TextGAIL: Generative Adversarial Imitation Learning for Text Generation

Qingyang Wu [1], Lei Li [2], Zhou Yu [1],
[1]University of California, Davis,
[2]ByteDance AI Lab,
{wilwu, joyu}@ucdavis.edu, lileilab@bytedance.com

Abstract

Generative Adversarial Networks (GANs) for text generation have recently received many criticisms, as they perform worse than their MLE counterparts (Caccia et al. 2020; Tevet et al. 2019; Semeniuta, Severyn, and Gelly 2018). We suspect previous text GANs' inferior performance is due to the lack of a reliable guiding signal in their discriminators. To address this problem, we propose a generative adversarial imitation learning framework for text generation that uses large pre-trained language models to provide more reliable reward guidance. As previous text GANs suffer from high variance of gradients, we apply contrastive discriminator, and proximal policy optimization (PPO) to stabilize and improve text generation performance. For evaluation, we conduct experiments on a diverse set of unconditional and conditional text generation tasks. Experimental results show that TextGAIL achieves better performance in terms of both quality and diversity than the MLE baseline. We also validate our intuition that TextGAIL's discriminator demonstrates the capability of providing reasonable rewards with an additional task.

Introduction

Automatic text generation has been used in tremendous applications such as machine translation, question answering, and dialog system. The most widely used approach for neural text generation is to maximize the probability of the target text sequence (Bengio, Ducharme, and Vincent 2000), which is also referred to as maximum likelihood estimation (MLE). However, MLE suffers from the exposure bias problem which is due to the discrepancy between training and inference. During training, the model is trained on the ground truth, but during inference, the model needs to autoregressively predict the next word conditioned on its own previously generated words. This discrepancy hurts generalization of unseen data and leads to lower quality of generated text (Welleck et al. 2019; Wiseman and Rush 2016). Therefore, solving the exposure bias problem becomes an promising approach to improve text generation quality.

Generative Adversarial Networks (GAN) are one of the directions to solve the exposure bias problem. The main idea is to alternately train between a discriminator to distinguish real samples from generated samples and the generator to improve its generated samples against the discriminator. Along this direction, there have been many studies. Nevertheless, there are increasing criticisms (Caccia et al. 2020; Tevet et al. 2019; Semeniuta, Severyn, and Gelly 2018) of text GANs showing that GAN generated text is substantially worse than the text generated by MLE. Especially, Caccia et al. (2020) find that MLE has a better quality-diversity trade-off when using the temperature sweep method for evaluation. More recently, large generative pre-trained language models have greatly improved the quality of MLE generations (Radford and Sutskever 2018; Radford et al. 2019), which further increases the gap of performance between MLE and text GANs.

In this work, we investigate whether large pre-trained language models can improve GANs in text generation. We propose TextGAIL, a generative adversarial training framework that leverages guidance from the large-scale pre-trained language models RoBERTa (Liu et al. 2019) and GPT-2 (Radford et al. 2019). We find that it does not work by simply combining the previous adversarial approaches with large pre-trained language models due to the high variance in gradients and the architecture limitations.

To reduce variance and improve performance, we apply generative imitation learning (GAIL) (Ho and Ermon 2016) and proximal policy optimization (PPO) (Schulman et al. 2017) for the optimization. We also introduce contrastive discriminator to better serve the conditional generation tasks.

For a fair comparison, we adopt temperature sweep approach (Caccia et al. 2020) to evaluate the quality-diversity trade-off. Previous text GANs often only perform experiment on unconditional generation tasks: COCO and EMNLP2017 News. We extend the experiments to conditional generation tasks, as more practical applications. Specifically, we experiment our model on CommonGEN and ROCStories.

We make several contributions: (1) We propose a generative adversarial imitation learning framework TextGAIL, which leverages large pre-trained language models. (2) We conduct extensive evaluations to show TextGAIL achieves better quality and diversity compared to an MLE fine-tuned baseline. (3) We show that large pre-trained language models can help the discriminator to provide useful rewards during the adversarial training process.

Related Work

Exposure bias is often considered to attribute to low-quality text generations by having generic and repetitive sentences (Welleck et al. 2019; Holtzman et al. 2019). Even after the emergence of large-scale pretrained language model GPT-2, this problem is still prevalent (Welleck et al. 2019). Many works tried to use Generative Adversarial Networks (GAN) (Goodfellow et al. 2014) to eliminate the exposure bias problem caused by MLE (Ranzato et al. 2016; Welleck et al. 2019).

