A Theoretical Analysis of the Repetition Problem in Text Generation*

Zihao Fu,[1] **Wai Lam,**[1] **Anthony Man-Cho So,**[1] **Bei Shi**[2]

[1]Department of Systems Engineering and Engineering Management,
The Chinese University of Hong Kong, Hong Kong
[2]AI Lab, Tencent
{zhfu, wlam, manchoso}@se.cuhk.edu.hk; beishi@tencent.com

Abstract

Text generation tasks, including translation, summarization, language models, and etc. see rapid growth during recent years. Despite the remarkable achievements, the repetition problem has been observed in nearly all text generation models undermining the generation performance extensively. To solve the repetition problem, many methods have been proposed, but there is no existing theoretical analysis to show why this problem happens and how it is resolved. In this paper, we propose a new framework for theoretical analysis for the repetition problem. We first define the Average Repetition Probability (ARP) to characterize the repetition problem quantitatively. Then, we conduct an extensive analysis of the Markov generation model and derive several upper bounds of the average repetition probability with intuitive understanding. We show that most of the existing methods are essentially minimizing the upper bounds explicitly or implicitly. Grounded on our theory, we show that the repetition problem is, unfortunately, caused by the traits of our language itself. One major reason is attributed to the fact that there exist too many words predicting the same word as the subsequent word with high probability. Consequently, it is easy to go back to that word and form repetitions and we dub it as the high inflow problem. Furthermore, we extend our analysis to broader generation models by deriving a concentration bound of the average repetition probability for a general generation model. Finally, based on the theoretical upper bounds, we propose a novel rebalanced encoding approach to alleviate the high inflow problem and thus reducing the upper bound. The experimental results show that our theoretical framework is applicable in general generation models and our proposed rebalanced encoding approach alleviates the repetition problem significantly in both the translation task and the language modeling task. The source code of this paper can be obtained from https://github.com/fuzihaofzh/repetition-problem-nlg.

1 Introduction

Text generation tasks aim at generating human-readable text for specific tasks including machine translation (Sutskever,

Vinyals, and Le 2014; Bahdanau, Cho, and Bengio 2014; Luong, Pham, and Manning 2015), summarization (Nallapati et al. 2016), data-to-text generation (Lebret, Grangier, and Auli 2016; Wiseman, Shieber, and Rush 2017; Fu, Bing, and Lam 2020; Fu et al. 2020c,a), language modeling (Radford et al. 2018, 2019; Brown et al. 2020) and etc. Despite the remarkable growth, nearly all existing generation systems suffer from the repetition problem (Fan, Lewis, and Dauphin 2018; Holtzman et al. 2020; Welleck et al. 2020) which unavoidably undermines the overall performance. The repetition problem refers to an undesirable effect that the results of the generation system always contain duplicate fragments. For example, a generation system is likely to generate text like "Tough it is still unfinished, but I like it but I like it but I like ...", which contains the useless repeating text. This problem hurts the quality of generation systems severely.

There are many conjectures for the reason of the repetition problem, but the real cause is still unknown (Welleck et al. 2020). Some conjectures think that it is caused by the model architectures (Holtzman et al. 2020; Vig 2018). Some of them suggest that it may be caused by the gap between sampling methods and the real human language (Holtzman et al. 2020). Choi (2018) argues that reliance on the fixed corpora cannot fulfill the real goal of using the language. Moreover, Welleck et al. (2020) attribute the reason to the likelihood maximizing decoding. However, there is no existing theory to quantitatively explain how and why the repetition occurs. Based on these conjectures, many out-of-the-box methods have been proposed to alleviate the repetition problem. Ackley, Hinton, and Sejnowski (1985); Ficler and Goldberg (2017); Caccia et al. (2019) propose to utilize the Temperature sampling in the decoding phase. Fan, Lewis, and Dauphin (2018) propose the Topk sampling method while Holtzman et al. (2020) propose the Nucleus sampling method. Though these methods are shown to successfully alleviate the repetition problem, there still needs a convincing explanation.

To determine the root cause of the repetition problem and give a convincing understanding of why existing methods work, we propose a novel theoretical analysis framework. We first define the general generation model and the Markov

generation model as the core text generation models for our analysis. Then, we define the Average Repetition Probability (ARP) to characterize the repetition problem quantitatively on the Markov generation model. Subsequently, we conduct an extensive analysis of the Markov generation model and derive several upper bounds of the ARP with intuitive understanding. We show that most of the existing methods are essentially minimizing the theoretical upper bounds explicitly or implicitly. Guided by our theory, we show that the repetition problem is caused by, unfortunately, our language itself. One major reason is attributed to the fact that there exist too many words predicting the same word as the subsequent word with high probability. Consequently, it is easy to go back to that word and form repetitions and we dub it as the high inflow problem. We call the predicted word as high inflow word and call the previous word together with the high inflow word as high inflow pair. Furthermore, we extend our results to broader generation models by deriving a concentration bound of the average repetition probability for a general generation model.

Based on the theoretical upper bounds, we propose a novel rebalanced encoding approach to alleviate the high inflow problem and thus reducing the upper bound. We merge each high inflow pair as a new word and thus decrease the high inflow probability. We conduct experiments on two fundamental tasks namely the Neural Machine Translation (NMT) task and the Language Modeling (LM) task. The experimental results show that our proposed new method alleviates the repetition significantly and outperforms existing prevalent out-of-the-box methods. We also conduct several experiments to verify the correctness and applicability of our theory when applied in NMT and LM tasks.

