MASKGAN: TEXTUAL GENERATIVE ADVERSARIAL NETWORKS FROM FILLING-IN-THE-BLANK

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural text generation models are often autoregressive language models or seq2seq models. Neural autoregressive and seq2seq models that generate text by sampling words sequentially, with each word conditioned on the previous model, are state-of-the-art for several machine translation and summarization benchmarks. These benchmarks are often defined by validation perplexity even though this is not a direct measure of sample quality. Language models are typically trained via maximum likelihood and most often with teacher forcing. Teacher forcing is well-suited to optimizing perplexity but can result in poor sample quality because generating text requires conditioning on sequences of words that were never observed at training time. We propose to improve sample quality using Generative Adversarial Networks (GANs), which explicitly train the generator to produce high quality samples and have shown a lot of success in image generation. GANs were originally designed to output differentiable values, so discrete language generation is challenging for them. We introduce an actor-critic conditional GAN that fills in missing text conditioned on the surrounding context. We show qualitatively and quantitatively, evidence that this produces more realistic text samples compared to a maximum likelihood trained model.

1 INTRODUCTION

Recurrent Neural Networks (RNNs) (Graves et al., 2012) are the most common generative model for sequences as well as for sequence labeling tasks. They have shown impressive results in language modeling (Mikolov et al., 2010), speech recognition (Graves et al., 2013), machine translation (Wu et al., 2016) and text classification (Miyato et al., 2016). Text is typically generated from these models by sampling from a distribution that is conditioned on the previous word and a hidden state that consists of a representation of the words generated so far. These are typically trained with maximum likelihood in an approach known as *teacher forcing*, where ground-truth words are fed back into the model to be conditioned on for generating the following parts of the sentence. This causes problems when, during sample generation, the model is often forced to condition on sequences that were never conditioned on at training time. This leads to unpredictable dynamics in the hidden state of the RNN. Methods such as Professor Forcing (Lamb et al., 2016) and Scheduled Sampling (Bengio et al., 2015) have been proposed to solve this issue. These approaches work indirectly by either causing the hidden state dynamics to become predictable (Professor Forcing) or by randomly conditioning on sampled words at training time, however, they do not directly specify a cost function on the output of the RNN encouraging high sample quality. Our proposed method does so.

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) are a framework for training generative models in an adversarial setup, with a generator generating images that is trying to fool a discriminator that is trained to discriminate between real and synthetic images. GANs have had a lot of success in producing more realistic images than other approaches but they have only been seen limited use for text sequences. This is due to the discrete nature of text making it infeasible to propagate the gradient from the discriminator back to the generator as in standard GAN training. We overcome this by using Reinforcement Learning (RL) to train the generator while the discriminator is still trained via maximum likelihood and stochastic gradient descent. GANs also commonly suffer from issues such as training instability and mode dropping, both of which are exacerbated in a textual setting. Mode dropping occurs when certain modalities in the training set are rarely generated by the

generator, for example, leading all generated images of a volcano to be the multiple variants of the same volcano. This becomes a significant problem in text generation since there are many complex modes in the data, ranging from bigrams to short phrases to longer idioms. Training stability is also an issue since unlike image generation, text is typically generated autoregressively and thus the loss from the discriminator is only observed after a complete sentence has been generated. This problem compounds when generating longer and longer sentences.

We reduce the impact of these problems by training our model on a text fill-in-the-blank or in-filling task. This is similar to the task proposed in Bowman et al. (2015) but we set up this task more robustly. In this task, portions of a body of text are deleted or redacted. The goal of the model is to then infill the missing portions of text so that it is indistinguishable from the original data. While in-filling text, the model operates autoregressively over the tokens it has thus far filled in, as in standard language modeling, while conditioning on the true known context. If the entire body of text is redacted, then this reduces to language modeling.

Designing error attribution per time step has been noted to be important in prior natural language GAN research (Yu et al., 2017; Li et al., 2017). The text infilling task naturally achieves this consideration since our discriminator will evaluate each token and thus provide a fine-grained supervision signal to the generator. Consider, for instance, if the generator produces a sequence perfectly matching the data distribution over the first $t-1$ time-steps, but then produces an outlier token y_t, $(x_{1:t-1}y_t)$. Despite the entire sequence now being clearly synthetic as a result of the errant token, a discriminative model that produces a high loss signal to the outlier token, but not to the others, will likely yield a more informative error signal to the generator.

This research also opens further inquiry of conditional GAN models in the context of natural language. There have been numerous impressive results in the continuous domain using conditional GAN approaches, beginning with the work of (Mirza & Osindero, 2014). In the work of Odena et al. (2016), the model is able to produce all 1000 classes of ImageNet Russakovsky et al. (2015) while conditioned on the class label. Another example of success of conditional GAN models was the transfer of style, object transfiguration and photo enhancement recently demonstrated by CycleGAN Zhu et al. (2017).

