} \parallel \pi_{\text{SFT}}]$, we use the log-space proportional controller of the β coefficient introduced by Ziegler et al. (2019):

$$e_t = \text{clip} \left(\frac{D_{\text{KL}}[\pi_{\phi} \parallel \pi_{\text{SFT}}] - D_{\text{KL}}^{\text{target}}}{D_{\text{KL}}^{\text{target}}}, -0.2, 0.2 \right)$$

$$\beta_{t+1} = \beta_t(1 + K_{\beta} e_t)$$

We additionally enforce a minimum value of β equal to 10^{-4} . This is because while increasing β is expected to decrease the KL-divergence, decreasing β towards zero might not immediately lead to KL-divergence increase. This means that β might become extremely small and take too long to recover once the KL-divergence starts increasing.

Length Penalty: Recall-oriented metrics such as ROUGE are difficult to compare across output summaries of diverse lengths. This is also the case for metrics such as SummaQA (Scialom et al., 2019) and BLANC (Vasilyev et al., 2020) which use the summary as an information source when performing language understanding tasks on the source document since longer summaries trivially allow for more information content.

At the same time, optimizing our reward function tends to lead to longer summaries, especially when $D_{\text{KL}}^{\text{target}}$ is large.

For this reason, we penalize output tokens beyond certain length by decreasing their reward, excluding the end-of-sequence token. Specifically, the following penalty p_i is subtracted from the i -th

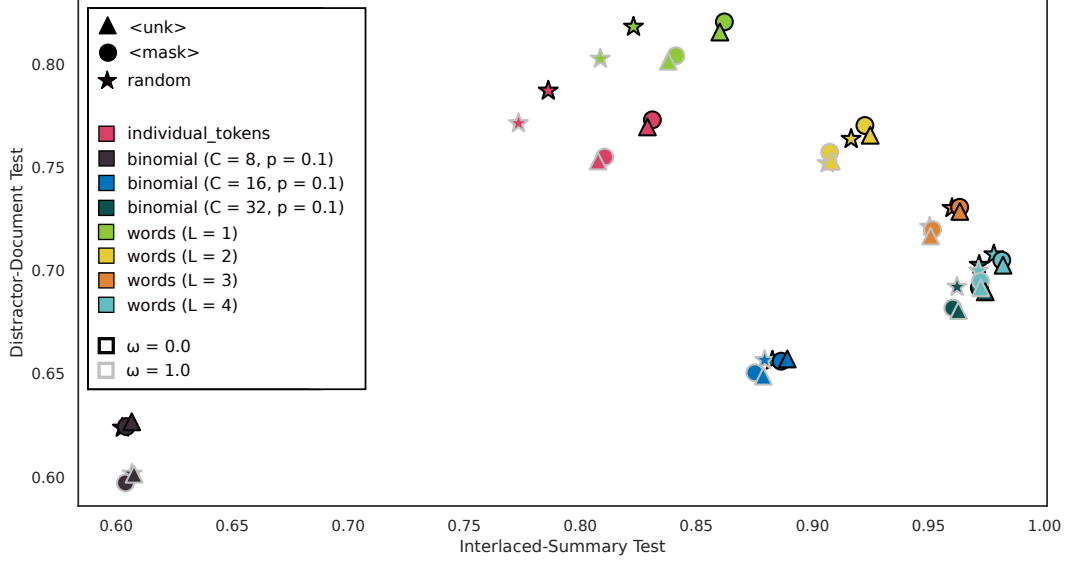


Figure 4: Grid search results for $p_{\text{samples}} = 2^{-1}$, $p_{\text{likelihood}} = 2^{1/2}$.

token’s reward:

$$p_i = \begin{cases} 0 & \text{if } i < k \\ 0 & \text{if the } i\text{-th token is EOS} \\ cb^{i-k} & \text{otherwise} \end{cases} \quad (7)$$

In our experiments, we use $b = 1.01$ and $c = 1$.

6 Experiments

In Section 6.1, we search for a suitable reward function configuration on the XSum dataset. In Section 6.2, we use the reward function in a reinforcement learning setting on both the XSum dataset and the CNN/Daily Mail dataset. In both cases, we use the first half of the dataset for supervised fine-tuning (SFT) of the models and the second half (with labels removed) for reinforcement learning.

The training of the models was performed on a single 40GB A100 GPU; the SFT model fine-tuning and the predictor model fine-tuning took 24 hours each, and the RL models trained for 6 hours.

