Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets

Zhen Yang[1,2]**, Wei Chen**[1*]**, Feng Wang**[1,2]**, Bo Xu**[1]

[1]Institute of Automation, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
`{yangzhen2014, wei.chen.media, feng.wang, xubo}@ia.ac.cn`

Abstract

This paper proposes an approach for applying GANs to NMT. We build a conditional sequence generative adversarial net which comprises of two adversarial sub models, a generator and a discriminator. The generator aims to generate sentences which are hard to be discriminated from human-translated sentences (i.e., the golden target sentences); And the discriminator makes efforts to discriminate the machine-generated sentences from human-translated ones. The two sub models play a mini-max game and achieve the win-win situation when they reach a Nash Equilibrium. Additionally, the static sentence-level BLEU is utilized as the reinforced objective for the generator, which biases the generation towards high BLEU points. During training, both the dynamic discriminator and the static BLEU objective are employed to evaluate the generated sentences and feedback the evaluations to guide the learning of the generator. Experimental results show that the proposed model consistently outperforms the traditional RNNSearch and the newly emerged state-of-the-art Transformer on English-German and Chinese-English translation tasks.

1 Introduction

Neural machine translation (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014; Cho et al., 2014; Bahdanau et al., 2014) which directly leverages a single neural network to transform the source sentence into the target sentence, has drawn more and more attention in both academia and industry (Shen et al., 2015; Wu et al., 2016; Johnson et al., 2016; Gehring et al., 2017; Vaswani et al., 2017). This end-to-end NMT typically consists of two sub neural networks. The encoder network reads and encodes the source sentence into the context vector representation; and the decoder network generates the target sentence word by word based on the context vector. To dynamically generate a context vector for a target word being generated, the attention mechanism which enables the model to focus on the relevant words in the source-side sentence is usually deployed. Under the encoder-decoder framework, many variants of the model structure, such as convolutional neural network (CNN) and recurrent neural network (RNN) are proposed (Bahdanau et al., 2014; Gehring et al., 2017). Recently, (Gehring et al., 2017) propose the Transformer, the first sequence transduction model based entirely on attention, achieving state-of-the-art performance on the English-German and English-French translation tasks. Despite its success, the Transformer, similar to traditional NMT models, is still optimized to maximize the likelihood estimation of the ground word (MLE) at each time step. Such an objective poses a hidden danger to NMT models. That is, the model may generate the best candidate word for the current time step yet a bad component of the whole sentence in the long run. Minimum risk training (MRT) (Shen et al., 2015) is proposed to alleviate such a limitation by adopting the sequence level objective, i.e., the sentence-level BLEU, for traditional NMT models. Yet somewhat improved, this objective still does not guarantee the translation results to be natural and sufficient. Since the BLEU point is computed as the geometric mean of the modified n-gram precisions (Papineni et al., 2002), almost all of the existing objectives essentially train NMT models to generate sentences with n-gram precisions as high as possible (MLE can be viewed to generate sentences with high 1-gram precisions). While n-gram precisions largely tell the good sentence apart from the bad one, it is widely acknowledged that higher n-gram precisions do not guarantee better sentences (Callison-Burch and Osborne, 2006; Chatterjee et al., 2007). Addition-

ally, the manually defined objective, i.e., the n-gram precision, is unable to cover all crucial aspects of the data distribution and NMT models may be trained to generate suboptimal sentences (Luc et al., 2016).

In this paper, to address the limitation mentioned above, we borrow the idea of generative adversarial training from computer vision (Goodfellow et al., 2014; Denton et al., 2015) to directly train the NMT model generating sentences which are hard to be discriminated from human translations. The motivation behind is that while we can not manually define the data distribution of golden sentences comprehensively, we are able to utilize a discriminative network to learn automatically what the golden sentences look like. Following this motivation, we build a conditional sequence generative adversarial net where we jointly train two sub adversarial models: A generator generates the target-language sentence based on the input source-language sentence; And a discriminator, conditioned on the source-language sentence, predicts the probability of the target-language sentence being a human-generated one. During the training process, the generator aims to fool the discriminator into believing that its output is a human-generated sentence, and the discriminator makes efforts not to be fooled by improving its ability to distinguish the machine-generated sentence from the human-generated one. This kind of adversarial training achieves a win-win situation when the generator and discriminator reach a Nash Equilibrium (Zhao et al., 2016; Arora et al., 2017; Guimaraes et al., 2017). Besides generating the desired distribution, we also want to directly guide the generator with a static and specific objective, such as generating sentences with high BLEU points. To this end, the smoothed sentence-level BLEU (Nakov et al., 2012) is utilized as the reinforced objective for the generator. During training, we employ both the dynamic discriminator and the static BLEU objective to evaluate the generated sentences and feedback the evaluations to guide the learning of the generator. In summary, we mainly make the following contributions:

- To the best of our knowledge, this work is among the first endeavors to introduce the generative adversarial training into NMT. We directly train the NMT model to generate sentences which are hard to be discriminated from human translations. The proposed model can be applied to any end-to-end NMT systems.

- We conduct extensive experiments on English-German and Chinese-English translation tasks and we test two different NMT models, the traditional RNNSearch (Bahdanau et al., 2014) and the state-of-the-art Transformer. Experimental results show that the proposed approach consistently achieves great success.

- Last but not least, we propose the smoothed sentence-level BLEU as the static and specific objective for the generator which biases the generation towards achieving high BLEU points. We show that the proposed approach is a weighted combination of the naive GAN and MRT.

2 Background and Related Work

2.1 RNNSearch and Transformer

The RNNSearch is the traditional NMT model which has been widely explored. We follow the de facto standard implementation by (Bahdanau et al., 2014). The encoder is a bidirectional gated recurrent units that encodes the input sequence $x = (x_1, \ldots, x_m)$ and calculates the forward sequence of hidden states $(\overrightarrow{h_1}, \ldots, \overrightarrow{h_m})$, and a backward sequence of hidden states $(\overleftarrow{h_1}, \ldots, \overleftarrow{h_m})$. The final annotation vector h_j is calculated by concatenating $\overrightarrow{h_j}$ and $\overleftarrow{h_j}$. The decoder is a recurrent neural network that predicts a target sequence $y = (y_1, \ldots, y_n)$. Each word y_i is predicted on a recurrent hidden state s_i, the previously predicted word y_{i-1} and a context vector c_i. The c_i is computed as a weighted sum of the encoded annotations h_j. The weight a_{ij} of each annotation h_j is computed by the attention mechanism, which models the alignment between y_i and x_j.

The Transformer, recently proposed by (Vaswani et al., 2017), achieves state-of-the-art results on both WMT2014 English-German and WMT2014 English-French translation tasks. The encoder of Transformer is composed of a stack of six identical layers. Each layer consists of a multi-head self-attention and a simple position-wise fully connected feed-forward network. The decoder is also composed of a stack of six identical layers. In addition to the two sub-layers

in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. The Transformer can be trained significantly faster than architectures based on recurrent or convolutional layers since it allows for significantly more parallelization.

2.2 Generative adversarial nets

Generative adversarial network, has enjoyed great success in computer vision and has been widely applied to image generation (Zhu et al., 2017; Radford et al., 2015). The conditional generative adversarial nets (Gauthier, 2014) apply an extension of generative adversarial network to a conditional setting, which enables the networks to condition on some arbitrary external data. Some recent works have begun to apply the generative adversarial training into the NLP area: (Chen et al., 2016) apply the idea of generative adversarial training to sentiment analysis and (Zhang et al., 2017) use the idea to domain adaptation tasks. For sequence generation problem, (Yu et al., 2016) leverage policy gradient reinforcement learning to back-propagate the reward from the discriminator, showing presentable results for poem generation, speech language generation and music generation. Similarly, (Zhang et al., 2016) generate the text from random noise via adversarial training. A striking difference from the works mentioned above is that, our work is in the conditional setting where the target-language sentence is generated conditioned on the source-language one. In parallel to our work, (Li et al., 2017) propose a similar conditional sequence generative adversarial training for dialogue generation. They use a hierarchical long-short term memory (LSTM) architecture for the discriminator. In contrast to their approach, we apply the CNN-based discriminator for the machine translation task. Furthermore, we propose to utilize the sentence-level BLEU as the specific objective for the generator. Detailed training strategies for the proposed model and extensive quantitative results are reported. We noticed that (Wu et al., 2017) is exploring the potential of GAN in NMT too.