In the following sections,

- We introduce a text generation model trained on in-filling (MaskGAN).
- Consider the actor-critic architecture in extremely large action spaces.
- Consider new evaluation metrics and the generation of synthetic training data.

2 Related Work

Research into reliably extending GAN training to discrete spaces and discrete sequences has been a highly active area. GAN training in a continuous setting allows for fully differentiable computations, permitting gradients to be passed through the discriminator to the generator. Discrete elements break this differentiability, leading researchers to either avoid the issue and reformulate the problem, work in the continuous domain or to consider RL methods.

SeqGAN (Yu et al., 2017) trains a language model by using policy gradients to train the generator to fool a CNN-based discriminator that discriminates between real and synthetic text. Both the generator and discriminator are pretrained on real and fake data before the phase of training with policy gradients. During training they then do Monte Carlo rollouts in order to get a useful loss signal per word.

Professor Forcing (Lamb et al., 2016) is an alternative to training an RNN with teacher forcing by using a discriminator to discriminate the hidden states of a generator RNN that is conditioned on real and synthetic samples. Since the discriminator only operates on hidden states, gradients can be passed through to the generator so that the hidden state dynamics at inference time follow those at training time.

Discrete GANs have been applied for translation (Yang et al., 2017) where they produced an improvement in BLEU score for Chinese-English translation. GANs have also been applied to dialogue generation (Li et al., 2017) showing improvements in adversarial evaluation and good results with

human evaluation compared to a maximum likelihood trained baseline. Their method applies REINFORCE with Monte Carlo sampling on the generator.

Replacing the non-differentiable sampling operations with efficient gradient approximators (Jang et al., 2016; Maddison et al., 2016) led to some work applying GANs in toy tasks, (Kusner & Hernández-Lobato, 2016), however, there has not been strong evidence of the efficacy of these tools yet. Recent unbiased and low variance gradient estimate techniques such as Tucker et al. (2017) may prove more effective.

WGAN-GP (Gulrajani et al., 2017) avoids the issue of dealing with backpropagating through discrete nodes by generating text in a one-shot manner using a 1D convolutional network. Hjelm et al. (2017) proposes an algorithmic solution and uses a boundary-seeking GAN objective along with importance sampling to generate text. In Rajeswar et al. (2017), the discriminator operates directly on the continuous probabilistic output of the generator. However, to accomplish this, they recast the traditional autoregressive sampling of the text since the inputs to the RNN are predetermined. Che et al. (2017) instead optimize a lower-variance objective using the discriminator's output, rather than the standard GAN objective.

Reinforcement learning methods have been explored successfully in natural language. Using a REINFORCE and cross entropy hybrid, MIXER, Ranzato et al. (2015) directly optimized BLEU score Papineni et al. (2002) and demonstrated improvements over baselines. More recently, actor-Critic methods in natural language were explored in Bahdanau et al. (2016) where instead of having rewards supplied by a discriminator in an adversarial setting.

Conditional text generation via GAN training has been explored in Rajeswar et al. (2017); Li et al. (2017).

Our work is distinct in that we employ an actor-critic training procedure on a task designed to provide rewards at every time step Li et al. (2017). We believe the in-filling may mitigate the problem of severe mode-collapse. This task is also harder for the discriminator which reduces the risk of the generator contending with a near-perfect discriminator. The critic in our method also helps the generator converge more rapidly by reducing the high-variance of the gradient updates in an extremely high action-space environment when operating at word-level in natural language.

3 MASKGAN

3.1 NOTATION

Let (x_t, y_t) denote pairs of input and target tokens. Let <m> denote a masked token (indicates that the original token is replaced with a hidden token) and let $\hat{x}_t$ denote the filled-in token. Finally, $\tilde{x}_t$ is a filled-in token passed to the discriminator which may be either real or fake.

3.2 ARCHITECTURE

The task of imputing missing tokens requires that our MaskGAN architecture condition on information from both the past and the future. We choose to use a seq2seq Sutskever et al. (2014) architecture. Our generator consists of an encoding module and decoding module. For a discrete sequence $\boldsymbol{x} = (x_1, \cdots, x_T)$, a binary mask is generated (deterministically or stochastically) of the same length $\boldsymbol{m} = (m_1, \cdots, m_T)$ where each $m_t \in \{0, 1\}$, selects which tokens will remain. The token at time t, x_t is then replaced with a special mask token <m> if the mask is 0 and remains unchanged if the mask is 1.

