Teacher Forcing Recovers Reward Functions for Text Generation

Yongchang Hao[†], **Yuxin Liu**[†], **Lili Mou**[†‡]
[†]Dept. Computing Science, Alberta Machine Intelligence Institute (Amii)
University of Alberta, Canada
[‡]Canada CIFAR AI Chair, Amii
{yongcha1,yliu17}@ualberta.ca, doublepower.mou@gmail.com

Abstract

Reinforcement learning (RL) has been widely used in text generation to alleviate the *exposure bias* issue or to utilize non-parallel datasets. The reward function plays an important role in making RL training successful. However, previous reward functions are typically task-specific and sparse, restricting the use of RL. In our work, we propose a task-agnostic approach that derives a step-wise reward function directly from a model trained with teacher forcing. We additionally propose a simple modification to stabilize the RL training on non-parallel datasets with our induced reward function. Empirical results show that our method outperforms self-training and reward regression methods on several text generation tasks, confirming the effectiveness of our reward function.[1]

1 Introduction

Teacher forcing [7] is the common training method for text generation models. Although this practice has been widely applied [7, 13, 58], there are two main issues: 1) Teacher-forcing training is data-hungry because parallel datasets are usually expensive to obtain. On the other hand, there are numerous unlabeled, non-parallel datasets available. This poses an urge to efficiently exploit non-parallel data. 2) Teacher forcing introduces a discrepancy between training and inference because the model learns to predict the next word based on the partial groundtruth reference during training, whereas in inference the model predicts the next word based on its self-generated previous words. This undesired discrepancy is known as *exposure bias* [44, 6, 28, 59].

To address the first problem, a straightforward method is to generate pseudo-parallel sentences for data augmentation, such as self-training [2], sequence-level knowledge distillation [23], and back-translation [49]. However, the exposure bias remains in such cases.

To address the second problem, the model should be trained on self-generated sentences. Common solutions are often based on reinforcement learning (RL). In text generation, however, there does not exist a naturally defined reward function for RL. Researchers have proposed various heuristic scores as the reward, such as BLEU [38] for translation and ROUGE [31] for summarization. These reward functions are task-specific, restricting the application of RL in other tasks. Further, these rewards require parallel data, failing to address the first problem above; they are typically sparse (only non-zero at the end of a sentence), making RL training difficult.

The goal of this paper is to address these two problems in one framework with a learned, dense reward function. Our approach has two steps: we first train a sequence-to-sequence (seq2seq) model on the parallel dataset and induce a reward function from the model. Then, we apply RL on non-parallel data based on our induced reward function.

[1]Our code is publicly available at `https://github.com/MANGA-UOFA/LMReward`.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Our method is task-agnostic and does not require handcrafted engineering or heuristics. Further, our reward function provides dense (step-wise) training signals, which makes RL training much easier than sparse rewards. Additionally, the reward function derived from the seq2seq model does not directly participate in the generation, which allows the model to explore based on its own prediction and thus alleviates the exposure bias.

We conduct experiments on dialogue generation and paraphrase generation. The empirical results suggest that our method leads to better performance compared with several baselines, including self-training and task-specific heuristic reward learning, on both datasets. This confirms the effectiveness and generality of our framework.

2 Approach

Our approach trains the seq2seq model on non-parallel data with reinforcement learning, whose foundation is the Markov decision process (MDP). In this section, we first introduce the MDP formulation for text generation. Then we describe our method to derive the reward function from a seq2seq model trained by teacher forcing. Finally, we describe the policy gradient method for RL training with our induced reward function.

2.1 Reinforcement Learning Formulation of Text Generation

Text Generation as a Markov Decision Process (MDP). We formulate the text generation process as an (undiscounted) MDP, which can be represented as a tuple $(\mathcal{S}, \mathcal{A}, T, r)$. At every step, a decision $a \in \mathcal{A}$ is made based on its state $s \in \mathcal{S}$. The transition dynamic $T(s'|s,a)$ is the probability of the next state being s', given the current state s and the action a. A function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defines the reward based on a state and an action.

