BRIDO: Bringing Democratic Order to Abstractive Summarization

Junhyun Lee*†, Harshith Goka*, Hyeonmok Ko

Samsung Research
{junhyun8.lee, h9399.goka, felix.ko}@samsung.com

Abstract

Hallucination refers to the inaccurate, irrelevant, and inconsistent text generated from large language models (LLMs). While the LLMs have shown great promise in a variety of tasks, the issue of hallucination still remains a major challenge for many practical uses. In this paper, we tackle the issue of hallucination in abstract text summarization by mitigating exposure bias. Existing models targeted for exposure bias mitigation, namely BRIO, aim for better summarization quality in the ROUGE score. We propose a model that uses a similar exposure bias mitigation strategy but with a goal that is aligned with less hallucination. We conjecture that among a group of candidate outputs, ones with hallucinations will comprise the minority of the whole group. That is, candidates with less similarity with others will have a higher chance of containing hallucinated content. Our method uses this aspect and utilizes contrastive learning, incentivizing candidates with high *inter-candidate* ROUGE scores. We performed experiments on the XSum and CNN/DM summarization datasets, and our method showed 6.25% and 3.82% improvement, respectively, on the consistency G-Eval score over BRIO.

Introduction

Large language models (LLMs) have achieved great success in many natural language processing tasks, such as machine translation (Vaswani et al. 2017; Arivazhagan et al. 2019), question answering (Karpukhin et al. 2020; Zhu et al. 2021), and text summarization (Lewis et al. 2020; Zhang et al. 2020). However, despite their impressive performance, it has been shown that LLMs exhibit "hallucination," in which they generate texts that are factually inaccurate, irrelevant, or inconsistent with the provided context. Hallucination persists even in nowadays much larger LLMs (Achiam et al. 2023; Anil et al. 2023; Touvron et al. 2023) which are known, or believed, to have an order of tens or hundreds of billions of parameters. Since such models can be potentially used in safety-critical applications, such as medical diagnosis or financial forecasting, hallucination poses a serious threat to the reliability and trustworthiness of these systems.

*These authors contributed equally.

†Corresponding author.

Copyright © 2025, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Metric	Model	XSum	CNN/DM
QAFactEval (LERC)	base	**2.047**	**4.551**
	BRIO	1.927	4.070
	ref.	*1.713*	*3.460*
QAFactEval (F1)	base	**20.3**	**80.6**
	BRIO	19.0	65.9
	ref.	*15.1*	*48.6*
QAFactEval (EM)	base	**11.5**	**66.2**
	BRIO	10.5	50.7
	ref.	*7.8*	*33.4*
QAFactEval (IsAns)	base	**63.0**	**97.1**
	BRIO	60.8	92.4
	ref.	*57.0*	*85.2*
G-Eval (consistency)	base	*3.62*	*4.42*
	BRIO	**3.70**	**4.45**
	ref.	**3.70**	4.43

Table 1: QAFactEval and G-Eval consistency results on Xsum and CNN/DM datasets for BRIO outputs, the base model outputs, and the reference summary. The base model is PEGASUS for the XSum dataset and BART for the CNN/DM dataset. The highest score is in bold and the lowest is in italics.

Among the variety of reasons researchers attribute to hallucination, Wang and Sennrich (Wang and Sennrich 2020) first claimed that exposure bias (Bengio et al. 2015; Ranzato et al. 2016) plays an important role. The mitigation of exposure bias is a subject that has been extensively investigated, and BRIO (Liu et al. 2022a) was one of them introduced in the scope of abstractive text summarization. Though not specifically in the context of hallucination, it has achieved the state-of-the-art ROUGE (Lin 2004) score at the time. Following the logic that exposure bias is a cause of hallucination, one would expect that BRIO would have less hallucination than its base models, PEGASUS (Zhang et al. 2020) and BART (Lewis et al. 2020).

As a preliminary experiment, we first check the aforementioned conjecture by comparing the hallucination metrics of BRIO and its base models on two text summarization datasets, XSum (Narayan, Cohen, and Lapata 2018)

and CNN/DM (Hermann et al. 2015). (Details on the metrics and the datasets to follow in the Experimental Setup section.) We show the results in Table 1, together with the score of the reference summary. Contrary to the expectation that BRIO would have mitigated hallucination, it scored less than the base models in most evaluations. We argue that this is due to the low performance of the reference summary, which scored worst among the three in most metrics. Note that BRIO (the abbreviation of "Bringing Order to Abstractive Summarization") utilizes the ranking of candidate summaries calculated as the similarity with respect to the reference summary.

With this information, we conclude that BRIO is not well-suited for hallucination mitigation, although its exposure bias reduction scheme is still valid. Thus, we introduce an alternative to BRIO, whose goal is to handle hallucination better while retaining the ability to alleviate exposure bias. With the observation that the reference summary scores poorly in hallucination metrics, we avoid using the reference as the sole grader of the candidate summaries. Additionally, we hypothesize that when we have a diverse group of candidates the hallucinated content would consist of the minority among them. Then we can order the candidates by the similarity among them (this group may or may not include the reference summary), with a higher rank for candidates more similar to others collectively. With this, we were able to achieve a 6.25% and 3.82% increase in the LLM-based hallucination measure for the XSum and CNN/DM datasets, respectively. This paper builds upon the sequence-to-sequence (seq2seq) model BRIO and compares it with other seq2seq baselines. However, the method can be applied to decoder-based LLMs and even greatly benefit from the large group of contemporary LLMs.

The main contribution of this work is as follows:

- We introduce BRIDO, an abstract summarization model that mitigates hallucination via decreasing exposure bias. BRIDO employs rank-based contrastive learning where the ranking of the candidates is determined by the similarity among the candidate group.
- We test BRIDO with two summarization datasets XSum and CNN/DM, and find a significant increase in the LLM-based hallucination measure.

Related Work

Hallucination of LLMs is a very active area of research and there are many useful survey papers on the subject matter (Ji et al. 2023; Zhang et al. 2023; Rawte, Sheth, and Das 2023; Huang et al. 2023; Tonmoy et al. 2024). Among the various aspects of hallucination, we would like to concentrate on the hallucination caused by exposure bias and its mitigation strategy.