2 Theoretical Framework

Our theoretical framework mainly encompasses three parts. We first define the Average Repetition Probability (ARP) to characterize the repetition problem quantitatively. Then, we analyze the repetition problem in a Markov generation model. We derive several upper bounds for ARP with intuitive understanding and we show that most of the existing approaches are trying to minimize the upper bounds explicitly or implicitly. Finally, we show that our analysis can be adopted into general generation models by proving that if each step's transition matrix does not deviate a lot from the Markov generation model, the deviation of the ARP value will be controlled with a concentration upper bound.

2.1 Definitions

To facilitate a quantitative analysis of the repetition problem, we first give some definitions which will be used in the later sections of this paper. We denote the generated sequence $s = [w_1, w_2, \cdots, w_{|s|}]$, in which w_i is the ith word and $|s|$ is the length of s. We denote $s_{p:q} = [w_p, w_{p+1}, \cdots, w_q], (1 \leq p < q \leq |s|)$ as a continuous subsequence of s. We say a continuous subsequence $s_{r:t}$ is a **loop** if $w_r = w_t$ (it is allowed to contain sub-loops inside). We say a continuous subsequence $s_{p:q} = [s_p, s_{p+1}, \cdots, s_q]$ is a **repetition subsequence** if $\exists 1 \leq p < q \leq |s| - q + p, \ s.t. \ w_{i+q-p} =$
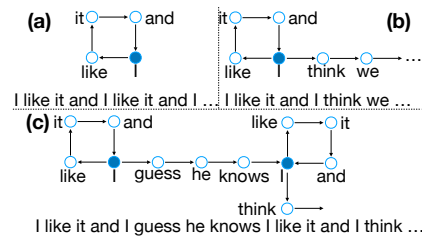

Figure 1: Loop and repetition subsequence.

$w_i, \forall i \in [p, q]$. It can be easily shown that a sequence has repetition subsequence if and only if it has at least two adjacent identical loops. As shown in Fig. 1 (a), the sequence contains repetition subsequences with adjacent loops while in Fig. 1 (b), it contains a loop without any repetition subsequence. It should be noted that in Fig. 1 (c), even though it has two identical loops, it does not contain any repetition subsequence since the loops are not adjacent.

Definition 2.1 (General Generation Model). *A general generation model is defined as $p_\theta(w_i | w_{i-1}, \cdots, w_1, x)$, in which θ stands for all the model parameters; w_i is the current word to be predicted; $w_{i-1}, \cdots, w_1$ are the previous words that have been generated and x is the input information.*

The general generation model is an abstract description of nearly all text generation models. For language models (Radford et al. 2018, 2019; Brown et al. 2020), the input x is empty while w_i is the word generated at the ith step. For translation models (Sutskever, Vinyals, and Le 2014; Bahdanau, Cho, and Bengio 2014; Luong, Pham, and Manning 2015), x stands for the source language. For summarization models, x is the original text while w_i is the ith word in the corresponding summary. However, analyzing the general generation model is almost impossible due to its complicated structure with many high non-linear layers. To facilitate effective analysis, we propose to first analyze a simplified model and study the relationship between them.

Definition 2.2 (Markov Generation Model). *A Markov generation model predicts a word only based on the previous word, which can be denoted as $p_\xi(w_i | w_{i-1}) \approx p_\theta(w_i | w_{i-1}, \cdots, w_1, x)$, where ξ stands for parameters of the Markov generation model.*

The Markov generation model simplifies the general generation model with the intuition that the current word is often most influenced by the previous word. We conduct our analysis on the Markov generation model first and then extend to the general generation model. Since the probability of the current word only depends on the previous word, it becomes a Markov chain. We denote the transition matrix as $A \in \mathbb{R}^{(n+1) \times (n+1)}$, in which n is the vocabulary size. A is a stochastic matrix and it is guaranteed that $A_{ij} \geq 0, \sum_{j=1}^{n+1} A_{ij} = 1, \forall i \in [1, n+1]$. We denote the $(n+1)$th word as the End-Of-Sentence (EOS) tag and the generation stops if it generates the EOS tag. Therefore, the EOS state is an absorbing state and A can be written as $A = \begin{bmatrix} B, b \\ 0, 1 \end{bmatrix}$, in
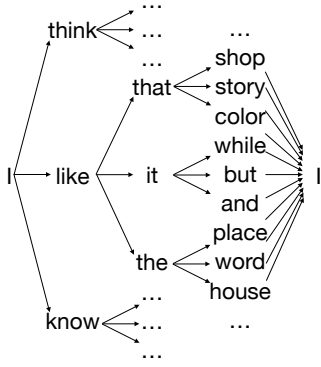
Figure 2: Illustration of loops starting and ending with the same word "I". On average, each word transits to three subsequent words.

which $B \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^{n \times 1}$. B is a sub-transition matrix without the EOS tag and $B_{ij} \geq 0, \sum_{j=1}^{n} B_{ij} \leq 1, \forall i, j \in [1, n]$.