Typically, the decision making is assisted by a policy π, which is a predicted distribution over actions and is trained to maximize the expected total reward, also known as an action value function:

$$q^{\pi}(s,a) := \mathop{\mathbb{E}}_{\substack{a_t \sim \pi(\cdot|s_t) \\ s_{t+1} \sim T(\cdot|s_t, a_t)}} \left[\sum_{t=1}^{H} r(s_t, a_t) | s_1 = s, a_1 = a \right] \tag{1}$$

where H is the number of steps. Theoretical results show that the optimal policy π^* satisfies the Bellman optimality equation:

$$q^{\pi^*}(s,a) = r(s,a) + \sum_{s' \in \mathcal{S}} T(s'|s,a) \max_{a'} q^{\pi^*}(s',a') \tag{2}$$

For text generation, the MDP state can be defined as the partial generated sequence $\boldsymbol{y}_{<t} := (y_1, \cdots, y_{t-1})$, and the action as the next token y_t in the vocabulary $\mathcal{V}$. The transition dynamic $T(\cdot|s,a)$ here is deterministic, since every state–action pair $(\boldsymbol{y}_{<t}, y_t)$ leads to a unique state $\boldsymbol{y}_{<t+1}$ for the next step.

In previous RL-based text generation, there lacks a naturally defined reward function $r(s,a)$. While researchers have applied various heuristics as the reward [1, 50], they suffer from several shortcomings (e.g., sparsity and task specificity) as mentioned in Section 1. To address these problems, we propose to induce a reward function for text generation tasks in a principled approach by inverse reinforcement learning.

Inverse Reinforcement Learning (IRL). The goal of IRL is to learn a reward function $r(s,a)$. Especially, we wish the resulting action value function q computed by Eqn. (1) could satisfy $q(s,a) \geq q(s,a')$ for every $a' \in \mathcal{A}$ and every (s,a) pair in the training set $\mathcal{D}$. In other words, the decisions in $\mathcal{D}$ are made greedily by $\operatorname{argmax}_a q(s,a)$ given any state s. Unfortunately, Ng and Russell [36] show that this is an ill-posed problem since the desirable reward function r is not unique. Therefore, we follow a common assumption [4, 43, 69] that resolves the ambiguities:

Assumption 1. Given an action value function q, the policy π takes the form of $\pi_q(a|s) := \exp(q(s,a)) / \sum_{a'} \exp(q(s,a'))$.

In traditional IRL [4, 43, 69], reward learning is difficult and this assumption does not directly yield a reward function due to the stochastic state transition $T(s'|s,a)$. However, our insight is that the transition is deterministic for text generation tasks, and thus we may utilize Assumption 1 to induce an action value function q, and then a reward function r, from some learned policy π, as explained in the next part.

2.2 Teacher Forcing Recovers IRL

One of our main contributions is that we show the seemingly complicated reward learning in Section 2.1 can be recovered by teacher forcing, the *de facto* common practice of supervised text generation. Our discovery leads to a convenient approach that derives a step-wise reward function simply from general seq2seq models, without the need for task-specific heuristics. This makes RL more general for text generation, and our step-wise reward largely simplifies RL training.

Maximum Likelihood Estimation (MLE) for IRL. Following Assumption 1, we let the policy $\pi_{q_\omega}(\cdot|s) \propto \exp(q_\omega(s,\cdot))$, where q_ω is a parameterized action value function. Under such a policy, the probability of each trajectory $\tau := ((s_1,a_1),\ldots,(s_H,a_H))$ in the dataset is given by the trajectory distribution P^π. The likelihood of the dataset is given by

$$P_{\text{IRL}}(\mathcal{D}|\omega) := \prod_{\tau\in\mathcal{D}} P^{\pi_{q_\omega}}(\tau) \tag{3}$$

Teacher-Forcing Training. For text generation, the standard teacher-forcing seq2seq training is to minimize the loss:

$$L_{\text{TF}}(\omega;\mathcal{D}) := -\sum_{\boldsymbol{y}\in\mathcal{D}}\sum_{t=1}^{|\boldsymbol{y}|} \log p_\omega(y_t|\boldsymbol{y}_{<t}) \tag{4}$$

where the predicted probability of the next token being v is $p_\omega(v|\boldsymbol{y}_{<t}) = \frac{\exp(f_\omega(\boldsymbol{y}_{<t},v))}{\sum_{v'\in\mathcal{V}}\exp(f_\omega(\boldsymbol{y}_{<t},v'))}$ for the logit function f_ω with parameters ω. In seq2seq training, an additional input $\boldsymbol{x}$ may be added to the conditional probabilities but is omitted here for simplicity.

