

# BFGAN: Backward and Forward Generative Adversarial Networks for Lexically Constrained Sentence Generation

Dayiheng Liu<sup>†</sup>, Jie Fu<sup>‡</sup>, Qian Qu<sup>†</sup>, Jiancheng Lv<sup>†\*</sup>

<sup>†</sup>Data Intelligence Laboratory, Sichuan University

<sup>‡</sup>MILA, IVADO, Polytechnique Montreal

losinuris@gmail.com

lvjiancheng@scu.edu.cn

## Abstract

Incorporating prior knowledge like *lexical constraints* into the model’s output to generate meaningful and coherent sentences has many applications in dialogue system, machine translation, image captioning, etc. However, existing RNN-based models incrementally generate sentences from left to right via beam search, which makes it difficult to directly introduce lexical constraints into the generated sentences. In this paper, we propose a new algorithmic framework, dubbed BFGAN, to address this challenge. Specifically, we employ a backward generator and a forward generator to generate lexically constrained sentences together, and use a discriminator to guide the joint training of two generators by assigning them reward signals. Due to the difficulty of BFGAN training, we propose several training techniques to make the training process more stable and efficient. Our extensive experiments on two large-scale datasets with human evaluation demonstrate that BFGAN has significant improvements over previous methods.

## 1 Introduction

In many real-world natural language generation scenarios, it is required to introduce *lexical constraints* into the generated sequences, which is called *lexically constrained* sentence generation (Hokamp and Liu, 2017). It generates a sentence that must contain specific words or phrases and is an active research topic in natural language generation. For examples, to avoid universal responses

in the dialogue, a contextual keyword can be introduced into the reply (Mou et al., 2016a). For machine translation, some specific domain terminologies may need to be introduced into the translation results (Post and Vilar, 2018). For image captioning, to mitigate the out-of-domain issue, image tag words can be incorporated into the output sentences (Anderson et al., 2017).

In recent years, the recurrent neural network (RNN) based models have made remarkable progress in several natural language generation tasks, including neural machine translation (Wu et al., 2016), product review generation (Dong et al., 2017), abstractive summarization (See et al., 2017), and affective text generation (Ghosh et al., 2017), etc. However, existing RNN based models usually generate sentences incrementally from left to right by beam search (Och and Ney, 2004). It is difficult for these models to directly generate sentences containing lexical constraints. Replacing an arbitrary word in output with the desired word will damage the fluency of the sentence. Given words as additional inputs, there is no guarantee that they will appear in the outputs (Anderson et al., 2017; Yin et al., 2016).

Various lexically constrained decoding methods (Anderson et al., 2017; Hokamp and Liu, 2017; Post and Vilar, 2018) have been proposed for lexically constrained sentence generation, some of which extends beam search to allow the inclusion of lexical constraints. However, these methods have high computational complexity (Post and Vilar, 2018). Moreover, these methods do not consider what specific words need to be included at the beginning of generation, but try to force specific words into sentences at every time step of the generation process. This unnatural way of generating may affect the quality of the generated sentence. We find that this problem is more serious when applying these methods for the uncon-

\*Correspondence to Jiancheng Lv.

ditional language generative models.

Another more natural way of generating lexically constrained sentences is based on a Backward and Forward Language Model (B/F-LM) introduced in (Mou et al., 2016b), where a backward and a forward language model work together to generate lexically constrained sentences. The backward language model takes the lexical constraint as input to generate the first half of the sentence. Then the forward language model takes this first half as input to generate the whole sentence. These two language models are trained using maximum likelihood estimation (MLE) objectives. As argued in (Bengio et al., 2015), they suffer from the exposure bias due to the discrepancy between training and inference stage: each language model generates a sequence conditioned on its previously generated ones during inference but they are trained to generate a sequence given ground-truth words.

More importantly, during preliminary experiments, we observed that the first half of the sentence generated by the backward language model and the second half of the sentence generated by the forward model tend to be semantically inconsistent or incoherent. This issue stems from the fact that the two language models are trained *separately*, but are required to output together. In other words, there is no interaction between the two language models during training, and neither of them can get access to the outputs from each other. We believe a better way is to jointly train the two language models and let them get access to each other’s output during training.

To this end, we propose a new method, dubbed Backward-Forward Generative Adversarial Network (BFGAN), to address the above issues for lexically constrained sentence generation. Our method contains three modules: a **Backward Generator**, a **Forward Generator**, and a **Discriminator**. We extend the B/F-LM (Mou et al., 2016b) in the following ways: In order to improve the coherence of the generation, we introduce the dynamic attention mechanism which is similar to the self-attention in (Mei et al., 2017; Vaswani et al., 2017) into the forward generator. With the dynamic attention mechanism, the scope of attention increases with the recursive operation from the beginning to the end of the sentence, and the forward generator can attend the first half sentence which is generated by the backward gener-

ator during inference. To solve the issue of discrepancies between training and inference in B/F-LM, we employ a discriminator to learn to distinguish real sentences and generated constrained sentences. This discriminator outputs reward signals to guide the joint training of the two generators. The two generators try to cooperate to generate lexically constrained sentences that can fool the discriminator.

Following SeqGAN (Yu et al., 2017) and LeakGAN (Guo et al., 2018), we model the lexically constrained sentence generation as a sequential decision making process and train the two generators with policy gradient. To our best knowledge, this is the first time that GANs have been used for generating lexically constrained sentences. Practically, it has been difficult to train GANs for text generation, especially to train two generator jointly for lexically constrained sentence generation. We propose several training techniques to make the training process more stable and efficient. With these training techniques, the BFGAN can be trained on large-scale datasets and generate long lexically constrained sentences (more than 20 words), and the BFGAN can be trained without additional labels.