Exposure bias (Bengio et al. 2015; Ranzato et al. 2016) is a major source of hallucination in the pre-training stage (Wang and Sennrich 2020). LLMs have different procedures in training and inference. While training, the model learns the probability distribution between words (tokens) by a teacher-forced maximum likelihood estimation (MLE). That is, regardless of the predicted token, the next token used in the following step is determined from the ground truth example. In contrast, during the autoregressive inference, the actual predicted (generated) token is included in the context of the next step. This discrepancy between the training and inference is referred to the exposure bias.

Wang, Liu, and Lam (2023) address hallucination in neural machine translation in the context of exposure bias. They demonstrate that introducing intermediate supervision signals from the source-like to target-like structure alleviates exposure bias and thus reduces hallucination. Other studies devoted to mitigating exposure bias include methods such as reinforcement learning (Chen, Wu, and Zaki 2019), unlikelihood training (Welleck et al. 2019), and minimum Bayes risk (Bertsch et al. 2023), although they do not directly address the issue of hallucination.

One particularly important study on exposure bias mitigation related to our work is BRIO (Liu et al. 2022a). BRIO is an abstractive summarization model, whose goal is to maximize the ROUGE score by reducing exposure bias through contrastive learning. The method was successful in the sense that it had achieved the state-of-the-art ROUGE score at the time. Since our proposed method is closely related to BRIO, we will present its mechanism in detail in the following section.

Many other hallucination reduction strategies exists, which do not involve exposure bias, in all stages of the LLMs including dataset preparation, training, and inference. We refer the readers to the survey papers for more information.

Bringing the Democratic Order

Here we introduce our proposed method **BRIDO**, which stands for "Bringing Democratic Order to Abstractive Summarization." Unlike BRIO, where the "order" is dictated by the ROUGE score only with the reference, BRIDO orders the candidates by collectively comparing the similarities with all other candidates, thus introducing a democratic taste to the ordering algorithm.

Motivation Let us first revisit the scheme of BRIO. BRIO uses a contrastive loss in addition to the usual MLE loss of abstractive summarization. For contrastive learning, N candidates were generated through diverse beam search (Vijayakumar et al. 2016) and ranked. The ranking is based on the ROUGE score with the reference summary, which measures the lexical similarity. It is the additional contrastive learning that complements the MLE loss, that mitigates the exposure bias. The method achieves the state-of-the-art ROUGE score (with the reference). This is exactly the information the contrastive loss is providing to the model – to value a high ROUGE score.

We now consider this strategy in the sense of hallucination mitigation and figure out points of modifications. First, we questioned whether the reference summary is good enough so that aligning with the reference also reduces hallucination. A preliminary experiment (Table 1) has been done to answer this question. The result is – the reference consistently scored the lowest, and PEGASUS/BART scored the highest in all QAFactEval metrics. The intuition that BRIO

would be placed in between PEGASUS/BART and the reference was correct, but counter-intuitively, the reference score was much worse than PEGASUS/BART. For G-Eval, BRIO performed better than the base models, but only by a small margin.

This suggests that the reference is *not* hallucination-free as assumed but is actually quite unfaithful. This is a characteristic of the XSum and CNN/DM datasets, as the reference there was collected from the bullet points of the news article. The bullet points are intended to complement the main article and are not an intended summary of the article. Therefore they often include information that is not in the source, which is a hallucination when considered as a summary of the article. This is also shown in previous works (Liu et al. 2023a) which compare the human evaluation of the reference and various model outputs. Consistent with our observation, the reference scored last in comparison with various models.

For example, in some op-ed articles the reference starts with "[author]: ..." which explicitly indicates this is the author's opinion and not the news outlet's official view. However, the author's name only shows up as a separate entity that is not part of the dataset and does not appear in the article (which is the source). In the perspective of the dataset, this is hallucination since it is an information that appeared in the summary which is not present in the source. Other examples include articles that are intended to be read after the title and the highlight. In those articles the title/highlight serves as a crucial context and contains information not included in the article (source). Instead, when the order is reversed where the article is the source and the highlight is its reference summary, then the reference will be inconsistant and inaccurate.

To summarize, BRIO succeeded in the alignment with the reference and scored well in the ROUGE (which is a comparison with the reference), but unfortunately, the reference had a high level of hallucination. Therefore it indeed reduced exposure bias, but not hallucination. If we could find a better alternative to align with, we would be able to both reduce exposure bias and mitigate hallucination.

Method

Similar to BRIO, our method BRIDO is based on coordinating the model so that the model can assign a higher estimated probability to the "better" candidates during inference. This is not directly captured in the standard MLE objective used in training – there could be multiple reasonable generations, and the MLE loss alone does not say anything about the order of the two imperfect references. We require the model to be able to predict the ranking order of the set of most probable candidate summaries, which are its own beam search results.

To achieve this objective, we train the model with a contrastive loss (Hopkins and May 2011; Zhong et al. 2020; Liu and Liu 2021):

$$\mathcal{L}_{\text{ctr}} = \sum_i \sum_{j>i} \max(0, f(S_j) - f(S_i) + \lambda_{ij}). \quad (1)$$

S_i and S_j are candidate summaries that satisfy $Score(S_i) > Score(S_j)$, that is, i, j are the rankings of the candidates measured in $Score$. $f(S_i)$ is the length-normalized estimated log probability, i.e., the estimated log probability normalized by $|S_i|^\beta$. (β is the length penalty hyperparameter) λ_{ij} defines the target margin which is related to either the difference in score, $\lambda_{ij} = (Score(S_i) - Score(S_j)) * \lambda$, or difference in ranking, $\lambda_{ij} = (j - i)) * \lambda$, with the margin hyperparameter λ.

Eq. (1) is common for both BRIO and BRIDO. The main technical difference between the two is the scoring scheme, $Score$. BRIO uses the ROUGE (Lin 2004) score with the reference summary S^* to score the candidate summaries. Defining $R(x, y)$ as the ROUGE score between x and y, $Score_{\text{BRIO}}(S) = R(S, S^*)$. In BRIDO, we instead score these candidates by calculating the ROUGE score with the rest of the candidates and references, i.e., we use the *inter-candidate* ROUGE:

$$Score_{\text{BRIDO}}(S_i) = \frac{1}{N-1+\alpha}\left(\sum_{j\neq i} R(S_i, S_j) + \alpha R(S_i, S^*)\right). \quad (2)$$

α is a hyperparameter controlling the contribution between other candidate summaries and the reference. For the ROUGE calculation, we use the Python implementation (https://github.com/pltrdy/rouge) with the default settings.