6.1 Reward Functions Evaluation

Reducers Parameters: We first search for suitable values of p_{samples} and $p_{\text{likelihood}}$. To this end, we run a grid search with $p_{\text{samples}} \in \{2^{i/4}\}_{i=-16}^4$ and $p_{\text{likelihood}} \in \{2^{i/4}\}_{i=-12}^8$. We also include a setting where a *signed maximum* is used in Equations 1 and 2 instead of the generalized mean, meaning that the value with the largest absolute value is selected. As mentioned in Section 3, this is the limit

behavior when p_{samples} or $p_{\text{likelihood}}$ go to ∞ . Standard (non-signed) *maximum* reducer is included as well. Furthermore, we include *minimum* and *signed minimum* in the case of samples reducer with the hope of reducing variance of the reward.

For the purpose of this grid search, values of the other hyperparameters were selected based on preliminary experiments.

The resulting scores on Distractor-Document Test and Interlaced-Summary Test are shown in Figure 3. We selected optimal values for further use by maximizing the harmonic average of both metrics. These optimal values are $p_{\text{samples}} = 2^{-1}$ and $p_{\text{likelihood}} = 2^{1/2}$.

Masking and IDF Weights: After fixing p_{samples} and $p_{\text{likelihood}}$, we run a grid search over selected masking strategies, masking tokens, and IDF weights exponents ω . Results of this grid search are visualized in Figure 4 where each evaluated configuration is represented with a symbol. The symbol type (triangle, circle or star) represents the masking token, its inner color represents the masking strategy, and its outline color represents the exponent of the IDF weights. We observe the following trends:

- Masking individual tokens is strictly outperformed by masking individual words.
- Masking tokens completely at random (with the probability of masking equal to 0.1) is a viable strategy as long as a sufficient number of masks is generated.
- When masking words, increasing the length of masked word bursts improves the Interlaced-

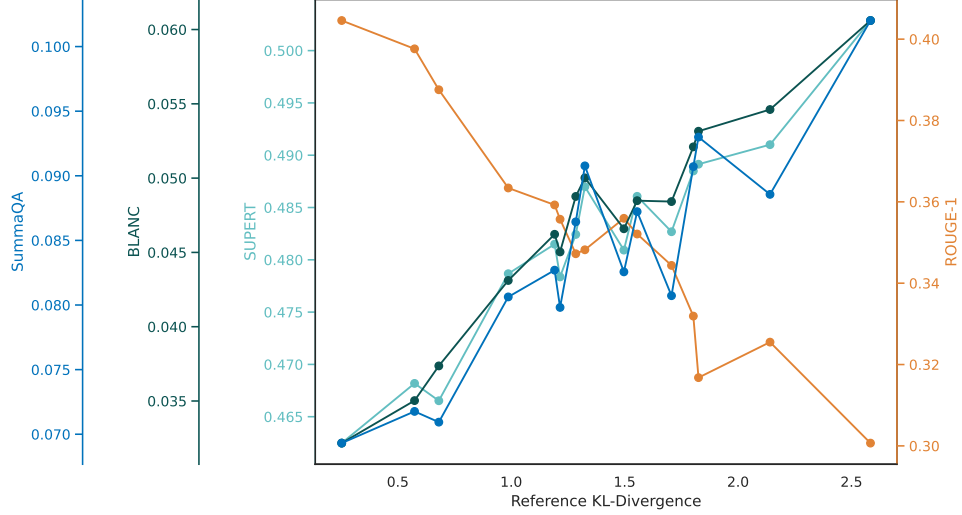


Figure 5: Allowing the model to deviate further from the initial behavior as measured by reference KL-divergence achieves higher scores on reference-free metrics at the cost of lower scores on reference-based metrics. ROUGE-1 is shown as a representative example. Each checkpoint was obtained by training the previously fine-tuned BART-base for 1000 updates using our reward function.

Summary score at the cost of worsening the Distractor-Document score. We hypothesize that the performance increase stems from more samples being calculated for each word (since each word is masked `burst_length-times`) while the performance decrease is due to each sample being affected by multiple words, leading to less specific signal.

- IDF weighting impairs reward function performance in all cases. This is true for all values $\omega \in \{0.5, 1.0, 1.5, 2.0\}$, most of which are not shown for clarity.
- The choice of the masking token is inconsequential in most cases except for when single tokens or single words are masked (in those cases, the random masking token performs worse). We offer no theoretical explanation of this.

