Local Temperature Beam Search: Avoid Neural Text *De*Generation via Enhanced Calibration

Dongkyu Lee[1,2*] **Gyeonghun Kim**[2] **Janghoon Han**[2] **Taesuk Hong**[2]
Yi-Reun Kim[2] **Stanley Jungkyu Choi**[2] **Nevin L. Zhang**[1]
[1]Department of Computer Science and Engineering, HKUST
[2]Language Lab, LG AI Research
[1]{dleear, lzhang}@cse.ust.hk
[2]{ghkayne.kim, janghoon.han, lino.taesuk,
yireun.kim, stanleyjk.choi}@lgresearch.ai

Abstract

Previous studies have constantly observed that a language model repeats itself, creating repetitions in an output sequence. To cope with the issue, stochastic decoding schemes have been the *de facto* approaches; the strategies add randomness in inference, hence avoiding the "self-loop". However, the remedy comes at the cost of sacrificing output quality due to the randomness involved. In this work, we introduce a *deterministic* decoding scheme, **local temperature beam search**. This inference algorithm is an *embarrassingly* simple variant of beam search, yet it reduces repetition, whose level is superior to that of a sampling-based decoding algorithm, while maintaining the level of coherence as in beam search. Our idea is rooted in the concept of model calibration; we view a repetition as a casualty from overconfidence in a model. Therefore, our work mitigates the miscalibration present in the course of inference with a post-calibration approach applied in beam-specific manner. Our inference scheme is validated on text completion tasks, in which the repetition problem is seen most clearly, and is exhaustively compared with existing inference schemes.

1 Introduction

Neural language models have gained much attention with ground-breaking performances (Vaswani et al., 2017; Lewis et al., 2020), and accordingly, decoding algorithms have been studied extensively along with such models (Holtzman et al., 2020; Kool et al., 2019; Meister et al., 2021; Fan et al., 2018). An inference algorithm aims to find an optimal hypothesis from a search space, which the level of optimum is commonly approximated and mapped by a language model. A choice of decoding/search algorithm can result in significant differences in model outputs, such as in diversity and coherence (Ippolito et al., 2019). For this reason, a large body of research has been done to find an optimal search algorithm (Cho, 2016; Ippolito et al., 2019; Meister et al., 2022; Holtzman et al., 2020; Fan et al., 2018).

In a broad view, there are two branches in the inference algorithm: deterministic and stochastic. Deterministic branch includes greedy decoding and beam search which are maximization-based inference strategies; the methods select tokens that maximize a sequence probability, hence generating coherent and high quality sequences. However, it has been constantly reported that the maximization-based schemes generate highly repetitive outputs (Welleck et al., 2020; Fu et al., 2021). Therefore, stochastic decoding algorithms, such as top-p (Holtzman et al., 2020) and top-k (Fan et al., 2018), have been the *de facto* options in an environment where a language model is likely to repeat itself, such as text completion. However, with the randomness introduced, the stochastic methods are at risk of generating incoherent sequence (Holtzman et al., 2020).

In this work, we view the repetition problem of a language model from the standpoint of model calibration. Our intuition is rooted in two interesting findings: 1) a language model assigns high probabilities to repeating tokens (Holtzman et al., 2020), but 2) human texts hardly contain repetition within a sequence (Paulus et al., 2017). In summary, we hypothesize that *a language model is* ***overconfident*** *when repeating itself, assigning spuriously high predictive scores to predictions that are less likely to appear.* The calibration of a language model is of importance especially in beam search (Müller et al., 2019); beam search keeps only a finite number of "likely" beams, whose likelihood is the (log) probabilities mapped by a model in inference. Therefore, when a probability in a beam is overestimated due to overconfidence, *the search will be biased towards the overconfident*

*This work was done while Dongkyu was an intern at LG AI Research

Findings of the Association for Computational Linguistics: ACL 2023, pages 9903–9915
July 9-14, 2023

beam, leading to the degeneration in text, i.e. repetition.

In this light, we propose *local temperature beam search* which is a *deterministic* algorithm that mitigates the long-standing repetition problem in a deterministic search algorithm while fully enjoying the strengths of a maximization-based approach. We mitigate the bias in beams, caused by overconfidence, by introducing local temperature scaling, in which a temperature value is decided in a context-specific manner. Accordingly, the repetition problem of a language model with our decoding scheme is reduced as much as a sampling-based scheme, whereas coherence score of generated outputs is as high as that of a maximization-based strategy. We attribute such improvement to its connection to n-gram blocking (Paulus et al., 2017); we illustrate how the proposed beam search is an *implicit* version of n-gram blocking. In this sense, the proposed idea brings more freedom in a beam search process while reducing repetition as n-gram blocking. The proposed decoding scheme is thoroughly tested in text completion, in which the stochastic decoding schemes have been dominant until now.

The contributions are as follows:

- Our work views the repetition problem of a language model from the standpoint of model calibration.
- We conduct a preliminary experiment that empirically illustrates the link between overconfidence and repetition.
- Our work bridges n-gram blocking to the proposed decoding scheme, and we attribute the success of local beam search to implicit penalty presents in our algorithm.
- The proposed beam search is robust to a wide choice of beam width and temperature.