The intuition behind this choice of scoring scheme is that hallucinations would consist of a minority of information among a large enough group. Consider a sentence with a single fact masked. If several models try to predict the fact, there would be one correct answer and a nearly infinite number of incorrect answers. Therefore, if the fact can be inferred from the context and not from a random guess, the true fact would consist of the majority of the answers. In contrast, in the worst-case scenario of a random guesses, the correct answer and each of the different incorrect answers would occur in equal frequency. Thus using the scoring scheme of Eq. (2), we conjecture that the candidates with high factuality will get high scores since they will agree more with other candidates. Consequently, if we perform contrastive learning based on this ranking, we will be able to reduce both exposure bias and hallucination. Let us reiterate the conceptual similarities and distinctions between BRIO and BRIDO at this point. They both use contrastive learning (as the multi-task together with the cross-entropy loss) to teach the models which summary is better, thus alleviating both exposure biases. However, the "better" summary is the candidate with the higher $Score$ in each model. BRIO considers a higher $Score_{\text{BRIO}}$ as better, which implies a higher ROUGE score but not necessarily less hallucination. In contrast, a high $Score_{\text{BRIDO}}$ is considered better in BRIDO, and together with the aforementioned conjecture this directly implies less hallucination.

An alternative choice of scoring can be a direct measure of hallucination, for example using the hallucination metric used in this paper (QAFactEval and G-Eval) as the scoring scheme. This is similar in spirit to BRIO – to maximize ROUGE, use ROUGE as the score – and has the advan-

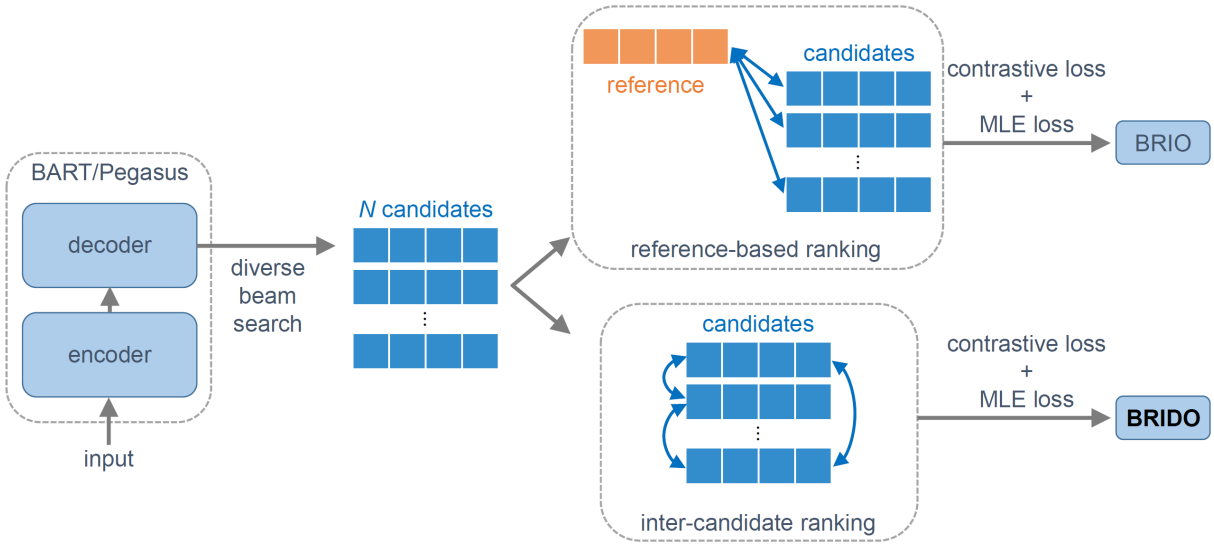

Figure 1: The BRIDO framework compared with BRIO (Liu et al. 2022a) and the base models (BART/Pegasus). BRIDO ranks the candidates based on the inter-candidate similarity, while BRIO uses the reference-based similarity score for ranking. In both cases, the similarity is measured by the ROUGE score.

tage that it does not need any additional assumptions. However, hallucination metrics require additional models which means much more computation and the possibility of error propagation. Here, to limit the additional cost and to ensure the efficiency of the method, we choose to proceed with a simple scheme (ROUGE) together with a reasonable assumption. Note that ROUGE can be calculated between *any* two texts, although it has been conventionally used as that between summary and reference. In the BRIDO scoring scheme, we are considering ROUGE between different summary candidates.

The contrastive loss (Eq. (1)) is added to the cross-entropy loss. The total loss becomes:

$$\mathcal{L} = \mathcal{L}_{\text{xent}} + \gamma \mathcal{L}_{\text{ctr}}. \tag{3}$$

Here, γ is the weight of the contrastive loss compared to that of the cross-entropy loss. With this multi-task objective, BRIDO learns the ability to judge which candidate is "better" (less exposure bias) than just to summarize. And, the "better" summarization for BRIDO is that with less hallucination.

Note that apart from the scoring scheme, our method is conceptually analogous to BRIO. A schematic illustration of BRIDO, in comparison with BRIO and the base models, is shown in Figure 1.

Hyperparameters

We enumerate the hyperparameters in training BRIDO. We leave the inference parameters such as temperature, top-k, and top-p unchanged from BRIO and do not list them here. While generating the candidates from the pre-trained models, the diverse beam search (Vijayakumar et al. 2016) is used. It has three main parameters – the diversity penalty(η), the number of beam groups(N_g), and the number of candidates (N). When defining the contrastive loss (Eq. (1)) and the score within, there are two more hyperparameters – the margin value (λ) and the weight of reference summary (α).

Diversity penalty (η) The diversity penalty ensures that the outputs of the diverse beam search are different enough. The larger the value, the more diverse the outputs.

Number of beam groups (N_g) The total number of candidates is divided into N_g beam groups. The diversity penalty applies to outputs between distinct beam groups. Candidates within the beam group are conventional beam search results.