In order to derive a quantitative definition of the repetition problem, we first study loops in the Markov generation model. It is easy to observe that the matrix B is very sparse since most words only have a limited number of subsequent words. We define the sparsity of matrix B as $\zeta = \frac{1}{n^2} \sum_i^n \sum_j^n \mathbb{1}(B_{ij} > 0)$, in which $\mathbb{1}(C)$ is the indicator function. It equals to 1 if C is true and equals to 0 otherwise. Therefore, one word can transit to ζn words on average at a single step. As illustrated in Fig. 2, for a word to transit back to itself at the kth step, it generates $(\zeta n)^k$ paths and has $\frac{(\zeta n)^k}{n}$ paths that transit back to itself on average. On the other hand, the probability for the ith word loops back to itself after k steps is the ith diagonal value of B^k which is denoted as B_{ii}^k. The average probability for each path is $\frac{n B_{ii}^k}{(\zeta n)^k}$ and the average probability for a loop to repeat again is $(\frac{n B_{ii}^k}{(\zeta n)^k})^2$. The average probability for all loops to repeat again is $(\frac{n B_{ii}^k}{(\zeta n)^k})^2 \cdot \frac{(\zeta n)^k}{n}$. Since $(B_{ii}^k)^2 \leq (B^{2k})_{ii}$, to simplify the analysis, we consider the upper bound for all states and the probability sum as $\text{tr}(\frac{n B^{2k}}{(\zeta n)^k})$, in which tr stands for the matrix trace. The average probability sum for all steps is $\sum_{k=1}^{\infty} \text{tr}(\frac{B^{2k}}{\zeta^k n^{k-1}})$. The average probability for each word is $R = \frac{1}{n} \sum_{k=1}^{\infty} \text{tr}(\frac{B^{2k}}{\zeta^k n^{k-1}}) = \sum_{k=1}^{\infty} \text{tr}(\frac{B^{2k}}{\zeta^k n^k})$. Therefore, we define the average repetition probability as follows:

Definition 2.3 (Average Repetition Probability). *Given a Markov generation model with sub-transition matrix $B \in \mathbb{R}^{n \times n}$ and non-zero probability ζ as defined above, the Average Repetition Probability (ARP) is defined as:*

$$R = \sum_{k=1}^{\infty} \text{tr}(\frac{B^{2k}}{\zeta^k n^k})$$

The Average Repetition Probability (ARP) gives a quantitative description of the repetition problem. Intuitively, if ARP is high, the repetition problem occurs frequently and if ARP is small, the repetition problem seldom happens.

To better understand how existing methods work, we show that most of the existing out-of-the-box methods are equivalent to applying a transformation $\mathcal{T}$ to the word probability distribution p and then applying a stochastic sampling on the transformed probability for obtaining the word $w \sim \mathcal{T}(p)$. For the **Stochastic** sampling (Fan, Lewis, and Dauphin 2018; Holtzman et al. 2020), $\mathcal{T}_s(p) = p$ is the identity operator. The transformation for the **Greedy** sampling (Klein et al. 2017; Ott et al. 2019) changes the probability of the most probable word to 1 and sets others' probabilities to 0 which can be denoted as $\mathcal{T}_g(p)_i = \mathbb{1}(i = \arg\max_j p_j)$. The **Topk** sampling (Fan, Lewis, and Dauphin 2018) method sets all word probabilities to 0 except top k words and rescales it to a new distribution. It can be denoted as $\mathcal{T}_k(p)_i = \mathbb{1}(i \in K)p_i / \sum_{j \in K} p_j$, in which K is the Topk probability word set: $K = \arg\max_K \sum_{j \in K} p_j, s.t. |K| = k$. The **Nucleus** sampling (Holtzman et al. 2020) sets all word probabilities to 0 except for words with probabilities sum to larger than p. It can be denoted as $\mathcal{T}_n(p)_i = \mathbb{1}(i \in N)p_i / \sum_{j \in N} p_j$, in which N is the smallest word set with probability sum larger than p: $N = \arg\min_N |N|, s.t. \sum_{j \in N} p_j \geq p$. The Temperature sampling (**Temp**) (Ackley, Hinton, and Sejnowski 1985; Ficler and Goldberg 2017; Caccia et al. 2019) rescales the probability with a temperature parameter t and can be denoted as $\mathcal{T}_t(p)_i = \exp((\log p_i)/t) / \sum_j \exp((\log p_j)/t)$. The Length Penalty (**LP**) (Wu et al. 2016; Klein et al. 2017) method prefers shorter sentences than long sentences. Without loss of generality, we simply enlarge the probability of the EOS tag with a constant β which can be denoted as $\mathcal{T}_l(p)_i = \exp(\tilde{\ell}_i) / \sum_j \exp(\tilde{\ell}_j)$, in which $\tilde{\ell}_i = \ell_i + \mathbb{1}(i = n+1)\beta$ and ℓ is the logits vector calculated by the model.

It can be directly concluded from the definition that the Greedy sampling always has serious repetition problem. In greedy sampling, each word only takes a fixed subsequent word and thus $\zeta n = 1$. Therefore, ARP can be very large and even diverges to infinity which is consistent with the observation that Greedy sampling always causes repetition sentences in many generation systems (Fan, Lewis, and Dauphin 2018; Holtzman et al. 2020).

2.2 Average Repetition Probability Upper Bounds

To provide more intuitive understanding of the repetition problem directly, we simplify ARP and derive several upper bounds for further investigation.

Theorem 1. *If $\zeta n > \rho(B^2)$,*

$$R = \text{tr}(B^2(\zeta n I - B^2)^{-1}) \leq \frac{\|B^2\|_*}{\sigma_n(\zeta n I - B^2)},$$

in which $\| \cdot \|_*$ is the nuclear norm; σ_n denotes the smallest singular value; $\rho(\cdot)$ is the spectral radius of the matrix; I is the identity matrix.