3 The Approach

3.1 Model overview

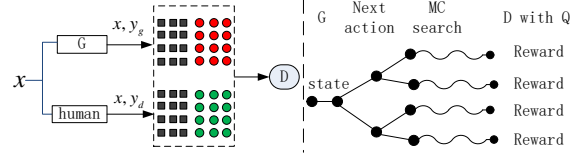In this section, we describe the architecture of the proposed BLEU reinforced conditional se-

Figure 1: The Illustration of the proposed BR-CSGAN. Left: D is trained over the real sentence pairs translated by the human and the generated sentence pairs by G. Note that D is a conditional discriminator. Right: G is trained by police gradient where the final reward is provided by D and Q.

quence generative adversarial net (referred to as BR-CSGAN) in detail. The sentence generation process is viewed as a sequence of actions that are taken according to a policy regulated by the generator. In this work, we take the policy gradient training strategies following (Yu et al., 2016). The whole architecture of the proposed model is depicted in figure 1. The model mainly consists of three sub modules:

Generator Based on the source-language sentences, the generator G aims to generate target-language sentences indistinguishable from human translations.

Discriminator The discriminator D, conditioned on the source-language sentences, tries to distinguish the machine-generated sentences from human translations. D can be viewed as a dynamic objective since it is updated synchronously with G.

BLEU objective The sentence-level BLEU Q serves as the reinforced objective, guiding the generation towards high BLEU points. Q is a static function which will not be updated during training.

3.2 Generator

Resembling NMT models, the generator G defines the policy that generates the target sentence y given the source sentence x. The generator takes exactly the same architecture with NMT models. Note that we do not assume the specific architecture of the generator. To verify the effectiveness of the proposed method, we take two different architectures for the generator, the RNNSearch [1] and Transformer [2].

---

[1]https://github.com/nyu-dl/dl4mt-tutorial
[2]https://github.com/tensorflow/tensor2tensor

3.3 Discriminator

Recently, the deep discriminative models such as the CNN and RNN have shown a high performance in complicated sequence classification tasks. Here, the discriminator is implemented based on the CNN architecture.

Since sentences generated by the generator have variable lengths, the CNN padding is used to transform the sentences to sequences with fixed length T, which is the maximum length set for the output of the generator. Given the source-language sequence $x_1, \ldots, x_T$ and target-language sequence $y_1, \ldots, y_T$, we build the source matrix $X_{1:T}$ and target matrix $Y_{1:T}$ respectively as:

$$X_{1:T} = x_1; x_2; \ldots; x_T \qquad (1)$$

and

$$Y_{1:T} = y_1; y_2; \ldots; y_T \qquad (2)$$

where $x_t, y_t \in R^k$ is the k-dimensional word embedding and the semicolon is the concatenation operator. For the source matrix $X_{1:T}$, a kernel $w_j \in R^{l \times k}$ applies a convolutional operation to a window size of l words to produce a series of feature maps:

$$c_{ji} = \rho(BN(w_j \otimes X_{i:i+l-1} + b)) \qquad (3)$$

where $\otimes$ operator is the summation of element-wise production and b is a bias term. ρ is a non-linear activation function which is implemented as ReLu in this paper. Note that the batch normalization (Ioffe and Szegedy, 2015) which accelerates the training significantly, is applied to the input of the activation function (BN in equation 3). To get the final feature with respect to kernel w_j, a max-over-time pooling operation is leveraged over the feature maps:

$$\widetilde{c}_j = max\{c_{j1}, \ldots, c_{jT-l+1}\} \qquad (4)$$

We use various numbers of kernels with different window sizes to extract different features, which are then concatenated to form the source-language sentence representation c_x. Identically, the target-language sentence representation c_y can be extracted from the target matrix $Y_{1:T}$. Finally, given the source-language sentence, the probability that the target-language sentence is being real can be computed as:

$$p = \sigma(V[c_x; c_y]) \qquad (5)$$

where V is the transform matrix which transforms the concatenation of c_x and c_y into a 2-dimension embedding and σ is the logistic function.

3.4 BLEU objective

We apply the smoothed sentence-level BLEU as the specific objective for the generator. Given the generated sentence y_g and the the ground true sentence y_d, the objective Q calculates a reward $Q(y_g, y_d)$, which measures the n-gram precisions of the generated sentence y_g. Identical to the output of the discriminator, the $Q(y_g, y_d)$ also ranges from zero to one, which makes it easier to fuse Q and D.