Number of candidates (N) The total number of candidates to generate. We have two settings on the relation between N and N_g: one is "number of candidates / 8 = number of beam groups" i.e., $\frac{N}{8} = N_g$ used in XSum experiments, and the other is "number of candidates = number of beam groups." i.e., $N = N_g$ used in CNN/DM experiments. That is, each beam group has 8 candidates in XSum, and a single candidate in CNN/DM.

The role of reference summary (α) We can choose to include the reference summary in the scoring process. Without the reference, the model's outcome is very much dependent on the base model's performance and, therefore, may be unstable. We control the involvement of reference with a hyperparameter α, which is defined in Eq. (2). $\alpha = 0$ is the reference-less limit, and $\alpha \to \infty$ is the BRIO limit. In the experiments, we consider two more cases where $\alpha = 1$ (reference is just another candidate) and $\alpha = N - 1$ (the final score is the macro-average of the inter-candidate ROUGE and the reference ROUGE).

Dataset	Model	ROUGE			QAFactEval				G-Eval	
		R-1	R-2	R-L	LERC	F1	EM	IsAns	cons.	ave.
XSum	PEGASUS	47.18	24.62	39.37	**2.047**	20.29	11.46	62.97	3.62	3.31
	BRIO	**48.86**	**25.54**	**40.48**	1.956	18.98	10.47	61.50	3.68	3.37
	BRIDO	47.69	24.62	39.19	2.028	**20.44**	**11.71**	**63.08**	**3.80**	**3.47**
CNN/DM	BART	44.24	21.30	41.06	**4.551**	**80.56**	**66.17**	**97.08**	4.42	3.82
	BRIO	**47.74**	**23.75**	**44.55**	4.070	65.88	50.72	92.38	4.45	3.91
	BRIDO	45.81	22.95	42.51	4.392	75.45	60.64	95.45	**4.55**	**3.93**

Table 2: The base results of BRIDO, in comparison with the baseline models PEGASUS/BART and BRIO, for the XSum and CNN/DM datasets. BRIDO improves over BRIO in all G-Eval and QAFactEval metrics.

Margin value (λ) The margin value is defined in λ_{ij} which is mentioned in Eq. (1). We consider two types of margins: fixed margin (the type used in BRIO) and difference margin. Fixed margin between two candidates S_i and S_j is defined as $\lambda_{ij} = (j - i) * \lambda$, where the subscripts i and j indicates the ranking based on the score ($Score(S_i) > Score(S_j)$). Whereas the difference margin between these two candidates is defined as $\lambda_{ij} = (Score(S_i) - Score(S_j)) * \lambda$. The fixed margin drives the model to have the length-normalized estimated log probability at least equality separated based on their ranking. In the difference margin scheme the separation is proportional to the score of the candidates, and thus can be considered to have higher resolution.

Experimental Setup

The BRIDO model is fine-tuned with the XSum (Narayan, Cohen, and Lapata 2018) and CNN/DM (Hermann et al. 2015) datasets to reduce the loss of Eq. (3) from the corresponding pre-trained models, PEGASUS (Zhang et al. 2020) and BART (Lewis et al. 2020). The same hyper-parameters and optimizer (learning rate = 2×10^{-3}, epoch = 2, and Adam optimizer (Kingma and Ba 2014)) are used for all datasets to validate the universality of our method except for the following differences – we use a batch size of 16 for PEGASUS - XSum experiments and a batch size of 24 for BART - CNN/DM experiments. We keep the same generation parameters as BRIO during the evaluation for each dataset. All training and evaluation are conducted using 8 Nvidia Ampere A100 GPUs (80GB).

Dataset

We use the following two datasets widely used in abstractive text summarization in our experiments.

XSum (Narayan, Cohen, and Lapata 2018) is a highly abstractive dataset of articles from the British Broadcasting Corporation (BBC), and the introductory sentences (generally one sentence) in these articles are regarded as summaries.

CNN/DM (Hermann et al. 2015) is a news dataset from the Cable News Network (CNN) and the Daily Mail (DM). Following (Nallapati et al. 2016), we treat the news articles as the source and the associated highlights as the summaries.

However, as previously noted, considering the introductory sentence (XSum) or the article highlight (CNN/DM) as the article summary has hallucination issues.

Metric

We choose the metrics to cover the various summarization and hallucination metrics.

ROUGE Recall-Oriented Understudy for Gisting Evaluation (Lin 2004) measures the lexical similarity between summaries and human-written references by comparing the overlap of n-grams, which are sequences of n words. The most common variants of ROUGE include ROUGE-N (for n-gram matching) and ROUGE-L (for longest common subsequence matching). Among these, we use ROUGE-1 (R-1), ROUGE-2 (R-2), and ROUGE-L (R-L) for evaluation. For the ranking, we use 2 R-1 $\times$ R-2$/$(R-1 $+$ R-2) for XSum and (R-1 $+$ R-2 $+$ R-L)$/3$ for CNN/DM following Liu et al. (2022a). ROUGE's limitation includes the lexical bias and the questionable quality of the reference. In this work, we generalize this conventional definition of ROUGE and use it to measure the lexical similarity between summary candidates as well.

QAFactEval (Fabbri et al. 2022) is a QA-Based Factual Consistency Evaluation without a reference. Given a summary and the source document, it uses a question generation model to generate questions for each of the named entities. It then uses the source document as a reference for the question-answering model to generate the answers for each question. It evaluates the answers with the true answer and measures the consistency of a summary with its source document. The submetrics (LERC, F1, EM, and IsAns) are the methods used in comparing the model answer with the true answer. LERC has a maximum score of 5 and F1, EM, and IsAns are measured in percent. QAFactEval, like all other QA-based factuality metrics, suffers from bias towards extractive summaries.