We conduct extensive experiments on two large-scale datasets. For evaluation, we use BLEU score and self-BLEU score (Zhu et al., 2018) to test the fluency and diversity of the generated sentences. In addition, we build a user-friendly web-based environment and launch a crowd-sourcing online study for human evaluation. In all cases, BFGAN shows significant improvements compared to the previous methods.

## 2 Related Work

### 2.1 Lexically Constrained Decoding

To incorporate pre-specified words or phrases in the output, one solution is lexically constrained decoding, which relies on a variant of beam search. For example, (Hokamp and Liu, 2017) propose grid beam search to force certain words to appear in the output for machine translation. This method can be seen as adding an additional constrained dimension to the beam. The hypothesis must meet all constraints before they can be considered to be completed. To make existing image captioning models generalize to out-of-domain images containing novel scenes or objects, (Anderson et al., 2017) propose constrained beam

search based on the finite-state machine, which is an approximate search algorithm capable of enforcing image tag over resulting output sequences. (Post and Vilar, 2018) point out that these two methods are time-consuming, and their computation graphs are difficult to be optimized when running on GPUs. They propose a fast lexically-constrained decoding based on dynamic beam allocation (DBA), which can be seen as a fast version of grid beam search.

Nevertheless, these methods still generate sentences from left to right. They do not consider what specific words need to be included at the beginning of generation, but try to force specific words into sentences at every time step of the generation process, which may affect the quality of the generated sentence. We find that this problem is more serious when applying these methods for the unconditional language generative models. Because our method starts with the given word, it has already known the lexical constraint before the generation, which is a more natural way. Our experimental results show that the BFGAN significantly outperforms the grid beam search (Hokamp and Liu, 2017) based unconditional language generative models.

## 2.2 Backward and Forward Language Modeling

To generate lexically constrained sentences, (Mou et al., 2016b) first propose three variant backward and forward language models: separated, synchronized, and asynchronously backward and forward language model (called sep-BF, syn-BF, and asyn-BF). Their experiments show that asyn-BF is the most natural way of modeling forward and backward sequences. (Mou et al., 2016a) incorporate the asyn-BF model into the seq2seq framework to generate conversation reply containing the given keyword.

The way BFGAN generates lexically constrained sentences is similar to asyn-BF as discussed in Introduction section. However, as discussed above, training backward and forward models separately with MLE will suffer from serious exposure bias, especially for unconditional language models. The two models in asyn-BF have never been trained together. The essential difference of our method is that it incorporates a discriminator to guide the joint training of the two generators. The experimental results show that

BFGAN performs constantly better than state-of-the-art models.

## 2.3 GANs for Text Generation

Applying adversarial learning (Goodfellow et al., 2014) to text generation has recently drawn significant attention, which is a promising framework for alleviating the exposure bias issue (Huszár, 2015). (Yu et al., 2017) take the sequence generation as a sequential decision making process. They propose SeqGAN which uses the output of the discriminator as reward and trains generator with policy gradient. (Li et al., 2017) extend SeqGAN model for dialogue generation. Instead of training binary classification as discriminator, (Lin et al., 2017) propose RankGAN that uses a ranker to make better assessment of the quality of the samples. MaliGAN (Che et al., 2017) uses importance sampling combined with the discriminator output to deal with the reward vanishing problem. To address the sparse reward issue in long text generation, (Guo et al., 2018) propose LeakGAN which leaks the feature extracted by the discriminator to guide the generator.

BFGAN is distinct in that it employs two generators for lexically constrained sentence generation, and employs a discriminator to guide the joint training of the two generators. In order to solve the problem of difficult training of the model. We propose several training techniques to make the training process more stable and efficient. To the best of our knowledge, this is the first time a GAN has been successfully used for generating lexically constrained sentences.

## 3 Methodology

### 3.1 Overall Architecture

The process of generation is shown in Figure 1, which involves three modules:

**Discriminator.** It learns to distinguish real sentences from machine-generated lexically constrained sentences. It guides the joint training of two generators by assigning them reward signals.

**Backward Generator.** Given a lexical constraint<sup>1</sup>, the backward generator takes it as the sentence’s starting point, and generates the first half sentence backwards.

**Forward Generator.** The sequence produced by the backward generator is reversed and fed into the forward generator. It then learns to generate the

<sup>1</sup>It can be a word, a phrase or a sentence fragment.

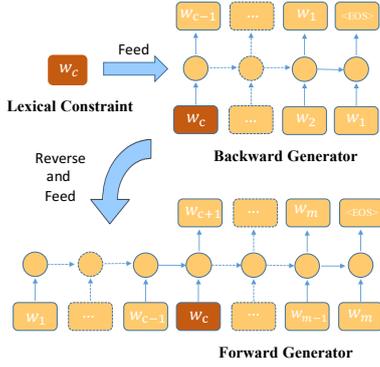


Figure 1: The generation process of backward and forward generators.

whole sentence with the aim of fooling the discriminator.

