Discrete Text Generation using Generative Adversarial Networks

Jianguo Zhang

June 20, 2018

Abstract

Language generation plays an important role in natural language processing, which is essential to many applications such as machine translation and dialogue systems. Recent studies show that the recurrent neural networks (RNNs) and the long short-term memory networks (LSTMs) can achieve impressive performance for the task of language generation. The most common approach to train an RNN is to maximize the log predictive likelihood of each true token in the training sequence given the previous observed tokens. However, this method suffers from so-called exposure bias in the inference stage: the model generates a sequence iteratively and predicts next token conditioned on its previously predicted ones that may be never observed in the training data. To address this problem, Scheduled Sampling method is proposed, where the generative model is partially fed with its own synthetic data as observed tokens rather than the true data when deciding the next token in the training state. Nevertheless, it cannot solve the problem fundamentally. Another solution is to build specific loss function like BLEU on the entire generated sequence instead of each transition. However, a task specific loss may not be generalized to many different tasks.

Generative adversarial networks (GANs) have drawn great attentions since Goodfellow et al. introduced the framework for generating the synthetic data that is similar to the real one. GANs do not need to design specific loss, as the discriminator just distinguishes the generated data from the real data. GANs have had a lot of success in producing more realistic images than other approaches, but they have only 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. Previous works such as SeqGAN addresses this problem using policy gradient method. However, their model does not consider different entities insider the sentences, and the

generated texts lack variety. In this proposal, we focus on two aspects: Firstly, improving the quality and diversity of generated sentences. Secondly, generating sentences based on conditions (e.g. labels and entities).

Contents

1 Introduction

The ability to generate coherent and semantically meaningful text plays a key role in many natural language processing applications such as machine translation [3] and dialogue system [28]. A typical approach is to train a recurrent neural network [12] that maximizes the log-likelihood of each ground-truth word given prior observed words. However, it suffers from so-called exposure bias [27] due to the discrepancy between training and inference stage: the model sequentially generates the next word based on previously generated words during inference but it is trained to generate words given ground-truth words. Therefore, the decoder can behave in undesired ways, such as generating truncated or repetitive outputs, outputting bland and generic responses, or in some cases, producing ungrammatical gibberish.

The sequential nature of reinforcement learning makes it applicable on sequence generation [2, 15, 27]. Moreover, the essence of deep reinforcement learning, assigning time-relayed rewards in the sequence generation process, is different to supervised learning and could make up the discrepancy between training and inference. Therefore, deep reinforcement learning could further guide the model to the improved quality and readability in generation. In addition, deep reinforcement learning usually uses diversity metrics as the objective to evaluate quality of generated sequence, such as BLEU [25] and ROUGEL [20], which respectively focuses on the precision and recall of n-grams. Beyond them, METEOR [4] uses a combination of both the precision and the recall of n-grams. As we can see, such metrics mostly rely on matching n-grams with the "ground-truths". As a result, sentences that contain frequent n-grams will get higher scores as compared to those using variant expressions. Recently, a new metric SPICE [18] was proposed. Instead of matching between n-grams, it focuses on those linguistic entities that reflect visual concepts (e.g. relationships and objects), while other qualities, e.g. the naturalness of the expressions, are not considered in this metric. Hence, deep reinforcement learning models taking these evaluation metrics as objectives are not enough to generate sequential sentences with high quality and diversity.

Generative adversarial network (GAN) was first proposed in [9], which is designed for generating real-valued, continuous data, and it has had a lot of success in producing more realistic images [14, 17, 26] than other approaches. Basically, GAN includes a generator and a discriminator. The generator tries to generate realistic images or texts and fool the discriminator. The discriminator discerns whether a generated image or sentence is real or fake. The discriminator evaluates the quality of whole generated sentence and only needs to discriminate whether the whole sentence is real or not. It avoids the problem of exposure bias and does not need a task-specific loss such as BLEU mentioned above. However, it has only seen limited use for text sequences in applications. There are several reasons. Firstly, recall that the generator does not have direct access to the training data and it receives signals from the discriminator. The discrete nature of text makes it infeasible to propagate the gradient from the discriminator back to the generator as in standard GAN