6.2 Reinforcement Learning

We use our reward function in a reinforcement learning setting, choosing a hyperparameter configuration based on the results of Section 6.1. The final hyperparameter configuration is listed in Table 3 in Appendix B. All metrics reported in this section were calculated using the SummEval package (Fabbri et al., 2021).

We first run multiple experiments on XSum (Narayan et al., 2018) with BART-base previously fine-tuned in a supervised fashion, varying the target reference KL-divergence and observing its ef-

fect on reference-free and reference-based metrics in Figure 5. Allowing the model to deviate further from the initial behavior as measured by reference KL-divergence achieves higher scores on reference-free metrics at the cost of lower scores on reference-based metrics.

Automatic Evaluation: One of the resulting checkpoints is then evaluated on 12 of the metrics provided by SummEval on the test set. The remaining two SummEval metrics (namely S^3 and SMS) were excluded due to technical problems which we detail in Appendix A. We repeat this for the BART-large model and also for the CNN/Daily Mail dataset (Nallapati et al., 2016).

All results on the test set are listed in Table 1. Our reinforcement learning approach is marked as RL. The models fine-tuned in a supervised fashion on the first half of the summarization dataset and used as a starting point for the reinforcement learning experiments are marked as SFT. We also run supervised fine-tuning on the whole dataset to offset the effect of the RL models seeing more data. These models are marked full-SFT.

The main result in Table 1 is that RL improves all reference-free metrics in all cases at the cost of deviating from the reference summaries, reflected in a decrease in all reference-based metrics. Interestingly, in the case of XSum, the base model achieves higher scores on reference-free metrics than the large model even after thorough hyperparameter tuning.

Model	Reference-based metrics				Reference-free metrics		
	ROUGE-1/2/3/4/L/su*/w	ROUGE-WE-3	BertScore	MoverScore	SummaQA	BLANC	SUPERT
XSum							
base SFT	41.0/18.2/ 9.6/5.55/33.1/17.3/19.7	22.9	0.429	0.247	0.059	0.025	0.454
base full-SFT	42.7/19.8/10.9/6.42/34.7/18.7/20.6	24.4	0.446	0.267	0.059	0.025	0.454
base RL	26.8/ 7.6/ 2.8/1.22/20.4/ 7.6/12.2	10.6	0.252	0.064	0.104	0.064	0.511
large SFT	44.5/21.2/11.9/7.12/36.3/20.1/21.6	26.1	0.467	0.291	0.058	0.025	0.454
large full-SFT	45.8/22.5/12.9/7.96/37.5/21.3/22.4	27.4	0.480	0.307	0.057	0.025	0.453
fb/bart-large-xsum *	45.3/22.2/12.7/7.76/37.1/20.8/22.1	27.1	0.475	0.298	0.059	0.025	0.454
large RL	33.0/11.2/4.64/2.19/24.9/10.8/14.9	15.4	0.322	0.138	0.089	0.049	0.494
CNN/Daily Mail							
base SFT	41.2/18.8/10.9/7.29/38.4/15.8/21.1	22.9	0.404	0.197	0.146	0.099	0.626
base RL ‡	39.1/17.2/ 9.9/6.60/36.3/14.3/20.0	21.0	0.377	0.173	0.154	0.108	0.642
large SFT	41.7/19.1/11.1/7.42/38.4/16.0/21.3	23.2	0.405	0.198	0.150	0.103	0.649
fb/bart-large-cnn †	44.1/21.3/12.9/8.80/41.0/18.2/22.7	25.6	0.426	0.226	0.146	0.0829	0.618
large RL ‡	41.1/18.4/10.7/7.16/37.4/15.3/20.9	22.4	0.389	0.187	0.162	0.114	0.681
Pegasus (huge news) †	44.1/21.5/13.0/8.89/41.0/18.2/22.7	25.7	0.438	0.229	0.150	0.085	0.619

(a) Model scores from summarization-specific evaluation metrics.