3.5 Policy Gradient Training

Following (Yu et al., 2016), the objective of the generator G is defined as to generate a sequence from the start state to maximize its expected end reward. Formally, the objective function is computed as:

$$J(\theta) = \sum_{Y_{1:T}} G_\theta(Y_{1:T}|X) \cdot R_{D,Q}^{G_\theta}(Y_{1:T-1}, X, y_T, Y^*)$$

where θ represents the parameters in G, $Y_{1:T} = y_1, \ldots, y_T$ indicates the generated target sequence, X is the source-language sentence, Y^* represents the ground true target sentence. $R_{D,Q}^{G_\theta}$ is the action-value function of a target-language sentence given the source sentence X, i.e. the expected accumulative reward starting from the state $(Y_{1:T-1}, X)$, taking action y_T, and following the policy G_θ. To estimate the action-value function, we consider the estimated probability of being real by the discriminator D and the output of the BLEU objective Q as the reward:

$$R_{D,Q}^{G_\theta}(Y_{1:T-1}, X, y_T, Y^*) =$$
$$\lambda(D(X, Y_{1:T}) - b(X, Y_{1:T})) + (1 - \lambda)Q(Y_{1:T}, Y^*)$$

where b(X,Y) denotes the baseline value to reduce the variance of the reward. Practically, we take b(X,Y) as a constant, 0.5 for simplicity. And the λ is a hyper-parameter. The question is that, given the source sequence, D only provides a reward value for a finished target sequence. If $Y_{1:T}$ is not a finished target sequence, the value of $D(X, Y_{1:T})$ makes no sense. Therefore, we cannot get the action-value for an intermediate state directly. To evaluate the action-value for an intermediate state, the Monte Carlo search under the policy of G_θ is applied to sample the unknown tokens. Each search ends until the end of sentence token is sampled or the sampled sentence reaches the maximum length. To obtain more stable reward and reduce the variance, we represent an N-time Monte

Carlo search as:

$$\{Y^1_{1:T_1}, \ldots, Y^N_{1:T_N}\} = MC^{G_\theta}((Y_{1:t}, X), N)$$

where T_i represents the length of the sentence sampled by the i'th Monte Carlo search. $(Y_{1:t}, X) = (y_1, \ldots, y_t, X)$ is the current state and $Y^N_{t+1:T_N}$ is sampled based on the policy G_θ. The discriminator provides N rewards for the sampled N sentences respectively. The final reward for the intermediate state is calculated as the average of the N rewards. Hence, for the target sentence with the length T, we compute the reward for y_t in the sentence level as:

$$R^{G_\theta}_{D,Q}(Y_{1:t-1}, X, y_T, Y^*) =$$
$$\begin{cases} \frac{1}{N}\sum_{n=1}^{N} \lambda(D(X, Y^n_{1:T_n}) - b(X, Y^n_{1:T_n})) + (1-\lambda)Q(Y_{1:T_n}, Y*) & t < T \\ \lambda(D(X, Y_{1:t}) - b(X, Y_{1:t})) + (1-\lambda)Q(Y_{1:t}, Y^*) & t = T \end{cases}$$

Using the discriminator as a reward function can further improve the generator iteratively by dynamically updating the discriminator. Once we get more realistic generated sequences, we re-train the discriminator as:

$$\min -\mathbb{E}_{X,Y \in P_{data}}[\log D(X, Y)] - \mathbb{E}_{X,Y \in G}[\log(1 - D(X, Y))]$$

After updating the discriminator, we are ready to re-train the generator. The gradient of the objective function $J(\theta)$ w.r.t the generator's parameter θ is calculated as:

$$\nabla J(\theta) = \frac{1}{T}\sum_{t=1}^{T}\sum_{y_t} R^{G_\theta}_{D,Q}(Y_{1:t-1}, X, y_T, Y^*) \cdot \nabla_\theta(G_\theta(y_t|Y_{1:t-1}, X))$$
$$= \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{y_t \in G_\theta}[R^{G_\theta}_{D,Q}(Y_{1:t-1}, X, y_T, Y^*) \cdot \nabla_\theta \log p(y_t|Y_{1:t-1}, X)]$$

3.6 Training strategies

GANs are widely criticized for its unstable training since the generator and discriminator need to be carefully synchronized. To make this work easier to reproduce, this paper gives detailed strategies for training the proposed model.

