Best-k Search Algorithm for Neural Text Generation

Jiacheng Xu Caiming Xiong Silvio Savarese Yingbo Zhou
Salesforce AI Research
{jiacheng.xu,cxiong,ssavarese,yingbo.zhou}@salesforce.com

Abstract

Modern natural language generation paradigms require a decoding strategy to obtain quality sequences out of the model. Beam search yields high-quality but low diversity outputs; stochastic approaches suffer from high variance and sometimes low quality. In this work, we propose a deterministic search algorithm balancing both quality and diversity. We first investigate the vanilla best-first search (BFS) algorithm and then propose the *best-k search* algorithm. Inspired by BFS, we greedily expand the top k nodes, instead of the first node, to boost efficiency and diversity. Upweighting recently discovered nodes accompanied by heap pruning ensures the completeness of the search procedure. Experiments on four NLG tasks show that best-k search yields more diverse and natural outputs compared to strong baselines, while our approach maintains high text quality. The proposed algorithm is parameter-free, lightweight, efficient, and easy-to-use.[1]

1 Introduction

Large-scale pre-trained language models (Devlin et al. (2019); Raffel et al. (2020); Brown et al. (2020); Nijkamp et al. (2022), *inter alia*) has significantly advanced the field of natural language generation. Despite the models' increasing capability in fluency, expressiveness and domain generalization, the generated outputs from these models are far from perfect (Gehman et al., 2020; Kryscinski et al., 2020; Fabbri et al., 2021). The decoding strategy is another crucial piece in this paradigm. If we form text generation as a search problem, decoding strategies are essentially search algorithms over the space composed by vocabulary $\mathcal{V}$. Beam search, a heuristic search algorithm, has been the go-to choice for many years. However, the generated sequences are usually repetitive because many diverse hypotheses are pruned at earlier stage of search (Eikema and Aziz, 2020). Sampling-based approaches (Fan et al., 2018; Holtzman et al., 2020) can indeed generate more diverse sequences, but they are hard to control due to their stochastic nature. Sometimes outputs are duplicate; sometimes a sampling choice breaks the whole sequence.

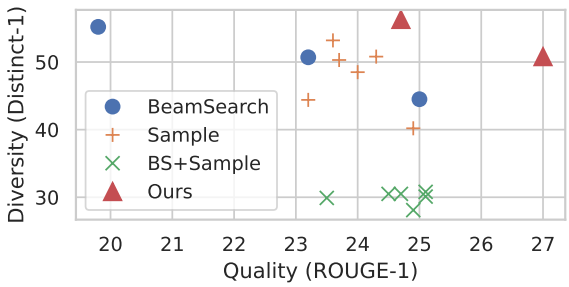

Figure 1: Generated text diversity and quality, measured by Distinctness-1(↑) and ROUGE-1(↑), on question generation. The dataset used is QuoRef and the model applied is MixQG. Our approach, best-k search with two configurations, beats baseline methods including beam search variations, sampling and BS+Sample methods on diversity and quality; see Sec. 4.4 for details.

We are looking for a decoding algorithm with high flexibility and controllability while it could also yield diverse outputs for certain use cases. We find that best-first search (BFS) algorithm satisfies these properties. First, it is a reproducible and deterministic algorithm. More importantly, since it theoretically does not prune hypotheses, it preserves a more diverse set of options and allows simultaneous expansion of hypotheses with different lengths. Despite these intriguing features, we identified two challenges, efficiency and completeness, of directly applying it to text generation.

In this work, we propose the **best-k search** for diverse and high-quality text generation. Our approach re-invents BFS with a few design changes to overcome the issues mentioned before. Parallel exploration is designed to explore the top k nodes from the search frontier each time instead of one

[1]The code implementation is available at https://jiacheng-xu.github.io/.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 12385–12401

in BFS. We also add a temporal decay mechanism to the algorithm to encourage search completions. A simple yet effective stateless scoring function as an alternative to more complicated length-adjusted counterparts is devised, and we show that it works well and helps further in finding diverse texts.

To verify the proposed algorithm, we conduct comprehensive experiments on four tasks, question generation, commonsense generation, text summarization and machine translation. Our results show that the proposed algorithm works well with a wide range of models on six datasets. Our approach yields high-fidelity, diverse and natural outputs while maintaining quality. Our contributions are (1) investigation of best-first search for text generation; (2) proposing an efficient, simple, and deterministic decoding algorithm, best-k search; (3) comprehensive experiments and strong results on six datasets with ablation study and analysis; (4) The algorithm is lightweight, easy-to-use, and compatible with any LLM. It is also orthogonal to many decoding techniques like sampling or rollout.

2 Revisiting Best-First Search

In this section, we will introduce the vanilla best-first search in the context of natural language generation as a decoding algorithm, and cover the first Research Question: **Is BFS adequate in searching hypotheses in text generation?**

Setup Text generation can be formulated as a sequence generation process given input $\mathbf{x}$ and a probabilistic language model[2] parameterized by θ.

$$p_\theta(\mathbf{y}|\mathbf{x};\theta) = \prod_{t=1}^{T} p_\theta(y_t|\mathbf{y}_{<t},\mathbf{x})$$

Traditionally, maximum *a posteriori* (MAP) decoding strategy is deployed to elicit highest-scoring output sequences $\arg\max_{\mathbf{y}^*} p_\theta(\mathbf{y}^*|\mathbf{x})$. Most previous work uses the log-likelihood of the sequence as the proxy for assessing the (partial) sequence quality. However, recent studies found discrepancies between model likelihood and quality assessed by humans (Stahlberg and Byrne, 2019; Holtzman et al., 2020; Eikema and Aziz, 2020; Zhang et al., 2021). Various approaches including length normalization (Wu et al., 2016), quality-aware decoding (Fernandes et al., 2022), and regularized decoding (Meister et al., 2020a) have attempted to modify the objective to mitigate the gap. In this work, we adopt $h(\cdot)$ as the scoring function, and $h(\mathbf{y_{1\cdots t}})$ is the score of a hypothesis $\mathbf{y_{1\cdots t}}$.

Graph Notation We frame the derivation of sequences as the expansion of a directed search graph, where BOS is the root node and EOS nodes are the leaf nodes. Any node n, except the root node, has exactly one parent node. The *score* of each node n is defined as the score of the hypothesis starting with BOS and ending with n. $h(\cdot)$ abstracts arbitrary scoring function. Each node n can be represented as a triplet $\langle s, w, t\rangle$ where the score is $s = h(n)$, token $w \in \mathcal{V}$ is the generated token, and t is the time of discovery. A completed sequence is defined as $\hat{y} = (\text{BOS}, \cdots, \text{EOS})$, and $\hat{Y}$ consists of all completed sequences. The search frontier $\mathcal{O}$ of the graph is a priority queue.[3]

Best-First Search Best-first search (BFS) is a greedy search algorithm which explores the graph according to the scoring function $h(\cdot)$. We describe the best-first search algorithm in the context of probabilistic NLG in Algorithm 2. For each iteration, BFS finds the most promising, expands it, adds newly discovered nodes to $\mathcal{O}$, and repeats until reaching the budget. *is-complete* is the conditional function for termination. P contains completed sequences. T counts the number of explored nodes. Recent work in decoding strategies (Meister et al., 2020b; Lu et al., 2022; Xu et al., 2022) was inspired and motivated by BFS, but none of them directly adopts BFS as the decoding algorithm.

Advantages & Challenges What are the potential advantages of using BFS? BFS is a deterministic and reproducible search algorithm with low pruning and no duplication. However, the vanilla best-first search suffers from efficiency and completeness issues. We present our preliminary study and discuss these issues in Appendix A.

3 Our Approach: Best-k Search

In this section, we will introduce best-k search, a novel search algorithm inspired by the vanilla best-first search. It features a few components: (1) parallel exploration enables batch-wise exploration in the search graph; (2) temporal decay yields a higher completion rate and fewer dangling nodes; (3) heap pruning improves the time and space efficiency of our approach. We describe the algorithm

[2]Language models (LM) discussed in this paper include unconditional and conditional models, where decoding algorithms could be applied ubiquitously.