The below theorem shows their equivalence up to an additional constant.

Theorem 1. *Suppose the value function q in Eqn.* (3) *and the seq2seq model f in Eqn.* (4) *have the same parametrization ω, we have*

$$L_{\text{TF}}(\omega;D) = -\log P_{\text{IRL}}(\mathcal{D}|\omega) + \text{const} \tag{5}$$

Proof. For the MLE of IRL under Assumption 1, the Ionescu–Tulcea theorem [22] asserts that there exists a unique trajectory distribution P_μ^π satisfying

$$\begin{aligned}
&P_\mu^\pi(s_1) = \mu(s_1)\\
&P_\mu^\pi(s_1,a_1,\ldots,s_t,a_t) = P_\mu^\pi(s_1,a_1,\ldots,s_t)\pi(a_t|s_t)\\
&P_\mu^\pi(s_1,a_1,\ldots,s_t,a_t,s_{t+1}) = P_\mu^\pi(s_1,a_1,\ldots,s_t,a_t)T(s_{t+1}|s_t,a_t)
\end{aligned}$$

for any $t \geq 1$, given the initial state distribution μ, transition probability T, and policy π.

The likelihood can thus be factorized by the multiplication of μ, T, and π:

$$P_{\text{IRL}}(\mathcal{D}|\omega) = \prod_{\tau\in\mathcal{D}} P_\mu^{\pi_{q_\omega}}(\tau) = \prod_{\tau\in\mathcal{D}}\left[\mu(s_1)\pi_{q_\omega}(a_1|s_1)\prod_{t=2}^{|\tau|} T(s_t|s_{t-1},a_{t-1})\pi_{q_\omega}(a_t|s_t)\right]$$

As mentioned, text generation has a deterministic transition, i.e., $T(s'|s,a) = 1$ for the next state $s' = s + [a]$. Taking the μ terms out, we have

$$-\log P_{\text{IRL}}(\mathcal{D}|\omega) = -\log\prod_{\tau\in\mathcal{D}}\prod_{t=1}^{|\tau|}\pi_{q_\omega}(a_t|s_t) - \log\prod_{\tau\in\mathcal{D}}\mu(s_1) \tag{6}$$

where the second term is a constant in terms of ω. In Section 2.1, text generation is modeled as an MDP with $s_t = \boldsymbol{y}_{<t}$ and $a_t = y_t$. Therefore, the first term of Eqn. (6) is the same as Eqn. (4) under the parametrization $\pi_{q_\omega} = p_\omega$, concluding the equivalence between MLE for IRL and the teacher-forcing training of a seq2seq model. □

Inducing the Reward Function. Theorem 1 shows that seq2seq training with teacher forcing actually learns an IRL model. Thus, we may derive a reward function as

$$r(s,a) = q_\omega(s,a) - \sum_{s' \in \mathcal{S}} T(s'|s,a) \max_{a' \in \mathcal{A}} q_\omega(s',a') = f_\omega(s,a) - \max_{a' \in \mathcal{A}} f_\omega(s+[a],a') \tag{7}$$

where the first equality is due to the Bellman optimality condition (2); the second equality is due to the parametrization of $q_\omega = f_\omega$ and the deterministic transition $T(s'|s,a) = 1$ for s' as the concatenation of the prefix s and token a.

Our use of the Bellman optimality condition is different from classic RL, where the reward is well-defined and the action value function is thus learned [55]. Instead, we induce the underlying reward assuming the action value function is known (given by Assumption 1). The following diagram shows the whole process of our derivation.

$$\mathcal{D} \xrightarrow{\text{Teacher Forcing}} \pi \xrightarrow{\text{Assumption 1}} q \xrightarrow{\text{Eqn. (7)}} r$$

In real-world applications, the learned action value function might be imperfect, and in this case, we may bound the error of our induced reward with the following theorem.