Firstly, we use the maximum likelihood estimation to pre-train the generator on the parallel training set until the best translation performance is achieved. Then, generate the machine-generated sentences by using the generator to decode the training data. We simply use the greedy sampling method instead of the beam search method for decoding. Next, pre-train the discriminator on the combination of the true parallel data and the machine-generated data until the classification accuracy achieves at ξ. Finally, we jointly train

the generator and discriminator. The generator is trained with the policy gradient training method. However, in our practice, we find that updating the generator only with the simple policy gradient training leads to unstableness. To alleviate this issue, we adopt the teacher forcing approach which is similar to (Lamb et al., 2016; Li et al., 2017). We directly make the discriminator to automatically assign a reward of 1 to the golden target-language sentence and the generator uses this reward to update itself on the true parallel example. We run the teacher forcing training for one time once the generator is updated by the policy gradient training. After the generator gets updated, we use the new stronger generator to generate η more realistic sentences, which are then used to train the discriminator. Following (Arjovsky et al., 2017), we clamp the weights of the discriminator to a fixed box ($[-\epsilon, \epsilon]$) after each gradient update. We perform one optimization step for the discriminator for each step of the generator. In our practice, we set ξ as 0.82, η as 5000, ϵ as 1.0 and the N for Monte Carlo search as 20.

4 Experiments and Results

We evaluate our BR-CSGAN on English-German and Chinese-English translation tasks and we test two different architectures for the generator, the traditional RNNSearch and the newly emerged state-of-the-art Transformer.

4.1 Data sets and preprocessing

English-German: For English-German translation, we conduct our experiments on the publicly available corpora used extensively as benchmark for NMT systems, WMT'14 En-De. This data set contains 4.5M sentence pairs [3]. Sentences are encoded using byte-pair encoding (Sennrich et al., 2015), which has a shared source-target vocabulary of about 37000 tokens. We report results on newstest2014. The newstest2013 is used as validation.

Chinese-English: For Chinese-English translation, our training data consists of 1.6M sentence pairs randomly extracted from LDC corpora [4]. Both the source and target sentences are encoded with byte-pair encoding and the tokens in the source and target vocabulary is about 38000 and

---

[3] http://nlp.stanford.edu/projects/nmt
[4] LDC2002L27, LDC2002T01, LDC2002E18, LDC2003E07, LDC2004T08, LDC2004E12, LDC2005T10

Model	Chinese-English				English-German newstest2014
	NIST03	NIST04	NIST05	average	
RNNSearch (Bahdanau et al., 2014)	33.93	35.67	32.24	33.94	21.2
Transformer (Vaswani et al., 2017)	42.23	42.17	41.02	41.80	27.30
RNNSearch+BR-CSGAN($\lambda = 1.0$)	35.21	36.51	33.45	35.05	22.1
RNNSearch+BR-CSGAN($\lambda = 0.7$)	35.97	37.32	34.03	**35.77**	**22.89**
RNNSearch+BR-CSGAN($\lambda = 0$)	34.57	35.93	33.07	34.52	21.75
Transformer+BR-CSGAN($\lambda = 1.0$)	42.67	42.79	41.54	42.30	27.75
Transformer+BR-CSGAN($\lambda = 0.8$)	43.01	42.96	41.86	**42.61**	**27.92**
Transformer+BR-CSGAN($\lambda = 0$)	42.41	42.74	41.29	42.14	27.49

Table 1: BLEU score on Chinese-English and English-German translation tasks. The hyper-parameter λ is selected according to the development set. For the Transformer, following (Vaswani et al., 2017), we report the result of a single model obtained by averaging the 5 checkpoints around the best model selected on the development set.

34000 respectively [5]. We choose the NIST02 as the development set. For testing, we use NIST03, NIST04 and NIST05 data sets.

To speed up the training procedure, sentences of length over 50 words are removed when we conduct experiments on the RNNSearch model. This is widely used by previous works (Ranzato et al., 2015; Shen et al., 2015; Yang et al., 2016).

4.2 Model parameters and evaluation

For the Transformer, following the base model in (Vaswani et al., 2017), we set the dimension of word embedding as 512, dropout rate as 0.1 and the head number as 8. The encoder and decoder both have a stack of 6 layers. We use beam search with a beam size of 4 and length penalty $\alpha = 0.6$. For the RNNSearch, following (Bahdanau et al., 2014), We set the hidden units for both encoders and decoders as 512. The dimension of the word embedding is also set as 512. We do not apply dropout for training the RNNSearch. During testing, we use beam search with a beam size of 10 and length penalty is not applied.