The proof is provided in Appendix A.1 (Fu et al. 2020b). Theorem 1 gives a very straightforward upper bound of the ARP. It can be easily proved that $\rho(B^2) < 1$ for a substochastic matrix B^2. Therefore, Theorem 1 fails only if $\zeta n \leq \rho(B^2) < 1$, which means that all words have less than

one subsequent word. It seldom happens in a normal language model. As neither the nuclear norm nor the smallest singular value has very intuitive physical meaning in generation models, we derive two corollaries based on Theorem 1 to get more practical and intuitive upper bounds.

Corollary 1.1.

$$R \leq \frac{\sqrt{r}(\sum_{i=1}^{n}\sum_{j=1}^{n}(B_{ij}-\mu_i)^2 + \sum_{i=1}^{n}(1-b_i)^2)}{\sigma_n(\zeta nI - B^2)},$$

where r is the rank of B^2 and $\mu_i = \frac{\sum_{k=1}^{n}B_{ik}}{n}$ is the mean of each row of B.

The proof is provided in Appendix A.2 (Fu et al. 2020b). It can be concluded from Corollary 1.1 that the upper bound of R decreases as the variance of B_{ij} decreases. The current methods including Nucleus sampling, Top-k sampling, and stochastic sampling are of this kind. Compared with the commonly used greedy sampling method in which only one word has a probability of 1 with others being 0, these methods significantly decrease the variance and thus lower the upper bounds. It should be noted that Stochastic sampling always achieves the smallest variance. However, due to the long tail effect, it cannot be used since it has a very high probability of sampling low probability words. It is also obvious that Temperature sampling alleviates the repetition problem because it controls the variance by changing the temperature parameter. It is interesting to observe that directly truncating words with low probabilities actually increases the variance since it sets a lot of words' probabilities to 0 while increasing the probabilities of the remaining words. Therefore, Topk/Nucleus sampling alone cannot alleviate the repetition problem. They should always work together with the Stochastic sampling method or the Temperature sampling method, and the real role of Topk/Nucleus sampling is to solve the long-tail problem of stochastic sampling or Temperature sampling. If the temperature is fixed, using Topk/Nucleus sampling can even make the repetition problem worse. On the other hand, the length penalty strategy also alleviates the repetition problem. The reason is that it is equivalent to increasing the probability of going to the EOS state which increases the elements in b. From Corollary 1.1, when $|b|$ increases, R decreases. We will conduct experiments to validate all the above claims.

Corollary 1.2. *If $\zeta nI - B^2$ is a diagonally dominant matrix,*

$$R \leq \frac{\|B^2\|_*}{\min_{1 \leq i \leq n}\{\frac{1}{2}(\zeta n - \underbrace{\sum_{j=1}^{n}(B^2)_{ij}}_{outflow}) + \frac{1}{2}(\zeta n - \underbrace{\sum_{k=1}^{n}(B^2)_{ki}}_{inflow})\}}.$$

The proof is provided in Appendix A.3 (Fu et al. 2020b). In Corollary 1.2, the inflow for a word is the probability sum of all words that take it as the subsequent word. If it is too big, the upper bound can be magnified extensively and fails to limit the ARP. This observation theoretically justifies the claim that high inflow words are more likely to go back to itself and cause the repetition problem. We

can also conclude that increasing $|b|$ can decrease the outflow term and thus decrease R. It also shows that the length penalty method can help alleviate the repetition problem. As is shown in many previous works, most repetition sentences contain high-frequency words. By Corollary 1.2, we can conclude that it is not the high-frequency words, but the high inflow words that really lead to repetition generation. We propose our novel rebalanced encoding based on this corollary to reduce the high inflow words.

2.3 Extension to General Generation Model

To extend our analysis to the general generation model, we define the ARP for the general generation model and show that if the transition matrix is controlled, the ARP will not deviate a lot.

Definition 2.4 (Average Repetition Probability for General Generation Model)**.** *For a general generation model, the sub-transition matrix for the kth step can be expressed as $B'_k = B + T_k$, in which $T_k \in \mathbb{R}^{n \times n}$ is a perturbation matrix and each element for $T_{k(ij)}$ is independently distributed with mean 0 and we assume that the variance is controlled as $\delta^2 < \frac{1}{n}$. The general average repetition probability is defined as:*

$$R' = \sum_{r=1}^{\infty} \text{tr}(\frac{\prod_{k=1}^{2r} B'_k}{\zeta^r n^r}).$$

In the general generation model, each step has its own sub-transition matrix B'_k. Since the current word is mostly affected by the previous word, we denote B'_k as applying a "small" perturbation matrix T_k on B. We show that if T_k is "small", R' will not deviate much from R.

Theorem 2. *For a general generation model, if $\sum_{i=1}^{n} B_{ij}^2 < 1, \zeta n > 4$, then for every constant $a > 0$ we have*

$$\text{Pr}(|R - R'| \geq a) \leq \frac{3\zeta n \delta^2}{a^2(\zeta n - 4)(\zeta n - 1)}.$$

The proof is provided in Appendix A.4 (Fu et al. 2020b). Theorem 2 shows that given a generation model that has a different transition matrix B'_k at each step, if the transition matrix does not deviate a lot from that in the Markov generation model, the deviation of ARP is bounded with high probability. It should be noted that the distribution of $T_{k(ij)}$ is influenced by B. Nevertheless, Theorem 2 only depends on the mean and the variance regardless of the specific value constraint.