2 Preliminaries

2.1 Neural Sequence Generation

Given a neural language model parameterized with θ, neural sequence generation is auto-regressively predicting a sequence of tokens. The following is an inference procedure at time step t.

$$\hat{y}_t = f(P(y|\hat{\boldsymbol{y}}_{<t}; \theta)) \tag{1}$$

where $P(y|\hat{\boldsymbol{y}}_{<t}; \theta)$ is the conditional probability distribution mapped by a model. f is a decoding/search algorithm, and $\hat{\boldsymbol{y}}_{<t}$ is a series of tokens preceding time step t. The inference is done in a left-to-right manner, predicting the next token given a context. In return, an output is a sequence of tokens $\hat{\boldsymbol{y}} \in V^+$, where V^+ is Kleene closure, a set of all possible strings from the vocabulary set V.

2.2 Maximization Decoding

A maximization decoding algorithm simply takes the most "*likely*" words in a context; a language model continues generating a sequence by appending tokens that maximize the sequence probability. Specifically, the log sentence probability, often referred to a score, is defined as the following:

$$\begin{aligned} s(\hat{\boldsymbol{y}}_{1:t}) &= \sum_{i=1}^{t} \log P(\hat{y}_i|\boldsymbol{y}_{<i}; \theta) \\ &= \log P(\hat{y}_t|\boldsymbol{y}_{<t}; \theta) + s(\hat{\boldsymbol{y}}_{<t}) \end{aligned} \tag{2}$$

The scoring function s intakes the inferred predictions $\hat{\boldsymbol{y}}_t$ and computes the log sentence probability by adding the log probability of the current prediction to the score of preceding inferences. Continuing a sequence with the most probable candidate is called greedy decoding. When a search space expands to multiple candidates, it becomes a beam search.

Beam search is a type of Breadth First Search that expands and keeps only B probable beams in the course of inference. The selection of B beams is based on their scores as follows.

$$\hat{Y}_t = \underset{\substack{[\hat{\boldsymbol{y}}_b:v] \\ |\hat{Y}_t|=B}}{\arg\max} \{s([\hat{\boldsymbol{y}}_b : v])|v \in V, \hat{\boldsymbol{y}}_b \in \hat{Y}_{t-1}\} \tag{3}$$

$[:]$ denotes concatenation. $\hat{Y}_t$ indicates a set of hypotheses/beams at time step t and the size of the set is the beam width B.

The maximization inference schemes excel in generating a coherent text and the application has been widely witnessed (Vaswani et al., 2017; Lee et al., 2021). However, it comes at the expense of diversity. Numerous studies have reported that a sequence generated with an algorithm that maximizes the likelihood tends to contain repetitive phrases (Fu et al., 2021; Holtzman et al., 2020).

2.3 Calibration

A model calibration indicates the trustworthiness of a model prediction (Guo et al., 2017); a well-calibrated model makes predictions whose predic-

tive scores match accuracy. Formally,

$$P(\hat{Y} = Y|\hat{P} = p) = p \tag{4}$$

where $\hat{Y}$ and $\hat{P}$ are predictions and probability assigned when making the prediction. Therefore, for instance, when a calibrated model makes a prediction with 0.8 probability, the chance of the prediction to be correct is 0.8. When the predictive score is greater than the accuracy, a model is overconfident; in the opposite case, the model is underconfident in making a prediction. There have been constant efforts to enhance calibration of a model (Lee et al., 2022), temperature scaling (Guo et al., 2017) being one of the early and effective attempts.

Temperature Scaling is a post-processing calibration method (Guo et al., 2017), which scales a logit vector with a single global temperature τ.

$$P(y^i|\boldsymbol{y}_{<t}; \tau, \theta) = \frac{\exp(\boldsymbol{z}^i/\tau)}{\sum_j^{|V|} \exp(\boldsymbol{z}^j/\tau)} \tag{5}$$

$P(y^i|\boldsymbol{y}_{<t}; \tau, \theta)$ refers to a conditional probability distribution over the output space scaled with a temperature τ, and $\boldsymbol{z}$ is a logit vector. When τ is set to ∞, the resulting probability distribution becomes a uniform distribution. On the contrary, when the temperature approaches a value close to 0, the distribution is mapped to a one-hot encoding with its whole probability mass assigned to an $\arg\max$ index. With this scaling mechanism, the temperature scaling has been found to mitigate the overconfidence problem in a neural network by setting the temperature greater than 1.0 (Müller et al., 2019; Guo et al., 2017).