G-Eval (Liu et al. 2023b) is a framework for using large language models with chain-of-thoughts (CoT) and a form-filling paradigm to assess the quality of NLG outputs. G-Eval consists of four sub-metrics which are coherence, consistency, fluency, and relevance. Among these four, consistency is the measure of hallucination. We use G-Eval consistency as a hallucination metric, and the average G-Eval score

Dataset	η	ROUGE			QAFactEval				G-Eval	
		R-1	R-2	R-L	LERC	F1	EM	IsAns	cons.	ave.
XSum	0.1	**47.69**	**24.62**	**39.19**	**2.028**	20.44	11.71	**63.08**	3.80	3.47
	0.3	47.33	24.11	38.57	2.010	20.50	**11.84**	62.96	**3.89**	**3.56**
	1.0	46.94	23.77	38.12	2.007	**20.52**	11.81	62.85	3.86	3.55
CNN/DM	0.1	45.98	22.97	42.51	4.414	76.55	**62.18**	95.62	4.55	3.94
	0.3	**46.06**	**23.07**	42.66	4.379	75.21	60.56	95.35	4.51	3.92
	1.0	45.81	22.95	42.51	4.392	75.45	60.64	95.45	4.55	3.93
	3.0	46.03	22.99	**42.69**	**4.417**	76.13	61.41	**95.81**	**4.58**	**3.95**
	10.0	45.42	22.62	42.05	4.413	**76.64**	62.17	95.65	4.54	3.94

Table 3: The performance of BRIDO when tuning the diversity penalty, η. $\eta = 0.1$ and $\eta = 1.0$ are the same parameters as in Table 2 for XSum and CNN/DM, respectively. Considering our main metric G-Eval the best parameters are $\eta = 0.3$ (XSum) and $\eta = 3.0$ (CNN/DM).

as a metric for the overall quality of the summarization. This will be our main metric for both quality and hallucination, as it is assisted by a large auxiliary model and does not suffer from the aforementioned deficiencies of ROUGE and QAFactEval. We use a slightly modified method by merging the quality assessment's various aspects into one prompt. We use GPT-4 (Achiam et al. 2023) as our LLM evaluator.

Baseline Models

Following Liu et al. (2022a), we use PEGASUS (Zhang et al. 2020) and the BRIO's XSum model as our baseline for the XSum experiments. Similarly, we use BART (Lewis et al. 2020) and the BRIO's CNN/DM model as our baseline for the CNN/DM dataset experiments. Both PEGASUS and BART are large pre-trained seq2seq LLMs standard in the literature. BRIO's XSum and CNN/DM models use PEGASUS and BART as their base models, respectively. In the following, we indicate both models as BRIO for simplicity.

Results

We trained two BRIDO models, each finetuned with XSum and CNN/DM. First, we fix the diversity penalty (η) to its corresponding BRIO value ($\eta = 0.1$ for XSum and $\eta = 1.0$ for CNN/DM) and also exclude the reference summary from the scoring ($\alpha = 0$). We perform an extensive search for the remaining hyperparameters: $N \in \{16, 32\}$, $\gamma \in [10, 100]$, $\lambda \in [0.001, 0.1]$, and fixed/difference margin scheme.

The best models were $N = 32$, $\gamma = 50$, $\lambda = 0.01$, difference margin for XSum; $N = 32$, $\gamma = 20$, $\lambda = 0.1$, difference margin for CNN/DM. Their performance, together with the baseline evaluations are shown in Table 2. The results of the other experiments for the hyperparameter search is presented in the appendix.

The ROUGE score is best in BRIO for both experiments, as expected. However, it scored worst in all QAFactEval metrics by a wide margin which reiterates the results in Table 1. For the QAFactEval, the base models (PEGASUS/BART) were on average the best, however in part due to its extractive-ness of the summaries. PEGASUS and BART outputs tend to be more extractive than BRIO or BRIDO, and this is a reason for their high score in QAFactEval measures which suffers from bias towards extractive summaries.

The fact that PEGASUS is not as dominant as BART in QAFactEval also supports that the extractiveness is the reason for the base models' high QAFactEval score. Note that XSum is intended to be more abstractive than CNN/DM (that is, the XSum reference summary is more abstract than that of CNN/DM). This is also evident from the exceptionally high values of QAFactEval scores for BART on CNN/DM. Scores such as 4.551 LERC and 97.08 IsAns cannot be explained without extractiveness having an effect. That said, although BRIDO was not the best among the three in QAFactEval, it showed significant improvement from BRIO and impressively scored better than PEGASUS in three submetrics in XSum.

G-Eval is where BRIDO was truly dominant. It scored best in both consistency and average for XSum and CNN/DM. The improvement of the consistency score is 4.97% (PEGASUS) and 3.26% (BRIO) for XSum; 2.94% (BART) and 2.25% (BRIO) for CNN/DM. Note that the improvement is more prominent for the more abstractive dataset XSum. We conclude here that our method BRIDO has mitigated hallucination (high G-Eval consistency score) without any degradation in summarization ability (high G-Eval average score).

Effect of diversity penalty One crucial hyperparameter in BRIDO is the diversity penalty (η). This is because η has a more sensitive optimal value than that in BRIO. η should not be too small that the candidate summary is not diverse enough. Then the probability of the generated candidate having a hallucinated content would be too small for the model to learn its pattern. In contrast, if η is too large, the candidate's summarization quality may not be good enough as the diversity beam search has traded diversity with summarization quality. The optimal η should be large enough to ensure the candidate summaries are diverse enough so that hallucinated content occurs regularly, and at the same time small enough that all candidates are above a certain quality.

We tune η to search for the optimal value in both datasets. For XSum we increase η up to 1.0, and for CNN/DM we search from a range of 0.1 to 10.0. The results are shown in

Dataset	α	ROUGE			QAFactEval				G-Eval	
		R-1	R-2	R-L	LERC	F1	EM	IsAns	cons.	ave.
XSum	0	47.33	24.11	38.57	2.010	**20.50**	**11.84**	62.96	3.89	**3.56**
	1	47.34	24.12	38.58	2.010	20.46	11.79	**63.02**	3.88	3.55
	31	47.44	24.25	38.78	2.008	20.27	11.55	62.89	**3.91**	3.54
	∞	**47.53**	**24.49**	**39.05**	**2.012**	20.15	11.49	62.85	3.77	3.46
CNN/DM	0	46.03	22.99	42.69	**4.417**	**76.13**	**61.41**	**95.81**	4.58	3.95
	1	46.10	23.03	42.76	4.411	75.81	61.07	95.60	4.53	3.93
	31	46.65	22.98	43.00	4.224	70.61	55.44	93.71	**4.62**	**4.02**
	∞	**47.21**	**23.74**	**43.67**	4.211	70.25	55.31	93.52	4.52	3.97

Table 4: The performance of BRIDO when tuning the weight on reference summary, α. The baseline results ($\alpha = 0$ row) are the same parameters as in Table 3 for $\eta = 0.3$ (XSum) and $\eta = 3.0$ (CNN/DM) which we have concluded the best parameters in the sense of hallucination mitigation. Considering our main hallucination metric G-Eval consistency, the best parameters are $\alpha = 31$ for both datasets.