### 3.2 Backward and Forward Generators

Let  $w_c$  denote the given lexical constraint. We denote the generated lexically constrained sentence with length  $m$  as  $s = w_1 \cdots w_c \cdots w_m$ <sup>2</sup>. The backward half sentence  $s_{<c} = w_{c-1} \cdots w_1$  is generated by the backward generator  $G_\theta^{(bw)}$ , and the forward half sentence  $s_{>c} = w_{c+1} \cdots w_m$  is generated by the forward generator  $G_{\theta'}^{(fw)}$ .  $\theta$  and  $\theta'$  represent the parameters of the backward generator and the forward generator respectively. The joint probability of the sentence can be written as:

$$G(s|w_c; \theta, \theta') = P_\theta^{(bw)}(s_{<c}|w_c)P_{\theta'}^{(fw)}(s_{>c}|s_{1:c}), \quad (1)$$

where  $s_{1:c} = w_1 \cdots w_c$ .

The backward generator  $G_\theta^{(bw)}$  models the probability of the backward half sentence:

$$P_\theta^{(bw)}(s_{<c}|w_c) = \prod_{i=1}^{c-1} P_\theta^{(bw)}(w_{c-i}|w_c \cdots w_{c-i+1}), \quad (2)$$

The forward generator  $G_{\theta'}^{(fw)}$  models the probability of the forward half sentence:

$$P_{\theta'}^{(fw)}(s_{>c}|s_{1:c}) = \prod_{i=1}^{m-c} P_{\theta'}^{(fw)}(w_{c+i}|w_1 \cdots w_{c+i-1}). \quad (3)$$

The two generators have the same structure but have distinct parameters. To improve the coherence of the constrained sentence, we employ an

<sup>2</sup>Note that  $w_c$  can also appear at the beginning, or the end.

LSTM-based language model with dynamic attention mechanism (called **attRNN-LM**) as generator. This attention mechanism is similar to attention in RNN-Seq2Seq (Bahdanau et al., 2014), but its scope of attention increases dynamically with the recursive operation from the beginning to the end of the sentence. In contrast, the RNN-Seq2Seq’s attention scope is fixed to the hidden states of the encoder.

The architecture of attRNN-LM is shown in Figure 2. As with the RNN-LM, the model encodes the input sequence  $w_1 \cdots w_t$  into a sequence of hidden states  $h_1 \cdots h_t$ . The attention context vector  $z_t$  at step  $t$  is computed as a weighted average of history hidden states  $h_{0:t-1}$ :

$$e_{ti} = v^T \tanh(W^a h_{t-1} + U^a h_i), \quad (4)$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=0}^{t-1} \exp(e_{tk})}, \quad (5)$$

$$z_t = \sum_{i=0}^{t-1} \alpha_{ti} h_i. \quad (6)$$

There are two variants of attRNN-LM: the first one called **attoRNN-LM** (see Figure 2(a)) is similar to A-RNN in (Mei et al., 2017), which uses the context vector  $z_t$  together with hidden state  $h_t$  to predict the output at time  $t$ :

$$h_t = \text{LSTM}(w_{t-1}, h_{t-1}), \quad (7)$$

$$P(w_t|s_{<t}) = \text{Softmax}\left(O^T(W^h h_t + W^z z_t)\right). \quad (8)$$

Another variant called **attinRNN-LM** (see Figure 2(b)) uses the linear combination of  $z_t$  and  $w_{t-1}$  as input to RNN to get the hidden state  $s_t$  and output at time  $t$ :

$$g_t = W^e w_{t-1} + W^z z_t \quad (9)$$

$$h_t = \text{LSTM}(g_t, h_{t-1}), \quad (10)$$

$$P(w_t|s_{<t}) = \text{Softmax}(O^T h_t). \quad (11)$$

The attRNN-LM allows for a flexible combination of history generated words for every individual word. The forward generator can attend the first half sentence which is generated by the backward generator during inference. Empirically, we find that both the attinRNN-LM and attoRNN-LM perform better than RNN-LM for our problems.

### 3.3 Discriminator

The discriminator  $D_\phi$  learns to distinguish real sentences and generated lexically constrained sen-

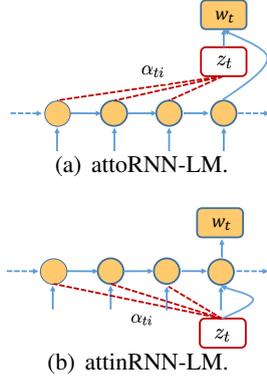


Figure 2: The architecture of two variants of attRNN-LM.

tences. It guides the joint training of two generators by assigning proper reward signals. This module can be a binary classifier or a ranker (Lin et al., 2017). We use CNN based model (Kim, 2014) as the discriminator which outputs a probability indicating whether the input is generated by humans or machines in the experiments.<sup>3</sup>

### 3.4 Training

#### 3.4.1 Pre-training

The two generators are firstly pre-trained by standard MLE loss. Note that our method can be trained without additional labels. Given a sentence, we randomly sample a notional word for it as the lexical constraint, then we take the first half and reverse its word order as a training sample for backward generator. We repeat this way to construct multiple training samples for each real sentence. The forward model is trained the same as the standard RNN language model.