The encoder reads in the masked sequence, which we denote as $\boldsymbol{m}(\boldsymbol{x})$, where the mask is applied element-wise. The encoder provides access to future context for the MaskGAN during decoding.

As in standard language-modeling, the decoder fills in the missing tokens auto-regressively, however, it is now conditioned on both the masked text $\boldsymbol{m}(\boldsymbol{x})$ as well as what it has filled-in up to that point. The generator decomposes the distribution over the sequence into an ordered conditional sequence $P\left(\hat{x}_1, \cdots, \hat{x}_T | \, \boldsymbol{m}(\boldsymbol{x})\right) = \prod_{t=1}^{T} P(\hat{x}_t | \hat{x}_1, \cdots, \hat{x}_{t-1}, \boldsymbol{m}(\boldsymbol{x}))$.

$$G(x_t) \equiv P(\hat{x}_t | \hat{x}_1, \cdots, \hat{x}_{t-1}, \boldsymbol{m}(\boldsymbol{x})) \tag{1}$$

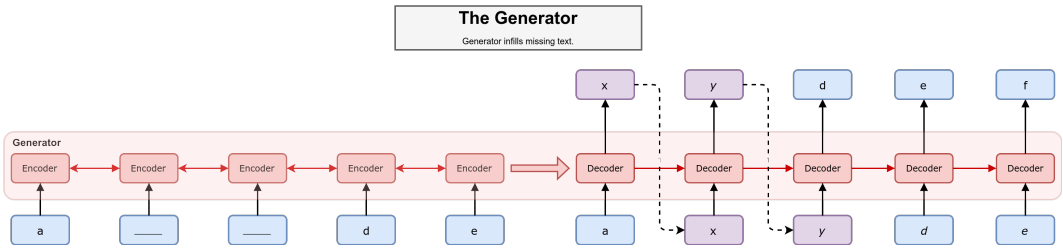


Figure 1: seq2seq generator architecture. Blue boxes represent known tokens and purple boxes are imputed tokens. We demonstrate a sampling operation via the dotted line. The encoder reads in a masked sequence, where masked tokens are denoted by an underscore, and then the decoder imputes the missing tokens by using the encoder hidden states. In this example, the generator should fill in the alphabetical ordering, (a,b,c,d,e).

The discriminator has an identical architecture to the generator except that the output is a scalar probability at each time point, rather than a distribution over the vocabulary size. The discriminator is given the filled-in sequence from the generator, but importantly, it is given the original real context $\boldsymbol{m}(\boldsymbol{x})$. We give the discriminator the true context, otherwise, this algorithm has a critical failure mode. For instance, without this context, if the discriminator is given the filled-in sequence `the director director guided the series`, it will fail to reliably identify the `director director` bigram as fake text, despite this bigram potentially never appearing in the training corpus (aside from an errant typo). The reason is that it is ambiguous which of the two occurrences of `director` is fake; `the *associate* director guided the series` or `the director *expertly* guided the series` are both potentially valid sequences. Without the context of which words are real, the discriminator was found to assign equal probability to both words. The result, of course, is an inaccurate learning signal to the generator which will not be correctly penalized for producing these bigrams. To prevent this, our discriminator D_ϕ computes the probability of each token $\tilde{x}_t$ being real given the *true context* of the masked sequence $\boldsymbol{m}(\boldsymbol{x})$.

$$D_\phi(\tilde{x}_t|\tilde{x}_{0:T}, \boldsymbol{m}(\boldsymbol{x})) = P(\tilde{x}_t = x_t^{\text{real}}|\tilde{x}_{0:T}, \boldsymbol{m}(\boldsymbol{x})) \tag{2}$$

In our formulation, the logarithm of the discriminator estimates are regarded as the reward

$$r_t \equiv \log D_\phi(\tilde{x}_t|\tilde{x}_{0:T}, \boldsymbol{m}(\boldsymbol{x})) \tag{3}$$

Our third network is the Critic network, which is implemented as an additional head off the discriminator. The critic estimates the value function, which is the discounted total return of the filled-in sequence $R_t = \sum_{s=t}^{T} \gamma^s r_s$, where γ is the discount factor.