Table 3.

We find some fluctuation in the performance. However, the performance difference between models was not significant compared to Table 2. The default value of η in Table 2 is the value used in BRIO, which is already an optimized value from Liu et al. (2022a). We expected that the optimal value would be different for BRIO and BRIDO, as the two has quite distinct scoring scheme. While this expectation was indeed true, the difference was not as prominent. We do observe some improvement in the G-Eval metrics and we chose $\eta = 0.3$ and $\eta = 3.0$ as the optimal parameter for XSum and CNN/DM, respectively.

Effect of the reference summary Now we turn to optimizing the scoring scheme and tune the role and weight of the reference summary, which is dictated by the hyperparameter α. Recall that up to this point, the score was pure inter-candidate ROUGE and the reference summary was not included ($\alpha = 0$).

We tune α in three additional values: 1, $N - 1$, and ∞. $\alpha = 1$ is where the weight of the reference summary is identical to the candidate summaries – each summary, including the reference summary, casts one vote. $\alpha = N - 1$ is the average of the score of the candidate summaries and that of the reference summary. That is, the average of the $N - 1$ candidate summaries gives a score, and the reference summary gives a score. The final score becomes a simple average between the two (similar to the macro-average). Finally, $\alpha = \infty$ is when the reference summary dominates the scoring. This is the same setting with BRIO. Table 4 shows the result of this experiment. Note that while $\alpha = \infty$ is the BRIO limit, the results of this is not identical to the BRIO result in Table 2 because of the different hyperparameters.

The result shows some performance improvement when the reference summary plays a role. For G-Eval consistency $\alpha = 31$ was the best for both XSum and CNN/DM. Notice that the $\alpha = \infty$ BRIO limit has the lowest G-Eval consistency score among the four models. While this is not the same BRIO model in Table 2, it demonstrates that the reference summary poses a negative role in hallucination reduction.

The other metrics showed some trends in this experiment. For example, ROUGE monotonically increased as α increased. This is a direct consequence of the fact that ROUGE here is the ROUGE with the reference. The $\alpha = \infty$ ROUGE is smaller than that of the BRIO, but this is because BRIO was optimized with the ROUGE score while BRIDO was not. For the QAFactEval, the trend is reversed and smaller α tends to have better values.

Our final best model parameters are $N = 32$, $\gamma = 50$, $\lambda = 0.01$, $\eta = 0.3$, $\alpha = 31$, difference margin for XSum; $N = 32$, $\gamma = 20$, $\lambda = 0.1$, $\eta = 3.0$, $\alpha = 31$, difference margin for CNN/DM. The G-Eval consistency scores for those models are 3.91 and 4.62, respectively. This is a 6.25% and 3.82% improvement over BRIO on XSum and CNN/DM respectively, and an 8.01% and 4.52% improvement over the base models.

Conclusion

We have introduced BRIDO, a method building upon BRIO and adding a democratic taste to its ordering. This retains the strength of BRIO, which mitigates exposure bias and teaches the model how to rank different imperfect models, andw align the model to value less hallucination. This goal is achieved by the scoring scheme which includes the inter-candidate ROUGE. We have conjectured that hallucinated content would consist the minority in a large enough candidate summaries, and therefore less hallucination would have a positive correlation with high inter-candidate ROUGE. The fact that the main objective of the model is aimed at less hallucination, and also the model results in less exposure bias both contributes to hallucination mitigation. We have shown the model play out with extensive hyperparameter search experiments and found that the hallucination measure (namely, the G-Eval consistency) has improved significantly.

While the hallucination metrics we have included in the manuscript, especially G-Eval, does measure effectively measure the degree of hallucination, human evaluation is still an important measure. The literature suggest good correlation between G-Eval and human evaluation (Liu et al.

2023b), and therefore we believe that our analysis would be in line with human evaluations as well. Regardless, directly showing the human evaluation results is worth the investigation, and we plan to study the human evaluation on BRIDO as a future work.

The BRIDO method is only applied to seq2seq models in the manuscript but a similar algorithm may be applied to, and greatly benefit from, decoder models as well. With the plethora of different decoder models, together with a combination of many different parameter sizes, the BRIDO algorithm may turn out very effective in large decoder models. Assuming that we have enough decoder models with above threshold output quality, we do not need to invoke the diverse beam search but rather use the outputs of various models to be the candidate summaries. This will ensure both the diversity and quality of the candidates, and result in a more powerful BRIDO model.

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anil, R.; Dai, A. M.; Firat, O.; Johnson, M.; Lepikhin, D.; Passos, A.; Shakeri, S.; Taropa, E.; Bailey, P.; Chen, Z.; et al. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Arivazhagan, N.; Bapna, A.; Firat, O.; Lepikhin, D.; Johnson, M.; Krikun, M.; Chen, M. X.; Cao, Y.; Foster, G.; Cherry, C.; et al. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *arXiv preprint arXiv:1907.05019*.

Bengio, S.; Vinyals, O.; Jaitly, N.; and Shazeer, N. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28.

Bertsch, A.; Xie, A.; Neubig, G.; and Gormley, M. R. 2023. It's MBR All the Way Down: Modern Generation Techniques Through the Lens of Minimum Bayes Risk. In *Proceedings of the Big Picture Workshop*, 108–122.

Chen, Y.; Wu, L.; and Zaki, M. J. 2019. Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation. In *International Conference on Learning Representations*.

Fabbri, A.; Wu, C.-S.; Liu, W.; and Xiong, C. 2022. QAFactEval: Improved QA-Based Factual Consistency Evaluation for Summarization. In Carpuat, M.; de Marneffe, M.-C.; and Meza Ruiz, I. V., eds., *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2587–2601. Seattle, United States: Association for Computational Linguistics.