[3]We use a max-heap for notation simplicity.

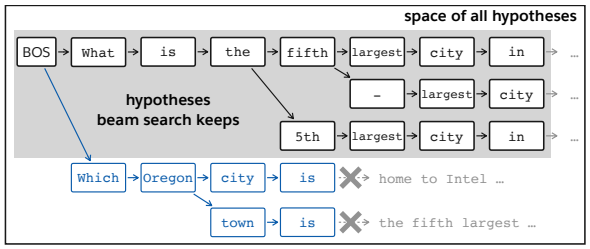

Figure 2: Pruning in beam search removes diverse hypotheses and reduces flexibility of search. This is an example of question generation and the reference contains the keyword *Intel*. Hypotheses in blue rectangle were discovered but pruned. A greedy completion of *Which Oregon city is* contains the information we want.

in Algorithm 1 and illustrate it in Figure 3.

3.1 Parallel Exploration

As suggested in Table 10, the wall clock running time of BFS is one order of magnitude slower than beam search under similar conditions. Given the same search budget, BFS is supposed to achieve similar time efficiency theoretically. However, multiple step-by-step operations are practically much slower than a batched one when GPUs are engaged. Hence, we propose a parallel exploration strategy to reduce the exploration time cost by popping k nodes from the priority queue each time and executing them in a batch. Current candidates are stored in the frontier $\mathcal{O}$. $\mathcal{PQ}$ is a priority queue after applying any scoring function to nodes in $\mathcal{O}$.

$$\mathcal{H} \leftarrow \mathcal{PQ}.\texttt{heappop(g)}$$

where $g = min(k, \mathcal{PQ}.\texttt{size()})$. The strategy serves as an approximation to best-first search as we pop the top-k most promising nodes instead of 1. This technique significantly improves the efficiency of best-k search compared to BFS, which will be discussed in Sec. 5.2.

3.2 Temporal Decay

Completion, measured by the number of outputs from the algorithm, has been another key challenge for BFS. In Table 10, increasing the search budget helps improve the completion rate but there is still a non-trivial portion of samples that fails. We propose a technique to fulfill the completition goal during the search process. For each node added to the search frontier $\mathcal{O}$, we keep the time stamp t. When we pop nodes, we modify the score of each node by adding an auxiliary score rewarding recently discovered nodes. The idea is to increase

Algorithm 1 Best-k Search with parallel exploration, heap pruning, and temporal decay.

Input: Generation model θ with vocabulary $\mathcal{V}$, search budget, $\mathcal{O}$ denotes open set (max priority queue). group size k. T is the number of explored steps; t is the time stamp.
Output: All completed paths P.

$\mathcal{O} \leftarrow \{\langle \infty, \text{BOS}, -1 \rangle\}, T \leftarrow 0, t \leftarrow 0.$
while $T <$ budget **do**
  $\mathcal{PQ} \leftarrow \emptyset$
  for $n \in \mathcal{O}$ **do**
    $\mathcal{PQ} \leftarrow \mathcal{PQ} + \langle n.\texttt{score} + \texttt{decay}(n.\texttt{time}, t), n \rangle$
  end for
  $g \leftarrow min(k, \mathcal{PQ}.\texttt{size()})$
  $\mathcal{H} \leftarrow \mathcal{PQ}.\texttt{heappop(g)}$ // $\mathcal{H}$ is the group of candidates to explore.
  $\mathcal{O} \leftarrow \mathcal{O} \setminus \mathcal{H}$
  for $\langle score, n \rangle \in \mathcal{H}$ **do**
    for $v \in \mathcal{V}$ **do**
      if is-complete$(n \circ v)$ **then**
        $P \leftarrow P \cup (n \circ v)$
        continue
      end if
      child $\leftarrow \langle h(n \circ v), v, t \rangle$ // Current time t of adding the node to $\mathcal{O}$.
      $\mathcal{O} \leftarrow \mathcal{O} \cup$ child
    end for
  end for
  $\mathcal{O} \leftarrow \mathcal{O}.\texttt{prune()}$
  $T \leftarrow T + g$
  $t \leftarrow t + 1$
end while

the score of recently discovered nodes so the algorithm prefers to continue them. The decay function needs to be monotonic. Hence, we define the decay function as a power function:

$$\texttt{decay}(n.\texttt{time}, t) = -\kappa(t - n.\texttt{time})^{\beta}$$

where $\kappa > 0$ controls the weight of the term and $\beta > 0$ controls the slope. t is the current time step and $n.\texttt{time}$ is a past time step, so $t - n.\texttt{time} > 0$. The older the node, the smaller the value of $\texttt{decay}(n.\texttt{time}, t)$. A more recent node will receive a higher incentive, so it's more likely to be popped and expanded. For example, a node discovered at $t = 1$ receives $\texttt{decay}(1, 5) = -4$ and a node discovered at $t = 4$ receives $\texttt{decay}(4, 5) = -1$, if we set $\kappa = \beta = 1$. In our experiment, we set $\beta = 0.5$ and explore different values of κ. We leave other forms of the decay function, i.e. logarithm, as future work, and discuss some design choices in Appendix F.

3.3 Heap Pruning

The size of the heap grows fast during exploration. For most of the time, however, our approach only utilizes top-ranked hypotheses. The temporal decay function is monotonic, so for any node in the

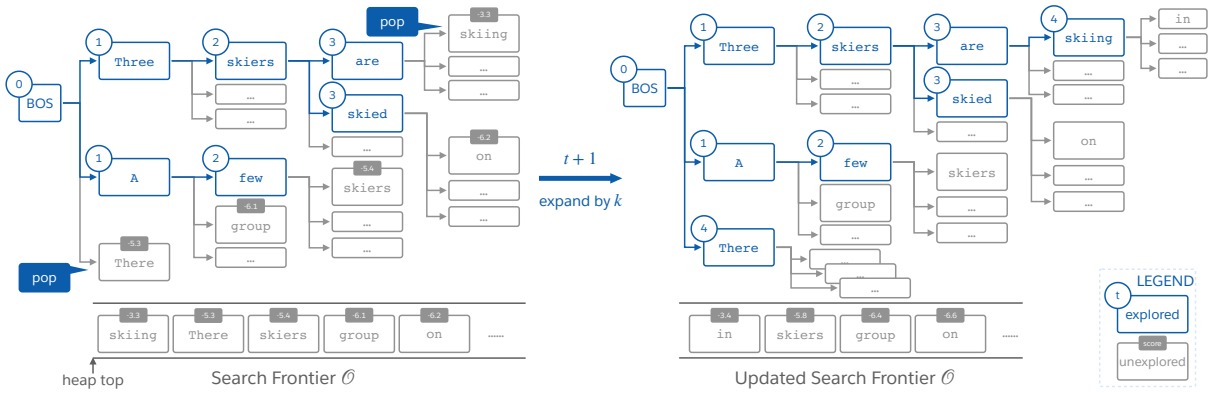

Figure 3: Illustration of best-k search with an example from CommonGen, where the input is "mountain ski skier". (left) the search graph before expansion; (right) the search graph after expansion with "skiing" and "There" expanded; (bottom) the search frontier. The upper left number of explored nodes (blue-bordered rectangle) indicates the time stamp of expansion. Grey rectangles are unexplored nodes in the frontier. For illustration purposes, we set $k = 2$, and only show the top 3 expansions for each node.

search frontier, the final score is always decreasing as the time moves forward. The usage of the temporal decay could affect the ranking, but we posit that if the margin of model score between a candidate node and the k-th highest node from the heap is larger than ϵ, it is unlikely that it will be used in future. The choice of the margin ϵ depends on factors including the intensity of temporal decay, remaining search budget, model calibration, and resource limitations. In practice, we set a sufficiently large maximum heap size to 500 to avoid tuning ϵ on different datasets. The expansion of each node could lead to $|\mathcal{V}|$ extension nodes, where $|\mathcal{V}|$ is the size of the vocabulary. As the conditional probability $p_\theta(y_t|\mathbf{y}_{<t}, \mathbf{x})$ is usually long-tailed, we discard those low-scoring nodes for efficiency. We set a threshold $\gamma = 0.05$ to filter out generations with probability lower than it.