3 Related Work

Recent studies have introduced decoding schemes that mitigate such weakness present in the popular decoding schemes. Diverse beam search (Vijayakumar et al., 2016) adds sequence dissimilarity between the beams in the score function, expecting diversity in the final outputs of B beams. Diverse sibling beam search (Li et al., 2016) penalizes beams that have the same root, so that outputs do not overlap with each other. Delayed beam search (Massarelli et al., 2020) transits from sampling to maximization, in which the first j time steps are inferred with a sampling-based scheme and the rest of the generation is done with beam search. Stochastic beam search (Kool et al., 2019) and conditional poisson stochastic beam search (Meister et al., 2021) bring randomness to beam search by sampling predictions at each time step. Cho (2016) proposes adding noise to a hidden state of a decoder, exploring the data manifold with the noise. This noise-based decoding has been shown to mitigate the low diversity in generated text and has been referred to as Noisy Parallel Approximate Decoding (NPAD). (Keskar et al., 2019) introduces repetition penalty, in which the penalty is given to a repeating token by discounting the corresponding logit value while leaving other logit values untouched. Lastly, contrastive search (Su et al., 2022) is a recent attempt that utilizes hidden representation in order to avoid repetition. The approach noticeably reduces repetitions, yet the core weaknesses are 1) high computation costs and 2) high dependency on isotropy level of an inference model.

Another popular branch of inference strategy is sampling. Top-k (Fan et al., 2018) considers only the k most probable tokens when sampling, truncating the sampling space to k indexes. Top-p sampling considers the smallest set of indexes whose sum amounts up to the p probability. Typical sampling (Meister et al., 2022) samples a token based on the expected information content of a token. For more detailed explanation, please refer to Appendix A.

4 Approach

4.1 Language Models Make Repetitions with Overconfidence

We make a connection between repetition and overconfidence with a preliminary experiment; we find that a language model is largely overconfident when making a repetition. We draw such observations by comparing predictive scores when generating n-gram repetitions and the probability of n-gram repetitions to appear in human-written text.

We compute the average predictive score of a language model when generating a unigram, bigram, trigram, and quadgram repetition in the course of text generation. For the probability of repetitions appearing, we compute the *probability* of unigram, bigram, trigram, and quadgram repetition in human-generated text. We compare the predictive scores and probability of such repetitions to appear, which the result is depicted in Table 1.

It is clear that *a language model assigns spuriously high probability mass, around 0.9, when making a repetition, even though there is only a*

Table 1: Preliminary experiment result on WebText corpus. The n-gram repetitions, dubbed as n-gram-rep hereafter, and the corresponding predictive scores, $P(\hat{y};\theta)$, are obtained from predictions made with beam search, with beam size set to 10. The likelihood of n-gram repetition appearing, $P(n\text{-gram-rep})$, is computed from ground truth, which is human-written text.

n-gram	n-gram-rep	$P(\hat{y};\theta)$	$P(n\text{-gram-rep})$
1-gram	0.66	0.89	0.35
2-gram	0.60	0.90	0.07
3-gram	0.58	0.91	0.03
4-gram	0.57	0.92	0.02

slim chance that such repetition exists. This is a clear sign of overconfidence as the model illustrates a clear gap between the predictive score and the likelihood of the predictions to be valid. The problem stands out more clearly in text generation; the calibration error accumulates along the inference (Müller et al., 2019), and hence beam search will be biased towards a beam with repetitions. In this light, this work proposes reducing the long-standing repetition problem of a language model **by mitigating overconfidence of a model in the course of inference**.

4.2 Local Temperature Beam Search

We propose an *embarrassingly* simple approach in beam search that handles the miscalibration issue in the course of inference; we apply a calibration method, namely temperature scaling (Guo et al., 2017), at every time step when a beam is found to be overconfident, thereby preventing the accumulation of miscalibration and degeneration in text.

4.2.1 Progressive Local Temperature

As witnessed in Table 1, the chance of n-gram repetition appearing in human-written text *exponentially decreases* **as** n **grows**, whereas predictive score of n-gram repetition mapped by a language model *increases linearly*. Accordingly, we propose to apply temperature scaling to handle such discrepancy in model calibration with **progressive local temperature**; a beam is assigned with a high temperature which is proportional to its level of overconfidence.

$$\tau_{t,b} = \max\{\tau^1_{t,b}, \tau^2_{t,b}, \cdots, \tau^n_{t,b}\}, \text{ where}$$
$$\tau^n_{t,b} = \begin{cases} \tau + \gamma * (n-1), & \text{if } \hat{y}^{max}_{t,b}{=}n\text{-gram-rep} \\ \tau, & \text{Otherwise} \end{cases} \quad (6)$$

where τ is the global temperature and γ is the temperature increasing factor. $\hat{y}^{max}_{t,b}$ is the `argmax` prediction of the probability distribution mapped by the inference model given context $\hat{\boldsymbol{y}}_{<t,b}$. Therefore, Equation 6 is validating whether the addition of $\hat{y}^{max}_{t,b}$ to $\hat{\boldsymbol{y}}_{<t,b}$ creates an n-gram repetition. If it is found to be the case of self-loop, the temperature is set according to the level of repetition. Therefore, the temperature $\tau_{t,b}$ is both **progressive and local**; the beam-specific and time step-specific temperature value is computed on-the-fly during an inference. It is worth noting that the proposed approach does not increase temperature with unigram repetitions, since a word is likely to appear multiple times in an utterance. In addition, when a generated text is free of repetition, then probability distributions over the inference remain untouched. Therefore, the proposed local temperature beam search becomes a vanilla beam search.