Hermann, K. M.; Kočiský, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; and Blunsom, P. 2015. Teaching machines to read and comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, 1693–1701. Cambridge, MA, USA: MIT Press.

Hopkins, M.; and May, J. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 1352–1362.

Huang, L.; Yu, W.; Ma, W.; Zhong, W.; Feng, Z.; Wang, H.; Chen, Q.; Peng, W.; Feng, X.; Qin, B.; et al. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*.

Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y. J.; Madotto, A.; and Fung, P. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12): 1–38.

Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W.-t. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6769–6781.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In Jurafsky, D.; Chai, J.; Schluter, N.; and Tetreault, J., eds., *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7871–7880. Online: Association for Computational Linguistics.

Lin, C.-Y. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, 74–81. Barcelona, Spain: Association for Computational Linguistics.

Liu, Y.; Fabbri, A.; Liu, P.; Zhao, Y.; Nan, L.; Han, R.; Han, S.; Joty, S.; Wu, C.-S.; Xiong, C.; et al. 2023a. Revisiting the Gold Standard: Grounding Summarization Evaluation with Robust Human Evaluation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 4140–4170.

Liu, Y.; Iter, D.; Xu, Y.; Wang, S.; Xu, R.; and Zhu, C. 2023b. G-Eval: NLG Evaluation using Gpt-4 with Better Human Alignment. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2511–2522. Singapore: Association for Computational Linguistics.

Liu, Y.; and Liu, P. 2021. SimCLS: A Simple Framework for Contrastive Learning of Abstractive Summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 1065–1072.

Liu, Y.; Liu, P.; Radev, D.; and Neubig, G. 2022a. BRIO: Bringing Order to Abstractive Summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2890–2903.

Liu, Y.; Liu, P.; Radev, D.; and Neubig, G. 2022b. BRIO: Bringing Order to Abstractive Summarization. https://github.com/yixinL7/BRIO.

Nallapati, R.; Zhou, B.; dos Santos, C.; Gulçehre, Ç.; and Xiang, B. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In Riezler, S.; and Goldberg, Y., eds., *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, 280–290. Berlin, Germany: Association for Computational Linguistics.

Narayan, S.; Cohen, S. B.; and Lapata, M. 2018. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In Riloff, E.; Chiang, D.; Hockenmaier, J.; and Tsujii, J., eds., *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 1797–1807. Brussels, Belgium: Association for Computational Linguistics.

Ranzato, M.; Chopra, S.; Auli, M.; and Zaremba, W. 2016. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016*.

Rawte, V.; Sheth, A.; and Das, A. 2023. A survey of hallucination in large foundation models. *arXiv preprint arXiv:2309.05922*.

Tonmoy, S.; Zaman, S.; Jain, V.; Rani, A.; Rawte, V.; Chadha, A.; and Das, A. 2024. A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv preprint arXiv:2401.01313*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Vijayakumar, A. K.; Cogswell, M.; Selvaraju, R. R.; Sun, Q.; Lee, S.; Crandall, D. J.; and Batra, D. 2016. Diverse Beam Search: Decoding Diverse Solutions from Neural Sequence Models. *ArXiv*, abs/1610.02424.

Wang, C.; Liu, Y.; and Lam, W. 2023. Progressive Translation: Improving Domain Robustness of Neural Machine Translation with Intermediate Sequences. arXiv:2305.09154.

Wang, C.; and Sennrich, R. 2020. On Exposure Bias, Hallucination and Domain Shift in Neural Machine Translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 3544–3552.

Welleck, S.; Kulikov, I.; Roller, S.; Dinan, E.; Cho, K.; and Weston, J. 2019. Neural Text Generation With Unlikelihood Training. In *International Conference on Learning Representations*.

Zhang, J.; Zhao, Y.; Saleh, M.; and Liu, P. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, 11328–11339. PMLR.

Zhang, Y.; Li, Y.; Cui, L.; Cai, D.; Liu, L.; Fu, T.; Huang, X.; Zhao, E.; Zhang, Y.; Chen, Y.; et al. 2023. Siren's song in the AI ocean: a survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*.

Zhong, M.; Liu, P.; Chen, Y.; Wang, D.; Qiu, X.; and Huang, X.-J. 2020. Extractive Summarization as Text Matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 6197–6208.

Zhu, F.; Lei, W.; Wang, C.; Zheng, J.; Poria, S.; and Chua, T.-S. 2021. Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv preprint arXiv:2101.00774*.

Sample Prompt for G-Eval

Below is a sample prompt for G-Eval used for the summarization task.

You will be given one summary written for a source text.

Your task is to rate the summary in four metrics.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Coherence (1-5) - the collective quality of all sentences. We align this dimension with the DUC quality question of structure and coherence whereby "the summary should be well-structured and well-organized. The summary should not just be a heap of related information, but should build from sentence to a coherent body of information about a topic."

Consistency (1-5) - the factual alignment between the summary and the summarized source. A factually consistent summary contains only statements that are entailed by the source document. Annotators were also asked to penalize summaries that contained hallucinated facts.

Fluency (1-3): the quality of the summary in terms of grammar, spelling, punctuation, word choice, and sentence structure.

Relevance (1-5) - selection of important content from the source. The summary should include only important information from the source document. Annotators were instructed to penalize summaries which contained redundancies and excess information.

Evaluation Steps:

1. Read the source text carefully and identify the main topic, main facts and details.

2. Read the summary and compare it to the news article.

3. Assign a score based on the Evaluation Criteria.

Source Text:
{{Document}}

Summary:
{{Summary}}

Evaluation Form (JSON only):
{“coherence”: 0, “consistency”: 0, “fluency”: 0, “relevance”: 0}

Code for Reference

The code is built upon BRIO's code (Liu et al. 2022b).