training [16, 34]. In other words, the gradient of the loss from D w.r.t. the outputs by G is used to guide the parameters of G to slightly change the generated value to make it more realistic. If the generated data is based on discrete tokens, the "slight change" guidance from the discriminator makes little sense because there is probably no corresponding token for such slight change in the limited dictionary space [7]. Secondly, GAN can only give the score for an entire sequence when it has been generated, it is nontrivial to balance how good it is now and the future score as an entire sequence, and this problem compounds when generating longer and longer sentences. Besides, it also suffers from mode collapse problems. Mode collapse occurs when certain modalities in the training set are rarely generated by the generator, e.g. leading all generated images of a volcano to be multiple variants of the same volcano. This problem becomes more severe in text generation since there are many complex modes in the data, ranging from bigrams to short phrases to longer idioms. In fact, due to the complex nature of text, it is very hard to handle mode collapse problem.

In this proposal, we will focus on two aspects: Firstly, how to improve the quality and diversity of generated sentences. Secondly, how to generate sentences based on conditions like labels and aspects inside the referential sentence.

2 Backgrounds

2.1 Deep Leaning on Sequence Generation

Traditional Markov model approaches are limited because their states must be drawn from a modestly sized discrete state space $\mathcal{S}$. Therefore Markov models are computationally impractical for modeling long-range dependencies [10]. While recurrent neural networks can capture long-range time dependencies, overcoming the major limitation of Markov models. The Sequence-to-Sequence (Seq2Seq) model [31] in the encoder-decoder format has been widely used in the sequence generation tasks, such as machine translation, dialogue generation and music generation. The Seq2Seq technique trains end-to-end neural network models which learns to encode a variable-length sequence into a fixed-length vector representation and to decode a given fixed-length vector representation back into a variable-length sequence.

2.2 Sequence-to-Sequence (Seq2Seq) Model

2.2.1 Seq2Seq Model

From a probabilistic perspective, Sequence-to-Sequence (Seq2Seq) model is a general method to learn the conditional distribution over a variable-length sequence conditioned on another variable-length sequence, namely, $P(y^1, ..., y^{T'}|x^1, ..., x^T)$, where the input and output sequence lengths T and T' may differ.

The encoder of Seq2Seq model is an RNN that reads each symbol of an input sequence x sequentially. As it reads each symbol, the hidden state of the RNN changes $h_t = f(h_{t-1}, x_t)$. After reading the end of the sequence, the hidden state of the RNN is a summary c of the whole input sequence.

The decoder of Seq2Seq model is another RNN which is trained to generate the output sequence by predicting the next symbol y_t given the hidden state s_t. Both y_t and s_t are also conditioned on y_{t-1} and on the summary c of the input sequence. Hence, the hidden state of the decoder at time t is computed by $s_t = f(s_{t-1}, y_{t-1}, c)$ and the conditional distribution of the next symbol is $P(y_t|y_{t-1}, y_{t-2}, ..., y_1, c) = g(s_t, y_{t-1}, c)$, where f and g are activation functions.

The best performing models also connect the encoder and decoder through an attention mechanism [3, 32]. An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

The Attentional Seq2Seq model in [24], at the j-th step of the decoder process, the attention score $\alpha_j(i)$ between the i-th source hidden unit h_i and the j-th target hidden unit s_j is calculated as follows:

$$\alpha_j(i) = \frac{exp(h_i \cdot s_j)}{\sum_{i=1}^{k} exp(h_k \cdot s_j)}, \tag{1}$$

where $h_i \cdot s_j$ is the inner product of h_i and s_j, which is used to directly calculate the similarity score between h_i and s_j. The j-th context vector c_j is calculated as the summation vector weighted by $\alpha_j(i)$:

$$c_j = \sum_{i=1}^{n} \alpha_j(i) h_i. \tag{2}$$

To incorporate the attention mechanism into the decoding process, the context vector is used for the the j-th word prediction by concatenating the context vector with hidden layer.