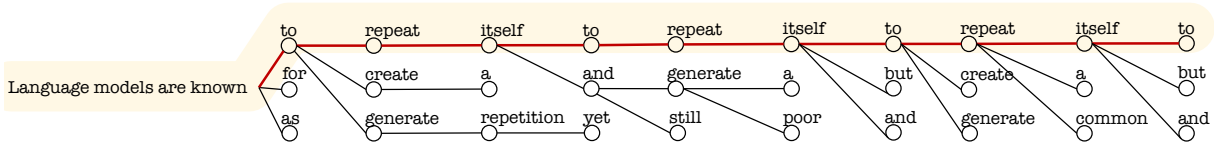
The core rationale behind the use of n-gram-based checking is to lessen model dependency within the proposed decoding scheme; recent studies, i.e. contrastive search (Su et al., 2022), utilize hidden representation to identify signs of repetitions. One major drawback is that such algorithms require a language model to have isotropic representation (Su and Collier, 2022). However, some language models, i.e. `gpt2-small` (Radford et al., 2019), are found to display anisotropic representations (Li et al., 2020; Su et al., 2022). Consequently, decoding schemes that involve hidden representations are not model-agnostic. On the contrary, by utilizing n-gram based matching logic, the proposed method alleviates the model dependency, and hence local temperature beam search can be coupled with off-the-shelf language models with less prerequisites.

4.2.2 Scaling Probability with Local Temperature

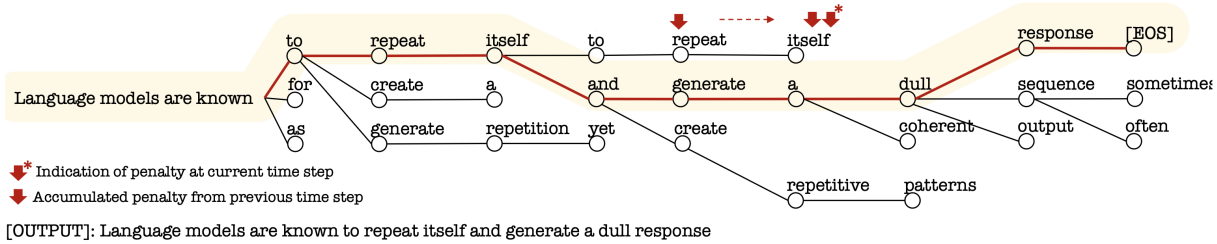
With the local temperature computed for each beam at each time step, each beam probability distribution is scaled with the local progressive temperature obtained.

$$P(y^i|\boldsymbol{y}_{<t,b};\tau_{t,b},\theta) = \frac{\exp\left(\boldsymbol{z}^i_{t,b}/\tau_{t,b}\right)}{\sum_j^{|V|}\exp\left(\boldsymbol{z}^j_{t,b}/\tau_{t,b}\right)} \quad (7)$$

We see repetition as overconfidence and set a temperature value *that surely increases entropy* of a distribution; this practice smooths the distribution, decreasing spuriously high predictive scores.



(a) Vanilla Beam Search



(b) Local Temperature Beam Search (Ours)

Figure 1: We illustrate inference with (a) vanilla beam search and (b) the proposed local temperature beam search in a text completion task with a prefix "*language models are known*". A vanilla beam search generates a sequence with a self-overlap pattern, such as "*to repeat itself*". On the contrary, local temperature beam search avoids degeneration in text as the mechanism penalizes a beam with a sign of repetition, and the penalty is accumulated to subsequent time steps. Therefore, the output inferred with the proposed decoding scheme is free of repetition, yet in high quality.

There exists two core differences between our temperature scaling and the vanilla temperature scaling (Guo et al., 2017), the first being how a temperature value is chosen. In (Guo et al., 2017), a temperature is a learned parameter on validation dataset, as a model can be both overconfident or underconfident. However, we remove the learning process, as repetition indicates overconfidence, thereby leaving the τ as a hyperparameter with the condition, $\tau \geq 1.0$. The second is the use of beam-specific progressive temperature. The proposed local temperature scaling has n number of multiple temperature options for each beam, $\{\tau, \tau+\gamma, \cdots, \tau+\gamma*(n-1)\}$, while vanilla temperature scaling maintains a single global temperature τ.

4.3 Connection to n-gram Blocking

Local temperature τ_b contributes more than simply smoothing a probability distribution; mitigating overconfidence of a beam with τ_b is *equivalent to penalizing the overall score of a beam*. We connect this aspect to n-gram blocking and illustrate how our approach is an *implicit* version of n-gram blocking.