After pre-training the generators, we pre-train the discriminator by providing positive examples from the human-written sentences and negative examples generated from the generators. Given a positive sample  $s$ , we construct multiple negative samples  $\hat{s}$  in the following ways: 1) We slice a random sentence fragment in the middle of the sample  $s$ , and take this sentence fragment as a negative sample. For example, given a sentence “my pastor recommended this book, and it was really insightful.”, we may get a sentence fragment like this: “recommended this book, and it was really”; 2) We take the sentence fragment as the lexical constraint for BFGAN to generate a complete sentence by beam search; 3) We use BFGAN to

<sup>3</sup>We also tried LSTM-based discriminators but found CNN-based model is more effective and efficient.

generate a complete sentence containing the lexical constraint from sampling; 4) We use the backward generator to generate a half sentence without any constraint by sampling and the forward generator to generate a complete sentence by sampling. By constructing different types of strong negative samples and weak negative samples, the generalization ability of the discriminator is increased. Therefore it can provide better reward signals for the generators. We train the discriminator using gradient ascend of the discriminator loss:

$$J(\phi) = \log D_\phi(s) + \log(1 - D_\phi(\hat{s})). \quad (12)$$

#### 3.4.2 Policy Gradient Training

The discriminator’s output  $D_\phi(\hat{s})$  is the probability that the generated sentence  $\hat{s}$  is written by human. We use  $D_\phi(\hat{s})$  as the reward  $r$  to encourage the two generators to work together to generate sentences which are indistinguishable from human-written sentences. The generators are trained by the REINFORCE algorithm (Williams, 1992):

$$J(\theta, \theta') = \mathbb{E}_{s \sim G(s|w_c; \theta, \theta')} D_\phi(s). \quad (13)$$

The gradient of Eq. (13) is approximated using the likelihood ratio trick (Sutton et al., 2000; Glynn, 1990):

$$\nabla J(\theta, \theta') \approx (D_\phi(s) - b) \nabla \log G(s|w_c; \theta, \theta'), \quad (14)$$

where  $b$  is the baseline value to reduce the variance of the estimate.

#### 3.4.3 Training Techniques

To construct the lexical constraints during policy gradient training, we randomly sample a batch of real sentence at each step. For each real sentence  $s$  with length  $m$ , we slice it from the middle into a sentence fragment  $s_c$  at length  $m - \Delta \times i$ . This sentence fragment is taken as the lexical constraint and fed into the generators to generate a fake example.

In order to gradually teach the model to produce stable sentences, we provide more and more difficult lexical constraints, which is a form of curriculum learning (Bengio et al., 2009). We increase the  $i$  to make the lexical constraints shorter and shorter. For generators, the part of lexical constraint  $s_c$  is trained with cross entropy loss and the

remaining generated words are trained with policy gradient. To make the training stable and prevent the perplexity value skyrocketing, we apply teacher forcing to giving the generators access to the gold-standard targets after each policy training step.

Monte Carlo (MC) search is used to estimate the reward for every generation step in (Yu et al., 2017). However this sampling process is time-consuming especially for long sentence generation (Li et al., 2017), thus making it hard to be applied to large-scaled datasets. In our experiment, given the lexical constraint, the generators use beam search to generate the whole sentence  $\hat{s}$ , which is consistent with the inference stage. Then we get the reward  $D_\phi(\hat{s})$  for each generated word from the discriminator. We empirically found this technique can improve the efficiency and stability of training.

The details of training procedure are summarized in Algorithm 1.

---

**Algorithm 1** Training Algorithm of BFGAN

---

**Require:** A backward generator  $G_\theta^{(bw)}$  with parameters  $\theta$ .  
 A forward generator  $G_{\theta'}^{(fw)}$  with parameters  $\theta'$ .  
 A discriminator  $D_\phi$  with parameters  $\phi$ .  
 A baseline  $b$ , step size  $\Delta$ , maximum length  $T$ .

- 1: Pre-train the  $G_{\theta'}^{(fw)}$ ,  $G_\theta^{(bw)}$  and  $D_\phi$  for certain epochs.
  - 2: **for**  $K = T, 1, -\Delta$  **do**
  - 3:   **for** number of training iterations **do**
  - 4:     Sample  $s$  from the dataset. Get the lexical constraints  $s_c$  with length  $K$ .
  - 5:     Given  $s_c$ , let  $G_{\theta'}^{(fw)}$  and  $G_\theta^{(bw)}$  generate the whole sentence  $\hat{s}$  using beam search.
  - 6:     Compute the reward  $r$  of  $\hat{s}$  using  $D_\phi$ .
  - 7:     Update  $G_\theta^{(bw)}$  on  $\hat{s}_{<c}$  with reward  $r$ .
  - 8:     Update  $G_{\theta'}^{(fw)}$  on  $\hat{s}_{>c}$  with reward  $r$ .
  - 9:     Update  $G_\theta^{(bw)}$  and  $G_{\theta'}^{(fw)}$  on  $s_c$  using cross entropy.
  - 10:    Teacher-Forcing: Update  $G_\theta^{(bw)}$  and  $G_{\theta'}^{(fw)}$  on  $s$ .
  - 11:    **for**  $i = 1, D$ -steps **do**
  - 12:     Sample  $s$  from real data, construct the negative samples  $\hat{s}$ .
  - 13:     Update  $D_\phi$  using  $s$  as as positive samples and  $\hat{s}$  as negative samples.
  - 14:    **end for**
  - 15:    **end for**
  - 16: **end for**
- 

## 4 Experiments

The experiments are designed for answering the following questions: **Q1:** Since we employ two variants attinRNN-LM as the generators, compared with RNN-LM, which model is the most effective one? **Q2:** What is the performance of the model after training with the proposed adversarial learn-

ing. **Q3:** Judging from the human views, how does the proposed approach compare with the existing methods?

### 4.1 Datasets

Two large-scale datasets are used here. The first one is the Amazon Product Reviews Corpus (APRC) (Dong et al., 2017) which is built upon Amazon product data (McAuley et al., 2015). We set the vocabulary size to 10K, and the maximum length of the sentences to 35. After preprocessing, the first corpus contains 350K review sentences.