SeqGAN (Yu et al. 2017) is the very first paper that adopts the adversarial training idea in text generation. As text sequence is discrete, SeqGAN applies REINFORCE (Williams 1992), which is a policy gradient algorithm, to train the generator with a reward defined by the discriminator's prediction on the generated sample. However, it suffers from a high variance of gradients. There are many other text GANs (Ke et al. 2019; Che et al. 2017; Guo et al. 2018; Lin et al. 2017; Nie, Narodytska, and Patel 2019; Zhou et al. 2020). However, as Caccia et al. (2020) has shown, many of them are worse than their MLE counterparts when evaluated in the quality-diversity trade-off setting. This is because many text GANs assume MLE-based models keep the softmax temperature to be 1.0 when sampling, but MLE-based models actually perform better with a lower temperature. Consequently, after using temperature sweep, MLE-based method has a better quality-diversity trade-off curve than many text GANs.

In another line of works, large scale pre-training (Radford et al. 2019) has shown significant improvement in text generation. Large pre-trained models such as GPT-2 can be fine-tuned with MLE on a specific task to achieve much better performance than the models without pre-training. Some papers even claim human-level text generation quality (Adiwardana et al. 2020). It is interesting to explore whether text GANs can be combined with large pre-trained language models to improve performance further. In this work, we propose an new imitation learning framework to combine pre-training models with GANs.

TextGAIL

In this section, we first give an overview of the generative adversarial imitation learning framework. Then we explain the discriminator and the generator in details. In the end, we summarize the entire training process. We show the overall architecture in Figure 1.

Generative Adversarial Imitation Learning

We extend the generative adversarial imitation learning (GAIL) (Ho and Ermon 2016) to text generation. The framework consists of a generator G_θ and a discriminator D_ϕ, which are parameterized with θ and ϕ, respectively. The goal of the generator is to output sequences similar to human written sequences. Meanwhile, the discriminator needs to distinguish the real sequences from the generated sequences, and provide a single sparse reward for each generated sequence.

Here, we replace the state s in GAIL with the text generation prompt x, and the corresponding action a with the target sequence y. Note that in the unconditional generation setting, x can be the start token. y can either be given from ground truth in the dataset as real data or sampled from the generator G_θ as fake data. GAIL finds a saddle point where together the generator and discriminator satisfy the following objective function:

$$\min_{G_\theta} \max_{D_\phi} \ \mathbb{E}_{p_{\text{real}}}[D_\phi(x, y)] \ + \mathbb{E}_{G_\theta}[1 - D_\phi(x, G_\theta(x))] \quad (1)$$

However, in text generation, the action space, which is the vocabulary size, is often vary large. The original GAIL has difficulty to remain stable with such a large action space (Paine et al. 2019). We introduce an imitation replay method inspired by the recent imitation learning algorithms (Paine et al. 2019; Reddy, Dragan, and Levine 2020) to stabilize the training.

We fill the experience replay buffer with a ratio λ of ground truth sequences when training the generator. Those ground truth sequences are treated the same as the generated sequences in the replay buffer. We set the reward (without normalization) to be a constant for the ground truth sequences. This approach is theoretically similar to mixing supervised MLE loss during the training, but in practice, it is much more efficient and easier to implement.

Contrastive Discriminator

The discriminator aims to distinguish between the real and generated samples. Standard discriminator utilizes logistic loss (sigmoid), but this loss saturates quickly after the model learns the difference between the real and the generated samples. We modify the discriminator to be a contrastive discriminator, which estimates the relative realness between generated sequences and real sequences. In other words, we let the discriminator estimate how much a real sequence is more realistic than a generated sequence. This can especially help conditional generation tasks. Here, we perform the prediction by utilizing softmax cross-entropy instead of logistic loss.

Instead of $D_\phi(x, y)$, the discriminator now takes a real sequence and its paired generated sequence as inputs, denoted as $D_\phi(\langle x, y_{\text{r}}\rangle, \langle x, y_{\text{g}}\rangle)$. The discriminator outputs a score to represent how good is the generated sequence y_{g} compared with the real sequence. Then we optimize it with the following objective function.

$$h_{\text{r}} = \text{Discriminator}(\langle x, y_{\text{r}}\rangle) \quad (2)$$

$$h_{\text{g}} = \text{Discriminator}(\langle x, y_{\text{g}}\rangle) \quad (3)$$

$$p_r, p_g = \text{softmax}(W_t[h_{\text{r}}; h_{\text{g}}]) \quad (4)$$

where W_t is the trainable weight to project the output embedding to a scalar logit. y_{r} is the real sequence, and y_{g} is the generated sequence. We can optimize the discriminator with cross-entropy loss to maximize the probability p_r for the real sequence. The probability prediction p_g for the generated sequence will be used as the reward signal to train the generator.