Let there be a repetitive n-gram at time step t in a beam b'. We describe the score of the beam in n-gram blocking (Equation 8) and in our approach (Equation 9):

$$s(\hat{\boldsymbol{y}}_{1:t,b'}) = s(\hat{\boldsymbol{y}}_{<t,b'}) - \infty \text{ (Explicit Penalty)} \quad (8)$$

$$s(\hat{\boldsymbol{y}}_{1:t,b'}) = \begin{cases} \text{(1) If } n\text{-gram-rep, (Implicit Penalty)} \\ \quad s(\hat{\boldsymbol{y}}_{<t,b'}) \\ \qquad + \log P(\hat{y}_{t,b'} | \boldsymbol{y}_{<t,b'}; \tau_{t,b'}, \theta) \\ \text{(2) Otherwise, (Zero Penalty)} \\ \quad s(\hat{\boldsymbol{y}}_{<t,b'}) \\ \qquad + \log P(\hat{y}_{t,b'} | \boldsymbol{y}_{<t,b'}; \tau, \theta) \end{cases} \quad (9)$$

The beam with repeating n-gram under n-gram blocking is *explicitly* penalized as the score of the beam is assigned with $-\infty$. Therefore, the beam is immediately removed from the search boundary due to the $\arg\max$ operations in Equation 3. On the other hand, local temperature beam search computes scores with a reduced amount that is proportional to the τ_b when the beam is found to contain repetitions; the beam is **implicitly penalized**, which the discount amount is $\log P(\hat{y}_{t,b} | \boldsymbol{y}_{<t,b}; \tau, \theta) - \log P(\hat{y}_{t,b} | \boldsymbol{y}_{<t,b}; \tau_b, \theta)$ from the original score $s(\hat{\boldsymbol{y}}_{1:t,b})$.

The core driving force of the proposed approach is not just implicit penalty, but the **accumulation of the penalty in the beam throughout the rest of the inference**. As seen in Equation 2, the scoring

Table 2: Quantitative evaluation on Webtext and Wikitext-103 test dataset. Following (Holtzman et al., 2020), in repetition-related metrics, bold numbers indicate scores that are the closest to those of ground-truth. In Sim-CSE and G-score, a bold number indicates best performance.

Dataset	Model	Algorithm	rep-2	rep-3	rep-4	diversity	PPL	Sim-CSE	G-Score
Webtext	GPT2-Small	Human	7.77	2.93	1.81	87.9	-	-	-
		Greedy	54.48	51.52	49.68	11.1	1.82	55.2	24.7
		Beam Search	60.02	58.04	56.67	7.3	1.38	**60.3**	20.9
		Top-p	10.04	5.61	4.095	81.4	4.31	56.4	67.7
		Contrastive	50.97	47.59	45.45	1.4	1.91	56.6	28.1
		Ours	**6.18**	**2.46**	**1.53**	**90.1**	2.56	54.5	**70.1**
	GPT2-Medium	Human	7.77	2.93	1.81	87.9	-	-	-
		Greedy	46.11	42.30	40.01	18.6	1.9	57.3	32.6
		Beam Search	51.98	49.48	47.84	12.7	1.45	**61.6**	27.9
		Top-p	**8.41**	4.34	**3.0**	**85**	4.1	58.0	70.2
		Contrastive	41.86	37.58	35.0	23.6	2.1	58.96	37.3
		Ours	4.827	**1.52**	0.526	93.2	2.6	57.1	**72.4**
	GPT2-Large	Human	7.77	2.93	1.81	87.9	-	-	-
		Greedy	44.04	39.99	37.61	20.9	2.1	57.5	34.7
		Beam Search	49.3	46.6	44.8	14.9	1.4	**61.7**	30.3
		Top-p	9.665	5.24	3.765	82.4	3.5	58.3	69.3
		Contrastive	4.7	**2.1**	**1.4**	**91.9**	3.69	55.9	71.7
		Ours	**5.5**	1.85	0.9	**91.9**	2.0	57.1	**72.4**
Wikitext-103	GPT2-Wiki	Human	3.6	0.85	0.28	95.3	-	-	-
		Greedy	47.63	41.41	37.44	19.2	3.68	54.8	32.4
		Beam Search	52.97	47.70	44.20	13.7	2.26	50.3	26.2
		Top-p	12.52	5.73	3.27	79.8	5.41	**56.5**	67.2
		Contrastive	9.41	4.22	2.75	84.4	2.90	53.3	67.1
		Ours	**5.91**	**1.13**	**0.30**	**92.8**	3.11	50.5	**68.45**

function has the shape of recursion; a score of a previous time step accumulates to following time steps. This indicates that a penalty in the proposed approach remains present in a beam along future decoding. Therefore, the beam with the penalty is at risk of dropping out from the beam search boundary. For instance, in Figure 1b, the second occurrence of the word "itself" receives a penalty not only within that time step but also from the previous time step as it is found to be a bi-gram repetition. Therefore, with such penalty, the beam is dropped from the beam search group, and thus other beams are explored in the inference.

5 Experiment

5.1 Dataset

The efficacy of the proposed decoding scheme is tested in text completion tasks, in which the self repetition problem has been widely witnessed (Holtzman et al., 2020). Given a prefix, a language model coupled with an inference algorithm generates a sequence of tokens conditioned on the prefix. We conduct experiments on the popular Wikitext-103 dataset (Merity et al., 2016) and WebText (Radford et al., 2019). The test datasets are composed of 2.2k sentences for Wikitext-103 and 5k for WebText[1].