2.2.2 Problems of Seq2Seq Model

Seq2Seq model usually use maximum likelihood estimation (MLE) as the objective function, which predicts the next word based on previous ground-truth words or generated words. Typically, if the target sequences with T words which is $[w_1, w_2, ..., w_T]$, then the objective can be written as:

$$L = -log\ p(w_1, ..., w_T) = -log \prod_{t=1}^{T} p(w_t|w_1, ..., w_{t-1}) = -log \sum_{t=1}^{T} p(w_t|w_1, ..., w_{t-1}). \tag{3}$$

This loss function trains the model to be good at greedily predicting the next word at each time without considering the whole sequence. It causes two problems: Firstly, during training stage, the model predicts the next word based on the previous ground-truth words, whereas during inference stages, the model predicts the next word based on the previous generated words. This leads to inconsistencies between the training and inference stages, and the error of predictions usually accumulated based on previous inaccurately predicted words. Secondly, The model predicts the next word based on previous words without considering the whole sequence. As a consequence, the prediction lacks global view of the sequence, and the generated sentences will bear high resemblance to training sentences in detailed wording, with very limited variability in expression [6].

2.3 Evaluation Methodologies for Generated Sentences

2.3.1 Evaluation Metrics

Human evaluation are the most accurate metric to measure the performance of generated corpus. But usually it is expensive and time-costing. Bilingual evaluation understudy (BLEU score) is designed to approximate human judgment at a corpus level [25]. It is the geometric mean of the n-gram precisions for all values of n between 1 and some upper limit N.

$$BLEU = B \cdot exp(\frac{1}{N}\sum_{n=1}^{N} log\ p_n), \tag{4}$$

where B is a penalty when the average length of the candidate translations c is larger than the average length of the reference translations r, B is equal to 1. Otherwise, B is equal to $e^{(1-r/c)}$. p_n is the modified n-gram precision, which is the number of n-grams in the candidate translation that occur in any of the reference translations, divided by the total number of n-gram in the candidate translation. There are other metrics like METEOR [4] and ROUGE [21, 20]. METEOR is based on explicit word to word matches between candidates and reference sentences. ROUGE score is the overlap of N-grams between the candidate and reference sentences.

2.3.2 Problems of General Evaluation Metrics

As we can see, the metrics discussed on above section mostly rely on matching n-grams with the ground-truths. As a result, sentences that contain frequent n-grams will get higher scores as compared to those using variant expressions. Besides, under these metrics, sentences that contain matched n-grams would get substantially higher scores than those using variant expressions [1]. This issue is manifested by the fact that human descriptions get considerably lower scores. Furthermore, some new evaluation metrics, e.g. SPICE [1], instead of matching between n-grams, which focuses on the linguistic entities that reflect

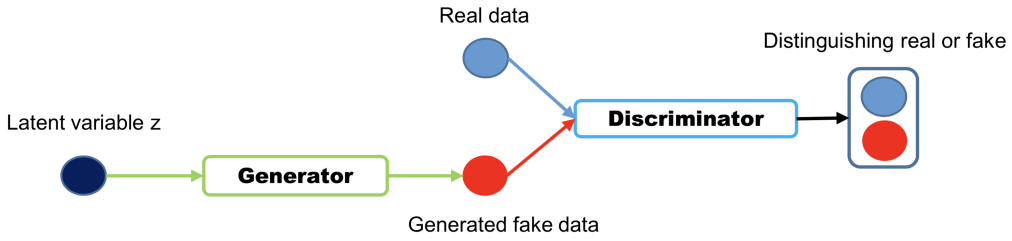

Figure 1: Generative Adversarial Network

visual concepts. However, SPICE still cannot consider some other qualities like naturalness and diversity of generated sentences.