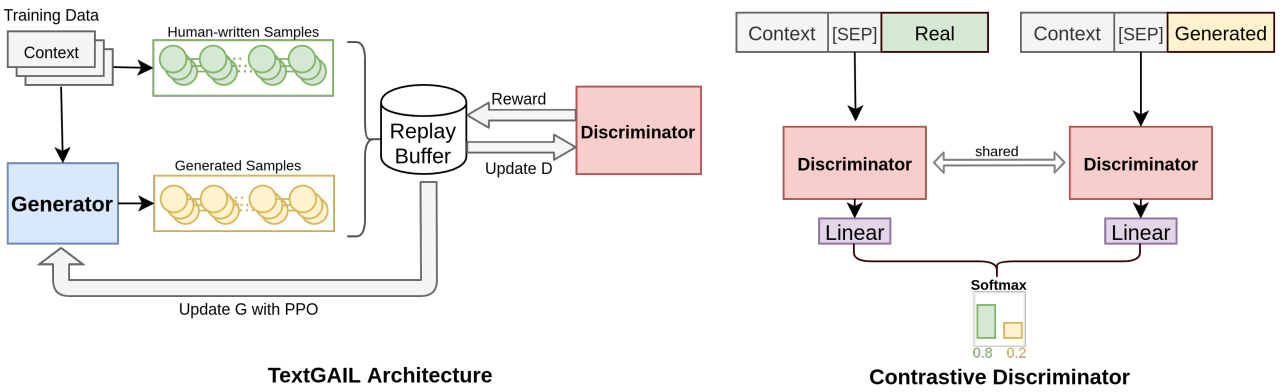


Figure 1: Left: Overall architecture of TextGAIL. Right: The contrastive discriminator..

Proximally Optimized Generator

For the generator, we begin by defining the probability of a text sequence as the joint probability of all the tokens:

$$G_\theta(y_{1:T}|x) = \prod_{t=0}^{T} G_\theta(y_t|y_{<t}, x) \tag{5}$$

where $y_{1:T}$ is a text sequence. T is the sequence length, and y_t is the word at the time step t. We sample from this distribution to acquire the generated sequences. Then we maximize the expected reward with policy gradient:

$$\mathbb{E}_{y\sim G_\theta}[\nabla_\theta \log G_\theta(x)\hat{R}_y] \tag{6}$$

where $\hat{R}_y$ is the advantage term that controls the update (which is the normalized reward here). Directly optimizing this objective suffers from high variance of gradients, because the D_ϕ is not stationary during adversarial training.

As a solution to reduce high variance, the original GAIL employs trust region policy optimization (TRPO) (Schulman et al. 2015), as it is crucial to ensure that $G_{\theta_{i+1}}$ does not move too far away from G_{θ_i}. However, TRPO needs to compute natural gradient which is computationally expensive. We replace it with a more recent and stable method, proximal policy optimization (PPO) (Schulman et al. 2017). Compared to TRPO, PPO is easier to implement and generalize. PPO has better sample complexity in practice as well.

PPO applies importance sampling by the likelihood ratio between the current and old policy for $y \sim G_{\theta_{\text{old}}}(\cdot|x)$:

$$r(\theta) = \frac{G_\theta(y_{1:T} \mid x)}{G_{\theta_{\text{old}}}(y_{1:T} \mid x)} \tag{7}$$

Then it maximizes the expected reward by optimizing the following surrogate:

$$L_{\text{G}}(\theta) = -\min \begin{cases} r(\theta)\,\hat{R}_y \\ \text{clip}\left(r(\theta), 1-\epsilon, 1+\epsilon\right)\hat{R}_y \end{cases} \tag{8}$$

This surrogate serves the same purpose as TRPO to have a trust region constraint on the gradient update. It will prevent the generator from moving too far away from the pre-trained language model.

TextGAIL Training Process

We warm-up the generator by training with a part of the training set using MLE as its loss function. We alternately train the discriminator and the generator to optimize Equation 1. A replay buffer stores temporarily generated outputs and human-written sequences. Next, we normalize the rewards in the buffer with running statistics to reduce variance. We update the generator G_θ with PPO using the replay buffer. In the meantime, we update the discriminator D_ϕ with the real and generated pairs. We repeat until the training stops. The summary of the algorithm is illustrated below.

Algorithm 1 TextGAIL

Initialize: Collect human-written sequences
Warm-up the generator G_θ
Replay Buffer B
for i = $1, 2, 3, \ldots$ **do**
  Sample p proportion of human-written sequences y
  Sample $1-p$ proportion of generator outputs $y \sim G(\cdot|x)$
  Put all sampled (x, y) pairs into B
  Collect rewards using discriminator D_ϕ for all $(x, y) \in B$
  Normalize all the rewards to get $\hat{R}$
  Replace rewards for human-written sequences with a constant
  Update the discriminator ϕ with Eq. 4
  Update the generator θ using the PPO with Eq. 8
  Clear Buffer B
end for

Experimental Settings

We will describe implementation details and automatic evaluation metrics in this section.