Model	BLEU	chrF	CIDEr	METEOR	Length	Stats (cov/comp/den)	Repeated (1/2/3)	Novelty (1/2/3)
XSum								
base SFT	12.015	36.182	1.291	0.184	21.228	0.763/20.916/1.637	0.068/0.003/0.000	0.250/0.720/0.883
base full-SFT	13.482	37.775	1.433	0.194	21.663	0.756/20.530/1.602	0.069/0.003/0.000	0.258/0.729/0.889
base RL	4.307	26.999	0.608	0.118	21.393	0.956/20.629/7.737	0.067/0.003/0.000	0.047/0.256/0.394
large SFT	14.692	39.373	1.512	0.206	22.260	0.748/20.030/1.549	0.069/0.003/0.000	0.267/0.741/0.897
large full-SFT	15.860	40.617	1.637	0.214	22.419	0.743/19.898/1.526	0.068/0.003/0.000	0.272/0.745/0.899
fb/bart-large-xsum *	15.231	39.596	1.571	0.209	21.417	0.757/20.597/1.642	0.066/0.002/0.000	0.257/0.730/0.889
large RL	6.687	30.712	0.843	0.148	22.744	0.877/18.876/3.049	0.071/0.002/0.000	0.129/0.525/0.719
CNN/Daily Mail								
base SFT	15.378	39.348	0.705	0.189	56.986	0.957/14.272/12.900	0.183/0.025/0.000	0.031/0.139/0.241
base RL ‡	14.379	38.733	0.617	0.181	60.519	0.963/13.261/16.586	0.175/0.023/0.000	0.025/0.095/0.167
large SFT	15.753	40.231	0.647	0.194	59.516	0.966/14.206/18.316	0.170/0.018/0.000	0.030/0.109/0.183
fb/bart-large-cnn †	17.101	42.710	0.757	0.211	62.299	0.977/12.881/15.300	0.163/0.013/0.000	0.019/0.118/0.217
large RL ‡	14.916	40.226	0.605	0.190	62.086	0.976/12.928/25.873	0.165/0.015/0.000	0.022/0.051/0.087
Pegasus (huge news) †	17.810	0.391	0.660	0.219	66.756	0.981/12.947/14.985	0.188/0.025/0.025	—

* Model obtained from Hugging Face.

† Scores taken from Fabbri et al. (2021).

‡ Evaluated using `exponential_decay_length_penalty = (55, 0.9)` of the Transformers library (Wolf et al., 2020).

(b) Model scores from general text generation evaluation metrics.

Table 1: Final model scores on 12 SummEval metrics. The two highest scores on each dataset for each metric (lowest for Repeated-1/2/3, coverage, and density) are bolded, ignoring the models marked with † and cases where the values are too small to be distinguishable. BART-base is listed as *base* and BART-large as *large*.

Model	Relevance	Consistency	Readability
base full-SFT	3.90	3.51	4.82
base RL	3.98	3.98	4.63
large full-SFT	3.88	3.80	4.88
large RL	4.04	3.98	4.86
reference	3.92	3.49	4.90

Table 2: XSum average human evaluation scores. The two highest scores on each metric are bolded.

Manual Evaluation: We conduct a small blind manual evaluation on the first 50 test set samples of the XSum dataset, which we detail in Appendix A. The evaluation was performed by a single referee. The final scores are listed in Table 2.

The evaluation confirms that the differences measured by automated metrics translate into meaningful improvements for human readers. This is important because RL may possibly exploit weaknesses in our reward function, which in turn could expose blind spots of the automated metrics. Even though

it is limited, our evaluation validates that this is not the case.

7 Conclusions

Our novel unsupervised token-based reward function for reference-free reinforcement learning improved the the summarization performance of BART in all reference-free automatic metrics of the SummEval package, as well as in manual metrics evaluated manually by human judges. The source code is available at [URL will be provided after review] under an open-source license.

Limitations

Although the reinforcement learning reward function itself is reference free, it still starts from a checkpoint fine-tuned on labeled training data. Overall, however, the demands of the entire pipeline on the training data are reduced due to reference-free reinforcement learning. A limitation of our evaluation is employing only one annota-

tor. Our experiments are also limited to only one pre-trained language model of the encoder-decoder architecture, although it is the standard choice for summarization. Finally, it would be interesting to conduct the experiments in languages other than English.

Ethical Considerations

All manual evaluations within the work described in this paper were done by one male member of the team, a senior student of a Master’s programme as part of his work on his Master’s thesis.

We used the scientific artifacts referenced here consistently with their intended use.

We are not aware of any substantial risks to the public or academic audience when using the software, whether in its working or malfunctioning state. Potential harm stems from biases present in the underlying pre-trained language model, but our work does not significantly increase these. As with any generative model, potential risk can come from generating misleading, irrelevant or incorrect summaries.