3.3 Training

Our model is not fully-differentiable due to the sampling operations on the generator's probability distribution for the next token. Therefore, to train the generator, we estimate the gradient with respect to its parameters θ via policy gradients (Sutton et al., 2000). Reinforcement learning was first employed to GANs for language modeling in Yu et al. (2017). Analogously, here the generator seeks to maximize the cumulative total reward $R = \sum_{t=1}^{T} R_t$. We optimize the parameters of the generator (θ) by performing gradient ascent on $\mathbb{E}[R]$. Using one of the REINFORCE family of algorithms, we can find an unbiased estimator of this as $\nabla_\theta \mathbb{E}_G[R_t] = R_t \nabla_\theta \log G_\theta(\hat{x}_t)$. The variance of this estimator may be reduced by using the learned value function as a baseline $b_t = V^G(x_{1:t})$. This results in the update equation

$$\nabla_\theta \mathbb{E}_G[R_t] = (R_t - b_t) \nabla_\theta \log G_\theta(\hat{x}_t) \tag{4}$$

In the nomenclature of RL, the quantity $(R_t - b_t)$ may be interpreted as an estimate of the advantage $A(a_t, s_t) = Q(a_t, s_t) - V(s_t)$. Here, the action a_t is the token chosen by the generator $a_t \equiv \hat{x}_t$ and

the state s_t are the current tokens produced up to that point $s_t \equiv \hat{x}_1, \cdots, \hat{x}_{t-1}$. This approach is an actor-critic architecture where G determines the policy and the baseline b_t is the critic (Sutton & Barto, 1998; Degris et al., 2012).

For this task, we design rewards at each time step for a single sequence in order to aid with credit assignment (Li et al., 2017). As a result, a token generated at time-step t will influence the rewards received at that time step and subsequent time steps. Our gradient update for the discounted total return $R = \sum_{t=1}^{T} R_t$ then becomes,

$$\nabla_\theta \mathbb{E}[R] = \mathbb{E}_{\hat{x}_t \sim G}\left[\sum_{t=1}^{T}(R_t - b_t)\nabla_\theta \log(G_\theta(\hat{x}_t))\right] \tag{5}$$

$$= \mathbb{E}_{\hat{x}_t \sim G}\left[\sum_{t=1}^{T}\left(\sum_{s=t}^{T}\gamma^s r_s - b_t\right)\nabla_\theta \log(G_\theta(\hat{x}_t))\right] \tag{6}$$

Intuitively, this shows that the gradient to the generator associated with producing $\hat{x}_t$ will depend on all the discounted subsequent rewards assigned by the discriminator. For a non-zero λ discount factor, the generator will be penalized for greedily selecting a token that earns a high reward at that time-step alone.

Finally, as in conventional GAN training, our discriminator will be updated according to the gradient

$$\nabla_\phi \frac{1}{m}\sum_{i=1}^{m}\left[\log D(x^{(i)})] + \log(1 - D(G(z^{(i)})\right] \tag{7}$$

3.4 Method Details

We have pretraining stages, first we train a language model using standard maximum likelihood training. We then use the pretrained language model weights for the seq2seq encoder and decoder modules. With these language models, we now pretrain the seq2seq model on the in-filling task using maximum likelihood, in particular, the attention parameters as described in Luong et al. (2015). We select the model producing the lowest validation perplexity on the masked task via a hyperparameter sweep over 500 runs.

4 Training Details

Our model was trained with the Adam method for stochastic optimization (Kingma & Ba, 2014) with the default Tensorflow exponential decay rates of $\beta_1 = 0.99$ and $\beta_2 = 0.999$. Our model uses 2-layers of 650 unit LSTMs for both the generator and discriminator, 650 dimensional word embeddings, variational dropout. We used Bayesian hyperparameter tuning to tune the variational dropout rate and learning rates for the generator, discriminator and critic. We perform 3 gradient descent steps on the discriminator for every step on the generator and critic.

We share the embedding and softmax weights of the generator as proposed in Bengio et al. (2003); Press & Wolf (2016); Inan et al. (2016). Furthermore, to improve convergence speed, we share the embeddings of the generator and the discriminator. Additionally, as noted in our architectural section, our critic shares all of the discriminator parameters with the exception of the separate output head to estimate the value. Both our generator and discriminator use variational dropout (Kingma et al., 2015).

5 Evaluation

Evaluation of generative models continues to be an open-ended research question. We seek heuristic metrics that we believe will be correlated with human-evaluation. BLEU score (Papineni et al., 2002) is used extensively in machine translation where one can compare the quality of candidate translations

from the reference. Motivated by this metric, we compute the number of unique n-grams produced by the generator that occur in the validation corpus for small n. Then we compute the geometric average over these metrics to get a unified view of the performance of the generator.

From our maximum-likelihood trained benchmark, we were able to find GAN hyperparameter configurations that led to small decreases in validation perplexity on $O(1)-$point. However, we found that these models did not yield considerable improvements to the sample quality so we abandoned trying to reduce validation perplexity. One of the biggest advantages of GAN-trained NLP models, is that the generator can produce alternative, yet realistic language samples, but not be unfairly penalized by not producing with high likelihood the single correct sequence. As the generator explores 'off-manifold' in the free-running mode, it may find alternative options that are valid, but do not maximize the probability of the underlying sequence. We therefore choose not to focus on architectures or hyperparameter configurations that led to small reductions in validation perplexity, but rather, searched for those that improved our heuristic evaluation metrics.