Implementation Details

TextGAIL takes advantage of large pre-trained language models. In particular, the generator uses the GPT-2 base (117M parameters) model, while the discriminator uses the RoBERTa-base (125M parameters) model. The human demonstrations

mix ratio p is set to 0.3 at the start of the training and linearly decay afterward. The constant reward for human demonstrations is set to 2.0. When generating outputs, we apply the recent nucleus sampling method (Holtzman et al. 2019) for decoding to avoid low probability words being sampled. We stop the training when the perplexity stops decreasing for both MLE and TextGAIL. The details of hyper-parameters are in the Appendix.

Baselines

Since previous text GANs are mainly for unconditional tasks, we only show their performance on unconditional tasks. For conditional generation tasks, we compare TextGAIL with GPT-2 fine-tuned on training dataset with a MLE loss. For a fair comparison between MLE models and TextGAIL models, we stop the training when TextGAIL reaches the perplexity of MLE baselines.

Evaluation Metrics

We measure model's quality and diversity from a range of temperatures between 0.1 to 1.0. This temperature sweep method ensures fair comparisons as described by Caccia et al. (2020)

For the quality metric, we use the n-gram matching metric BLEU. When using BLEU for unconditional generation tasks, the entire training corpus is used as references for BLEU (Yu et al. 2017). Since BLEU has its limitations, we further conduct human evaluations to measure models' generation quality. We also compare perplexity under different temperatures for more comprehensive comparisons.

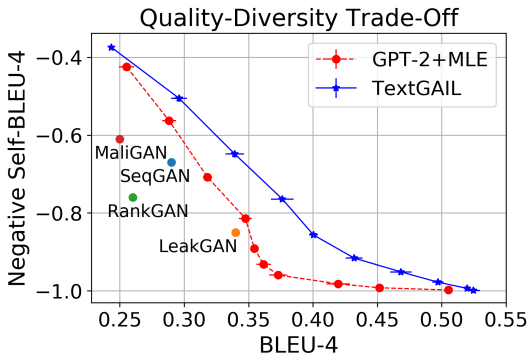
For the diversity metric, we use Self-BLEU (Yu et al. 2017) on unconditional generation tasks, and Distinct-n (Li et al. 2016) on conditional generation tasks. Self-BLEU evaluates how one generated sentence resembles the rest in a set of generated samples. Distinct-n is the number of distinct n-grams divided by the total number of n-grams in the test set. When decoding with beam search, we use Seq-Rep-n (Welleck et al. 2019) to measure sequence-level repetition inside a sentence. Seq-Rep-n is the portion of duplicate n-grams in a sequence.

Results and Analysis

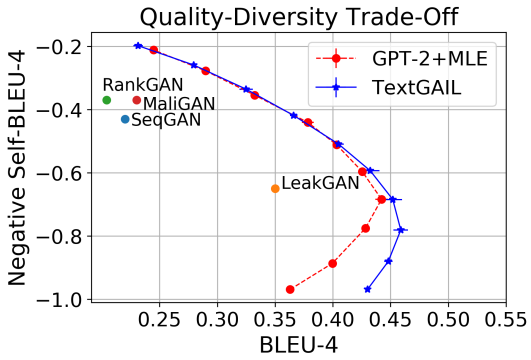
Automatic Evaluation Results

We first test if TextGAIL performs better than a model that fine-tunes GPT-2 on the training dataset with an MLE loss. As suggested by Caccia et al. (2020), temperature sweep can reflect the quality-diversity trade-off, which enables fair comparisons between two models. We sweep the softmax temperature from 0.1 to 1.0 to observe how the models behave accordingly. For unconditional generation tasks, we report BLEU vs. Self-BLEU as the quality-diversity metric (we use negative Self-BLEU for better visualization). For conditional generation tasks, we report BLEU vs. Distinct as the quality-diversity metric. The blue lines are TextGAIL, and the red lines are GPT-2 fine-tuned with MLE.

For the unconditional tasks, the results are shown in Figure 2. TextGAIL and GPT-2+MLE is much better than the previous text GANs. For COCO Captions, we can observe



(a) COCO Captions



(b) EMNLP2017 News

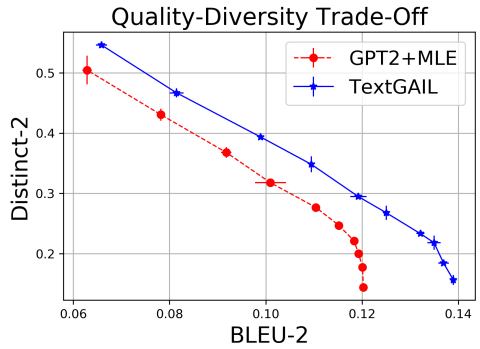
Figure 2: Quality-diversity trade-off on unconditional tasks. The curve closer to the top right corner has better performance. Error bars are from three random seed runs.

that TextGAIL achieves great improvement over MLE. However, for EMNLP2017 News, the difference is not significant. We suspect that the reason is because of the text length in the dataset. EMNLP2017 News has much longer text length than COCO Captions. This may lead to worse reward signals from the discriminator when training the generator.