3 Rebalanced Encoding

It can be concluded from Corollary 1.2 that the ARP upper bound decreases as the inflow term decreases. It can also be inferred from Corollary 1.1 that if the variance of B decreases, the ARP upper bound will also decrease. Inspired by these two properties, we propose a novel rebalanced encoding (RE) which merges the high inflow pairs into single words and thus reduces the inflow term and the variance of B simultaneously. Different from most of the existing methods that focus on the post-editing of the probability distribution, the RE method uses a novel word encoding scheme and

Algorithm 1 Rebalanced Encoding Algorithm

```
def learnRE(words : list, N : int, gamma : float):
  merges = []
  for step in range(N):
    id_to_word = list(set(words))
    word_to_id = {w:i for i,w in enumerate(id_to_word)}
    M = numpy.zeros([len(id_to_word), len(id_to_word)])
    for i in range(len(words) - 1):
      M[word_to_id[words[i]], word_to_id[words[i+1]]]+=1
    M =  M / M.sum(1).reshape(-1,1).clip(1)
    if M.max() <= gamma: break
    merges += [(id_to_word[i1], id_to_word[i2]) for
             i1, i2 in zip(*(M > gamma).nonzero())]
    words = applyRE(words, merges)
  return merges

def applyRE(words : list, merges : list):
  for merge in merges:
    for i in range(len(words) - 1):
      if tuple(words[i : i + len(merge)]) == merge:
        words[i : i + len(merge)] = [
          "==".join(merge).replace("@@==", "")]
        i -= 1
  return words
```

thus makes the model predict a good probability distribution instead of post-editing.

Our proposed RE method first applies traditional BPE encoding (Sennrich, Haddow, and Birch 2016) to the training text. Then, it makes a statistical transition matrix with the encoded training text. It picks high inflow pairs that have transition probability higher than a threshold γ and merges the word pair as a whole word. Specifically, if two words are split from BPE as subwords, we simply merge them and remove the BPE tags. For example, for the high inflow word pair "(de@@, crease)", we replace all "de@@ crease" to "decrease". If the two words are words that have not been split by BPE, we merge them by adding a "==" tag between them and treat them as a single new word. For example, for the high inflow word pair "(involved, in)", we replace all "involved in" to "involved==in". We rebuild the transition matrix and repeat the above procedure until all the probabilities in the transition matrix are less than γ or we reach a specific iteration epoch. The detailed algorithm is shown in Algorithm 1 and we follow the presentation of using Python code (Sennrich, Haddow, and Birch 2016) to make it more concise.

4 Experiments

4.1 Experimental Settings

We conduct our experiments on two common text generation tasks, namely, the neural machine translation task and the language modeling task.

Neural Machine Translation (NMT). The NMT task translates sentences from the source domain to the target domain with the most prevalent Transformer (Vaswani et al. 2017; Ott et al. 2019) architecture. We adopt the widely used IWSLT'14 English-German dataset containing 160K sentences pairs. We encode all the sentences with the byte-pair encoding (Sennrich, Haddow, and Birch 2016) with subword units of 10,000 for each language. The overall translation performance is evaluated by BLEU (Papineni et al. 2002) and ROUGE$_L$ (Lin 2004). The repetition problem is quantitatively evaluated with rep-w, rep-n (Welleck et al. 2020) and rep-r. rep-w measures the

Method	rep-w↓	rep-n↓	rep-r↓	BLEU↑	ROUGE$_L$ ↑
Greedy	0.0883	0.0330	0.0512	0.352	0.606
Stochastic	0.0783	0.0272	0.0337	0.222	0.472
Temp (t=0.15)	0.0879	0.0328	0.0511	0.351	0.605
Topk (k=10)	0.0882	0.0329	0.0507	0.350	0.605
Topk (k=40)	0.0881	0.0329	0.0511	0.350	0.604
Nucleus (p=0.9)	0.0878	0.0328	0.0508	0.349	0.603
Nucleus (p=0.95)	0.0882	0.0329	0.0510	0.350	0.604
LP (β=6)	0.0863	0.0322	0.0500	0.349	0.605
RE (γ=0.15)	0.0768	0.0281	0.0419	0.350	0.608
RE (γ=0.1)	0.0743	0.0275	0.0417	0.350	0.607
RE (γ=0.05)	0.0585	0.0211	0.0296	0.340	0.603
RE (γ=0.02)	0.0434	0.0158	0.0221	0.335	0.600

Table 1: Experimental results for NMT task.[1]

Method	rep-w↓	rep-n↓	rep-r↓	ppl-c
Greedy	0.590	0.733	0.917	0.150
Stochastic	0.120	0.092	0.155	7.320
Temp (t=0.75)	0.254	0.215	0.409	1.060
Topk (k=40)	0.235	0.188	0.363	0.969
Topk (k=10)	0.251	0.195	0.348	0.962
Nucleus (p=0.9)	0.234	0.195	0.368	1.020
Nucleus (p=0.95)	0.227	0.191	0.322	1.090
LP (β=7)	0.547	0.660	0.829	1.050
RE (γ=0.1)	0.196	0.180	0.321	0.974
RE (γ=0.08)	0.180	0.156	0.286	1.010