Theorem 2. *Let r^* be an underlying true reward function and q^* be the corresponding optimal value function. Given an approximate value function q, we denote by r the reward function derived from Eqn.* (7)*. Then, we must have $\|r - r^*\|_\infty$ bounded by $O(\|q - q^*\|_\infty)$. Here, $\|\cdot\|_\infty$ takes the maximum absolute value over all $s \in \mathcal{S}$ and $a \in \mathcal{A}$.*

Proof. See Appendix A. □

2.3 Periodically Synchronized Behavior Policy in Policy Gradient

In text generation, a neural network can be viewed as a policy π that predicts the word distribution given the state of a decoding step. The reward induced from Section 2.2 can be used to improve the policy through RL. To stabilize training, we propose a variant of off-policy policy gradient methods [10] with a periodically synchronized behavior policy.

Our RL training adopts the off-policy REINFORCE [63] as the backbone of our algorithm. Let π_φ be the model policy (i.e., the model's prediction) to be optimized, and π_b be the behavior policy (i.e., the sampling distribution during training). Through importance sampling, the gradient of the expected total reward with respect to φ can be obtained by the off-policy policy gradient theorem [10]

$$\nabla_\varphi \mathop{\mathbb{E}}_{\pi_\varphi}\left[\sum_t r(s_t,a_t)\right] = \mathop{\mathbb{E}}_{\pi_b}\left[\sum_t \rho_t \hat{q}_r(s_t,a_t) \nabla_\varphi \log \pi_\varphi(a_t|s_t)\right] \tag{8}$$

where $\rho_t := \pi_\varphi(a_t|s_t)/\pi_b(a_t|s_t)$ is the importance weight, and $\hat{q}_r(s_t,a_t) := \sum_{i \geq t} r(s_i,a_i)$ is the total reward of the trajectory. In practice, off-policy REINFORCE ($\pi_\varphi \neq \pi_b$) is more exploratory than the on-policy one ($\pi_\varphi = \pi_b$), since the model policy π_φ would become more concentrated during optimization and does not explore much, whereas π_b is typically chosen to cover more trajectories. However, Degris et al. [10] adopt a fixed behavior policy π_b, which does not exploit along with the current model policy π_φ. Trajectories sampled from the fixed behavior policy are unlikely to occur under the current model policy. This results in a lack of exploitation, and thus might lead to less informative training.

To balance exploration and exploitation, we would like the behavior policy to be close to the model policy but stay exploratory at the same time. We thus propose a periodically updating schedule, where the behavior policy is frozen for a long period to preserve exploration but keeps track of the current model policy to enhance exploitation. Particularly, we synchronize the behavior policy with the model policy for every $k = 5000$ gradient updates of the latter. Our remedy is a simple method to overcome the instability of REINFORCE. It shares a common ground with a number of policy gradient methods like the proximal policy optimization (PPO) [48], especially in that both methods involve multiple updates with a fixed behavior policy. As the main contribution of this paper is reward induction, we resort to this simple fix and leave the mathematical connection as an interesting future direction.

Algorithm 1 summarizes our approach. In the implementation, we are able to execute the loops in parallel, which speeds up the training process. Our periodically synchronized behavior policy further enables us to parallelize sampling and model updates to reduce the awaiting time.

2.4 Application to Semi-Supervised Learning

Our approach naturally aligns with the paradigm of semi-supervised learning, as it involves training a seq2seq model to induce the reward function, which requires (at least a small volume of) parallel data $\mathcal{D}_p$. Additionally, we assume there is a non-parallel dataset $\mathcal{D}_u$ containing input sentences only for RL training with the induced reward.

Our semi-supervised approach consists of two stages. We first train a seq2seq model f_ω on the parallel dataset $\mathcal{D}_p$ to induce the reward function r by Eqn. (7). The procedure is described in Section 2.2. The reward function then facilitates RL training on the non-parallel dataset $\mathcal{D}_u$, which is shown in Algorithm 1.