3 Generative Adversarial Networks (GANs)

3.1 Framework of GANs

Goodfellow et al. [9] proposed GAN, which includes a generator and a discriminator and they formulate a game of two players: the discriminator, $D(x)$, takes a point x in data space and computes the probability that x is sampled from data distribution P_{data}, rather than generated by the generator G. At the same time, the generator first maps a noise vector z drawn from a prior $P(z)$ to the data space, obtaining a sample $G(z)$ that resembles the training data, and then uses this sample to challenge the discriminator. The mapping $G(z)$ induces a generator distribution P_G in data domain with probability density function $P_G(x)$. The goal of GAN is to produce and approximation to a target distribution, this is done iteratively via G and D. Both G and D are parameterized by neural networks and learned by solving the following minimax optimization:

$$\min_G \max_D \mathcal{J}(D,G) = E_{x\sim P_{data}(x)}[log\ D(x)] + E_{z\sim P_z}[log(1-D(G(z)))]. \tag{5}$$

Fig. 1 shows the framework of GAN. The discriminator aims to accurately discriminate between samples from target distribution and those realistic images or sentences produced by the generator, whereas the generator tries to fool the discriminator. They compete with each other and iteratively update, when they reach the Nash Equilibrium, the discriminator cannot distinguish whether the generated samples are real or not, and the generator can keep generating realistic samples.

3.2 Strengths of GANs

Compared with seq2seq model (Sec. 2.2) and general evaluation metrics (Sec. 2.3). GAN has two important advantages.

Firstly, Many models, e.g. Seq2Seq models and recurrent neural networks [12], which usually need to design specific loss functions based on different tasks and datasets. For example, to improve the quality of generated samples, one may need to design specific loss to compare the similarity between hidden features of the generated sample and the real sample on final neural network layers [29]. However, such task-specific loss may not generalize to other tasks such as dialogue generation [19]. While the discriminator in GAN only need to discriminate whether a generated sample is real or not, we don't need to design task-specific loss for different tasks. Hence, GAN is more suitable to many applications.

Secondly, since GAN evaluate the quality of the whole generated sentence, it naturally avoids the exposure bias problem caused by Seq2Seq model, and it can improve the quality of the generated sentences as it does not predict and optimize the prediction word by word.

Thirdly, GAN avoids the problems caused by general evaluation metrics such as BLEU, METEOR, since it evaluates the generated sentence as a whole sentence, and doesn't focus on the the n-grams or specific tokens inside the sentences. Hence, GAN has advantages to consider the naturalness and diversity of generated sentences and generate realistic samples in a human-written way.

3.3 Challenges of GANs

3.3.1 Lack of Diversity

GAN suffers from mode collapse problem. Mode collapse 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 multiple variants of the same volcano, or over generating some unseen images (an image of pig contains two heads). 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. For more details, please refer to [8, 35].

3.3.2 Limited Progress of Generating Sentences Based on Conditions

So far, there are several works applying GANs to discrete text generations [11, 22, 34, 35], and most of them focus on generating sentences directly from referential sentences and paragraphs only. However, in real scenarios, like dialogue systems, we want to generate questions based on answers. Besides, in addition to the given sentences, it is also useful to specify a relevant aspect contained in text to further improve diversity and quality of the generated sentences [13]. However, it may contains many aspects and labels inside a long sentences. For instance, a sentence "The weather is pretty good today. I take some medicine because of my headache, and I give the presentation for CS590. It is a

really a good course. I learn a lot from it." with a label "positive", includes several verbs and nouns, and the label "positive" is not enough. Moreover, sentences like this are very time-consuming for humans to labels and identify aspects in large datasets. Hence, how to extract the aspects and labels is not a trivial problem. As a consequence, it also makes the research of text generation move very slowly compared with image generations.

4 Proposals

In Sec. 3.1, we discussed the general idea of GAN and its significant strengths over Seq2Seq model (Sec. 2.2) and general evaluation metrics (Sec. 2.3). We also presented some challenges of GAN (Sec. 3.3). In this section, we will propose some ideas to tackle these challenges.

4.1 Improving Quality and Diversity of Generated Sentences

4.1.1 Methodologies

Eq. 5 is usually the objective of standard GAN in many text generation works [11, 34, 35]. The output of the discriminator is a binary prediction indicating whether the given sentence is written by human or machine.

However, for a large variety of natural language expressions, this binary prediction is too restrictive, since the diversity and richness inside the sentences are constrained by the degenerated distribution due to binary classification.