3.4 Model Score

The depth of a BFS search graph is not aligned while the that of beam search remains the same during the search. As the scoring function plays a crucial role in finding ideal sequences $\hat{Y}$, we investigate whether existing scoring functions are still compatible with the best-k search algorithm. Here are a few common ways to define the scoring function h regarding the length l of the (partial) sequence: 1. original: $h(\mathbf{y}) = \sum_{t=0}^{l} \log p_\theta(y_t|\mathbf{y}_{<t}, \mathbf{x})$. This is the original way of defining the score of a sequence with its sequence log-likelihood. 2. length-adjusted scoring function: $h(\mathbf{y}) = \frac{1}{|\mathbf{y}|^\alpha} \sum_{t=0}^{l} \log p_\theta(y_t|\mathbf{y}_{<t}, \mathbf{x})$. The tunable hyper-parameter α controls the preference of length (Meister et al., 2020a). The hypotheses in BFS have different length so it's tricky to pick a good hyper-parameter for length-adjusted functions across samples and datasets. In this work, we also propose a memoryless scoring function $h(\mathbf{y}) = \log p_\theta(y_t|\mathbf{y}_{<t}, \mathbf{x})$. It approximates the score of the whole hypothesis $\mathbf{y}$ with the probability of the last node. It satisfies the Markov property that only the last state's probability is considered for the next continuation. When we use this scoring function together with best-k search, we term the approach as BKS_{last} . We conduct ablation studies to understand different scoring functions in Sec. 5.3. We found that the length-biased scoring function typically works the best while the memoryless function generates more diverse outputs with slightly lower quality.

4 Evaluation

4.1 Tasks, Models & Datasets

We investigate four conditional text generation tasks, ranging from more precision-oriented tasks like machine translation to more open-ended tasks like commonsense generation and question generation. MT is a use case where diverse outputs are not always required, so in Section 6 we devise our algorithm followed by reranking to see how much we can benefit from diverse and high-quality outputs. We describe the detail of the tasks, models and datasets in Appendix C.

4.2 Baselines

Beam earch (BS) is the long-standing choice for decoding sequences for decades (Reddy, 1977) and

Method	Stat S	Stat |S|	Diversity (↑) D-1	Diversity (↑) D-2	Diversity (↑) D-3	Oracle (↑) R1	Oracle (↑) R2	Oracle (↑) RL	Natural (↑) MV	Quality (↑) R1	Quality (↑) R2	Quality (↑) RL	Quality (↑) MTR	GRM
BS	10	10	44.8	48.7	46.9	32.6	12.9	30.1	59.5	25.9	9.2	23.7	20.9	88.9
DBS	10	9	52.3	52.2	47.6	30.1	9.5	26.4	41.5	24.2	7.3	21.3	18.7	85.2
DBS+	10	9	55.8	53.1	45.8	26.1	6.8	23.4	13.7	20.3	4.5	17.8	14.9	85.7
BTYP$_{0.2}$	10	1	29.9	27.8	24.3	24.2	7.0	22.0	53.5	23.5	6.7	21.4	18.3	90.8
BTYP$_{0.5}$	10	2	30.5	28.4	24.7	25.0	7.5	22.7	48.1	24.7	7.2	22.4	19.2	92.5
BTYP$_{0.95}$	10	2	30.9	28.9	25.4	26.9	8.6	24.8	61.4	25.0	7.6	23.1	19.5	92.3
BNCLS$_{0.5}$	10	1	28.1	25.0	21.1	24.9	7.0	22.7	51.2	24.9	7.0	22.8	18.2	92.3
BNCLS$_{0.8}$	10	2	30.1	28.0	24.4	25.6	7.9	23.7	49.9	25.1	7.3	23.1	19.1	92.5
BNCLS$_{0.9}$	10	2	30.8	28.7	25.2	26.0	8.2	24.0	58.6	25.1	7.5	23.0	19.3	91.2
TYP$_{0.2}$	10	5	44.4	46.2	42.4	26.5	7.3	23.9	50.9	23.2	6.3	21.1	18.1	88.3
TYP$_{0.5}$	10	7	48.5	52.0	47.9	30.1	10.6	27.3	71.0	24.0	6.5	21.9	18.0	91.8
TYP$_{0.95}$	10	9	54.3	59.4	55.7	31.2	11.2	28.5	84.3	22.1	6.1	20.1	17.1	89.5
NCLS$_{0.5}$	10	5	40.2	41.4	37.7	29.2	9.9	26.3	58.3	24.9	7.3	22.9	18.6	93.9
NCLS$_{0.8}$	10	8	50.8	55.1	51.3	30.5	10.2	27.3	47.7	24.3	6.2	21.6	18.2	91.1
NCLS$_{0.9}$	10	9	53.2	58.2	53.7	31.2	11.5	28.7	46.0	23.6	6.9	21.4	18.0	90.9
MixQG	-	-	-	-	-	-	-	-	-	24.9	8.0	22.3	-	-
BKS$_{mean}$	20	20	50.8	56.1	54.0	33.9	14.2	31.0	83.0	27.0	8.9	24.5	21.3	86.8
BKS$_{last}$	19	19	53.4	59.4	55.9	32.7	13.4	30.1	69.4	26.0	8.4	23.2	19.7	91.7

Table 1: Experiments result on QuoRef question generation. S and |S| stand for the number of sentences and the unique number of sentences. D-1, -2, and -3 stand for unigram, bigram and trigram distinctness. MV is the MAUVE score measuring the naturalness of the generated outputs. MTR is METEOR score. GRM measures the grammaticality. We highlight the best, second best, and the worst for each column. A visualized comparison with D-1 and R1 is presented in Figure 1.

diverse beam search is a diversity-promoting variant of beam search (Vijayakumar et al., 2018). We experiment with different numbers of beam groups for diverse beam search: 5 for DBS and 10 for DBS+. **Sample** is represented by two widely-adopted strong stochastic sampling methods, nucleus sampling (NCLS) (Holtzman et al., 2020) and typical sampling (TYP) (Meister et al., 2022a). **Beam sample** includes a collection of beam search multinomial sampling methods. We experiment with the integration of beam search with typical sampling and nucleus sampling, denoted as BNCLS and BTYP respectively. Implementation of baseline approaches is available at Transformers/-GenerationMixin/generate.

Ours We use two typical configurations to represent our approach: BKS$_{last}$ where the scoring function is memoryless, and BKS$_{mean}$ where $\alpha = 1$. In BKS$_{mean}$, the score of the sequence is the average log-likelihood of individual time steps. We experiment with $k = \{5, 10\}$ and the weight of temporal decay in $\{0.0, 0.01, 0.05, 0.1, 0.2\}$, and report the configuration with the best combination of diversity ($\overline{\text{D}}$) and quality ($\overline{\text{R}}$).

4.3 Metrics

We measure the generated outputs from multiple aspects including text quality, relevance, diversity, and naturalness. 1. **Statistics**: we report the number of completed strings and the number of unique completed strings as S and |S|. 2. **Diversity**: following Li et al. (2016); Yang and Klein (2021), we report the distinctness of completions, measured as the number of unique n-grams divided by the number of words, denoted as D-1, D-2 and D-3. 3. **Text quality**: we adopted two relevance based metrics, ROUGE (R1, R2, RL) (Lin, 2004) and METEOR (MTR) (Banerjee and Lavie, 2005), for assessing the surface similarity between the generated strings and the reference. 4. **Naturalness**: We measure the naturalness of the generated sequences with MAUVE (Pillutla et al., 2021), a metric for open-ended text generation.