Algorithm 1: Our Algorithm

Input: A non-parallel dataset $\mathcal{D}_u$, learned logit (value) function f_ω, policy π_φ with the initial parameter φ, total update steps U, and synchronizing period k
Output: A policy π_φ parameterized by φ
begin
  for $i \leftarrow 1...U$ **do**
    if $i \equiv 0 \pmod{k}$ **then**
      $\pi_b \leftarrow \pi_\varphi$; ▷ `Behavior policy update`
    Sample a source sentence $\boldsymbol{x} \in \mathcal{D}_u$
    Construct the initial state $s \leftarrow (\boldsymbol{x}, [\text{BOS}])$; ▷ `[BOS] is the beginning token`
    Sample a trajectory τ from the behavior policy π_b
    $\hat{q}^r_{h+1} \leftarrow 0$ and $g \leftarrow \mathbf{0}$
    for $t \leftarrow |\tau|...1$ **do**
      if $t = |\tau|$ **then**
        $r_t \leftarrow f_\omega(s_t, a_t)$; ▷ `Termination step, no` s_{t+1}
      else
        $r_t \leftarrow f_\omega(s_t, a_t) - \max_{a'} f_\omega(s_{t+1}, a')$; ▷ `By Eqn. (7)`
      $\hat{q}^r_t \leftarrow r_t + \hat{q}^r_{t+1}$; ▷ `Accumulating rewards`
      $\rho_t \leftarrow \pi_\varphi(a_t|s_t)/\pi_b(a_t|s_t)$; ▷ `Importance weight`
      $g \leftarrow g + \rho_t \hat{q}^r_t \nabla_\varphi \log \pi_\varphi(a_t|s_t)$; ▷ `By Eqn. (8)`
    $\varphi \leftarrow \varphi + \eta g$; ▷ `Gradient ascent`
  return π_φ

3 Experiments

3.1 Datasets and Metrics

Dialogue Generation. We adopt two widely used datasets, DailyDialog [30] and OpenSubtitles [57], for the dialogue experiment. The DailyDialog dataset is constructed from English dialogues crawled from the Internet, whereas the OpenSubtitles dataset is constructed from movie subtitles based on IMDB identifiers. We split a dialogue session into single-turn context–response pairs in our experiment. In our setting of semi-supervised learning, we use the smaller dataset, DailyDialog, as the parallel corpus $\mathcal{D}_p$, and the larger dataset, OpenSubtitles, as the non-parallel corpus $\mathcal{D}_u$ (i.e., we only retain the context sentence in the OpenSubtitles dataset). This follows the common setup for semi-supervised learning, where the unlabeled dataset is larger than the labeled one.

It should be emphasized that a recent study [62] shows more than 20% of test samples are identical to some training samples in both DailyDialog and OpenSubtitles. This results in meaningless comparison and inflated performance of previous methods, e.g., a BLEU4 of 11.01 in AdaLabel [60] and 14.61 in DialogBERT [16]. Therefore, we use the deduplicated datasets in [62], containing

Table 1: Main results. $^{\uparrow/\downarrow}$The higher/lower, the better. †Quoted from Wen et al. [62] on deduplicated dialogue datasets. ‡Quoted from [29]. §Quoted from [11]. For the paraphrase generation metric, we have iBLEU = α BLEU $-(1-\alpha)$ SBLEU.

(a) Dialogue generation.

Method	BLEU2$^{\uparrow}$	BLEU4$^{\uparrow}$
Parallel DailyDialog		
AdaLabel† [60]	6.72	2.29
DialogBERT† [16]	5.42	2.16
T5-Base [42]	**8.96**	**3.69**
+ Parallel OpenSubtitles		
[T5-Base] Fully Supervised	8.75	3.06
+ Non-Parallel OpenSubtitles		
[T5-Base] Self-Training	9.10	3.73
[T5-Base] R-Regression	10.34	4.18
[T5-Base] Ours	**11.02**	**4.30**

(b) Paraphrase generation. "Copy" refers to direclty copying the input sentence.

Method	BLEU4$^{\uparrow}$	SBLEU4$^{\downarrow}$	iBLEU4$^{\uparrow}$
Copy	29.88	100.0	16.89
Parallel Quora Generation			
Dagger‡ [12]	28.42	66.98	18.88
RL-NN‡ [40]	20.98	**40.52**	14.83
T5-Base [42]	**30.83**	44.77	**23.27**
+ Non-Parallel Quora Generatoin			
LTSL§ [11]	29.25	71.25	19.20
[T5-Base] Self-Training	31.39	48.02	23.44
[T5-Base] R-Regression	30.77	**44.23**	23.27
[T5-Base] Ours	**31.47**	45.43	**23.78**

60K/6.5K/7K samples for training/validation/test in DailyDialog and 1M non-parallel samples in OpenSubtitles. Although our scores will be lower than previous inflated ones, we follow the correct setting for research.