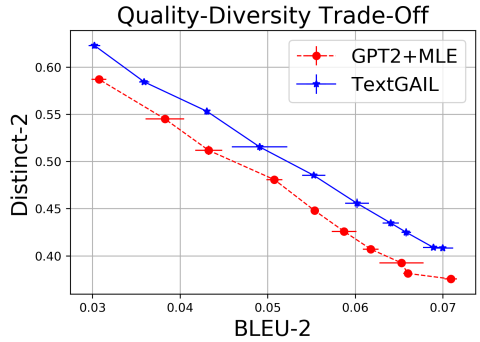
In Figure 3, we show the results of conditional tasks. Since previous text GANs are not designed for conditional generation tasks, we are unable to report them here. From the figure, we can observe more consistent improvement of TextGAIL compared to the improvement in unconditional tasks. While in unconditional generation tasks, without the context given, it is even hard for humans to classify if the text is real or generated.

One advantage of testing with conditional tasks is that we can apply deterministic decoding methods such as beam search rather than stochastic decoding used in unconditional generation tasks. This can provide more reliable interpretation of our model.

We perform beam search with a beam size of four on the two conditional generation tasks. We run the experiments with three different random seeds. The results on beam search generations are shown in Table 1. We observe that TextGAIL performs better than MLE-based method in both quality and diversity metrics. When we examine the generated text, we



(a) CommonGEN



(b) ROCStories

Figure 3: Quality-diversity trade-off on conditional tasks. The curve closer to the top right corner has better performance. Note that there are no results from the most previous GANs, as they are not designed for conditional tasks. Error bars are from three random seed runs.

find that MLE produces many repetitions, which is possibly caused by text degeneration with exposure bias (Welleck et al. 2019). As TextGAIL mitigates exposure bias, both the quality metric BLEU-2 and the diversity metric Distinct-2 improves over the baseline MLE method.

We suspect the main contribution of improvement is due to the discriminator being more effective in providing useful reward signals in conditional generation tasks. As in CommenGEN and ROCStories, the contrastive discriminator can better classify the realness of the generations conditioned to the context by comparing the real and generated sentences. In Section 6, we will conduct additional experiments to interpret what the discriminator has learned.

Human Evaluation Results

Automatic evaluation metrics have their limitations in measuring overall performance. Therefore, we also conduct human evaluations to measure the quality of TextGAIL compared to GPT-2 fine-tuned with MLE. The MLE model with a low temperature generates large amount of repetitions. We observe the model has less repetition and better quality with nucleus sampling with hyper-parameters top-p 0.9 and temperature 0.8. So we use this setting for human evaluation. For each task, we randomly select 100 samples in test sets, and for

	CommonGEN	
	MLE	TextGAIL
BLEU-2 ↑	15.74 ± 0.04	**16.21** ± 0.17
Distinct-2 ↑	8.66 ± 0.05	**10.00** ± 0.08
Seq-Rep-2 ↓	65.85± 2.30	**60.78** ± 2.41
	ROCStories	
	MLE	TextGAIL
BLEU-2 ↑	7.44 ± 0.13	**7.77** ± 0.27
Distinct-2 ↑	41.05 ± 2.39	**46.83** ± 3.83
Seq-Rep-2 ↓	76.92 ± 2.18	**74.71** ± 2.07

Table 1: Beam search results on the conditional tasks. Since beam search results are deterministic, it is not applicable to the unconditional tasks.

	TextGAIL vs MLE		
	Win	Lose	Tie
COCO	31.2%	27.6%	41.2%
EMNLP2017	44.2 %	44.2%	11.6%
CommonGEN	59.6%*	34.6%*	5.8%
ROCStories	60.5%*	23.4%*	16.1%

Table 2: Human evaluation results. *denotes statistical significance (binomial test, $p < 0.05$)

each sample, we ask five Amazon Mechanical Turk workers to select which model's result is better to reduce the variance. In total, we have 500 data points for each task. For conditional generation tasks, the context is shown to all workers. The workers are instructed to select the model with better logic and commonsense. The evaluators can select "Cannot determine" when the two models are similar in quality.

The human evaluation results are shown in Table 2. There is no statistical difference between TextGAIL and MLE on unconditional generation tasks in this pairwise comparison evaluation. One possible reason is that when comparing two completely different sentences in this unconditional generation setting, it is difficult for human evaluators to make consistent decisions. In contrast, TextGAIL significantly outperforms MLE in human evaluation on two conditional generation tasks. Since these tasks expect models to produce similar content with respect to the ground truth. It is easier for human to select the output with better quality. The results are in agreement with automatic evaluation results. In the Case Study, we will further analyze the examples .

Case Study

We further analyze the generated outputs from the two conditional generation tasks. We show some examples in Table 3 and Table 4.