4.4 Question Generation

For QuoRef and SQuAD, we present the experiment results in Table 1 and 2. Due to the space limit, we present the results of SQuAD in 13 in Appendix E. Our methods achieve significantly higher MAUVE score than peer methods. To visualize the *trade-off* in quality and diversity, we also visualize these two metrics in Figure 1, which shows our approach significantly surpasses all baseline methods on both diversity and text quality, measured by D-1 and R1. There is a typical trade-off curve for diversity and quality by controlling hyper-parameters (p

	|S|	$\overline{\text{D}}$	$\overline{\text{OR}}$	$\overline{\text{R}}$	Mv	Grm	Mtr
BS	10	23.0	34.2	25.1	9.6	88.1	29.8
DBS	9	26.6	32.8	23.1	13.0	82.3	27.9
DBS+	9	30.9	32.6	19.8	9.0	80.6	23.1
BTyp$_{0.2}$	1	9.7	25.6	25.1	15.6	88.4	29.7
BTyp$_{0.5}$	1	10.3	25.8	25.0	36.2	93.4	30.1
BTyp$_{0.95}$	2	11.0	28.0	26.6	11.9	89.2	31.1
BNcls$_{0.5}$	1	9.2	26.7	26.4	13.0	90.2	30.7
BNcls$_{0.8}$	2	10.5	27.5	26.5	9.3	89.6	30.9
BNcls$_{0.9}$	2	10.8	27.9	26.5	10.0	89.2	30.9
Typ$_{0.2}$	6	22.6	29.1	23.7	17.0	86.6	28.1
Typ$_{0.5}$	8	28.0	34.8	24.9	13.3	88.7	29.4
Typ$_{0.95}$	10	36.2	34.9	23.3	18.8	84.0	27.8
Ncls$_{0.5}$	5	18.7	32.1	26.5	13.8	89.8	30.8
Ncls$_{0.8}$	9	30.2	35.4	24.8	16.5	86.6	29.3
Ncls$_{0.9}$	9	33.8	35.6	24.0	16.2	86.2	28.6
BKS$_{mean}$	29	30.3	35.8	25.5	16.5	88.6	30.5
BKS$_{last}$	24	36.5	36.1	21.7	22.6	86.4	25.8

Table 2: Results of question generation on DROP. $\overline{\text{D}}$ is the average of D-1, D-2 and D-3. $\overline{\text{OR}}$ and $\overline{\text{R}}$ are the average of Oracle ROUGE and ROUGE.

	|S|	$\overline{\text{D}}$	$\overline{\text{OR}}$	$\overline{\text{R}}$	Mv	Grm	Mtr
BS	10	40.6	42.1	40.3	23.4	88.3	42.7
DBS	10	48.2	42.6	37.9	21.6	79.2	37.3
DBS+	10	54.1	42.4	36.4	15.9	77.5	35.8
BTyp$_{0.2}$	2	27.4	36.3	38.0	27.0	83.8	40.0
BTyp$_{0.5}$	2	26.7	37.7	40.4	17.1	88.9	43.0
BTyp$_{0.95}$	2	27.9	38.4	40.7	14.6	89.2	43.3
BNcls$_{0.5}$	1	24.3	37.1	40.5	11.9	87.9	43.1
BNcls$_{0.8}$	2	27.0	38.5	41.0	16.9	89.5	43.5
BNcls$_{0.9}$	2	27.4	38.5	40.9	15.4	89.6	43.6
Typ$_{0.2}$	9	55.4	40.5	34.9	37.9	79.3	37.8
Typ$_{0.5}$	10	55.0	42.4	37.1	37.8	82.2	39.3
Typ$_{0.95}$	10	61.0	39.5	33.7	41.7	74.9	36.1
Ncls$_{0.5}$	8	44.6	41.1	39.2	24.6	86.2	41.5
Ncls$_{0.8}$	10	55.7	41.7	36.5	31.7	82.0	38.6
Ncls$_{0.9}$	10	59.7	41.2	35.6	41.5	79.4	38.0
BKS$_{mean}$	27	45.7	41.4	38.2	19.2	83.8	41.0
BKS$_{last}$	22	51.4	43.3	37.6	34.6	84.7	39.3

Table 3: Results on commonsense generation.

value for nucleus sampling, group size for diverse beam search, etc.), but our approaches go beyond the established curve by a significant margin. We posit that 10x more output (2 vs. 29) and substantial gain of diversity (10.8 vs. 30.3) could unlock tons of applications and choices in many real-world applications. For example, on DROP, BNcls$_{0.9}$ achieves $\overline{\text{D}} = 10.8$ and $\overline{\text{R}} = 26.5$ while BKS$_{mean}$ comes with $\overline{\text{D}} = 30.3$ and $\overline{\text{R}} = 25.5$.

4.5 Commonsense Generation

We present the experimental result of commonsense generation in Table 3. Sampling based approaches are overall good at diversity but the quality of generated text is lower than other methods. For example, Typ$_{0.95}$ has the best average distinctness score and the worst ROUGE score at the same time. Our approach BKS$_{last}$ also has the highest oracle ROUGE score, which indicates high search quality over human annotations.

4.6 Text Summarization

As reranking text summarization system outputs has gained increasing interest, the fruitfulness and diversity of generated summaries are valuable attributes to look at. We present the result of text summarization in Table 5. Our approach remains competitive in quality, diversity, and naturalness. Our approach achieves an average ROUGE of 31.9 and MAUVE of 99.5, higher than any other methods. $\overline{\text{D}}$ of our approach is lower than sampling due to the longer sequence lengths and more dangling nodes.

5 Analysis

5.1 Examples

We show one example output of CommonGen in Table 4. We list outputs provided by Lu et al. (2022)[4] and the outputs from our experiments. The outputs from our model are more diverse since multiple types of subjects exists, including *a dog*, *the dogs*, and *two dogs*.

We also present one example from QuoRef question generation in Table 6. In this example, we can observe the duplication issue rooted in sampling based methods. Most of the generated questions from sampling are duplicate, covering the easiest question to ask. However, our approaches yield diverse and high-quality questions, covering broader spectrum of facts and knowledge like *Intel*, *Silicon Forest*, *country seat of Washington County*.

5.2 Efficiency

We test the wall-clock running time of our algorithms and the standard beam search. We follow the same configuration in Sec. 2. The result is presented in Table 7. Although our approach is still slower than beam search, due to all the overhead cost including padding sequences, scoring hypotheses and heap management, the speed is reasonable for many applications. The heap size could be

[4]The model we use is different from the ones in Lu et al. (2022), so their outputs are only for reference.

GBS / DBA / NEUROLOGIC*
G: A dog is run over by a ball and mouth agape. D: A dog is run over by a ball and bites his mouth. N: A dog running with a ball in its mouth.
NCLS$_{0.8}$ / TYP$_{0.5}$ / Ours
A dog running around with a ball in his mouth. The dog is running with a ball in his mouth. The dog runs away with the ball out of the mouth. A dog running on its mouth with a ball A dog with a ball running around his mouth.
A dog with a ball in its mouth running around the pond. A dog runs to the door, eating a ball, and another dog in the mouth. A dog running away with a ball in its mouth. A dog running with a ball in his mouth. A dog is running around its mouth catching a ball.
A dog is running around with a ball in its mouth. a dog running around with a ball in its mouth The dogs are running around with balls in their mouths. Two dogs running around in the same room with a ball in their mouths. Two dogs running with balls in their mouths.