We use BLEU scores [38] as main evaluation metrics, which are widely used in dialogue generation [62]. In particular, BLEU-n evaluates the geometric average of i-gram precision scores for $i = 1, \cdots, n$. Following previous work [62], we lowercase all sentences and tokenize them with the NLTK library [33].

Paraphrase Generation. We follow previous studies [11, 32, 34] and use the Quora Question Pair dataset[2] for the paraphrasing experiment. The Quora dataset is originally designed for paraphrase classification, containing both paraphrase and non-paraphrase pairs. The paraphrase pairs naturally form a parallel dataset for the generation purpose; following the common practice [34], we split it into 124K/4K/20K samples for training/validation/test. The non-paraphrase pairs, containing 510K sentences, are discarded in previous work, but we are able to utilize them in a semi-supervised manner.

We use the standard iBLEU score [54] as the main evaluation metric. It involves a penalty of Self-BLEU (SBLEU) between the generated and input sentences, as the paraphrasing task requires using different lexicons. Specifically, we have iBLEU = α BLEU $-(1-\alpha)$ SBLEU, where α is typically set to 0.1 [11, 32, 34]. For clarity, we also report BLEU and Self-BLEU scores in our experiment.

3.2 Settings and Competing Methods

For each task, we first fine-tune a T5-Base model [42] on the parallel data by Eqn. (4). Then we apply our proposed method to induce the reward and further train the model by Algorithm 1 on the non-parallel data. We compare our approach with the following semi-supervised methods.

Self-Training. We apply the supervised model to the non-parallel dataset and generate pseudo-target sentences, which are used to continue training the model. This is a commonly used semi-supervised approach in text generation literature [21, 68].

R-Regression. Wu et al. [66] propose a reward regression (R-Regression) approach, where the reward is defined as the BLEU score. Since their reward is the same as the evaluation metric, such a method may achieve higher BLEU scores without actually improving the generation quality. By contrast, our reward is induced in a principled way and is agnostic to evaluation metrics. In our experiment, we

[2] `https://www.kaggle.com/c/quora-question-pairs`

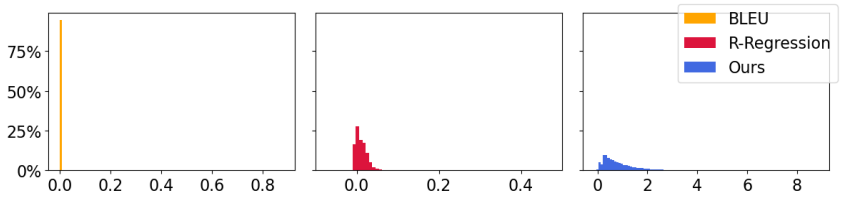

Figure 1: The distributions of token-level estimation of future rewards ($\hat{q}_t^r$ in Algorithm 1) on the DailyDialog valid set. The BLEU score of a sentence is shared among tokens in the same sentence.

replicate the R-regression method, which constitutes a controlled comparison to our approach, as the only difference is the reward function.

Appendix B provides implementation details and hyperparameters of our approach.

3.3 Main Results

Results of Dialogue Generation. Table 1a shows the results of the dialogue generation task. We notice that our fine-tuned T5-Base model [42] has already outperformed dedicated methods, AdaLabel [60] and DialogBERT [16]. This is consistent with the findings of [62, 61] in that the alleged "state-of-the-art" dialogue systems do not outperform standard pretrained language models on deduplicated datasets, highlighting the importance of working with the correct setting.

We then apply semi-supervised learning (Self-Training, R-Regression, and our approach) with the non-parallel OpenSubtitles dataset. We achieve higher performance than T5-Base trained only on parallel DailyDialog. Interestingly, the fully supervised model—trained on both parallel DailyDialog and parallel OpenSubtitles—does not achieve high performance, even lower than the one trained with DailyDialog only. It is noticed that the OpenSubtitles dataset is noisy [8], which likely causes the performance degradation. This signifies the need of semi-supervised learning, as it can explore the data space with non-parallel corpora.

Among semi-supervised approaches, RL-based methods (R-Regression and ours) are generally better than Self-Training. This is within our expectation because Self-Training learns from its own generation and may be overconfident, whereas RL approaches are able to explore different parts of the data space, being a more effective way of semi-supervised learning.