5.2 Experiment Details

For the model used in experiments, we have utilized pretrained language models available at `transformers` by huggingface (Wolf et al., 2020a). To be specific, we have conducted experiments with GPT2-small, GPT-medium, GPT-large, and GPT2-small finetuned on Wikitext-103. Links to the model checkpoints are listed in Appendix D Given the prefix length of 32, a language model generates a maximum number of 128 tokens that follows the prefix.

For all of the experiments conducted, we set the global temperature to 1.0. For Top-p sampling, p is set to 0.8, and for contrastive search, k and α are set to 4 and 0.6 respectively. For beam search and ours, the beam width is set to 10, otherwise explicitly mentioned. For the hyperparameter n within the proposed approach, we set the value to 4, hence checking repetitions from bigram to quadgram. For the temperature increasing factor

[1] https://s3.amazonaws.com/research.metamind.io/wikitext/wikitext-103-v1.zip

γ, we set the value to 0.3 for WebText and 0.5 for Wikitext-103.

For evaluation metrics, we report rep-n ratio, which indicates the average distinct n-gram ratio in sentences. A high rep-n indicates a sequence is filled with n-gram repetitions. In addition to the rep-n score, following (Su et al., 2022), we also report the diversity score. Aside from the repetition-related metrics, we report perplexity, Sim-CSE score for coherence, and G-Score which indicates the overall quality of the generated text both considering coherence and diversity. For more details, please refer to Appendix B.

5.3 Evaluation Result

From Table 2, it is evident that the proposed strategy has the lowest repetitions and the most similar n-gram frequencies to those of humans across multiple datasets and with different language models. Beam search and greedy decoding clearly suffer from repetition shown with high rep-n, and low diversity scores. The problem is mitigated to a certain level in the sampling-based schemes, such as top-p. However, the most noticeable gain in handling the problem is seen from the proposed method. Though our method is a deterministic algorithm, it outperforms the stochastic decoding strategies in minimizing the self-loop. Furthermore, the G-score is higher than other decoding strategies across every dataset tested. This demonstrates the well-balance of coherence and diversity in the generated text by the proposed decoding scheme. Lastly, unlike contrastive search, our approach is free from the causality brought by the anisotropic representation of language models. Despite the fact that contrastive search works well with the GPT2 large model, the decoding scheme makes trivial difference to vanilla beam search and greedy decoding when paired with the GPT2 small and medium size models. This is in sharp contrast to the results of ours, as the proposed local temperature beam search fulfills its purpose with any language model.

5.4 Analysis & Ablation Study

5.4.1 Beam Width and Temperature

It has been constantly reported that an increase in beam width leads to degeneration in generated text, making the overlap within a generated sequence bigger (Holtzman et al., 2020). Our experiment results in Table 3 further support the claim; vanilla beam search suffers significantly from increased beam width. For instance, we observe an increase in every n-rep metric with the increase in beam width. This clearly shows that vanilla beam search is vulnerable to the choice of beam width. On the contrary, our strategy is robust to a choice of beam size. In fact, we observe a drop on each repetition metric with increased beam width. Therefore, unlike vanilla beam search, our search algorithm can be equipped with a varying size of search boundary.

Furthermore, increasing the global temperature does not guarantee prevention of repetitions. Beam search with temperature set to 2.0 achieves meaningful gain in terms of reducing repetition, compared to those of beam search with temperature 1.0. However, even with the enhanced ability, the diversity score still stays around 29. This implies that simply increasing the global temperature does not mitigate the repetition problem of language models, and incorporating local temperature is necessary in preventing text degeneration in terms of repetition.

5.4.2 Progressive Temperature

The proposed method applies progressive temperature, higher temperature for beams with longer n-gram repetitions. In this ablation study, we perform non-progressive temperature setting, where any beam with n-gram repetition or longer receives the same temperature value. As demonstrated in Table 3, progressive temperature setting is one of the core aspects of local temperature beam search; when temperatures are non-progressive and shared to the beams with repetitions, we observe clear drop in diversity scores. We find that outputs with non-progressive temperatures are filled with short repetitions, as demonstrated with the increase in rep-2 scores.

5.4.3 Computation Cost

Identifying and handling overconfident beams inevitably adds computation cost, yet the cost is trivial. The proposed approach computes n-gram matching for every time step. However, the n-gram matching is simply *counting n-grams of a sentence*. Therefore, the addition of "counting" operations at each time step does not add much computation cost to the vanilla beam search.

5.4.4 Broader Application

One major drawback shared among the repetition-handling decoding schemes is the limited scope of application; the usage of the schemes have been mainly confined to open-ended text generation

Table 3: Ablation study on Wikitext-103 test dataset. Δ indicates changes in hyperparameters. τ^n indicates non-progressive local temperature. B indicates beam width and γ indicates the temperature increasing factor.