Table 4: An example from CommonGen where the input is "ball dog mouth run". We first present the outputs on GBS, DBA, and NEUROLOGIC*, provided in Lu et al. (2022). Then we show five sample outputs from NCLS$_{0.8}$, TYP$_{0.5}$ and BKS$_{last}$, respectively.

	|S|	$\overline{\text{D}}$	$\overline{\text{OR}}$	$\overline{\text{R}}$	MV	GRM	MTR
BS	8	16.3	36.6	31.2	98.0	96.4	36.9
DBS	8	20.5	36.3	28.9	64.6	95.2	32.3
DBS+	7	21.5	35.6	27.8	22.3	92.0	29.8
BTYP$_{0.2}$	2	12.3	29.5	27.6	98.8	96.0	34.2
BTYP$_{0.5}$	3	13.4	33.0	30.4	98.2	96.3	36.5
BTYP$_{0.95}$	3	13.3	33.7	30.9	98.5	96.4	37.0
BNCLS$_{0.8}$	3	13.2	33.5	30.8	98.5	96.3	37.1
BNCLS$_{0.9}$	3	14.0	34.1	31.0	98.5	96.4	37.1
TYP$_{0.2}$	7	30.9	34.2	26.7	97.8	94.7	31.3
TYP$_{0.5}$	8	34.7	38.8	28.8	97.9	95.1	32.7
TYP$_{0.95}$	8	35.7	38.5	28.1	98.4	95.1	32.3
NCLS$_{0.8}$	8	35.3	38.8	28.7	98.1	95.1	32.9
NCLS$_{0.9}$	8	37.2	37.7	27.3	98.5	94.4	31.4
BKS$_{mean}$	22	21.4	39.0	31.9	99.5	95.3	35.9
BKS$_{last}$	17	24.3	37.5	28.9	98.5	95.7	33.3

Table 5: Results on XSum with BART-XSum.

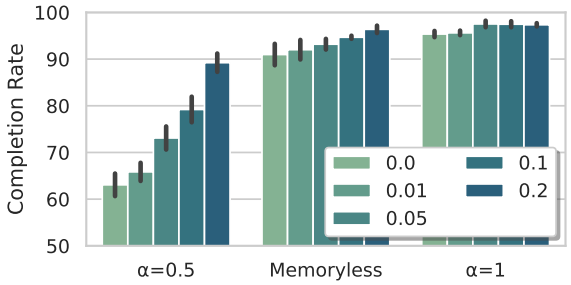

Figure 4: Evaluation of the weight term κ for temporal decay. We increase the weight κ in the objective from 0.0 (no decay) to 0.2 and evaluate how it relates to the completion rate for different scoring functions. In the case of $\alpha = 0.5$, increased weight significantly helps the completion rate.

shrunk and the heap management could be optimized for even better efficiency.

5.3 Choice of Scoring Function

In this paper, we experimented with two families of scoring functions: length-normalized sequence log-likelihood and a new memoryless greedy score. We studied how the scoring function works in practice. More particularly, we looked into whether some form of scoring function will cause significant incompletion or search failure. We present the result and discuss the choice of scoring function in Appendix B.

5.4 Effect of Temporal Decay

We evaluate how temporal decay helps the completion rate in different settings in Figure 4. As the result in Table 12 indicates a high incomplete rate when $\alpha = 0$, we only evaluate three scoring schemas, $\alpha = 0.5$, $\alpha = 1$ (BKS$_{mean}$), and the memoryless setting (BKS$_{last}$). Temporal decay helps the completion when the scoring function itself struggles with completion. For example, when $\alpha = 0.5$, increasing κ improves the completion rate from 66% to 92%.

6 Application: Reranking Diverse Outputs

Machine translation is typically considered as a precision-oriented task, where typically only a few translations are considered as correct. In this section, we would like to answer the RQ: **Do we benefit by selecting from a pool of high-quality diverse outputs, even when the task does not necessarily require such?**

Setup We use a popular machine translation dataset with multiple references (Ott et al., 2018), based on WMT'14 En-Fr and En-De test sets (Bojar et al., 2014). The model for this task is the mBART[5] model (Tang et al., 2021). In order to rerank decoded outputs, we adopt a state-of-the-art quality estimation model for MT,

[5] https://huggingface.co/facebook/mbart-large-50-many-to-many-mmt

Sampling	BKS_{last}	BKS_{mean}
NCLS$_{0.8}$	What city is the fifth largest?	What city in OR is the fifth largest in OR?
What is the fifth largest city in OR? (x5)	What city is the fifth-largest city in the State?	What city is the fifth largest [∅ / city in OR / city in the State / in OR / in the state]?
What is the fifth-largest city in OR?	What is the 5th largest city in OR?	What city is the fifth-largest in the State?
What is the fifth-largest city in OR?	What is the fifth largest city in [OR / the State of OR / the State]?	What is the fifth largest city in OR?
What is the fifth-largest city in the State of OR?	What is the fifth-largest city in the State?	What is the fifth largest city in the State?
What is the fifth-largest city in the State of OR?	Which city is the fifth largest city in OR?	Which city in OR has the largest population?
Which city in OR is the county seat of Washington County?	Which city is the fifth largest city?	Which city in OR hosts Intel?
TYP$_{0.5}$	Which is the fifth largest city?	Which city in OR is known as the Silicon Forest?
What is the fifth largest city in OR? (x5)	Which OR city is the fifth largest in the state?	Which city in OR is the fifth largest in OR?
What is the fifth-largest city in OR? (x3)	Which OR city is the fifth largest?	Which city in OR is the fifth largest in the state?
Which city in OR is the county seat of Washington County?	Which OR town is home to Intel?	Which city is the fifth largest [∅ / city in OR / city / in the state]?
Which city is the county seat of Washington County?	Which OR town is home to the tech company Intel?	Which OR city is the county seat of Washington County?
	Which OR town is known as the Silicon Forest?	Which OR city is the fifth largest in size?
	Which OR town is the fifth largest [∅ / city / city in the state / in size / in the state]?	Which OR city is the fifth largest?

Input (Ans || Context): Hillsboro || Hillsboro is the fifth-largest city in the State of Oregon and is the county seat of Washington County. Lying in the Tualatin Valley on the west side of the Portland metropolitan area, the city hosts many high-technology companies, such as Intel, that comprise what has become known as the Silicon Forest. At the 2010 Census, the city's population was 91,611.For thousands of years before the arrival of ... **Reference Question**: What city is Intel located in?

Table 6: Example on QuoRef question generation. The duplication of sampling is high while our model generates a more diverse set of questions. Some outputs from our approach cover the entity Intel mentioned in the reference. We manually replace all the occurrences of *Oregon* with *OR* and combine some hypotheses due to the layout limit.

	Best-k Search		BS	BTYP$_{0.5}$
	$k = 5$	$k = 10$	$b = 10$	$b = 10$
Time	1.8s	1.2s	0.7s	1.4s
|S|	18.2	12.8	8.3	3.0

Table 7: Efficiency comparison of our approach and beam search. Time shows the decoding time used for each example.

COMET-QE (Rei et al., 2020). The quality estimation model we use to rerank all the outputs is `wmt21-comet-qe-da`. The QE model is a referenceless model $Q(\mathbf{s}, \mathbf{t})$ which judges whether the source input $\mathbf{s}$ and the hypothesis translation $\mathbf{t}$ form a matched pair based on regression metrics.

Result We present the result on MT En-De and En-Fr in Table 8 and 11. Our approach has a huge gain after reranking and surpasses all of the sampling based methods and beam search only methods while maintaining high diversity. BNCLS$_{0.8}$, the approach with best BLEU score, is 9.4 behind BKS_{last} on $\overline{\text{D}}$ while the human annotation reference is much higher than any of the machine generated hypothesis sets. The success of overgeneration-then-reranking paradigm has been witnessed in summarization (Song et al., 2021; Ravaut et al., 2022; Pernes et al., 2022) and translation (Fernandes et al., 2022), where the proposed algorithm could be valuable in searching high-quality diverse outputs.

7 Related Works

Best-first search BFS was widely used in structural prediction (Klein and Manning, 2003), statistical MT (Och et al., 2001), and for searching hypotheses (Saha et al., 2022). Recent work in decoding strategies (Meister et al., 2020b; Lu et al., 2022; Xu et al., 2022) conceptualized best-first search as part of their paradigm, but it was not the dominant component of any of these systems.

Text decoding algorithms Stochastic decoding algorithms have gained popularity in the past few years (Fan et al., 2018; Holtzman et al., 2020; Meister et al., 2022a; Suzgun et al., 2022). Rollout-based algorithms are capable of satisfying certain utility functions or constraints at the cost of efficiency (Leblond et al., 2021; Chaffin et al., 2022; Lu et al., 2022). Recombination-based search algorithm (Xu et al., 2022) can find thousands of hypotheses despite complicatedness.