Moreover, our approach outperforms RL with R-Regression, where the reward is the only difference. The controlled experiment confirms that the reward induced from models trained with teacher forcing is effective for RL training. It is also worth noting that R-Regression uses the evaluation metric as the reward, and thus may deliberately improve the metric rather than text quality. By contrast, our reward is induced in a principled manner and is agnostic to evaluation metrics, and our approach still achieves higher performance even with such a disadvantage.

In general, our approach achieves the best performance in both metrics. In particular, it significantly improves DailyDialog-trained T5-Base by +2.06 (+23.0%) in BLEU2 and +0.61 (+16.5%) in BLEU4. It also outperforms the second-best method, R-Regression, by 0.68 (+6.6%) in BLEU2 and 0.12 (+2.9%) in BLEU4, verifying the effectiveness of our approach.

Results of Paraphrase Generation. The results of paraphrase generation are shown in Table 1b. As seen, directly copying the input already achieves a high BLEU score against the reference. iBLEU addresses this by penalizing the Self-BLEU score (against input) and is considered the main metric.

We consider another semi-supervised baseline LTSL [11]. It performs retrieval-based paraphrase expansion and meta optimization, thus being task specific. We see that LTSL has an extremely high Self-BLEU, suggesting the generated paraphrase largely resembles the input. It achieves a lower iBLEU score than other semi-supervised approaches.

We also see that RL approaches generally achieve lower Self-BLEU than Self-Training. This is because Self-Training learns from its own predictions, which overlap the input more than groundtruth paraphrases do (Self-BLEU of groundtruth: 29.87); as a result, Self-BLEU increases to 48.02 from

Table 2: Comparing sparse reward functions and dense ones.

(a) Dialogue generation.

Sparse	Name	BLEU2↑	BLEU4↑
-	Self-Training [23]	9.10	3.73
Yes	R-Regression [66]	9.45	3.73
	Induced-R	**9.75**	**3.99**
No	R-Regression [66]	10.34	4.18
	Induced-R	**11.02**	**4.30**

(b) Paraphrase generation.

Sparse	Name	BLEU4↑	SBLEU↓	iBLEU4↑
-	Self-Training [23]	31.39	48.11	23.44
Yes	R-Regression [66]	30.78	**44.32**	23.27
	Induced-R	**31.28**	45.22	**23.63**
No	R-Regression [66]	30.77	**44.23**	23.27
	Induced-R	**31.47**	45.43	**23.78**

44.77 of T5-Base. By contrast, RL learns by exploring different possible paraphrases and is able to retain low Self-BLEU.

Overall, our approach achieves the highest BLEU and a reasonably low Self-BLEU, yielding the best iBLEU among all competing methods. The results are consistent with Table 1a, showing the generality of our approach.

3.4 Analyses

Step-Wise Reward. In Figure 1, we show the distributions of different reward functions. As seen, the BLEU score is mostly concentrated at 0, providing little information as the reward for training. R-Regression consequently suffers from a similar problem, as it is trained by the groundtruth BLEU scores. The distribution of our induced reward, on the other hand, has the lowest peak and is the most wide-spreading one.

We conduct another analysis to show the importance of step-wise rewards for RL training. We compare our approach with a sparse reward function that defers all rewards to the end of a sentence. In other words, the last step's reward is the sum of our step-wise rewards, whereas all previous steps have a reward of 0. This constitutes a rigorous analysis, as the total reward and thus the training objective, are the same in both cases. Results in Table 2 show that our step-wise reward outperforms the sparse reward in all cases. This suggests our approach serves as a meaningful credit assignment of the total reward, which is beneficial for RL training.

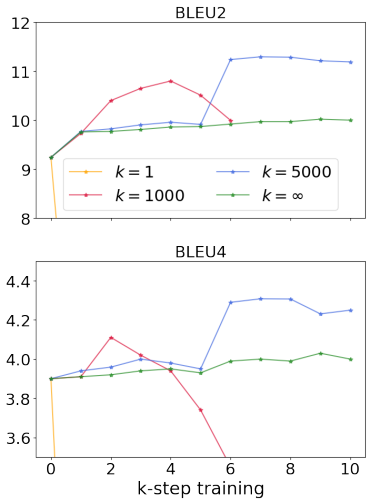

Figure 2: The learning curves by choosing different values of k. Scores are measured on the validation set of DailyDialog. Training is terminated when the BLEU4 score drops below 3.5.