Table 2: Experimental results for LM task.[1]

repetition words proportion by the fraction of the current token that occurs in the previous w tokens. It can be calculated as rep-w $= \frac{1}{|\mathcal{D}|} \sum_{s \in \mathcal{D}} \frac{1}{|s|} \sum_{t=1}^{|s|} \mathbb{1}[s_t \in s_{t-w-1:t-1}]$, in which s denotes generated sentences in the result set $\mathcal{D}$ and $|s|$ denotes the word count of s; s_t is the tth word in s while $s_{t-w-1:t-1}$ is the sub-sequence of s from the $(t-w-1)$th word to the $(t-1)$th word. The rep-w metric extends the rep/ℓ in Welleck et al. (2020) in which the sentence length is fixed. rep-n is defined as rep-n $= 1.0 - \frac{|\{\tilde{s}|\exists p \in [1,|s|-n+1], \tilde{s}=s_{p:p+n-1}\}|}{|s|-n+1}$. rep-n depicts the repetition problem for the specific gram n while rep-w characterizes the repetition problem in the view of words. To give a quantitative metric for the repetition problem by length and avoid setting specific parameters, we propose the rep-r metric. rep-r stands for the ratio of the repetition snippet in a sentence measured by length. It is defined as rep-r $= \frac{|\{i|(s_i=s_j \wedge s_{i+1}=s_{j+1}, \exists j \neq i) \vee (s_i=s_k \wedge s_{i-1}=s_{k-1}, \exists k \neq i)\}|}{|s|}$, in which $\wedge$ and $\vee$ are logical "and" and "or" respectively. The details of hyper-parameters settings are presented in Appendix A.6 (Fu et al. 2020b).

Language Modeling (LM). LM task is a fundamental task for natural language processing which predicts the probability of a word conditioned on previous words. We adopt the language model with Transformer decoder which has also been utilized by the most prevalent GPT mod-

---

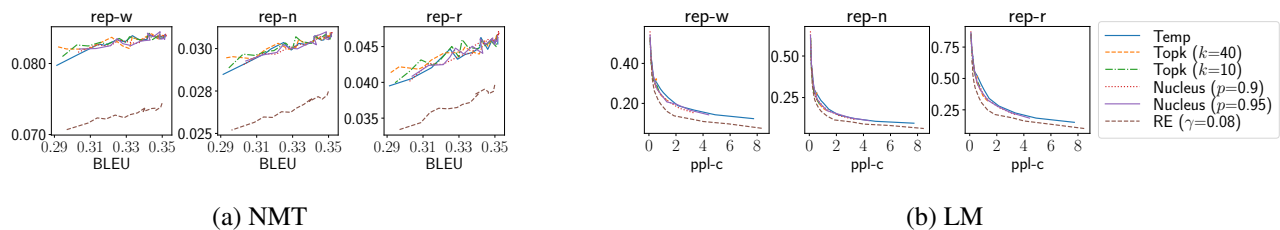[1]↑ means larger is better while ↓ means lower is better.

(a) NMT

(b) LM

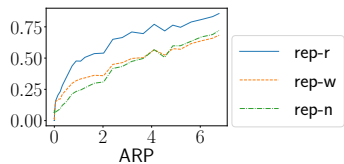Figure 3: Repetition performance balancing.
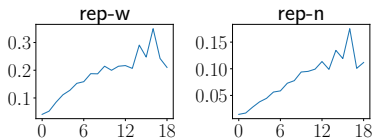

Figure 4: Correlation of ARP and repetition metrics.

Figure 5: Influence of high inflow pair. x axis is the high inflow pair count.

Method	Overall↑	Repetition↑
Greedy	3.642	3.342
Nucleus (p=0.9)	3.650	3.383
RE (γ=0.08)	3.800	3.400

Table 3: Human Evaluation.[1]

els (Radford et al. 2018, 2019; Brown et al. 2020). We use the Wiki-103 dataset (Merity et al. 2017) and encode the text with byte pair encoding with subword units around 10,000. We use the same language model as Ott et al. (2019) and keep all hyper-parameters the same. Since the perplexity score strongly depends on the encoding approaches we use for making subword units, it is not directly comparable when different encoding approaches are applied. We propose a new metric called perplexity coefficient (ppl-c) which divides the perplexity of the generated text with the average perplexity of the text in the test set. Therefore, a sentence is more likely to be written by humans if its ppl-c score closes to 1.0. To evaluate the model, we take words in the dictionary to feed them into the language model and generate subsequent sequences. We guarantee that the initial words for all models are the same. The repetition problem is evaluated with rep-w, rep-n (Welleck et al. 2020), and rep-r. The Topk, Nucleus, and RE methods are combined with the Temperature sampling method together. To make the models comparable with each other, we tune all the models to make the ppl-c scores as close to 1.0 as possible which is closest to human-written text. The details of hyper-parameters settings are presented in Appendix A.6 (Fu et al. 2020b).

Comparison Methods. As our proposed RE method does not need any additional modification of the generation model. we compare it with the most widely used out-of-the-box methods including Greedy sampling, Stochastic sampling, Temperature sampling, Topk sampling, Nucleus sampling, and Length Penalty (LP) method. The details of each method have been discussed in Section 2.1.