Diversity in text generation The diversity of text generation has been a key challenge for applications like dialogue (Li et al., 2016; Zhang et al.,

			BLEU		
	|S|	$\overline{\text{D}}$	ORIGIN	COMET	Δ
Reference	11	36.9	-	-	-
BS	10	15.4	30.4	**32.3**	1.9
DBS	10	18.7	25.0	27.8	**2.8**
DBS+	10	24.6	20.8	22.9	2.1
$\text{BTYP}_{0.2}$	3	11.0	26.5	26.1	-0.4
$\text{BTYP}_{0.5}$	3	10.2	34.3	34.6	0.3
$\text{BTYP}_{0.95}$	3	10.7	32.9	33.4	**0.5**
$\text{BNCLS}_{0.5}$	2	9.0	33.0	33.3	0.3
$\text{BNCLS}_{0.8}$	3	10.2	34.9	**34.9**	0.0
$\text{BNCLS}_{0.9}$	3	10.4	32.6	33.8	1.2
$\text{TYP}_{0.2}$	9	27.2	19.9	19.5	-0.3
$\text{TYP}_{0.5}$	9	28.6	25.6	27.0	1.4
$\text{TYP}_{0.95}$	10	36.5	19.2	22.1	**2.9**
$\text{NCLS}_{0.5}$	8	18.6	31.1	**32.2**	1.1
$\text{NCLS}_{0.8}$	10	30.2	25.9	27.0	1.0
$\text{NCLS}_{0.9}$	10	35.0	23.2	25.8	2.6
BKS_{mean}	35	19.6	30.1	**33.3**	3.2
BKS_{last}	33	20.5	26.1	31.1	**5.0**

Table 8: Machine translation from English to German. ORIGIN and COMET are the BLEU score before and after reranking; Δ indicates the change of BLEU score from reranking.

2020; Stasaski and Hearst, 2022), MT (Shen et al., 2019) and conditional text generation (Yang and Klein, 2021). Beam search has also been developed to generate more diverse outputs (Vijayakumar et al., 2018; Anderson et al., 2017; Post and Vilar, 2018). Prior work also studies the trade-off between diversity and quality in text generation (Zhang et al., 2021).

Degeneration of beam search Welleck et al. (2020b); Holtzman et al. (2020) addressed the degeneration issue in neural text generation and Cohen and Beck (2019) studies the beam search performance degradation in neural sequence models. The gap between high probability and quality has been observed and studied (Meister et al., 2022b; Freitag et al., 2022).

8 Conclusion

In this work, we propose best-k search, a novel decoding algorithm for text generation based on best-first search. The algorithm features a few technical components, and generates natural and diverse text while maintaining high quality. We conduct comprehensive experiments on four tasks to verify the approach. The algorithm is orthogonal to sampling methods and it is parameter-free, lightweight, and efficient.

Acknowledgements

We thank Greg Durrett, Tong Niu, Chen Xing, Hiroaki Hayashi, Katie Stasaski, Philippe Laban, Semih Yavuz and Shafiq Rayhan Joty for helpful proofreading and comments on this work. We also thank the Salesforce AI Research team for generous support and feedback.

Limitations

In this work, we propose a decoding algorithm for text generation. We present the algorithm with comprehensive discussion on design choices and mechanisms. We further verified our algorithm on four tasks and six datasets. However, we acknowledge the following limitations. First, we mainly apply the method to English data although we cover German and French in MT experiments. In future work, we could verify the approach on non-English languages, especially CJK, due to the possible gap of tokenization. Second, we did not cover open-ended generation tasks like story generation and long-form generation tasks in this paper. Third, we could conduct more experiments and analysis on the mechanism of our approach, and examine the outputs with human judgement and feedback.

References

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

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind

Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Antoine Chaffin, Vincent Claveau, and Ewa Kijak. 2022. PPL-MCTS: Constrained textual generation through discriminator-guided MCTS decoding. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2953–2967, Seattle, United States. Association for Computational Linguistics.

Eldan Cohen and Christopher Beck. 2019. Empirical analysis of beam search performance degradation in neural sequence models. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1290–1299. PMLR.

Pradeep Dasigi, Nelson F. Liu, Ana Marasović, Noah A. Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5925–5932, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Bryan Eikema and Wilker Aziz. 2020. Is MAP decoding all you need? the inadequacy of the mode in neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4506–4520, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. SummEval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

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.

Patrick Fernandes, António Farinhas, Ricardo Rei, José De Souza, Perez Ogayo, Graham Neubig, and Andre Martins. 2022. Quality-aware decoding for neural machine translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1396–1412, Seattle, United States. Association for Computational Linguistics.

Markus Freitag, David Grangier, Qijun Tan, and Bowen Liang. 2022. High quality rather than high model probability: Minimum Bayes risk decoding with neural metrics. *Transactions of the Association for Computational Linguistics*, 10:811–825.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

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*.

Dan Klein and Christopher D. Manning. 2003. A* parsing: Fast exact Viterbi parse selection. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 119–126.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Rémi Leblond, Jean-Baptiste Alayrac, Laurent Sifre, Miruna Pislar, Lespiau Jean-Baptiste, Ioannis Antonoglou, Karen Simonyan, and Oriol Vinyals. 2021. Machine translation decoding beyond beam search. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8410–8434, Online and Punta Cana, Dominican Republic. 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.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Ximing Lu, Sean Welleck, Peter West, Liwei Jiang, Jungo Kasai, Daniel Khashabi, Ronan Le Bras, Lianhui Qin, Youngjae Yu, Rowan Zellers, Noah A. Smith, and Yejin Choi. 2022. NeuroLogic a*esque decoding: Constrained text generation with lookahead heuristics. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 780–799, Seattle, United States. Association for Computational Linguistics.

Clara Meister, Ryan Cotterell, and Tim Vieira. 2020a. If beam search is the answer, what was the question? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2173–2185, Online. Association for Computational Linguistics.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2022a. Locally typical sampling.

Clara Meister, Tim Vieira, and Ryan Cotterell. 2020b. Best-first beam search. *Transactions of the Association for Computational Linguistics*, 8:795–809.

Clara Meister, Gian Wiher, Tiago Pimentel, and Ryan Cotterell. 2022b. On the probability–quality paradox in language generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 36–45, Dublin, Ireland. Association for Computational Linguistics.

Lidiya Murakhovs'ka, Chien-Sheng Wu, Philippe Laban, Tong Niu, Wenhao Liu, and Caiming Xiong. 2022. MixQG: Neural question generation with mixed answer types. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1486–1497, Seattle, United States. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. 2022. A conversational paradigm for program synthesis. *arXiv preprint arXiv:2203.13474*.

Franz Josef Och, Nicola Ueffing, and Hermann Ney. 2001. An efficient A* search algorithm for statistical machine translation. In *Proceedings of the ACL 2001 Workshop on Data-Driven Methods in Machine Translation*.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3956–3965. PMLR.

Diogo Pernes, Afonso Mendes, and André FT Martins. 2022. Improving abstractive summarization with energy-based re-ranking. *arXiv preprint arXiv:2210.15553*.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34:4816–4828.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics.

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

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Mathieu Ravaut, Shafiq Joty, and Nancy Chen. 2022. SummaReranker: A multi-task mixture-of-experts re-ranking framework for abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4504–4524, Dublin, Ireland. Association for Computational Linguistics.

Raj Reddy. 1977. Speech understanding systems: A summary of results of the five-year research effort at carnegie mellon university. *Pittsburgh, Pa*.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Swarnadeep Saha, Shiyue Zhang, Peter Hase, and Mohit Bansal. 2022. Summarization programs: Interpretable abstractive summarization with neural modular trees. *arXiv preprint arXiv:2209.10492*.