CommonGEN In the CommonGEN examples, the good target sentences should use the given three words as much as possible. We can observe that MLE's outputs do not seem to

CommonGEN	Example 1	Example 2
Context:	field look stand	ocean surf surfer
Ground Truth:	I stood and looked across the field, peacefully.	A surfer surfing in the ocean.
MLE:	(1) looks at the ground at the end of the driveway (2) looks at the wall and the wall stands (3) i stand on a bench in front of the field with a smile on my lips	(1) surfer and surfers walk the beach at the coast (2) surfer in the surf on the coast (3) surfer in the ocean.
TextGAIL:	(1) field looks like a soccer field with a few soccer players standing (2) a man stands in the middle of the field looking at the scoreboard (3) a small group of people stand in the field looking at a city	(1) the surfers wave their surfboards on the beach (2) Two surfers are surfing in the ocean and one is looking to the horizon. (3) a surf diver watches as a group of dolphins swim in the ocean

Table 3: Examples of MLE and TextGAIL on CommonGEN. TextGAIL follow the instruction by using the three given three words in the generations. The results are longer and more diverse than MLE.

ROCStories	Example 1	Example 2
Context:	I wanted to buy a video game console. I asked my parents, and they came up with an idea. They said if I did my chores, I would be given money to save. I did my chores without being asked every week for a whole summer.	Ben went to the DMV to get his License. The instructor gave Ben a passing grade at the end. Excited, Ben calls up his father to tell him the good news. Ben father never picked up, he died in a car accident that day.
Ground Truth:	My parents gave me enough money to buy the console.	Ben was devastated.
MLE:	(1) Now I have the video game console I asked for. (2) It was an awesome idea. (3) The next week, I had to buy a new gaming console!	(1) Ben was happy to learn his lessons about being smart. (2) Ben's father is now very sad, and he has a job to do. (3) Ben was happy that his dad was alive.
TextGAIL:	(1) I bought a PlayStation 4 to play with my parents. (2) I was so happy when my parents gave me a Wii U. (3) When I got my console, I played my favorite video games.	(1) Ben regrets going to the DMV. (2) Ben mourns the loss of his father but also the passing of a great man. (3) It seems like too much to bear.

Table 4: Examples of MLE and TextGAIL on ROCStories. TextGAIL generates better and more reasonable story endings. Also, note that named entities such as "PlayStation 4" and "Wii U" have never appeared in the training set.

follow that instruction, while TextGAIL is behaving better. This difference suggests that TextGAIL's discriminator might have learned to guide the generator to follow the implied instruction. Also, we can observe that the MLE's outputs have more repetitions and are less diverse than MLE's outputs. This is also partially reflected in the automatic evaluation metrics. These examples also correlates with our intuition that eliminating exposure bias alleviates dull and repetitive outputs (Welleck et al. 2019).

ROCStories For ROCStories, the good story endings should be as reasonable and interesting as possible. We can observe similar patterns in this task. MLE's generations lack of details and are universal, as the Example 1 MLE (2) appears more than once in other story contexts. We further find that TextGAIL can generate new named entities such as "PlayStation 4" and "Wii U", which never appeared in the training set. We speculate it might have appeared in GPT-2's pre-trained corpus. This task also provides the evidence that eliminating exposure bias improves generalization of unseen data. Moreover, from Example 2, we observe that TextGAIL seems to generate more reasonable and logical endings than MLE. We suspect that TextGAIL's discriminator has used some latent information to distinguish between the real and generated samples. In Section 6, we specifically analyze what the discriminator has learned to provide useful rewards to the generator.

Ablation Studies

Each component's contribution in TextGAIL's performance is shown on the CommonGEN task with an ablation study in Table 5. We use beam search with beam size four as the inference method.

Some previous text GANs involve architecture changes such as LeakGAN (Guo et al. 2018) and RelGAN (Nie, Nar-

	PPL ↓	BLEU-2 ↑	Distinct-2 ↑
SeqGAN+Pre-train	140.42	n/a	n/a
TextGAIL	14.85	16.23	9.50
w/o PPO*	111.08	n/a	n/a
w/o human demo	132.63	n/a	n/a
w/o Contrastive	17.26	13.72	8.40
w/o D pre-train	16.94	15.14	9.14

Table 5: Ablation studies results on CommenGEN. "w/o D pre-train" means randomly initialized discriminator. "n/a" indicates that the model diverges during the training. "PPL" stands for perplexity.

odytska, and Patel 2019), it is hard to directly apply them on the Transformer-based (Vaswani et al. 2017) models. Also, most of them are not designed for conditional generation tasks. Therefore, we only test incorporating pre-trained language models for SeqGAN. The model fails to converge. It is probably due to the large number of parameters that needs update and the high variance in gradients. This suggests that the optimization techniques of PPO and mixed human demonstrations are crucial for stably training the text GANs.