References

- Riad Akrou, Marc Schoenauer, and Michele Sebag. 2011. Preference-based policy learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 12–27, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Kristjan Arumae and Fei Liu. 2019. [Guiding extractive summarization with question-answering rewards](#). 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 2566–2577, Minneapolis, Minnesota. Association for Computational Linguistics.
- Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.
- Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. [Deep reinforcement learning from human preferences](#). In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.
- 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, Minneapolis, Minnesota. Association for Computational Linguistics.
- Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. [SummEval: Re-evaluating summarization evaluation](#). *Transactions of the Association for Computational Linguistics*, 9:391–409.
- Yang Gao, Wei Zhao, and Steffen Eger. 2020. [SUPERT: Towards new frontiers in unsupervised evaluation metrics for multi-document summarization](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1347–1354, Online. Association for Computational Linguistics.
- Amelia Glaese, Nat McAleese, Maja Trębacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, et al. 2022. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*.
- Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. 2019. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*.
- 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.
- Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.
- Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gülçehre, and Bing Xiang. 2016. [Abstractive text summarization using sequence-to-sequence RNNs and beyond](#). In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.
- Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. [Don’t give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.
- OpenAI. 2022. Introducing ChatGPT. <https://openai.com/blog/chatgpt>. Accessed: 2024-04-09.

- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.
- Ramakanth Pasunuru and Mohit Bansal. 2018. **Multi-reward reinforced summarization with saliency and entailment**. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 646–653, New Orleans, Louisiana. 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*.
- Thomas Scialom, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2019. **Answers unite! unsupervised metrics for reinforced summarization models**. 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 3246–3256, Hong Kong, China. Association for Computational Linguistics.
- Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020. Learning to summarize from human feedback. In *Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS ’20*, pages 3008–3021, Red Hook, NY, USA. Curran Associates Inc.
- Oleg Vasilyev, Vedant Dharnidharka, and John Bohannon. 2020. **Fill in the BLANC: Human-free quality estimation of document summaries**. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 11–20, Online. Association for Computational Linguistics.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. **Attention is all you need**. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.
- Ronald J. Williams. 1987. A class of gradient-estimating algorithms for reinforcement learning in neural networks. In *Proceedings of the IEEE First International Conference on Neural Networks*, San Diego, CA.
- Ronald J. Williams. 1992. **Simple statistical gradient-following algorithms for connectionist reinforcement learning**. *Machine Learning*, 8(3):229–256.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 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, Online. Association for Computational Linguistics.
- Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. **Bertscore: Evaluating text generation with bert**. In *International Conference on Learning Representations*.
- Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

A Evaluation Details

Output Generation: During evaluation, we use beam search with 4 beams and early stopping. We also ban the repetition of any trigram and limit the maximum generation length to 160 for CNN/Daily Mail and 128 for XSum.

Human Evaluation: We conduct blind manual evaluation of the generated summaries on the first 50 test set samples from XSum using our custom browser application built with Streamlit. A screenshot of this application is shown in Figure 6. The user is presented with general instructions, a definition for each evaluated metric, an XSum article, and five different summaries (four of them being generated from the evaluated models and the remaining one being the supervised reference summary). The presented summaries are shuffled randomly for each example so that the user cannot tell which one was generated from which model.

We take inspiration from Fabbri et al. (2021) in designing the axes on which summaries are evaluated. However, we discard their *Coherence* metric which measures inter-sentence coherence since XSum summaries mostly consist of just one sentence. In the end, we use the following three metrics:

- *relevance* – measures how well the summary captures key points of the article,
- *consistency* – measures whether the facts in the summary are consistent with the facts in the original article,
- *readability* – measures the grammatical, stylistic, and logical quality of the summary when read by itself (without the original article).

Instructions

Please follow these steps:

1. Carefully read the news article, be aware of the information it contains.
2. Carefully read the proposed summaries A-E (5 in total).
3. Rate each summary on a scale from 1 (worst) to 5 (best) by its *relevance*, *consistency*, and *readability*.

Definitions

Relevance:

This rating measures how well the summary captures the key points of the article.
Consider if the important aspects of the article are contained in the summary.

Consistency:

This rating measures whether the facts in the summary are consistent with the facts in the original article.
Consider whether the summary reproduces facts accurately and does not make up untrue information.

Readability:

This rating measures the quality of the summary when read by itself (without the original article).
Consider if it is well-written and grammatically correct, and if it sounds natural.