4.2 Experimental Results

Main Results. The NMT experimental results are shown in Table 1 while the LM Experimental results are shown in Table 2. We have the following conclusions from the results. (1) The RE method alleviates the repetition problem and outperforms existing methods significantly. (2) The rep-

etition problem of the Greedy method is more serious than other models while it hardly appears in the Stochastic sampling method. This result justifies our theoretical analysis in Corollary 1.1 that higher variance in the transition matrix leads to a higher probability of repetition. (3) The LP method and RE method both alleviate the repetition problem. The reason is that the RE method controls the inflow term while the LP method controls the outflow term and thus they limit the upper bound of ARP. This result justifies our theoretical analysis in Corollary 1.2. (4) Empirically, the performance metrics become slightly worse (BLEU and ROUGE$_L$ decrease or ppl-c increases when larger than 1.0) as the repetition problem is alleviated. This is because existing performance metrics are not very sufficiently sensitive to the repetition problem and they will not become significantly better even when the repetition problem is well alleviated. For example, the ppl-c metric will not become worse even if it adds many repetition sequences with high prediction probability. Therefore, the existing metrics are not enough to provide a comprehensive evaluation and thus we conduct a human evaluation.

Human Evaluation. We conduct a human evaluation and the results are shown in Table 3. We sample 120 generated text in the LM task for each model. The sampled text is scored by human helpers to evaluate the overall performance and repetition performance. The detailed questionnaire is shown in Appendix A.5 (Fu et al. 2020b). It can be concluded from the results that our proposed method outperforms the comparison methods in both alleviating the repetition problem and improving the overall score simultaneously.

Repetition Performance Balancing. As is shown in our main results, mitigating the repetition problem and improving the evaluation metrics are not simultaneously feasible since existing performance metrics are not sensitive to repetition. In order to show that our proposed RE method achieves the best balance between the two goals, we conduct

Method	Generated Text
Greedy	Battalion , the 1st Battalion , the 2nd Battalion , the 1st Battalion , the 1st Battalion , the 1st Battalion , the 1st Battalion , the 1st Battalion , the 1st Battalion , the 2nd Battalion , the 1st Battalion , the 1st Battalion , the 1st Battalion , the 2nd Battalion , the 1st Battalion , the 1st Battalion , the 1st Battalion , the 1st Battalion ...
Topk (k=10)	Battalion of the Royal Marines were also assigned to the 1st Division , the 1st Division , and the 1st Division . The 1st Division was assigned to the 1st Division , and the 1st Division was assigned to the 1st Division . The 1st Division was assigned to the 1st Division , and the 1st Division was assigned to the 1st Division . The 1st ...
RE	Battalion was the first unit to be deployed to Iraq in March 1971 . In April 1971=, the battalion was involved=in Operation Ira@@ q@@ i Fre@@ edom , a major operation in=the Ira@@ q@@ i conflict . In January 1971=, the battalion was involved=in Operation Ira@@ q@@ i Fre@@ edom , a major operation in=the Ira@@ q@@ i Gulf . In March 1971=, the battalion was involved=in Operation Ira@@ q@@ i Fre@@ edom , a major operation in=the Ira@@ q@@ i Gulf . In April 1971=, the battalion was involved=in Operation Ira@@ q@@ i Fre@@ edom , a major operation in=the Ira@@ q@@ i Gulf . ⟨eos⟩

Table 4: Experimental results for LM task.

a repetition performance balancing experiment. We tune the temperature parameter for the models to get different repetition metrics at different levels of performance (BLEU or ppl-c). The NMT task results are illustrated in Fig. 3 (a) while the LM task results are shown in Fig. 3 (b). It can be concluded from the results that (1) The RE method achieves the lowest repetition scores at the same performance score in both NMT and LM tasks. (2) The Topk sampling and Nucleus sampling alleviate the repetition problem of the Temperature sampling method because they help alleviate the long-tail effect and thus improve the overall performance. This experiment justifies the claims in Corollary 1.1.

Well-definedness of ARP. To show that our proposed definition of ARP is well defined, we conduct an experiment to study the relationship between the theoretical ARP and the repetition metrics. To calculate the theoretical ARP, we conduct this experiment on a Markov generation model. The Markov transition matrix is calculated by counting words in Wiki-103. We sample each word with the Temperature sampling method and we change the temperature parameter to study the relation of ARP and repetition metrics. The results are shown in Fig. 4. It can be concluded from the results that as the ARP grows, all repetition metrics are increasing. This positive correlation shows that ARP is well-defined.

Influence of High Inflow Pairs. To show that the high inflow pairs do cause the repetition problem, we conduct an experiment in the NMT task to show the relationship between the high inflow pair count and the repetition metrics. As shown in Fig. 5, we calculate the high inflow pair count and the repetition scores in each sentence. It can be concluded from the results that if one sentence contains too many high inflow pairs, it is more likely to get a higher repetition score. This result also justifies Corollary 1.2.

Case Study. To get a better understanding how the repetition is alleviated, we show a case study by sampling results from different methods. The results are shown in Table 4. It can be observed from the results that the Greedy sampling method has a severe repetition problem. The Topk sampling method alleviates the repetition problem by minimizing ppl-c and can thus increase the temperature when the ppl-c level is fixed. Our proposed RE method gives the best trade-off between the repetition problem and the overall performance.