We test replacing the contrastive discriminator with a normal discriminator that classifies a singe input with sigmoid. The BLEU-2 and Distinct-2 scores decreases significantly. We suspect that when the discriminator can only see one single input without comparing against the real example, it would be harmful for conditional generation tasks, as even if the generator outputs a sentence better than the ground truth, the generator cannot receive the accurate reward signal.

Moreover, we experiment the discriminator without any pre-training. The performance drops as expected. This suggests the importance of pre-training for TextGAIL to improve over the MLE method. In Section 6, we further explore what the discriminator has learned during adversarial learning.

What Has the Discriminator Learned

Models	Supervised?	Accuracy(%)
RoBERTa w/ extra data	✓	92.8 ± 0.28
GPT-2 + MLE	×	69.6 ± 0.35
TextGAIL D	×	79.1 ± 0.76
TextGAIL D w/o pre-train	×	51.2 ± 0.85

Table 6: Story Cloze Test results. "D" means the discriminator. TextGAIL's learned discriminator can classify the story ending with the correct commonsense.

We analyze the reward signal of the learned discriminator in TextGAIL, which is supposed to distinguish the real samples from the generated samples. We apply the learned discriminator in TextGAIL on a story ending classification task, Story Cloze Test, to identify story endings with the correct commonsense given the story prompt (Mostafazadeh et al. 2016). This task uses a different dataset from ROCStories but is in the similar domain. We report the Story Cloze Test results in Table 6.

TextGAIL's discriminator achieves 79.1% accuracy. This suggests the learned discriminator provides meaningful rewards to the generator in the training process. We compare our learned discriminator against a RoBERTa classifier fine-tuned on the Story Cloze Test's training data to explore if adding more supervision affects the performance. We find that the fine-tuned RoBERTa classifier achieves the best accuracy (92.8%). Clearly, direct supervision improves the performance, but our zero-shot TextGAIL discriminator is not too far from the supervised model's performance.

Inspired by Trinh and Le (2018), we also construct another baseline, the GPT-2 fine-tuned on ROCStories data with MLE (not Story Cloze Test), which also does not require extra supervision. The model selects the ending with a higher joint language model probability. It reaches 69.6% accuracy, which is significantly worse than TextGAIL discriminator (79.1%). This result, in another way, suggests TextGAIL has better reward guidance than the MLE language model.

We also compare TextGAIL's Discriminator against the model without pre-training (from ablation study) to see how much pre-training contributes to the performance. The accuracy drops from 79.1% to 51.2 % . This finding suggests that TextGAIL's discriminator is relying on the information obtained from pre-training to select the correct story ending.

Conclusion

We propose a generative adversarial imitation learning framework for text generation - TextGAIL, which leverages the large pre-trained language models. We extend the exploration of adversarial training on text generation by incorporating large scale pre-trained models. We use a contrastive discriminator and proximal policy optimization to improve the stability of the generator's training. We incorporate a large-scale pre-trained language model, GPT2 in our framework as the generator. We also use a pre-trained RoBERTa model to initialize the discriminator to provide reliable rewards to the imitation learning framework. Experiment shows that TextGAIL can generate not only more diverse but more accurate and reasonable outputs, and the discriminator can provide meaningful reward signals in various unconditional and conditional text generation tasks.

References

Adiwardana, D.; Luong, M.; So, D. R.; Hall, J.; Fiedel, N.; Thoppilan, R.; Yang, Z.; Kulshreshtha, A.; Nemade, G.; Lu, Y.; and Le, Q. V. 2020. Towards a Human-like Open-Domain Chatbot. *CoRR* abs/2001.09977. URL https://arxiv.org/abs/2001.09977.

Bengio, Y.; Ducharme, R.; and Vincent, P. 2000. A Neural Probabilistic Language Model. In *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, 932–938. URL http://papers.nips.cc/paper/1839-a-neural-probabilistic-language-model.

Caccia, M.; Caccia, L.; Fedus, W.; Larochelle, H.; Pineau, J.; and Charlin, L. 2020. Language GANs Falling Short. In *International Conference on Learning Representations*. URL https://openreview.net/forum?id=BJgza6VtPB.

Che, T.; Li, Y.; Zhang, R.; Hjelm, R. D.; Li, W.; Song, Y.; and Bengio, Y. 2017. Maximum-Likelihood Augmented Discrete Generative Adversarial Networks. *CoRR* abs/1702.07983. URL http://arxiv.org/abs/1702.07983.

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A. C.; and Bengio, Y. 2014. Generative Adversarial Networks. *CoRR* abs/1406.2661. URL http://arxiv.org/abs/1406.2661.

Guo, J.; Lu, S.; Cai, H.; Zhang, W.; Yu, Y.; and Wang, J. 2018. Long Text Generation via Adversarial Training with Leaked Information. In McIlraith, S. A.; and Weinberger, K. Q., eds., *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 5141–5148. AAAI Press. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16360.

Ho, J.; and Ermon, S. 2016. Generative Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, 4565–4573. URL http://papers.nips.cc/paper/6391-generative-adversarial-imitation-learning.