Tianxiao Shen, Myle Ott, Michael Auli, and Marc'Aurelio Ranzato. 2019. Mixture models for diverse machine translation: Tricks of the trade. In *International conference on machine learning*, pages 5719–5728. PMLR.

Kaiqiang Song, Bingqing Wang, Zhe Feng, and Fei Liu. 2021. A new approach to overgenerating and scoring abstractive summaries. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1392–1404, Online. Association for Computational Linguistics.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3356–3362, Hong Kong, China. Association for Computational Linguistics.

Katherine Stasaski and Marti Hearst. 2022. Semantic diversity in dialogue with natural language inference. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 85–98, Seattle, United States. Association for Computational Linguistics.

Mirac Suzgun, Luke Melas-Kyriazi, and Dan Jurafsky. 2022. Follow the wisdom of the crowd: Effective text generation via minimum bayes risk decoding. *arXiv preprint arXiv:2211.07634*.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2021. Multilingual translation from denoising pre-training. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3450–3466, Online. Association for Computational Linguistics.

Ashwin Vijayakumar, Michael Cogswell, Ramprasaath Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2018. Diverse beam search for improved description of complex scenes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Sean Welleck, Ilia Kulikov, Jaedeok Kim, Richard Yuanzhe Pang, and Kyunghyun Cho. 2020a. Consistency of a recurrent language model with respect to incomplete decoding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5553–5568, Online. Association for Computational Linguistics.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020b. Neural text generation with unlikelihood training. In *ICLR*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Jiacheng Xu, Siddhartha Jonnalagadda, and Greg Durrett. 2022. Massive-scale decoding for text generation using lattices. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4659–4676, Seattle, United States. Association for Computational Linguistics.

Kevin Yang and Dan Klein. 2021. FUDGE: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535, Online. Association for Computational Linguistics.

Hugh Zhang, Daniel Duckworth, Daphne Ippolito, and Arvind Neelakantan. 2021. Trading off diversity and quality in natural language generation. In *Proceedings of the Workshop on Human Evaluation of NLP*

Property	Det.	No Dup.	Low Pruning	Completeness
BS	✓	✓	✗	✓
Sample	✗	✗[6]	✗	✓
BFS	✓	✓	✓	✗

Table 9: Property comparison of search algorithms and approaches. *Det.* stands for deterministic search with reproducibility. *No Dup.* indicates the approach could guarantee no duplication of output sequences.

Algorithm 2 Best-First Search

Input: Language model abstracted as p_θ, search budget, and frontier $\mathcal{O}$.
Output: All completed paths P

$\mathcal{O} \leftarrow \{\langle \infty, \text{BOS}, -1 \rangle\}, T \leftarrow 0, t \leftarrow 0.$
while $T <$ budget **do**
  $n \leftarrow \mathcal{O}.\texttt{pop}()$
  for $v \in \mathcal{V}$ **do**
    if is-complete$(n \circ v)$ **then**
      $P \leftarrow P \cup (n \circ v)$
      continue
    end if
    child $\leftarrow \langle h(n \circ v), v, t \rangle$
    $\mathcal{O} \leftarrow \mathcal{O} \cup$ child
  end for
  $T \leftarrow T + 1$
  $t \leftarrow t + 1$
end while

Systems (HumEval), pages 25–33, Online. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

A Best-First Search Algorithm

We describe the best-first search algorithm in the context of probabilistic NLG in Algorithm 2.

A.1 Setup for investigating BFS

We use XSum (Narayan et al., 2018) and a BART model `BART-large-XSum`[7] (Lewis et al., 2020) fine-tuned on it as the testbed of our preliminary study. We sample 100 examples from the test set to measure the decoding quality. We set the beam size to 10 and the max sequence length to 30. For the machine configuration, we use Intel Xeon CPU @ 2.20GHz for CPU and NVIDIA A100-SXM4-40GB for GPU. We use Transformers (v4.23.1) (Wolf et al., 2020) and pytorch (v1.9.0) for baseline implementation and model calls.

A.2 Advantages & Challenges

What are the potential advantages of using BFS, compared to beam search and sampling approaches? We enumerate the inherent property of beam search, sampling, and best-first search in Table 9. BFS has many strengths to satisfy desired properties like diversity, quality, and controllability in text generation.

Deterministic BFS is a deterministic search algorithm with lower variance and higher controllability than stochastic sampling methods. This also indicates that BFS is compatible with sampling on top, similar to beam search.

No duplication BFS comes with no duplication, so it's guaranteed that the more search budget used, the more unique outputs there will be. Sampling methods with low truncation thresholds suffer from this issue.

No Pruning We illustrate the pruning issue in beam search in Figure 2. BS prunes the desired hypotheses. Unlike beam search, BFS never prunes,[8] and preserves all explored nodes. This also brings great flexibility that the generation could switch between different branches of search.

Diversity BFS yields diverse outputs with decent quality. The diversity of generated sequences is based on empirical lens, which will be covered in our experiments.

As we have discussed many strengths BFS enjoys, why has it not been the dominant approach? We implement a standard BFS algorithm, as described in Algorithm 2, and look into how it works on decoding text summaries from `BART-XSum`. We also define a notion of equivalent beam size[9]

[6] Duplication of sampling methods depend on the choice of hyper-parameter.

[7] https://huggingface.co/facebook/bart-large-xsum

[8] In practice, due to the large vocabulary $\mathcal{V}$, we only keep the highest k out of $|\mathcal{V}|$ ranked options for each expansion for efficiency. We posit that the long-tail low probability continuations won't be prioritized by the priority queue and it's fine to discard them anyway.

[9] Beam size and equivalent beam size are interchangeable for the rest of the paper for simplicity. We follow Xu et al. (2022) for the definition of equivalent beam size.

Beam Size	1	2	5	10
Incomplete Rate	58.1%	23.8%	3.9%	3.0%
Time (s)	1.0	1.9	5.6	13.7

Table 10: Search incomplete rate and speed for the vanilla BFS. Beam size denotes the equivalent beam size, which is a reflection of the total search budget. Incomplete rate measures how often a search does not reach any completed state (EOS). Time denotes the running time for running the search algorithm per example.

to calibrate the search budget for all methods. For beam search, we set a beam size b and a max decoding length T, and the total search cost is $C = bT$, which means there will be C times forward passes through LM. BFS also calls the LM for C times and discover C nodes.

While beam search iteratively gains depth, best-first search does not. Hence, we investigate how often BFS could (not) reach the search goal, which is at least one EOS token. In Table 10, we show that the vanilla BFS has a pretty high chance of failure when the search budget is very limited. Even in the case of beam size $b = 10$, there is a 3% of chance that the method won't reach any completed sequence, a sequence ending with EOS or other pre-defined termination tokens. This indicates that **the vanilla BFS struggles with the completeness**. Efficiency is another crucial factor for practical usage. We measure the time consumed for running the search for each example and report it in Table 10. For reference, beam search with $b = 10$ can be completed in 0.7s per example. The vanilla BFS is slower than BS since the step-wise exploration in BFS is not batched.

B Choice of Scoring Function

We test the incompletion rate in a very strict use case: decoding a summary with at most $T = 30$ tokens with a total budget $C = bT = 300$. If the model does not reach any EOS token before depth of 30, we consider it as a case of incompletion. We show the comparison of the incompletion rate in Table 12. The length-normalized sequence log-likelihood is formed as $\frac{1}{l^\alpha}\sum_{t=0}^{l}\log p_\theta(y_t|\mathbf{y}_{<\mathbf{t}}, \mathbf{x})$. The original definition of scoring function, $\alpha = 0$, is a failure in the context of best-first search. The reason behind is the monotonic relation of the hypothesis score and the length. Since shorter sequences always have higher score, the greedy property of best-first search will hinder the exploration of longer sequences. Although the weight of temporal decay could be increased, it will change the foundation of the algorithm if the decay is overwhelming the hypothesis score.