Hence, we can design a ranker R consisting of one synthetic sequence generated by the generator G, and multiple human-written sentences, given the reference sentence U. The learning objective of G is to produce a synthetic sentence that receives higher ranking score than those drawn from real data. The goal of R is to rank the synthetic sentence lower than human-written sentences. Thus, we can treat R and G as playing a minimax game with the objective function:

$$\min_{\theta} \max_{\phi} = E_{s\sim\mathcal{P}_h}[log\ R_\phi(s|U,\mathcal{C}^-)] + E_{s\sim G_\theta}[log(1 - \ R_\phi(s|U,\mathcal{C}^+))], \tag{6}$$

where θ and ϕ are the variable parameters in G and R, respectively. The E is the expectation operator, and $\mathcal{P}$ is the real data from human-written sentences, $s \sim \mathcal{P}_h$ and $s \sim G_\theta$ denote that s is from human-written sentences and synthesized sentences, respectively. When the input sentence s is from the real data, $\mathcal{C}^-$ is pre-sampled from generated data. If the input sentence s is from the synthetic data, the $\mathcal{C}^+$ is pre-sampled from human-written data.

For the discriminator, we can compare the relevance score to measure an overall quality of a generated sentence in contrast to a set of sentences in the datasets. The relevance score (cosine similarity) of an input sequence c given a reference u can be defined as:

Table 1: Performance of models on MSCOCO test set

Model	BLEU3	BLEU4
MLE	0.397	0.297
Our	0.305	0.215

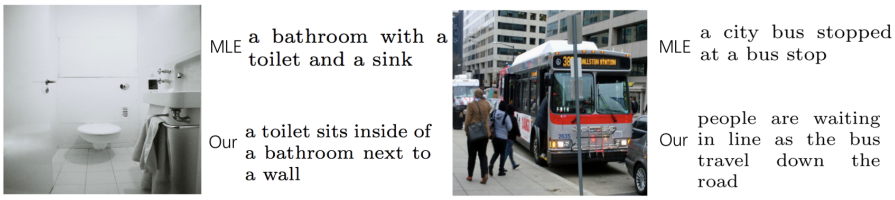

Figure 2: Quantitative results

$S(c|u) = (y_c^T \cdot y_u)/(\|y_c\| \, \|y_u\|)$, where y_c and y_u are the embedding hidden features of the input sequence and the reference. The overall quality of a generated sentence in contrast to a set of references C can be defined as:

$$D_\phi(c|u, C) = \frac{exp(S(c|u))}{\sum_{c' \in C} exp(S('c|u))}. \tag{7}$$

Note, the reference set C includes the generated sentence, unrelated sentences, and human-written sentences. The reason that we also use the unrelated sentences is that the generated sentences by the generator should have various expressions. At the same time, they should also be related to the context or reference.

4.1.2 Initial Experimental Results

We conduct the experiments on the MSCOCO dataset [23], which contains 82081 training images and 40137 validation images, each image has 5 human-written captions. We use 5000 images for both validation and testing, and the rest for training. For the baseline, we compare with the Maximum Likelihood Estimation (MLE) based method [33]. For the evaluation, we use BLEU score [25] to evaluate the performance of the model.

Table 1 shows the results, our model receive lower BLEU scores compared with MLE. However, as reported in previous works [5], higher BLUE score does not mean the generated sentences have good quality. Fig. 2 shows that our method performs better than MLE.

4.2 Generating Sentences Based on Conditions (e.g. labels, aspects)

Previous works discussed in Sec. 3.3.2 mainly generate sentences from referential sentences, paragraphs, or structured data only, and they did not consider different aspects and mul-

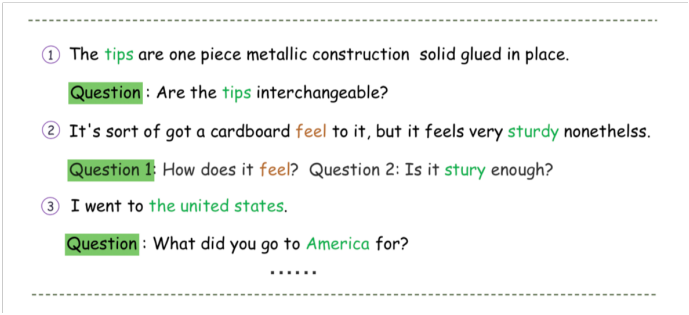

Figure 3: Examples of the observation [13].

tiple labels inside the context. In order to generate sentences based on conditions such as aspects and labels, we need to extract the aspects and multiple labels from the long sentences or paragraphs. However, it is very time-consuming to extract them by hand. To tackle this problem. we have two ways.