Evaluation

Article

Prison Link Cymru had 1,099 referrals in 2015-16 and said some ex-offenders were living rough for up to a year before finding suitable accommodation. Workers at the charity claim investment in housing would be cheaper than jailing homeless repeat offenders. The Welsh Government said more people than ever were getting help to address housing problems. Changes to the Housing Act in Wales, introduced in 2015, removed the right for prison leavers to be given priority for accommodation. Prison Link Cymru, which helps people find accommodation after their release, said things were generally good for women because issues such as children or domestic violence were now considered. However, the same could not be said for men, the charity said, because issues which often affect them, such as post traumatic stress disorder or drug dependency, were often viewed as less of a priority. Andrew Stevens, who works in Welsh prisons trying to secure housing for prison leavers, said the need for accommodation was "chronic". "There's a desperate need for it, finding suitable accommodation for those leaving prison there is just a lack of it everywhere," he said. "It could take six months to a year, without a lot of help they could be on the streets for six months. "When you think of the consequences of either being on the street, especially with the cold weather at the moment or you may have a roof over your head, sometimes there is only one choice." Mr Stevens believes building more one-bedroom flats could help ease the problem. "The average price is a hundred pounds a week to keep someone in a rented flat, prison is a lot more than that so I would imagine it would save the public purse quite a few pounds," he said. Official figures show 830 one-bedroom properties were built in the year to March 2016, of an overall total of 6,900 new properties in Wales. Marc, 50, who has been in and out of prison for the past 20 years for burglary offences, said he struggled to find accommodation each time he was released. He said he would ask himself: "Where am I going to stay? Where am I going to live? Have I got somewhere where I can see my daughter." "You're put out among the same sort of people doing the same sort of thing, and it's difficult, it's difficult to get away from it. It's like every man for himself, there's nothing." Marc has now found stable accommodation with homeless charity Emmaus and said it had been life changing. "You feel safe, you got hot food, you've got company of people in similar situations to yourself but all dealing with different issues. It's a constructive, helpful

Summary A

Prison leavers are living rough for up to a year after their release, a charity

Relevance

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

Consistency

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

Readability

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

Summary B

There is a "chronic" need for more housing for prison leavers in Wales, a

Relevance

Figure 6: Screenshot of the browser application used for human evaluation (cropped).

SummEval Technical Problems: During the evaluation using SummEval, we exclude the S^3 metric which is a model-based combination metric that uses other metrics as input features. The model in question is unfortunately serialized using an old version of Python not available in our execution environment.

We also opt to exclude the Sentence Mover's Similarity (SMS) as it reports the value of 1.0 for all our experiments. Since we did not modify the default configuration for this metric we suspect it might be a problem with the SummEval implementation.

Regarding the model scores listed in [Fabbri et al. \(2021\)](#), we believe there is a problem with the computation or reporting of the Repeated-3 metric as it is exactly equal to the Repeated-2 metric in all cases except for one which is hard to believe given the metrics' definitions. Nevertheless, we report their unaltered results in Table 1.

B Hyperparameters

Table 3 lists the final hyperparameters for the reinforcement learning task.

C Predictor Fine-Tuning

If the predictor is taken to be the unaltered BART-base model, the reward function performs significantly worse both according to quantitative analysis and when employed in reinforcement learning. We therefore fine-tune the predictor on a summarization dataset with inputs and outputs swapped, in other words, the predictor is fine-tuned to predict the source document from its summarization.

We also found the reward function's performance marginally improved when the predictor's inputs are masked during fine-tuning similarly to how they are masked during the reward function evaluation. Specifically, we employ the `random_bursts` strategy where the `burst_length` is also chosen at random, using the masking token that is then used during reinforcement learning. This means that three different predictors are em-

Hyperparameter	XSum		CNN/Daily Mail	
	BART-base	BART-large	BART-base	BART-large
Reward Function Configuration				
ω (IDF weight power)			0.0	
Samples reducer	$p_{\text{samples}} = 2^{-1}$			min
$p_{\text{likelihood}}$	$2^{1/2}$			1.0
Masking strategy			words	
Mask burst length			2	
Masking token	<mask>			random
Reinforcement Learning Configuration				
Batch size	16	12	12	5
Learning rate			10^{-5}	
γ (discount factor)			0.9	
Minimum episode length			16	
$D_{\text{KL}}^{\text{target}}$	2.1	2.7	0	0
β_0 (initial KL-divergence coefficient)	0.014	0.02	0.03	0.03
K_β	0.25	0.5	0	0
β_{min}		10^{-4}		0
k (length penalty start index)		36		69

Table 3: Final hyperparameters for the reinforcement learning task.

ployed during the experiments (one for each masking token).