5 Related Works

Text generation tasks aim at generating sentences based on given input. It can be divided into two main categories, namely, the sequence-to-sequence based systems (Sutskever, Vinyals, and Le 2014; Bahdanau, Cho, and Bengio 2014; Luong, Pham, and Manning 2015; Wu et al. 2016; Vaswani et al. 2017) and the language model based systems (Radford et al. 2018, 2019; Brown et al. 2020). The sequence-to-sequence framework has been applied to solve many tasks including neural machine translation (Sutskever, Vinyals, and Le 2014; Bahdanau, Cho, and Bengio 2014; Luong, Pham, and Manning 2015), summarization (Nallapati et al. 2016) and data-to-text generation (Lebret, Grangier, and Auli 2016; Wiseman, Shieber, and Rush 2017, 2018; Novikova, Dušek, and Rieser 2017; Fu, Bing, and Lam 2020; Fu et al. 2020c,a). On the other hand, language model based systems are applied in the open-ended generation tasks. For example, OpenAI proposes the GPT models (Radford et al. 2018, 2019; Brown et al. 2020) which are based on language models.

Many empirical out-of-the-box methods without too much model modification have been proposed to alleviate the repetition problem by designing proper sampling methods to replace the widely used greedy sampling method (Klein et al. 2017; Ott et al. 2019). The details of these methods have been discussed in Section 2.1. Apart from the out-of-the-box methods, Welleck et al. (2020) propose the unlikelihood training approach with new components to solve the problem of likelihood training.

6 Conclusions

We propose a novel theoretical analysis for the repetition problem. Our theory shows that the repetition problem is caused by the language itself. Too many high inflow words in the human language make it easy to go back to themselves and inevitably increase the repetition probability. Guided by this theory, we show that most of the existing methods are minimizing the ARP upper bounds explicitly or implicitly. Furthermore, we propose a novel rebalanced encoding method to tackle the repetition problem by reducing the high inflow terms. Our experiments validate our proposed theory and demonstrate the effectiveness of rebalanced encoding.

References

Ackley, D. H.; Hinton, G. E.; and Sejnowski, T. J. 1985. A learning algorithm for Boltzmann machines. *Cognitive science* 9(1): 147–169.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* .

Caccia, M.; Caccia, L.; Fedus, W.; Larochelle, H.; Pineau, J.; and Charlin, L. 2019. Language GANs Falling Short. In *International Conference on Learning Representations (ICLR)*.

Choi, Y. 2018. The Missing Representation in Neural Language Models. In *3rd Workshop on Representation Learning for NLP (RepL4NLP)*.

Fan, A.; Lewis, M.; and Dauphin, Y. 2018. Hierarchical Neural Story Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 889–898.

Ficler, J.; and Goldberg, Y. 2017. Controlling Linguistic Style Aspects in Neural Language Generation. In *Proceedings of the Workshop on Stylistic Variation*, 94–104.

Fu, Z.; Bing, L.; and Lam, W. 2020. Open Domain Event Text Generation. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 7748–7755.

Fu, Z.; Bing, L.; Lam, W.; and Jameel, S. 2020a. Dynamic topic tracker for kb-to-text generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, 2369–2380.

Fu, Z.; Lam, W.; So, A. M.-C.; and Shi, B. 2020b. A Theoretical Analysis of the Repetition Problem in Text Generation. *arXiv preprint arXiv:2012.14660* .

Fu, Z.; Shi, B.; Lam, W.; Bing, L.; and Liu, Z. 2020c. Partially-Aligned Data-to-Text Generation with Distant Supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 9183–9193.

Holtzman, A.; Buys, J.; Du, L.; Forbes, M.; and Choi, Y. 2020. The Curious Case of Neural Text Degeneration. In *International Conference on Learning Representations (ICLR)*.

Klein, G.; Kim, Y.; Deng, Y.; Senellart, J.; and Rush, A. 2017. OpenNMT: Open-Source Toolkit for Neural Machine Translation. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, System Demonstrations* 67–72.

Lebret, R.; Grangier, D.; and Auli, M. 2016. Neural Text Generation from Structured Data with Application to the Biography Domain. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1203–1213.

Lin, C.-Y. 2004. Rouge: A package for automatic evaluation of summaries. *Workshop on Text Summarization Branches Out* .

Luong, T.; Pham, H.; and Manning, C. D. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1412–1421.

Merity, S.; Xiong, C.; Bradbury, J.; and Socher, R. 2017. Pointer Sentinel Mixture Models. In *International Conference on Learning Representations (ICLR)*.

Nallapati, R.; Zhou, B.; dos Santos, C.; Guulçehre, Ç.; and Xiang, B. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, 280–290.

Novikova, J.; Dušek, O.; and Rieser, V. 2017. The E2E Dataset: New Challenges For End-to-End Generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, 201–206.

Ott, M.; Edunov, S.; Baevski, A.; Fan, A.; Gross, S.; Ng, N.; Grangier, D.; and Auli, M. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Annual meeting on Association for Computational Linguistics*, 311–318.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving Language Understanding by Generative Pre-Training .

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language Models are Unsupervised Multitask Learners .

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1715–1725.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, 3104–3112.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Vig, J. 2018. Deconstructing bert: Distilling 6 patterns from 100 million parameters. *Medium, December* .

Welleck, S.; Kulikov, I.; Roller, S.; Dinan, E.; Cho, K.; and Weston, J. 2020. Neural Text Generation With Unlikelihood Training. In *International Conference on Learning Representations (ICLR)*.

Wiseman, S.; Shieber, S.; and Rush, A. 2017. Challenges in Data-to-Document Generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2253–2263.

Wiseman, S.; Shieber, S.; and Rush, A. 2018. Learning Neural Templates for Text Generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 3174–3187.

Wu, Y.; Schuster, M.; Chen, Z.; Le, Q. V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* .