Holtzman, A.; Buys, J.; Forbes, M.; and Choi, Y. 2019. The Curious Case of Neural Text Degeneration. *CoRR* abs/1904.09751. URL http://arxiv.org/abs/1904.09751.

Ke, P.; Huang, F.; Huang, M.; and Zhu, X. 2019. ARAML: A Stable Adversarial Training Framework for Text Generation. *CoRR* abs/1908.07195. URL http://arxiv.org/abs/1908.07195.

Li, J.; Galley, M.; Brockett, C.; Gao, J.; and Dolan, B. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. In Knight, K.; Nenkova, A.; and Rambow, O., eds., *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, 110–119. The Association for Computational Linguistics. ISBN 978-1-941643-91-4. URL https://www.aclweb.org/anthology/N16-1014/.

Lin, K.; Li, D.; He, X.; Sun, M.; and Zhang, Z. 2017. Adversarial Ranking for Language Generation. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 3155–3165. URL http://papers.nips.cc/paper/6908-adversarial-ranking-for-language-generation.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692. URL http://arxiv.org/abs/1907.11692.

Mostafazadeh, N.; Chambers, N.; He, X.; Parikh, D.; Batra, D.; Vanderwende, L.; Kohli, P.; and Allen, J. F. 2016. A Corpus and Evaluation Framework for Deeper Understanding of Commonsense Stories. *CoRR* abs/1604.01696. URL http://arxiv.org/abs/1604.01696.

Nie, W.; Narodytska, N.; and Patel, A. 2019. RelGAN: Relational Generative Adversarial Networks for Text Generation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. URL https://openreview.net/forum?id=rJedV3R5tm.

Paine, T. L.; Gülçehre, Ç.; Shahriari, B.; Denil, M.; Hoffman, M. D.; Soyer, H.; Tanburn, R.; Kapturowski, S.; Rabinowitz, N. C.; Williams, D.; Barth-Maron, G.; Wang, Z.; de Freitas, N.; and Team, W. 2019. Making Efficient Use of Demonstrations to Solve Hard Exploration Problems. *CoRR* abs/1909.01387. URL http://arxiv.org/abs/1909.01387.

Radford, A.; and Sutskever, I. 2018. Improving Language Understanding by Generative Pre-Training. *OpenAI Blog* URL https://openai.com/blog/language-unsupervised.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language Models are Unsupervised Multitask Learners. *OpenAI Blog* URL https://openai.com/blog/better-language-models/.

Ranzato, M.; Chopra, S.; Auli, M.; and Zaremba, W. 2016. Sequence Level Training with Recurrent Neural Networks. In Bengio, Y.; and LeCun, Y., eds., *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. URL http://arxiv.org/abs/1511.06732.

Reddy, S.; Dragan, A. D.; and Levine, S. 2020. SQIL: Imitation Learning via Reinforcement Learning with Sparse Rewards. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. URL https://openreview.net/forum?id=S1xKd24twB.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M. I.; and Moritz, P. 2015. Trust Region Policy Optimization. In Bach, F. R.; and Blei, D. M., eds., *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, 1889–1897. JMLR.org. URL http://proceedings.mlr.press/v37/schulman15.html.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347. URL http://arxiv.org/abs/1707.06347.

Semeniuta, S.; Severyn, A.; and Gelly, S. 2018. On Accurate Evaluation of GANs for Language Generation. *CoRR* abs/1806.04936. URL http://arxiv.org/abs/1806.04936.

Tevet, G.; Habib, G.; Shwartz, V.; and Berant, J. 2019. Evaluating Text GANs as Language Models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language*

Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), 2241–2247. URL https://www.aclweb.org/anthology/N19-1233/.

Trinh, T. H.; and Le, Q. V. 2018. A Simple Method for Commonsense Reasoning. *CoRR* abs/1806.02847. URL http://arxiv.org/abs/1806.02847.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 5998–6008. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.

Welleck, S.; Kulikov, I.; Roller, S.; Dinan, E.; Cho, K.; and Weston, J. 2019. Neural Text Generation with Unlikelihood Training. *CoRR* abs/1908.04319. URL http://arxiv.org/abs/1908.04319.

Williams, R. J. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning* 8: 229–256. doi:10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Wiseman, S.; and Rush, A. M. 2016. Sequence-to-Sequence Learning as Beam-Search Optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, 1296–1306. URL https://www.aclweb.org/anthology/D16-1137/.

Yu, L.; Zhang, W.; Wang, J.; and Yu, Y. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, 2852–2858. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14344.

Zhou, W.; Ge, T.; Xu, K.; Wei, F.; and Zhou, M. 2020. Self-Adversarial Learning with Comparative Discrimination for Text Generation. In *International Conference on Learning Representations*. URL https://openreview.net/forum?id=B1l8L6EtDS.