6 EXPERIMENTS

6.1 THE PENN TREEBANK (PTB)

The Penn Treebank dataset Marcus et al. (1993) has a vocabulary of 10,000 unique words. The training set contains 930,000 words, the validation set contains 74,000 words and the test set contains 82,000 words. For our experiments, we train on the training partition.

We first pretrain a language model with parameter dimensions common to MaskGAN in the Zaremba et al. (2014) code base to a validation perplexity of 78. After then loading the weights of from the language model into the MaskGAN generator we further pretrain with masking rate of 0.5 (half the text blanked) to a validation perplexity of 55.3. Finally, we then pretrain the discriminator on the samples produced from the current generator and real training text.

We present both conditional and unconditional samples generated on this data set. In the following section, MaskGAN refers to our GAN-trained variant and MaskMLE refers to our maximum-likelihood trained variant.

6.1.1 CONDITIONAL SAMPLES

We produce samples conditioned on surrounding text. Underlined sections of text are portions here and throughout the text have been filled in via either the MaskGAN or MaskMLE algorithm.

Ground Truth	**the next day 's show <eos> interactive telephone technology has taken a new leap in <unk> and television programmers are**
MaskGAN	the next day 's show <eos> interactive telephone technology has taken a new leap <u>in its retail business <eos> a</u> the next day 's show <eos> interactive telephone technology has <u>long dominated the <unk> of the nation 's largest economic</u> the next day 's show <eos> interactive telephone technology has <u>exercised a N N stake in the u.s. and france</u>
MaskMLE	the next day 's show <eos> interactive telephone technology has taken a new leap <u>in the complicate case of the</u> the next day 's show <eos> interactive telephone technology has <u>been <unk> in a number of clients ' estimates mountain-bike</u> the next day 's show <eos> interactive telephone technology has <u>instituted a week of <unk> by <unk> <unk> wis. auto</u>

Table 1: Conditional samples from PTB for both MaskGAN and MaskMLE models.

6.1.2 LANGUAGE MODEL (UNCONDITIONAL) SAMPLES

We may also run MaskGAN in an unconditional mode, where the entire context is blanked out, thus making it equivalent to a language model. We present a set of language model samples below in Table 2.

MaskGAN	a <unk> basis despite the huge after-tax interest income <unk> from $ N million <eos> in west germany N N oct. N as the end of the year the resignations were approved <eos> the march N N <unk> was down the world 's most corrupt organizations act as a multibillion-dollar <unk> atmosphere or the metropolitan zone historic array with their

Table 2: Language model (unconditional) samples from PTB for both MaskGAN.

6.2 IMDB MOVIE DATASET

The IMDB dataset Maas et al. (2011) consists of 100,000 movie reviews taken from IMDB. Each review may contain several sentences. The dataset is divided into 25,000 labeled training instances, 25,000 labeled test instances and 50,000 unlabeled training instances. The label indicates the sentiment of the review and may be either positive or negative. We use the first 40 words of each review in the training set to train our models, which leads to a dataset of 3 million words.

Identical to the training process in PTB, we pretrain a language model to a validation perplexity of 105.6. After then loading the weights of from the language model into the MaskGAN generator we further pretrain with masking rate of 0.5 (half the text blanked) to a validation perplexity of 87.1. Finally, we then pretrain the discriminator on the samples produced from the current generator and real training text.

6.2.1 CONDITIONAL SAMPLES

Ground Truth	**Pitch Black was a complete shock to me when I first saw it back in 2000 In the previous years I**
MaskGAN	Pitch Black was a complete shock to me when I first saw it back in 1979 I was really looking forward Pitch Black was a complete shock to me when I first saw it back in 1976 The promos were very well Pitch Black was a complete shock to me when I first saw it back in the days when I was a
MaskMLE	Black was a complete shock to me when I first saw it back in 1969 I live in New Zealand Pitch Black was a complete shock to me when I first saw it back in 1951 It was funny All Interiors Pitch Black was a complete shock to me when I first saw it back in the day and I was in

Table 3: Conditional samples from IMDB for both MaskGAN and MaskMLE models.

6.2.2 LANGUAGE MODEL (UNCONDITIONAL) SAMPLES

As in the case with PTB, we generate IMDB samples unconditionally, equivalent to a language model. We present length-40 samples of both positive and negative sentiment in Table 4.