All models are implemented in TensorFlow (Abadi et al., 2015) and trained on up to four K80 GPUs synchronously in a multi-GPU setup on a single machine [6]. We stop training when the model achieves no improvement for the tenth evaluation on the development set. BLEU (Papineni et al., 2002) is utilized as the evaluation metric. We apply the script *mteval-v11b.pl* to evaluate the Chinese-English translation and utilize the script

multi-belu.pl for English-German translation [7].

4.3 Main results

The model of RNNSearch is optimized with the mini-batch of 64 examples. It takes about 30 hours to pre-train the RNNSearch on the Chinese-English data set and 46 hours on the English-German data set. During generative adversarial training, it takes about 35 hours on the Chinese-English data set and about 50 hours on the English-German data set. For the Transformer, each training batch contains a set of sentence pairs containing approximately 25000 source tokens and 25000 target tokens. On the Chinese-English data set, it takes about 15 hours to do pre-training and 20 hours to do generative adversarial training. On the English-German data set, it takes about 35 hours for the pre-training and 40 hours for the generative adversarial training.

Table 1 shows the BLEU score on Chinese-English and English-German test sets. On the RNNSearch model, the naive GAN (i.e., the line of RNNSearch+BR-CSGAN (λ=1) in table 1) achieves improvement up to +1.11 BLEU points averagely on Chinese-English test sets and +0.9 BLEU points on English-German test set. Armed with the BLEU objective, the BR-CSGAN (the line of RNNSearch+BR-CSGAN (λ=0.7)) leads to more significant improvements, +1.83 BLEU points averagely on Chinese-English translation and +1.69 BLEU points on English-German translation. We also test the translation performance when the RNNSearch is only guided by the static

---

[5]When doing BPE for Chinese, we need to do word segmentation first and the following steps are the same with BPE for English.

[6]We intend to share the code we used to train and evaluate our models available soon

[7]https://github.com/moses-smt/mosesdecoder/blob/617e8c8/scripts/generic/multi-bleu.perl;mteval-v11b.pl

BLEU objective (the line of RNNSearch+BR-CSGAN (λ=0)), and we only get +0.58 BLEU points improvement on Chinese-English translation and +0.55 BLEU points improvement on English-German. Experiments on the Transformer show the same trends. While the Transformer has achieved state-of-the-art translation performances, our approach still achieves +0.81 BLEU points improvement on Chinese-English translation and +0.62 BLEU points improvement on English-German.

These results indicate that the proposed BR-CSGAN consistently outperforms the baselines and it shows better translation performance than the naive GAN and the model guided only by the BLEU objective.

5 Analysis

5.1 Compared with MRT

We show that MRT (Shen et al., 2015) is an extreme case of our approach. Considering a sentence pair (x, y), the training objective of MRT is calculated as

$$\widehat{J}(\theta') = \sum_{y^s \in S(x)} p(y^s|x; \theta') \Delta(y^s, y)$$

where $\Delta(y^s, y)$ is a loss function (i.e., the sentence-level BLEU used in this paper) that measures the discrepancy between a predicted translation y^s and the training example y, $S(x)$ represents the set which contains all of the predictions given the input x, and θ' is the parameters of the NMT model. Unfortunately, this objective is usually intractable due to the exponential search space. To alleviate this problem, a subset of the search space is sampled to approximate this objective. In this paper, when we set λ as zero, the objective for the proposed BR-CSGAN comes to

$$J(\theta)_{\lambda=0} = \sum_{Y_{1:T}} G_\theta(Y_{1:T}|X) \cdot Q(Y_{1:T}, Y^*)$$

where the $Q(Y_{1:T}, Y^*)$ is also a loss function between the predicted translation $Y_{1:T}$ and the training example Y^*. It is easy to be found that, under this condition (i.e., λ set as zero), the proposed BR-CSGAN optimizes almost the same objective with MRT. The only difference is that the reinforcement learning procedure is utilized in BR-CSGAN to maximize the total reward and MRT instead applies random sampling to approximate