The second dataset is Chinese Sentence Making Corpus (CSMC).<sup>4</sup> We manually crawl and build this dataset which contains 107K Chinese common words or idioms. Each common word or idiom have several example sentences which contain that word or idiom. There are 1.34M example sentences. The maximum length of these sentences is 30. Compared with the APRC dataset, the CSMC is a more open-domain dataset and training the model on it will be more challenging. These two datasets are randomly split into train/valid/test sets following these ratios respectively: 85%, 5%, 10%.

### 4.2 Structure Comparison of Generator (Q1)

To answer the first question, we train the forward generators with RNN-LM, attinRNN-LM, and attorRNN-LM structures on CSMC dataset separately. All the generators are 2-layer char-level LSTMs with 1024 hidden units. The dimension of word embeddings is set to 1024. The batch size, dropout rate, threshold of element-wise gradient clipping and initial learning rate of Adam optimizer (Kingma and Ba, 2015) are set to 128, 0.5, 5.0 and 0.001. Layer Normalization (Ba et al., 2016) is also applied.

We use the perplexity (PPL) metric to evaluate the generators. In addition, we use BLEU score as an additional evaluation metric. Following (Yu et al., 2017; Lin et al., 2017), we randomly sample 10,000 sentences from the test set as the references. For each generator, we let it generate 10,000 sentences by sampling. We report the average BLEU-4 scores of the generated sentences. The results are shown in Table 1. From the results we can see that both attinRNN-LM and attorRNN-LM perform better than RNN-LM, and attinRNN-

<sup>4</sup>In order to make our results easy to reproduce, we will release all the datasets and codes upon acceptance of the paper.

LM achieves the best performance. In the following experiments, we use the attinRNN-LM structure as the backward and the forward generators.

Models	PPL (dev)	PPL (test)	BLEU-4
RNN-LM	17.95	18.15	0.193
attoRNN-LM	16.18	16.37	0.219
attinRNN-LM	<b>15.81</b>	<b>16.07</b>	<b>0.225</b>

Table 1: The evaluation results of different generators on CSMC.

Models	BLEU-2	BLEU-3	BLEU-4
B/F-LM	0.799	0.476	0.190
BF-MLE	0.798	0.482	0.203
BFGAN	0.836	0.543	0.248
BFGAN-Mali	0.829	0.536	0.246
BFGAN-rescale	<b>0.850</b>	<b>0.574</b>	<b>0.295</b>

Models	sBLEU-2	sBLEU-3	sBLEU-4
B/F-LM	0.807	0.498	0.236
BF-MLE	0.810	0.516	0.269
BFGAN	0.852	0.624	0.399
BFGAN-Mali	0.849	0.617	0.392
BFGAN-rescale	0.849	0.644	0.445

Table 2: BLEU and self-BLEU (sBLEU) scores on CSMC.

Models	BLEU-2	BLEU-3	BLEU-4
B/F-LM	0.843	0.653	0.527
BF-MLE	0.854	0.665	0.539
BFGAN	0.920	0.796	0.672
BFGAN-Mali	0.920	0.803	0.675
BFGAN-rescale	<b>0.926</b>	<b>0.814</b>	<b>0.698</b>

Models	sBLEU-2	sBLEU-3	sBLEU-4
B/F-LM	0.924	0.811	0.680
BF-MLE	0.925	0.810	0.681
BFGAN	0.937	0.837	0.721
BFGAN-Mali	0.936	0.834	0.717
BFGAN-rescale	0.937	0.840	0.729

Table 3: BLEU and self-BLEU (sBLEU) scores on APRC.

### 4.3 Evaluation of BFGAN (Q2)

In this experiment, we train the BFGAN on both CSMC and APRC datasets. The hyperparameters of the backward and the forward generators trained on CSMC are the same as those in the first experiment. Since the APRC dataset is smaller than CSMC, we set both backward and forward generators to one layered word-level LSTM with 1024 hidden units when training on APRC.

We take the BF-MLE (the backward and the forward generators without adversarial learning), and the B/F-LM (the backward and forward LSTM

based language models) (Mou et al., 2016b) as the baselines. In addition, we compare the basic BFGAN with BFGAN-rescale (the BFGAN which use Bootstrapped Rescaled Activation (Guo et al., 2018) to rescale the reward during training), and the BFGAN-Mali (the BFGAN that use the same reward rescaling as in the MaliGAN (Che et al., 2017) during training). To be fair, all models use roughly the same number of parameters.

As with the evaluation in (Zhu et al., 2018), we use BLEU score to measure the similarity degree between the generated sentences and the real sentences. We randomly sample 10,000 sentences from the test set as the references. Moreover, we use self-BLEU score to evaluate the diversity of the generated sentences. For each model, we let its backward generator generate 10,000 half sentences and its forward generator generate complete sentences from sampling.

The evaluation results are shown in Table 2 and Table 3. Firstly, we can see that the BF-MLE is better than B/F-MLE. This result shows the effectiveness of attinRNN-LM again. Secondly, we find that the BLEU-4 score of BF-MLE is lower than the forward generator (see Table 1). In other words, the fluency of the sentences generated by the two generators together is worse than that generated by the forward generator only. These results prove what we mentioned above, the two generators trained *separately* indeed suffer from more serious exposure bias than one generator. However, after training the two generators with the proposed adversarial learning, the BLEU scores of all BFGAN variants are much higher than those of BF-MLE and B/F-LM on both datasets. These results show that BFGAN performs much better than B/F-LM. In addition, the BFGAN-rescale achieves the highest BLEU scores.