For the first way, we can measure the cosine similarity between two sentences. For example, Fig. 3 shows the observation of conversations, the "the united states" and "America" although are represented by different words, they have the same meaning. We can use cosine similarity to easily extract these words.

Another way to better extract the aspects and labels from long sentences is to use self-attention [30], which is originally used for natural language inference. We can compare the input sentence with itself. The aspects or multiple labels inside the sentence will receive higher probability and other unimportant words will receive lower probability.

The experiments for this part are still in process.

References

[1] P. Anderson, B. Fernando, M. Johnson, and S. Gould. Spice: Semantic propositional image caption evaluation. In *European Conference on Computer Vision*, pages 382–398. Springer, 2016.

[2] D. Bahdanau, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, and Y. Bengio. An actor-critic algorithm for sequence prediction. In *ICLR*, 2017.

[3] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2014.

[4] S. Banerjee and A. Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

[5] B. Dai, S. Fidler, R. Urtasun, and D. Lin. Towards diverse and natural image descriptions via a conditional gan. *ICCV*, 2017.

[6] J. Devlin, S. Gupta, R. Girshick, M. Mitchell, and C. L. Zitnick. Exploring nearest neighbor approaches for image captioning. *arXiv preprint arXiv:1505.04467*, 2015.

[7] I. Goodfellow. Generative adversarial networks for text. *http://goo.gl/Wg9DR7*, 2016.

[8] I. Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[10] A. Graves, G. Wayne, and I. Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[11] J. Guo, S. Lu, H. Cai, W. Zhang, Y. Yu, and J. Wang. Long text generation via adversarial training with leaked information. *AAAI*, 2018.

[12] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[13] W. Hu, B. Liu, J. Ma, D. Zhao, and R. Yan. Aspect-based question generation. *ICLR Workshop*, 2018.

[14] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *ICCV*, 2017.

[15] N. Jaques, S. Gu, R. E. Turner, and D. Eck. Tuning recurrent neural networks with reinforcement learning. In *ICLR*, 2017.

[16] M. J. Kusner and J. M. Hernández-Lobato. Gans for sequences of discrete elements with the gumbel-softmax distribution. *arXiv preprint arXiv:1611.04051*, 2016.

[17] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2016.

[18] J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*, 2015.

[19] J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky. Adversarial learning for neural dialogue generation. *arXiv preprint arXiv:1701.06547*, 2017.

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

[21] C.-Y. Lin and E. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 71–78. Association for Computational Linguistics, 2003.

[22] K. Lin, D. Li, X. He, Z. Zhang, and M.-T. Sun. Adversarial ranking for language generation. In *Advances in Neural Information Processing Systems*, pages 3155–3165, 2017.

[23] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[24] M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[25] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[26] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *ARXIV*, 2016.

[27] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. Sequence level training with recurrent neural networks. In *ICLR*, 2016.

[28] K. Reschke, A. Vogel, and D. Jurafsky. Generating recommendation dialogs by extracting information from user reviews. In *NAACL*, volume 2, pages 499–504, 2013.

[29] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[30] T. Shen, T. Zhou, G. Long, J. Jiang, S. Pan, and C. Zhang. Disan: Directional self-attention network for rnn/cnn-free language understanding. *AAAI*, 2018.

[31] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[33] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 3156–3164. IEEE, 2015.

[34] L. Yu, W. Zhang, J. Wang, and Y. Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pages 2852–2858, 2017.

[35] Y. Zhang, Z. Gan, K. Fan, Z. Chen, R. Henao, D. Shen, and L. Carin. Adversarial feature matching for text generation. *ICML*, 2017.