Algorithm	Δ	rep-2	rep-3	rep-4	diversity	PPL	Sim-CSE	G-Score
Beam Search	B=20	57.73	53.02	49.82	10.0	2.29	48.28	21.97
Ours	B=20	5.626	1.11	0.31	93.0	3.28	49.38	67.77
Beam Search	τ=2.0	39.84	33.23	28.59	28.7	3.53	48.32	37.24
Ours	τ^2	22.119	15.27	11.49	58.4	2.85	50.19	54.14
Ours	τ^3	10.637	3.38	1.39	85.1	2.99	50.70	65.68
Ours	τ^4	11.9	2.89	0.57	85.1	2.92	51.36	66.11

Table 4: BLEU score on four WMT-19 tasks. "EN", "RU", and "DE" denote English, Russian, and German respectively. Numbers in parenthesis (Δ) are the absolute difference compared to the BLEU score of beam search.

Dataset	Beam	Top-p	Contrastive	Ours)
EN-RU	33.4	27.5 (-5.9)	19.9 (-13.5)	**33.2 (-0.2)**
RU-EN	39.0	32.0 (-7.0)	24.3 (-14.8)	**38.9 (-0.1)**
EN-DE	42.8	34.7 (-8.1)	29.5(-13.3)	**42.6 (-0.2)**
DE-EN	41.3	33.9 (-7.4)	26.9(-14.4)	**41.2 (-0.1)**

(Holtzman et al., 2020; Su et al., 2022). On the contrary, the proposed idea is a universal decoding scheme that generalizes well across different tasks. We illustrate the superior aspect in four machine translation tasks: WMT19 EN-DE, RU-EN, DE-EN, and EN-RU (Foundation). More information regarding translation model, dataset, and inference setting can be found in Appendix C.

Table 4 depicts how previous repetition-handling strategies fail to generalize in machine translation tasks. Top-p sampling faces a severe drop in BLEU score compared to that of vanilla beam search, with the drop amounts up to 8.1. The same applies in contrastive search; BLEU score, on average across the 4 corpora tested, is down by 14 score compared to that of beam search. On the contrary, the proposed decoding strategy generates text with the same level of quality as with beam search. This empirical finding widens the scope of potential application of our inference algorithm and illustrates its superior generalization ability to the existing methods.

6 Conclusion & Future Study

In this study, we view the repetition problem of a language model as a calibration issue; a language model repeats itself as the model is overconfident in predictions. In this light, we propose a local temperature scaling in which the post-calibration method is applied only to the overconfident beams. Our local temperature beam search is a deterministic decoding strategy that excels in reducing repetition while maintaining coherence level; we attribute the success to the implicit penalty given in the course of generation. Lastly, unlike existing inference strategies, the proposed idea is robust to a variety of choice of temperature and beam width.

The objective of this paper is centered around reducing the self-overlap in a sequence, and hence the subject of local temperature scaling is chosen accordingly. However, the local temperature beam search can be utilized in tasks with different aims, as local temperature has a role of implicitly penalizing a beam. For example, a language model can be penalized with the proposed idea when the model generates a sequence with gender bias in it. We believe that ways to utilize our approach can be explored in future research.

Limitations

Since the proposed method *scales probability distribution, not shift*, the proposed idea does not change an output when coupled with a greedy decoding strategy. Greedy decoding simply takes `argmax` of probability distribution at each time step, and hence, the output with or without the local temperature scaling will not be changed within greedy. Therefore, the proposed idea is required to be utilized with a beam search, or a variant of it.

References

Kyunghyun Cho. 2016. Noisy parallel approximate decoding for conditional recurrent language model. *CoRR*, abs/1605.03835.

Nouha Dziri, Andrea Madotto, Osmar Zaïane, and Avishek Joey Bose. 2021. Neural path hunter: Reducing hallucination in dialogue systems via path

grounding. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2197–2214, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Wikimedia Foundation. Acl 2019 fourth conference on machine translation (wmt19), shared task: Machine translation of news.

Zihao Fu, Wai Lam, Anthony Man-Cho So, and Bei Shi. 2021. A theoretical analysis of the repetition problem in text generation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Daphne Ippolito, Reno Kriz, João Sedoc, Maria Kustikova, and Chris Callison-Burch. 2019. Comparison of diverse decoding methods from conditional language models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3752–3762, Florence, Italy. Association for Computational Linguistics.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

Wouter Kool, Herke van Hoof, and Max Welling. 2019. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In *International Conference on Machine Learning*.

Dongkyu Lee, Ka Chun Cheung, and Nevin L. Zhang. 2022. Adaptive label smoothing with self-knowledge in natural language generation.

Dongkyu Lee, Zhiliang Tian, Lanqing Xue, and Nevin L. Zhang. 2021. Enhancing content preservation in text style transfer using reverse attention and conditional layer normalization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 93–102, Online. Association for Computational Linguistics.

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.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. A simple, fast diverse decoding algorithm for neural generation. *CoRR*, abs/1611.08562.

Luca Massarelli, Fabio Petroni, Aleksandra Piktus, Myle Ott, Tim Rocktäschel, Vassilis Plachouras, Fabrizio Silvestri, and Sebastian Riedel. 2020. How decoding strategies affect the verifiability of generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 223–235, Online. Association for Computational Linguistics.