From the results of self-BLEU score, compared with other models, we find that the BFGAN-rescale suffers from more serious mode collapse (Arjovsky et al., 2017). On both datasets, we can see that the self-BLEU scores of BF-MLE and B/F-LM are lower than those of other models. The BF-MLE and B/F-LM can generate sentences with higher diversity. These results are in line with the evaluation in (Zhu et al., 2018), the self-BLEU score of various text GAN is higher than that of model trained by MLE. We can see that the self-BLEU scores on APRC dataset of BFGAN are close to those of BF-MLE and B/F-LM,

but for CSMC dataset, the self-BLEU scores of B/F-LM are much lower than those of BFGAN variants. We further analyzed this reason. Because the CSMC dataset is a more open-domain dataset, the B/F-LM suffers from more serious exposure bias on this dataset and often generates unreal samples. These low quality samples may have greater diversity.

#### 4.4 Human Evaluation (Q3)

To further evaluate the proposed method, we conduct several human evaluations. In this experiment, we compare the BFGAN with BF-MLE and grid beam search (Hokamp and Liu, 2017). In addition, we compare BFGAN with human, which can be regarded as a Turing test. All comparisons are blind paired comparisons.

We randomly pick 50 common words or idioms as lexical constraints, and let all models generate a lexically constrained sentence for each lexical constraint. For grid search, we use forward generator with grid beam search algorithm to force the lexical constraint appear in the output. To be fair, all models are the same structure. For each lexical constraint, we randomly pick a real sample that contains this constraint from the dataset as a human-written sample. Then we launch a crowdsourcing online study asking evaluators to decide which generated sentence is better. We build a user-friendly web-based environment for this experiment, and the interface for human interaction is illustrated in Appendix. For each round, the human evaluation interface presents two sentences which are generated by two different methods with the same lexical constraint, then asks evaluators to choose the better one. Ties are permitted. A total of 50 evaluators<sup>5</sup> participate in the evaluation.

Table 4 presents the results on CSMC dataset. Only 4% sentences generated by the grid beam search and 6% sentences generated by BF-MLE were considered better than those produced by the BFGAN. We can observe a significant quality improvement from the BFGAN. From the result of comparison between BFGAN and human-written samples, we can see that 56% sentence groups make the evaluators unable to tell which is better. Surprisingly, there is 6% sentences generated by the BFGAN are considered even better than human-written samples. These results show

<sup>5</sup>All evaluators are well educated and have Bachelor or higher degree. They are independent of the authors' research group.

that our method can generate high quality lexically constrained sentences.

BFGAN won	BF-MLE won	Tied
<b>76%</b>	6%	18%
BFGAN won	Grid Beam Search won	Tied
<b>82%</b>	4%	14%
BFGAN won	Human won	Tied
6%	38%	<b>56%</b>

Table 4: Human evaluations on CSMC. The comparison result between BFGAN and BF-MLE is shown at the top. The comparison result between BFGAN and grid beam search is shown in the middle. At the bottom of the table is the comparison result between BFGAN and human-written samples.

## 5 Qualitative Analysis

We show some evaluation examples in Figure 3. We can see that the BFGAN can generate high-quality lexical constrained sentences. Some of them are even indistinguishable from sentences written by human. While most sentences generated by BF-MLE are lack of coherence and fluency. In addition, the quality of the sentence generated by grid search is the worst. Since we introduce the dynamic attention mechanism into the generators, we qualitatively evaluate the effectiveness of it through the visualizations of the attention weights. Figure 4 provides two visualization samples of the attention weights in the BFGAN. Figure 4 (a) and (b) present the generated sentences and we highlight in red two output words for two time steps. For each highlighted generated word, we visualize the attention weights of each previous generated word, where darker shades indicate larger attention weights. We can see that the dynamic attention mechanism helps learn a better generator that promotes the coherence, by learning to associate the currently-generated word with informative contextual words. We show some examples of APRC dataset in Appendix.

## 6 Conclusion and Future Work

In summary, our key contributions are as follows:

- We propose a new method, called BFGAN, to address the challenge of generating sentences with lexical constraints. BFGAN significantly improves the quality of lexically con-

**BFGAN:** 听到地震的[消息],他踉踉跄跄跑出去,几乎是楼梯上滚下来。  
Hearing the earthquake [news], he staggered out and almost rolled down the stairs.  
**BF-MLE:** 你没听过的好[消息],大家都听不懂他的话。  
You haven't heard your good [news], no one could understand him.  
**GRID:** 因为[消息]已经开始,我们的生命已经来临。  
Because [news] has begun, our lives have arrived.  
**REAL:** 老虎下山的[消息]使整个山村骚动起来。  
The [news] of the tiger's downhill caused the whole village to stir up.

**BFGAN:** 忍,是一种[坚强],痛苦也是一种智慧。  
Tolerance is a kind of [strong], and pain is also a kind of wisdom.  
**BF-MLE:** 一个人的生活就像海洋,只有意志[坚强]的海湾。  
One's life is like an ocean, with only a bay of [strong] will.  
**GRID:** 本发明[坚强],耐磨性能强,耐碱性强,耐碱性强,耐碱性强。  
The invention of [strong] has strong abrasion resistance and strong alkali resistance.  
**REAL:** [坚强],在常人眼中似乎难以理解。  
In the eyes of ordinary people, it seems hard to understand [strong].