the risk. Actually, the BR-CSGAN is a weighted sum of the naive GAN (λ=1) and MRT (λ=0), and it incorporates the advantages of the two approaches. Specifically, compared to naive GAN which is trained without specific objective guidance, BR-CSGAN utilizes the BLEU objective to guide the generator to generate sentences with higher BLEU points. And compared to MRT which is trained only with the static objective, the BR-CSGAN applies a dynamic discriminator which updates synchronously with the generator, to feedback the dynamic rewards for the generator. Table 2 compares the translation performance between the MRT and BR-CSGAN on Chinese-English and English-German translation tasks. We only conduct experiments on the RNNSearch because we only get the open-source implementation of MRT on the RNNSearch [8]. Results show that the proposed BR-CSGAN consistently outperforms the MRT on the Chinese-English and English-German translations.

Model	Chinese-English average	English-German newstest2014
RNNSearch	33.94	21.2
MRT (Shen et al., 2015)	34.64	21.6
BR-CSGAN($\lambda = 0.7$)	35.77	22.89

Table 2: BLEU score on Chinese-English and English-German translation tasks for MRT and BR-CSGAN.

5.2 When to stop pre-training

The initial accuracy ξ of the discriminator which is viewed as a hyper-parameter, can be set carefully during the process of pre-training. A natural question is that when shall we end the pre-training. Do we need to pre-train the discriminator with the highest accuracy? To answer this question, we test the impact of the initial accuracy of the discriminator. We pre-train five discriminators which have the accuracy as 0.6, 0.7, 0.8, 0.9 and 0.95 respectively. With the five discriminators, we train five different BR-CSGAN models (with the generator as RNNSearch and λ set as 0.7) and test their translation performances on the development set at regular intervals. Figure 2 reports the results and we can find that the initial accuracy of the discriminator shows great impacts on the translation performance of the proposed BR-CSGAN. From figure 2, we show that the initial accuracy of the

---

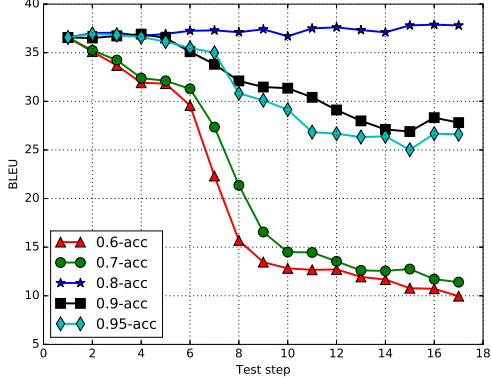[8] The open-source implementation can be found at: https://github.com/EdinburghNLP/nematus

Figure 2: BLEU score on the development set for the BR-CSGAN where the discriminators have different initial accuracy. "0.6-acc" means the initial accuracy is 0.6. We report the results on the Chinese-English translation tasks. RNNSearch is taken as the generator.

discriminator needs to be set carefully and either it is too high (0.9 and 0.95) or too low (0.6 and 0.7), the model performs badly [9]. This suggests that it is important for the generator and discriminator to keep a balanced relationship at the beginning of the generative adversarial training. If the discriminator is too strong, the generator is always penalized for its bad predictions and gets no idea about right predictions. Hence, the generator is discouraged all the time and the performance gets worse and worse. On the other hand, if the discriminator is too weak, the discriminator is unable to give right guidance for the generator, i.e. the gradient direction for updating the generator is random. Empirically, we pre-train the discriminator until its accuracy reaches around 0.8.

5.3 Sample times for Monto Carol search

We are also curious about how the sample times N for Monte Carlo search affects the translation performance. Intuitively, if N is set as a small number, the intermediate reward for each word may be incorrect since there is a large variance for the Monto Carol search when the sample time is too small. And if otherwise, the computation shall be very time consuming because we need to do much more sampling. Therefore, there is a trade-off between the accuracy and computation complexity here. We investigate this problem on the Chinese-English translation task. Table

---

[9]To make the illustration simple and clear, we only depict the results when the RNNSearch acts as the generator.

N	NIST02	NIST03	NIST04	NIST05
0	36.87	33.93	35.67	32.24
5	-	-	-	-
10	-	-	-	-
15	37.34	34.91	36.09	33.45
20	38.58	35.97	37.32	34.03
25	38.65	36.04	37.52	33.91
30	38.74	36.01	37.54	33.76

Table 3: The translation performance of the BR-CSGAN with different N for Monte Carlo search. "-" means that the proposed model shows no improvement than the pre-trained generator or it can not be trained stably. With N set as 0, it is referred to as the pre-trained generator. Similarly, we only report results on the RNNSearch and λ is set as 0.7.