			BLEU		
	|S|	$\overline{\text{D}}$	Original	COMET	Δ
Reference	11	29.2	-	-	-
BS	10	14.6	39.6	**38.4**	-1.2
DBS	10	18.4	32.1	32.1	0.0
DBS+	10	21.7	32.0	33.3	**1.3**
$\text{BTYP}_{0.2}$	2	10.2	35.4	35.4	-0.1
$\text{BTYP}_{0.5}$	2	9.2	44.3	**44.2**	**0.0**
$\text{BTYP}_{0.95}$	3	9.9	39.9	39.7	-0.2
$\text{BNCLS}_{0.5}$	2	8.9	40.6	40.6	-0.1
$\text{BNCLS}_{0.8}$	2	9.5	38.5	38.4	-0.1
$\text{BNCLS}_{0.9}$	3	9.8	39.5	38.9	-0.6
$\text{TYP}_{0.2}$	8	26.4	23.9	25.0	1.1
$\text{TYP}_{0.5}$	9	27.0	31.2	32.6	**1.4**
$\text{TYP}_{0.95}$	10	37.2	24.1	24.1	0.0
$\text{NCLS}_{0.5}$	8	17.1	35.6	**36.3**	0.7
$\text{NCLS}_{0.8}$	10	28.9	28.9	28.7	-0.2
$\text{NCLS}_{0.9}$	10	33.4	25.4	26.6	1.2
BKS_{mean}	18	16.8	38.0	**39.0**	1.0
BKS_{last}	26	18.1	33.5	37.2	**3.6**

Table 11: Machine translation from English to French. We highlight the best BLEU score after reranking and the improvement Δ for each sector. Numbers are rounded after calculation for display simplicity.

	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 1.0$	BKS_{last}
Rate	79.5%	8.8%	1.8%	2.1%

Ref.	BS	DBS	BS+Sample	Sample
Rate	5.0%	1.1%	6.4%	0.8%

Table 12: Incomplete rate (↓) with different choices of scoring function in best-k search (top) and reference baselines (bottom). $\alpha = 0$ stands for sequential log-likelihood without length adjustment; $\alpha = 1.0$ represents BKS_{mean} . We show the lowest incomplete rate under various configurations of temporal decay. See Sec. 5.3 for the definition and discussion.

Performance of BKS_{last} and BKS_{mean} is overall good across all datasets. We also notice an interesting difference that BKS_{mean} prioritizes the quality slightly more than BKS_{last} while BKS_{last} enjoys more diversity. For example, BKS_{last} on QuoRef achieves higher distinctness score but a slightly lower ROUGE score. The difference of scoring function will definitely impact the search strategy and we treat it as a handle of controllability for our algorithm.

C Experiment Setup

Question Generation We adopt a state-of-the-art question generation model, MixQG (Murakhovs'ka et al., 2022), as the testbed to verify whether our approach could elicit more diverse, larger number and high-quality questions compared to baseline approaches. We use the variant `mixqg-large` in this paper. For datasets, we select a range of seen and unseen QA datasets, including SQuAD (Rajpurkar et al., 2016), DROP (Dua et al., 2019), and QuoRef (Dasigi et al., 2019). We set the maximum decoding length to 25 BPEs for SQuAD and QuoRef, and 20 for DROP.

Commonsense Generation CommonGen is a dataset for generative commonsense reasoning (Lin et al., 2020). The input is a few keywords and the target is a sentence satisfying commonsense and covering these keywords. We adopt a T5-based model[10] fine-tuned on the training set of CommonGen. Since CommonGen has multiple references for each input, we utilize multiple references for each example by evaluating outputs against them. The maximum decoding length is set to 20.

Text Summarization We use XSum (Narayan et al., 2018) as the dataset for abstractive text summarization. The model we use for this task is the BART[11] model (Lewis et al., 2020) fine-tuned on XSum. The maximum decoding length is 30.

D Experiment: Machine Translation En→Fr

We present the machine translation result from English to French in Table 11. The dataset we use here is an extended version of newstest2014. We can see a significant improvement over BLEU in our approach after using COMET-QE reranking.

We obtained similar results on En-Fr compared to En-De in Table 8. Our approach achieves a good combination of diversity and quality compared to baseline methods. One of the beam search + sampling method, BTYP$_{0.5}$, achieves 44.2 after COMET-QE reranking, which surpasses any other methods by a decent margin. Our approach, BKS$_{mean}$, beats strong baselines including beam search and sampling-only approaches. What worth noticing is the significant jump after reranking, which shows a great success of overgeneration + reranking as a paradigm.

	|S|	$\overline{\text{D}}$	$\overline{\text{OR}}$	$\overline{\text{R}}$	MV	GRM	MTR
BS	10	21.8	55.7	41.3	91.4	87.0	48.5
DBS	9	25.1	50.7	36.5	72.1	80.9	42.1
DBS+	9	29.6	50.5	31.9	37.7	81.3	35.6
BTYP$_{0.2}$	1	9.8	41.9	41.2	96.8	87.6	46.8
BTYP$_{0.5}$	1	10.2	46.1	44.9	94.3	88.7	50.2
BTYP$_{0.95}$	2	10.7	46.6	44.7	95.0	88.5	50.4
BNCLS$_{0.5}$	1	9.2	44.8	44.5	97.0	88.7	49.6
BNCLS$_{0.8}$	1	10.2	46.3	44.6	94.8	88.4	49.9
BNCLS$_{0.9}$	2	10.7	46.8	44.7	96.0	88.1	50.4
TYP$_{0.2}$	5	21.1	45.8	37.7	94.2	87.1	43.4
TYP$_{0.5}$	7	26.2	54.3	40.6	97.1	88.1	45.7
TYP$_{0.95}$	9	34.8	55.6	39.2	97.8	86.5	44.6
NCLS$_{0.5}$	9	31.9	55.6	39.0	95.1	86.3	44.3
NCLS$_{0.8}$	8	28.3	55.4	40.8	98.4	87.8	46.1
NCLS$_{0.9}$	9	31.6	55.9	39.0	97.7	85.9	44.3
BKS$_{mean}$	19	29.5	54.8	38.2	99.0	86.0	44.0
BKS$_{last}$	18	32.8	54.2	35.8	98.8	84.4	40.6

Table 13: Results of question generation on SQuAD.

E Experiment: Question Generation on SQuAD

We present the result of question generation on SQuAD in Table 13. Our approach achieves the best MAUVE score and a good combination of diversity and quality metrics. Our approach outperforms baseline models in either diversity or quality on SQuAD.

F Design Choice for Completion

In our paper, we design a temporal decay function to encourage the completion of our search algorithm. We have also considered a depth-based auxiliary term to encourage the completion. For instance, we can define $\texttt{aux}(n) = n\texttt{.length()}$ where a longer sequence will receive a higher score if we assume a longer sequence is more likely to terminate (Welleck et al., 2020a). The problem of this function is that it always prefer longer sequences. Once there exists one single long sequence, the rest of the search will focus on this string because it is longer than any other strings. The search will be shaped into a depth-first search while what we expect is to discover a diverse set of strings with various length and prefix.

[10]The model is available at https://huggingface.co/mrm8488/t5-base-finetuned-common_gen.

[11]https://huggingface.co/facebook/bart-large-xsum

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Sec. Limitations after Sec. Conclusion.

☒ A2. Did you discuss any potential risks of your work?
We do not see a substantial risk of our work. Although the output could contain toxic or biased content, we posit that it does not attribute to the search algorithm we propose.

☑ 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?

We create scientific artifacts. See Section 4 and Appendix.

☐ B1. Did you cite the creators of artifacts you used?
Not applicable. Left blank.

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
It is currently under discussion and processing and we will release the code.

☐ 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 4, Section 5 and Appendix.

C ☑ Did you run computational experiments?

Section 4, 5 and 6, and Appendix.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
We develop an inference algorithm, which does not require substantial computational resource. We provide related information in Appendix.

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?
We discuss the experimental setup in Section 4 and Appendix A.

☑ 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?
We report the descriptive statistics in Section 4, 6 and Appendix.

☑ 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.)?
We report the usage of existing packages in Section 4 and Appendix. There are also some footnotes providing instructions and details throughout the paper.

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.