Clara Meister, Afra Amini, Tim Vieira, and Ryan Cotterell. 2021. Conditional Poisson stochastic beams. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 664–681, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2022. Typical decoding for natural language generation. *CoRR*, abs/2202.00666.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *CoRR*, abs/1705.04304.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Yixuan Su and Nigel Collier. 2022. Contrastive search is what you need for neural text generation.

Yixuan Su, Tian Lan, Yan Wang, Dani Yogatama, Lingpeng Kong, and Nigel Collier. 2022. A contrastive framework for neural text generation. In *Advances in Neural Information Processing Systems*.

Zhiliang Tian, Wei Bi, Dongkyu Lee, Lanqing Xue, Yiping Song, Xiaojiang Liu, and Nevin L. Zhang. 2020. Response-anticipated memory for on-demand knowledge integration in response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 650–659, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ashwin K. Vijayakumar, Michael Cogswell, Ramprasaath R. Selvaraju, Qing Sun, Stefan Lee, David J. Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *CoRR*, abs/1610.02424.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*.

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. 2020a. 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.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. Rush. 2020b. 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.

A Sampling Decoding

Different from the deterministic approaches, a sampling generation scheme requires a stochastic process; a prediction is done by sampling a token from a predicted categorical distribution, *exploring a search space*.

$$\hat{y}_t \sim P(\tilde{V}|\boldsymbol{c}_t;\theta) \tag{10}$$

where $\tilde{V} \subseteq V$. When $\tilde{V} = V$, it is referred to as pure sampling. This practice, however, introduces high randomness in inference as the sampling space is large (i.e. $|V|$=32k in WMT14 (Vaswani et al., 2017)), and thus variants have been introduced; a sampling space is truncated to a certain subset of the space. Top-k sampling (Fan et al., 2018) limits the sampling space to top k probable indexes. However, considering a fixed number of candidates is found to be suboptimal, such as when dealing with a flat probability distribution (Holtzman et al., 2020). Therefore, top-p sampling (nucleus sampling) (Holtzman et al., 2020) is proposed, where the sampling space is truncated to the smallest set, such that the sum of their probabilities is equal or greater than a pre-defined p. Due to its flexibility, top-p is known to perform well on varying shapes of distributions.

With randomness injected, the sampling-based methods have been utilized in a diversity-promoting environment, such as in dialogue (Tian et al., 2020), and story generation (Fan et al., 2018). However, the diversity comes at a price; it has been reported that the stochastic decoding algorithms are positively correlated with an increase in hallucination (Dziri et al., 2021). Furthermore, (Ippolito et al., 2019) witnesses a trade-off between diversity and quality in such algorithms.

B Metric

Sim-CSE We utilize sim-CSE sentence embedding model (Gao et al., 2021) to measure the coherency between prefix and generated text; prefix and generated text are both fed to sim-CSE model, and cosine similarity is computed between their sentence embeddings.

G-Score In this paper, we compute the geometric mean between diversity and sim-CSE score. The reason is that sim-CSE score is highly correlated with repetitions. If the generated text is repeating a given prefix, the sentence representation is likely to be similar. This is empirically seen in Table 2, where beam search generates sequences with repetitions, yet the sim-CSE score is the highest among those of the baselines. Therefore, we compute the geometric mean between diversity and coherence scores, so that G-score indicates the overall quality of the generated text.

C Translation

We report the datasets used on machine translation tasks. For inference models, we utilize pretrained models available on huggingface, which the urls are provided in the following section. The number of test instances is around 2k for all four machine translation datasets.

D Model Checkpoints

We list out the urls to the model checkpoints used in the experiments.
GPT2-Small: https://huggingface.co/gpt2
GPT2-Medium: https://huggingface.co/gpt2-medium
GPT2-Large: https://huggingface.co/gpt2-large
GPT2-Finetuned on Wikitext-103: https://huggingface.co/neulab/gpt2-finetuned-wikitext103
EN-DE Translation Model: https://huggingface.co/facebook/wmt19-en-de
DE-EN Translation Model: https://huggingface.co/facebook/wmt19-de-en
EN-RU Translation Model: https://huggingface.co/facebook/wmt19-en-ru
RU-EN Translation Model: https://huggingface.co/facebook/wmt19-ru-en

E n-gram blocking

n-gram blocking is a simple, yet *aggressive*, technique to handle the repetition problem in beam search. When an n-gram appears more than once in a beam, the beam is removed from the shortlist.[2] Therefore, for instance, with non-repeating n-gram set to 2, a bi-gram named entity or expression (i.e. New York) can only appear once in a sequence. Despite the uncompromising stand of the technique, it has been used for blocking repetition and is implemented in popular libraries (Ott et al., 2019; Wolf et al., 2020b; Klein et al., 2017).

[2] by setting the logit of the n-gram to $-\inf$.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
After Conclusion

☐ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and Section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☑ B1. Did you cite the creators of artifacts you used?
Section 5.1

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 5.1

C ☑ Did you run computational experiments?

Section 5.2

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 5.2

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 5.2

☐ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Not applicable. Left blank.

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Not applicable. Left blank.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.