6.3 PERPLEXITY OF GENERATED SAMPLES

As of this date, GAN training has not achieved state-of-the-art word level validation perplexity on the Penn Treebank dataset. Rather, the top performing models are still maximum-likelihood trained

MaskGAN	**Positive**: Follow the Good Earth movie linked Vacation is a comedy that credited against the modern day era yarns which has helpful something to the modern day s best It is an interesting drama based on a story of the famed **Negative**: I really can t understand what this movie falls like I was seeing it I m sorry to say that the only reason I watched it was because of the casting of the Emperor I was not expecting anything as **Negative**: That s about so much time in time a film that persevered to become cast in a very good way I didn t realize that the book was made during the 70s The story was Manhattan the Allies were to

Table 4: Language model (unconditional) samples from IMDB for both MaskGAN.

Model	Perplexity of samples under a pretrained LM
MaskMLE	273.1 ± 3.5
MaskMLE overfitting at 500k steps	155.3 ± 3.3
MaskGAN	108.3 ± 3.5

Table 5: The samples generated by the MaskGAN model consistently lower perplexity than samples generated by the MaskMLE model. The perplexity is calculated using a pre-trained language model that is equivalent to the decoder (in terms of architecture and size) used in the MaskMLE and MaskGAN models. This language model was used to initialize both models.

models, such as the recently well-regularized multilayer LSTM by Melis et al. (2017) followed by the architectures found via neural architecture search in Zoph & Le (2016). An extensive hyperparameter search with MaskGAN further supported that GAN training does not improve the validation perplexity results set via state-of-the-art models. However, we instead seek to understand the quality of the *sample generation*. As highlighted earlier, a fundamental problem of generating in free-running mode potentially leads to 'off-manifold' sequences which can result in poor sample quality for teacher-forced models. We seek to quantitatively evaluate this dynamic present only during sampling. This is commonly done with BLEU but as shown by Wu et al. (2016), BLEU is not necessarily correlated with sample quality. We believe the correlation may be even less in the in-filling task since there are many potential valid in-fillings and BLEU would penalize valid ones.

Instead, we calculate the perplexity of the generated samples by MaskGAN and MaskMLE by using the language model that was used to initialize MaskGAN and MaskMLE. Both MaskGAN and MaskMLE produce samples autoregressively (free-running mode), building upon the previously sampled tokens to produce the distribution over the next.

The MaskGAN model produces samples which are more likely under the initial model than the MaskMLE model. The MaskMLE model generates improbable sentences, as assessed by the initial language model, during inference as compounding sampling errors result in a recurrent hidden states that are never seen during teacher forcing Lamb et al. (2016). Conversely, the MaskGAN model operates in a free-running mode while training and this supports that it is more robust to these sampling perturbations.

6.4 MODE COLLAPSE

In contrast to image generation, mode collapse can be measured by directly calculating certain n-gram statistics. In this instance, we measure mode collapse by the percentage of unique n-grams in a set of 100K generated IMDB samples. We generate each sample (consisting of 40 words) conditioned only on the label. This results in almost 4M bi/tri/quad-grams.

In addition, all complete samples (taken as a sequence) are unique. We also observed during RL training an initial small drop in perplexity on the ground-truth validation set but then a steady increase in perplexity as training progressed. Despite this, sample quality remained relatively consistent. The final samples were generated from a model that had a perplexity on the ground-truth of 400. We

Model	Unique bigrams	Unique trigrams	Unique quadgrams
MaskMLE	24.6	57.6	81.2
MaskMLE overfitting at 500k steps	13.7	40.3	66.6
MaskGAN	16.7	41.5	64.6

Table 6: As often witnessed in GAN literature, our discrete GAN is subject to some mode collapse.

hypothesise that mode dropping is occuring near the tail end of sequences since generated samples are unlikely to generate all the previous words correctly in order to properly model the distribution over words at the tail. Theis et al. (2015) also shows how validation perplexity is not a good measurement of sample quality.

6.5 HUMAN EVALUATION

We evaluate the quality of the generated samples of the MaskGAN and the MaskMLE in a blind heads-up comparison using Amazon Mechanical Turk. We pay raters to compare the quality of two extracts along 3 axes (grammaticality, topicality and overall quality). They are asked if the first extract, second extract or neither is higher quality.

Model	Grammaticality	Topicality	Overall
MaskGAN	54.1	44.8	50.9
MaskMLE	25.8	29.4	26.2

Table 7: A Mechanical Turk heads-up evaluation between MaskGAN and MaskMLE IMDB samples when asked which model's samples raters preferred out of 100 samples. 279 ratings were obtained.

Model	Grammaticality	Topicality	Overall
MaskGAN	44.7	27.7	45.7
MaskMLE	37.2	36.3	40.4

Table 8: A Mechanical Turk heads-up evaluation between MaskGAN and MaskMLE PTB samples when asked which model's samples raters preferred out of 100 samples. 94 ratings were obtained.