The following code is the `build_diverse_beam` function to be replaced in `preprocess.py` file to calculate various scores for the generated candidates.

```
def build_diverse_beam(input):
    (
        src_line,
        tgt_line,
        cands,
        src_line_untok,
        tgt_line_untok,
        cands_untok,
        tgt_dir,
        dataset,
        ranking_metric,
    ) = input
    cands = [sent_tokenize(x) for x in cands]
    abstract = sent_tokenize(tgt_line)
    _abstract = "\n".join(abstract)
    article = sent_tokenize(src_line)

    cache_initialized = False
    cache = None

    if dataset == "xsum":
        def joint_metric(score):
            return (
                2
                * score["rouge1"].fmeasure
                * score["rouge2"].fmeasure
                / (score["rouge1"].fmeasure + score["rouge2"].fmeasure)
                if (score["rouge1"].fmeasure + score["rouge2"].fmeasure)
                else 0
            )

    else:
        def joint_metric(score):
            return (
                score["rouge1"].fmeasure
                + score["rouge2"].fmeasure
                + score["rougeLsum"].fmeasure
            ) / 3

    if ranking_metric == "":
        def compute_ranking_metric(hyp, *args, **kwargs):
            score = all_scorer.score(_abstract, "\n".join(hyp))
            return joint_metric(score)

    elif ranking_metric == "intra_rouge":
        def compute_ranking_metric(
            hyp, index: int, all_hyps: List[str], *args, **kwargs
        ):
            nonlocal cache, cache_initialized
            if not cache_initialized:
                cache = np.zeros((len(cands), len(cands)), dtype=np.float32) - 1
                for idx1, hyp1 in enumerate(all_hyps):
                    for idx2, hyp2 in enumerate(all_hyps):
                        if idx1 == idx2:
                            continue
                        score = all_scorer.score("\n".join(hyp2), "\n".join(hyp1))
                        cache[idx1, idx2] = joint_metric(score)
                cache_initialized = True
            return float(cache[index][np.arange(len(cands)) != index].mean())

```

```
    elif ranking_metric == "intra_rouge_plus_ref":
        def compute_ranking_metric(
            hyp, index: int, all_hyps: List[str], *args, **kwargs
        ):
            all_hyps = all_hyps + [abstract]
            nonlocal cache, cache_initialized
            if not cache_initialized:
                cache = np.zeros((len(all_hyps), len(all_hyps)), dtype=np.float32) - 1
                for idx1, hyp1 in enumerate(all_hyps):
                    for idx2, hyp2 in enumerate(all_hyps):
                        if idx1 == idx2:
                            continue
                        score = all_scorer.score("\n".join(hyp2), "\n".join(hyp1))
                        cache[idx1, idx2] = joint_metric(score)
                cache_initialized = True
            return float(cache[index][np.arange(len(all_hyps)) != index].mean())

    elif ranking_metric == "intra_rouge_plus_ref_weighted":
        def compute_ranking_metric(
            hyp, index: int, all_hyps: List[str], weights: List[float], *args, **kwargs
        ):
            all_hyps = all_hyps + [abstract]
            weights_sum = sum(weights)
            weights = np.array([w / weights_sum for w in weights])
            nonlocal cache, cache_initialized
            if not cache_initialized:
                cache = np.zeros((len(all_hyps), len(all_hyps)), dtype=np.float32) - 1
                for idx1, hyp1 in enumerate(all_hyps):
                    for idx2, hyp2 in enumerate(all_hyps):
                        if idx1 == idx2:
                            continue
                        score = all_scorer.score("\n".join(hyp2), "\n".join(hyp1))
                        cache[idx1, idx2] = joint_metric(score)
                cache_initialized = True
            return float(
                ((cache[index] * weights)[np.arange(len(all_hyps)) != index]).mean()
            )

    weights = [1.0 for c in cands] + [float(len(cands) - 1)]
    candidates = [
        (x, compute_ranking_metric(x, idx, cands, weights))
        for idx, x in enumerate(cands)
    ]
    cands_untok = [sent_tokenize(x) for x in cands_untok]
    abstract_untok = sent_tokenize(tgt_line_untok)
    article_untok = sent_tokenize(src_line_untok)
    candidates_untok = [
        (cands_untok[i], candidates[i][1]) for i in range(len(candidates))
    ]
    output = {
        "article": article,
        "abstract": abstract,
        "candidates": candidates,
        "article_untok": article_untok,
        "abstract_untok": abstract_untok,
        "candidates_untok": candidates_untok,
    }
    with open(tgt_dir, "w") as f:
        json.dump(output, f)
```

The following code shows the changes made for using score difference in contrastive loss function. This functions is from the file `model.py`.

```python
def RankingLoss(
    score,
    summary_score=None,
    margin=0,
    gold_margin=0,
    gold_weight=1,
    no_gold=False,
    no_cand=False,
    use_ranking_score_as_margin=False,
    ranking_scores=None,
):
    ones = torch.ones_like(score)
    loss_func = torch.nn.MarginRankingLoss(0.0)
    TotalLoss = loss_func(score, score, ones)
    # candidate loss
    n = score.size(1)
    if not no_cand:
        for i in range(1, n):
            pos_score = score[:, :-i]
            neg_score = score[:, i:]
            pos_score = pos_score.contiguous().view(-1)
            neg_score = neg_score.contiguous().view(-1)
            ones = torch.ones_like(pos_score)
            if use_ranking_score_as_margin:
                pos_ranking_scores = ranking_scores[:, :-i]
                neg_ranking_scores = ranking_scores[:, i:]
                pos_ranking_scores = pos_ranking_scores.contiguous().view(-1)
                neg_ranking_scores = neg_ranking_scores.contiguous().view(-1)
                loss = torch.clamp(
                    -(ones * (pos_score - neg_score))
                    + (pos_ranking_scores - neg_ranking_scores) * margin,
                    0.0,
                ).mean()
            else:
                loss_func = torch.nn.MarginRankingLoss(margin * i)
                loss = loss_func(pos_score, neg_score, ones)
            TotalLoss += loss
    if no_gold:
        return TotalLoss
    # gold summary loss
    pos_score = summary_score.unsqueeze(-1).expand_as(score)
    neg_score = score
    pos_score = pos_score.contiguous().view(-1)
    neg_score = neg_score.contiguous().view(-1)
    ones = torch.ones_like(pos_score)
    loss_func = torch.nn.MarginRankingLoss(gold_margin)
    TotalLoss += gold_weight * loss_func(pos_score, neg_score, ones)
    return TotalLoss
```