**BFGAN:** 真正的[天才],只不过是把别人喝咖啡的功夫都用在工作上了。  
The real [genius] is simply putting other people's coffee time to work.  
**BF-MLE:** 他是一个好[天才],吃得不能不类,没什么事情都干不出来。  
He is a good [genius], eating neither fish nor fowl, no matter what he can not do.  
**GRID:** 我最喜欢秋[天才]更喜欢秋意绵绵的河海校园。  
I like autumn best. I prefer the river and sea campus with autumn scenery.  
**REAL:** 在要求[天才]的产生之前,应该先要求可以使天才生长的民众。  
Before asking for [genius], we should ask for the people who can make it grow.

**BFGAN:** 敌人被我军打得敌人[抱头鼠窜],丢盔弃甲。  
The enemy threw away shield and armor, [covered their head and scurried away].  
**BF-MLE:** 敌人要给[抱头鼠窜]的敌人以喘息之机。  
Enemies will give [scared] enemies a chance to kick their breath.  
**GRID:** 只有[抱头鼠窜]的匪徒,才能有所不为。  
Only the [scared] bandits are able to do nothing.  
**REAL:** 英勇的我军打得敌人[抱头鼠窜]。  
The valiant army sent the enemy [on a run].

**BFGAN:** 我们要珍惜水资源,保护[农作物],我们共同创造。  
We must cherish the water resources and protect [crops]. We create together.  
**BF-MLE:** 但是这种[农作物]的发展也很大。  
But the development of this kind of [crop] is also great.  
**GRID:** 刹那间,他看到[农作物]发出一副楚楚的样子。  
He saw [crop] and for a moment he looked pitiful.  
**REAL:** 引进[农作物]新品种,先要小面积试种,然后才能大面积推广。  
The introduction of new varieties of [crops] requires a small area for trial production before they can be popularized in large areas.

**BFGAN:** 夏天的夜空中,[星星]们闪烁着自己的光辉。  
In the summer sky, [stars] shine their own brilliance.  
**BF-MLE:** 天上的[星星]就像一颗小宝石一样晶莹剔透。  
The [stars] in the sky are as clear as a little gem.  
**GRID:** 那[星星]也跳起了舞,唱起了歌,为月的满而挥洒自己的光芒。  
The [stars] also danced, sang songs, and spread their radiance for the full moon.  
**REAL:** 最前面的是它的头,有着[星星]般的小眼睛和一张棕色的小嘴。  
At the front is its head, with small eyes like [stars] and a small brown mouth.

**BFGAN:** 雨水过后,[小河]两岸一片新绿的新叶,又像一个新绿的绸子。  
After the rain, new leaves appeared on both sides of the [river], like a new green silk.  
**BF-MLE:** 在寒冷的冬天,[小河]是成熟的。  
In the cold winter, the [river] is mature.  
**GRID:** 被[小河]弯弯曲曲,像一条丝带在飘扬。  
Twisted by a [river], like a ribbon flying.  
**REAL:** 家乡的[小河],美丽的小河。  
The river in my hometown is a beautiful [river].

**BFGAN:** 他既是一个[大人物],也该有一颗善良心灵的心。  
He is a [great man] and should have a good heart.  
**BF-MLE:** 我们都是时代的伟[大人物],我们应该尊重自己的尊严。  
We are all [great men] of the times. We should respect our dignity.  
**GRID:** 接着,我们按照同样的方法,把[大人物]的地方扫干净。  
Then, we follow the same method to clean up the place where the [great man] live.  
**REAL:** 追随伟[大人物]的思想,是最富有趣味的一门科学。  
Following the idea of a [great man] is the most interesting science.

**BFGAN:** 英勇的战士们[夜以继日]地工作者,为祖国的事业奋斗。  
The valiant soldiers worked [day and night] for the cause of their country.  
**BF-MLE:** 漫长[夜以继日],闻鸡起舞,闻鸡起舞,无所不能。  
Long [night and day], rising up upon hearing the crow of a rooster and practice with the sword. All things are possible.  
**GRID:** 一个[夜以继日]的夜晚,一个人静静的等待着冬天的到来。  
On a night after [day and night], one is waiting for the winter to come.  
**REAL:** 工人体育馆东门有个信号灯,[夜以继日]的工作者。  
There is a signal lamp on the east gate of the worker's gymnasium, working [day and night].

**BFGAN:** 即使[徒劳无功],也绝不会因此而丧失自己的尊严。  
Even if it is [fruitless], one will never lose dignity.  
**BF-MLE:** 设计师[徒劳无功],坐卧不安坐卧。  
The designer is [in vain]. He is restless and sleeps.  
**GRID:** 与其说它是[徒劳无功],它不如说它是不说话。  
It is not so much silent as [useless].  
**REAL:** 伸手摘星,即使[徒劳无功],亦不致一手污泥。  
Reaching out for a star, even if it [does not work], will not cause any sludge.

Figure 3: Some generated examples. GRID: grid beam search, REAL: real samples. We use [-] to highlight the lexical constraints. Since the original generated sentences are Chinese, we manually translate them into English alongside.



Figure 4: Visualization of attentions. The blocks under the words represent the attention weights.

strained sentence generation compared with state-of-the-art methods.

- To ease the training of BFGAN, we propose an efficient and stable training algorithm. Furthermore, it does not need additional labels.
- Extensive experiments based on two large-scale datasets and human evaluations demonstrate significant improvements over previous methods.

For future work, we plan to extend our method

to sequence-to-sequence frameworks and apply it to more nature language generation applications like dialogue systems. It might also be interesting to explore how to generate sequences with multiple lexical constraints.

## Acknowledgment

This work was supported by the National Science Foundation of China (Grant No.61625204), partially supported by the State Key Program of National Science Foundation of China (Grant No.61432012 and 61432014).