3 presents the translation performance of the BR-CSGAN on the test sets when the N are set from 5 to 30 with interval 5. From table 3, the proposed model achieves no improvement than the baseline (i.e., the pre-trained generator) when N are set less than 15 and the BLEU scores are not reported on the table. As a matter of fact, the translation performance of the model gets worse and worse. We conjecture that the approximated reward is far from the expected reward due to the large variance when N is set too small, and gives wrong gradient directions for model updating. Since the training for GAN is not stable, the wrong gradient direction exacerbates the unstableness and results in the BLEU getting worse and worse. With the increasing of N, the translation performance of the model gets improved. However, with N set larger than 20, we get little improvement than the model with N set as 20 and the training time exceeds our expectation.

6 Conclusion and Future Work

In this work, we propose the BR-CSGAN which leverages the BLEU reinforced generative adversarial net to improve the NMT. We show that the proposed approach is a weighted combination of the naive GAN and MRT. To verify the effectiveness of our approach, we test two different architectures for the generator, the traditional RNNSearch and the state-of-the-art Transformer. Extensive experiments on Chinese-English and English-German translation tasks show that our approach consistently achieves significant improvements. In the future, we would like to try multi-adversarial framework which consists of

multi discriminators and generators for GAN.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems .

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875* .

Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. 2017. Generalization and equilibrium in generative adversarial nets. *ICML* .

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Chris Callison-Burch and Miles Osborne. 2006. Re-evaluating the role of bleu in machine translation research .

Niladri Chatterjee, Anish Johnson, and Madhav Krishna. 2007. Some improvements over the bleu metric for measuring translation quality for hindi. In *Computing: Theory and Applications, 2007. ICCTA'07. International Conference on*. IEEE, pages 485–490.

Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie, and Kilian Weinberger. 2016. Adversarial deep averaging networks for cross-lingual sentiment classification. *arXiv preprint arXiv:1606.01614* .

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* .

Emily L Denton, Soumith Chintala, Rob Fergus, et al. 2015. Deep generative image models using a? laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*. pages 1486–1494.

Jon Gauthier. 2014. Conditional generative adversarial nets for convolutional face generation. *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester* 2014(5):2.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning .

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

Gabriel Lima Guimaraes, Benjamin Sanchez-Lengeling, Pedro Luis Cunha Farias, and Alán Aspuru-Guzik. 2017. Objective-reinforced generative adversarial networks (organ) for sequence generation models. *arXiv preprint arXiv:1705.10843* .

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* .

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2016. Google's multilingual neural machine translation system: Enabling zero-shot translation. *arXiv preprint arXiv:1611.04558* .

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. *EMNLP* pages 1700–1709.

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

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

Pauline Luc, Camille Couprie, Soumith Chintala, and Jakob Verbeek. 2016. Semantic segmentation using adversarial networks. *arXiv preprint arXiv:1611.08408* .

Preslav Nakov, Francisco Guzman, and Stephan Vogel. 2012. Optimizing for sentence-level bleu+ 1 yields short translations. In *COLING*. volume 12, pages 1979–1994.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. *Association for Computational Linguistics* pages 311–318.

Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* .

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *Computer Science* .

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2015. Minimum risk training for neural machine translation. *arXiv preprint arXiv:1512.02433* .

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need .

Lijun Wu, Yingce Xia, Li Zhao, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2017. Adversarial neural machine translation. *arXiv preprint arXiv:1704.06933* .

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* .

Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. 2016. A character-aware encoder for neural machine translation. In *COLING*.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2016. Seqgan: Sequence generative adversarial nets with policy gradient. *The Association for the Advancement of Artificial Intelligence 2017* .

Yizhe Zhang, Zhe Gan, and Lawrence Carin. 2016. Generating text via adversarial training. *NIPS* .

Yuan Zhang, Regina Barzilay, and Tommi Jaakkola. 2017. Aspect-augmented adversarial networks for domain adaptation. *arXiv preprint arXiv:1701.00188* .

Junbo Zhao, Michael Mathieu, and Yann LeCun. 2016. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126* .

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