The Mechanical Turk results show that MaskGAN generates superior human-looking samples to MaskMLE on the IMDB dataset. However, on the smaller PTB dataset, the results are closer. We also show results with SeqGAN (trained with the same network size and vocabulary size) as MaskGAN, which show similar results to MaskGAN on PTB but with a smaller margin. In additional tests of generated samples, we found that 20.6% of 4-grams in SeqGAN samples appeared in the training set compared to 10.8% of 4-grams in MaskGAN samples, showing that our model is less prone to mode collapse well generating high-quality samples.

7 DISCUSSION

We generally found training with contiguous masks to produce better samples. We believe this is simply an issue of error attribution, especially if the discriminator is not conditioned on the mask. For instance, suppose a random mask is sampled that has masked-out words with many real words in-between, an easy solution for the generator is to simply repeat the preceding or following word. It would then be very difficult for the discriminator to determine which word in the repeat was the real word and which was generated. Indeed, we observed this behavior in early samples.

We also found the use of attention was important for the in-filled words to be sufficiently conditioned on the input context. Without attention, the in-filling would fill in reasonable subsequences that became implausible in the context of the surrounding words.

Model	Grammaticality	Topicality	Overall
SeqGAN	45.8	34.4	45.8
MaskMLE	42.7	35.4	42.7

Table 9: A Mechanical Turk heads-up evaluation between SeqGAN and MaskMLE PTB samples when asked which model's samples raters preferred out of 100 samples. 96 ratings were obtained. We were unable to obtain sufficient responses on Mechanical Turk and thus we are unable to rule out the null hypothesis $p = 0.5$ with high significance.

In general we think the proposed contiguous in-filling task is a good approach to reduce mode collapse and help with training stability for textual GANs. We show that MaskGAN samples on a larger dataset (IMDB reviews) is significantly better than the corresponding tuned MaskMLE model as shown by human evaluation. We also show we can produce high-quality samples despite the MaskGAN model having much higher perplexity on the ground-truth test set.

REFERENCES

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086*, 2016.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pp. 1171–1179, 2015.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

Tong Che, Yanran Li, Ruixiang Zhang, R Devon Hjelm, Wenjie Li, Yangqiu Song, and Yoshua Bengio. Maximum-likelihood augmented discrete generative adversarial networks. *arXiv preprint arXiv:1702.07983*, 2017.

Thomas Degris, Patrick M Pilarski, and Richard S Sutton. Model-free reinforcement learning with continuous action in practice. In *American Control Conference (ACC), 2012*, pp. 2177–2182. IEEE, 2012.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pp. 6645–6649. IEEE, 2013.

Alex Graves et al. *Supervised sequence labelling with recurrent neural networks*, volume 385. Springer, 2012.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017.

R Devon Hjelm, Athul Paul Jacob, Tong Che, Kyunghyun Cho, and Yoshua Bengio. Boundary-seeking generative adversarial networks. *arXiv preprint arXiv:1702.08431*, 2017.

Hakan Inan, Khashayar Khosravi, and Richard Socher. Tying word vectors and word classifiers: A loss framework for language modeling. *arXiv preprint arXiv:1611.01462*, 2016.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, pp. 2575–2583, 2015.

Matt J Kusner and José Miguel Hernández-Lobato. Gans for sequences of discrete elements with the gumbel-softmax distribution. *arXiv preprint arXiv:1611.04051*, 2016.

Alex M Lamb, Anirudh Goyal ALIAS PARTH GOYAL, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances In Neural Information Processing Systems*, pp. 4601–4609, 2016.

Jiwei Li, Will Monroe, Tianlin Shi, Alan Ritter, and Dan Jurafsky. Adversarial learning for neural dialogue generation. *arXiv preprint arXiv:1701.06547*, 2017.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 142–150. Association for Computational Linguistics, 2011.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

Gábor Melis, Chris Dyer, and Phil Blunsom. On the state of the art of evaluation in neural language models. *arXiv preprint arXiv:1707.05589*, 2017.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, pp. 3, 2010.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. Virtual adversarial training for semi-supervised text classification. *ICLR 2017*, 1050:25, 2016.

Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318. Association for Computational Linguistics, 2002.

Ofir Press and Lior Wolf. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859*, 2016.

Sai Rajeswar, Sandeep Subramanian, Francis Dutil, Christopher Pal, and Aaron Courville. Adversarial generation of natural language. *arXiv preprint arXiv:1705.10929*, 2017.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112, 2014.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.

George Tucker, Andriy Mnih, Chris J Maddison, and Jascha Sohl-Dickstein. Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models. *arXiv preprint arXiv:1703.07370*, 2017.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016. URL `http://arxiv.org/abs/1609.08144`.

Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. Improving neural machine translation with conditional sequence generative adversarial nets. *arXiv preprint arXiv:1703.04887*, 2017.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pp. 2852–2858, 2017.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.

Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

A APPENDIX

In our appendix, we explore various failure modes of the MaskGAN model.

A.1 MODE COLLAPSE

As widely witnessed in GAN-training, we also find a common failure of mode collapse across various n-gram levels. The mode collapse may not be as extreme to collapse at a 1-gram level (`ddddddd`$\cdots$) as described by Gulrajani et al. (2017), but it may manifest as grammatical, albeit, inanely repetitive phrases, for example,

MaskGAN	It is a very funny film that is very funny It s a very funny movie and it s charming It

Table 10: Mode collapse failure mode occurring at a higher order n-gram.

Of course the discriminator may discern this as an out-of-distribution sample, however, in certain failure modes, we observed the generator to move between common modes frequently present in the text.

A.2 MATCHING SYNTAX FOR FUTURE CONTEXT

We notice that the MaskGAN architecture often struggles to produce syntactically correct sequences when it must condition on future context. This is also a relatively challenging task for humans, because the filled in text must not only be contextual but also match syntactically at the boundary between the blank and where the text is present.

MaskGAN	Cartoon is one of those films me when I first saw it back in 2000

Table 11: Demonstration of syntax not being correct at the boundary of generated and present text.

As noted in this failure mode, the intersection between the filled in text and the present text is non grammatical.

A.3 LOSS OF GLOBAL CONTEXT

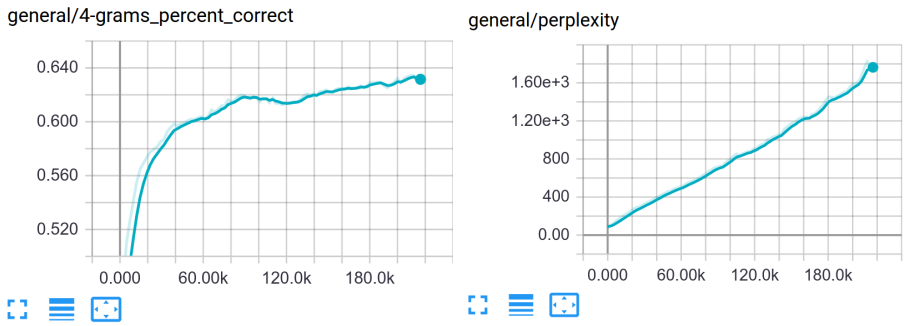
Similar to failure modes present in GAN image generation, the produced samples often can lose global coherence, despite being sensible locally.

MaskGAN	This movie is terrible The plot is ludicrous The title is not more interesting and original This is a great movie Lord of the Rings was a great movie John Travolta is brilliant

Table 12: Unconditional and conditional MaskGAN samples failing to maintain global coherence in the IMDB dataset.

A.4 n-GRAM METRICS MAY BE MISLEADING PROXIES

In the absence of a global scalar objective to optimize while training, we monitor various n-gram language statistics to assess performance. However, these only are crude proxies of the quality of the produced samples.



(a) 4-gram (b) Perplexity

Figure 2: Particular failure mode succeeding in the optimization of a 4-gram metric at the extreme expense of validation perplexity. The resulting samples are shown below.

For instance, MaskGAN models that led to improvements of a particular n-gram metric at the extreme expense of validation perplexity as seen in Figure 2 could devolve to a generator of very low sample diversity. In Table 13, we produce several samples from this particular model which, despite the dramatically improved 4-gram metric, has lost diversity.

MaskGAN	It is a great movie It s just a tragic story of a man who has been working on a home It s a great film that has a great premise but it s not funny It s just a silly film It s not the best movie I have seen in the series The story is simple and very clever but it

Table 13: Demonstration of low sample diversity when the MaskGAN model over-optimizes n-gram metrics.

Capturing the complexities of natural language with these metrics alone is clearly insufficient.

A.5 SEQGAN SAMPLES

We modified SeqGAN to train and generate PTB samples using the same size architecture for the generator as in the MaskGAN generator.

```
are removed <eos> another takeover target lin 's directors
attempted through october <unk> and british airways is allowed
three funds
poor utilization of the company 's facilities and its headquarters
for virtually most of its previous age but remains small
cineplex odeon corp. shares made fresh out of the group purchase
one part of a revised class of <unk> british
include its $ N billion in bonds and by far short interest <eos>
on friday the company said it offered
there are <unk> <unk> and <unk> about the <unk> seed <eos> they
use pcs are <unk> and their performance <eos>
```