## References

- Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2017. Guided open vocabulary image captioning with constrained beam search. In *EMNLP*.
- Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875*.

- Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450* .
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .
- Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *NIPS*.
- Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ICML*.
- Tong Che, Yanran Li, Ruixiang Zhang, R Devon Hjelm, Wenjie Li, Yangqiu Song, and Yoshua Bengio. 2017. Maximum-likelihood augmented discrete generative adversarial networks. *arXiv preprint arXiv:1702.07983* .
- Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. 2017. Learning to generate product reviews from attributes. In *ACL*.
- Sayan Ghosh, Mathieu Chollet, Eugene Laksana, Louis-Philippe Morency, and Stefan Scherer. 2017. Affect-Im: A neural language model for customizable affective text generation. In *ACL*.
- Peter W Glynn. 1990. Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM* .
- Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NIPS*.
- Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. 2018. Long text generation via adversarial training with leaked information. In *AAAI*.
- Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *ACL*.
- Ferenc Huszár. 2015. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101* .
- Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*.
- Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.
- Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. 2017. Adversarial learning for neural dialogue generation. In *EMNLP*.
- Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, and Ming-Ting Sun. 2017. Adversarial ranking for language generation. In *NIPS*.
- Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *SIGKDD*.
- Hongyuan Mei, Mohit Bansal, and Matthew R Walter. 2017. Coherent dialogue with attention-based language models. In *AAAI*.
- Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016a. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. In *COLING*.
- Lili Mou, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016b. Backward and forward language modeling for constrained sentence generation. *Computer Science* .
- Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational linguistics* .
- Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *NAACL-HLT*.
- Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*.
- Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.
- Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*.
- 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* .
- Jun Yin, Xin Jiang, Zhengdong Lu, Lifeng Shang, Hang Li, and Xiaoming Li. 2016. Neural generative question answering. In *IJCAI*.
- Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*.
- Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Tegygen: A benchmarking platform for text generation models. In *SIGIR*.

## 7 Appendix

### 7.1 Interface of Human Evaluation

We build a user-friendly web-based environment based on Flask<sup>6</sup> for human evaluation experiment. The interface for human interaction is illustrated in Figure 5.



Figure 5: The human evaluation interface presents two sentences which are generated by two different methods with the same lexical constraint at a time. It asks users to choose the better one. Ties are permitted.

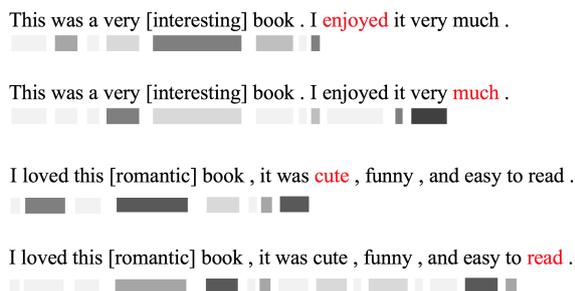


Figure 6: Visualization of attentions. The blocks under the words represent the attention weights. Darker shades indicate larger attention weights. We use [·] to highlight the lexical constraints.

### 7.2 Samples

We show some generated examples of different models trained on ARPC in Figure 7.

### 7.3 Visualization of Attentions

Figure 6 provides four visualization samples of the attention weights in the BFGAN trained on ARPC dataset. We highlight in red four output words for four time steps. For each highlighted generated word, we visualize the attention weights of each

**BFGAN:** I highly recommend this book. you wo n't be [disappointed].  
**BF-MLE:** I might try others , but I do prefer her to be [disappointed] a bit.  
**GRID:** Three of the three books were fabulous, you will not be [disappointed].  
**REAL:** I was [disappointed] but will keep looking for the right book.

**BFGAN:** All of her books are [outstanding], she is one of my favorite authors.  
**BF-MLE:** Outstanding writing, [outstanding] story .  
**GRID:** This is an [outstanding] book, I could n't put it down.  
**REAL:** Once again the ilona andrews team creates an [outstanding] novel .

**BFGAN:** This is a [beautiful story] of love, loss, and finding love. I ca n't wait to read the next one.  
**BF-MLE:** Karen rose always writes [beautiful story], found this one being boring.  
**GRID:** I do wish the book were a little longer, but it was a [beautiful story] to read.  
**REAL:** [Beautiful story] of survival and love, this is one of the best stories I have read in a long time.

**BFGAN:** I really enjoyed this book. I could n't wait to get back [to the beginning] to read it again.  
**BF-MLE:** I liked this book right up [to the beginning] to the end.  
**GRID:** This book kept me going back [to the beginning] to see what was going to happen next.  
**REAL:** All james patterson books are great reads. Going back [to the beginning] and reading them all.

**BFGAN:** An interesting story set in [San Francisco], the characters were well developed.  
**BF-MLE:** The history of [San Francisco] during century England during the late 19th century England.  
**GRID:** I enjoyed this story set in my hometown [San Francisco] area .  
**REAL:** It was an interesting story. It was also interesting to learn how [San Francisco] was in earlier times.

**BFGAN:** This was my first [CJ Lyons] book and I really enjoyed it, I will definitely read more.  
**BF-MLE:** I ca n't wait to read more books by [CJ Lyons], but I was immediately hooked.  
**GRID:** I could n't put it down. I highly recommend [CJ Lyons].  
**REAL:** Very dramatic suspense story. Looking forward to more from [CJ Lyons]!

Figure 7: Some generated examples on ARPC. GRID: grid beam search, REAL: real samples. We use [·] to highlight the lexical constraints.

previous generated word, where darker shades indicate larger attention weights.

<sup>6</sup><http://flask.pocoo.org/>