The Effect of the Synchronizing Period. We analyze the effect of the synchronizing period k introduced in Section 2.3. In Figure 2, we see that the training will be unstable if $k = 1$ (on-policy), in which case the model generates uninformative and meaningless sentences (illustrated in Appendix C). When $k = 1000$, the performance increases quickly at the beginning, but it starts to decrease with further training. We hypothesize that this is due to the lack of exploration (Section 2.3). When k is infinitely large (the behavior policy is fixed), the performance grows slowly and stops improving after a certain number of steps. Based on this analysis, we choose $k = 5000$ to balance exploitation and exploration. Although the experiment is conducted on DailyDialog only due to the limit of time and resources, we directly apply the setting to other experiments, showing the robustness of our approach.

Data Efficiency. We analyze data efficiency by sampling different numbers of data points from the non-parallel corpus. In Figure 3, the trends demonstrate how the performance evolves with more non-parallel data. As shown, our method consistently outperforms self-training, even with only 0.1% (the leftmost points) of the training set. Additionally, the performance of our method quickly increases with more data, whereas self-training grows slowly. This is expected because RL training explores different parts of the sentence space and learns from their rewards, whereas self-training only learns from the single generated sentence by the model itself (given an input).

We also investigate how performance changes according to the size of the parallel dataset, which reflects the quality of the learned policy. Results are shown in Figure 4, Appendix D.

Figure 3: Trends of self-training and our method given different sizes of the non-parallel data. Scores are measured on the Daily-Dialog test set.

4 Related Work

Semi-Supervised Learning for Text Generation. In text generation, popular ways to utilize both parallel and non-parallel data include self-training [21, 68] and back-translation [49]. Both methods first train a model on the parallel sentences and then generate pseudo-parallel pairs for the non-parallel sentences. The difference is that self-training generates pseudo-parallel pairs from source to target, whereas back-translation generates from target to source. We mainly consider self-training as a baseline because it does not require an additional model in the reversed direction, making the comparisons fairer. Our implementation of self-training is also similar to "sequence-level knowledge distillation" [23, 15, 20], except that the latter augments the parallel data instead of the non-parallel ones. In Section 3.4, we show that self-training cannot efficiently utilize the data because of the lack of exploration. Additionally, the exposure bias issue remains because they are trained with the teacher-forcing objective.

Text Generation beyond Teacher Forcing. Teacher forcing is known to have the *exposure bias* issue. A line of work uses Generative adversarial networks (GAN) [14] to alleviate the issue. For example, Yu et al. [67] and Guo et al. [18] propose to use GAN-style training to generate text similar to the training set in an on-the-fly manner. This practice reduces the discrepancy between training and inference because the inputs of the model are its own generation rather than the groudtruth sentences. Shi et al. [51] further formulate the adversarial training using the IRL interpretation. These GAN-style methods are different from ours in two main ways. First, GAN-style training requires parallel corpora and thus cannot be directly applied to semi-supervised learning on non-parallel datasets. Second, GAN-style training involves the optimization of an adversarial objective, making the training unstable, e.g., suffering from mode collapse [14].

Another paradigm to alleviate the exposure bias is RL. For instance, Sokolov et al. [53] and Kreutzer et al. [26] leverage the bandit-structured prediction framework for text generation with BLEU as the heuristically defined reward. Bahdanau et al. [1] and Shen et al. [50] utilize different variants of policy gradient for RL training. However, these methods are task-specific and suffer from the sparse reward as mentioned in Section 3.4. More importantly, these approaches require parallel data to calculate the reward and cannot utilize non-parallel data either. To address this, Wu et al. [66] propose to learn a reward regression model on the parallel dataset and perform RL on the non-parallel data with the learned reward. As mentioned, such a method is still task-specific because it requires the human heuristics of the task to define the proper reward function. Additionally, it also suffers from the reward-sparsity problem, as seen in Section 3.4.

Searching is also a popular way to replace teacher forcing. The Learning to Search (L2S) framework [5, 9] enables the model to search for a better score during learning and is widely applied to text generation. For example, Wiseman and Rush [64] propose to optimize the beam search results through training. Li et al. [29] develop an unsupervised learning approach to text generation based on local search. In addition to the L2S framework, Leblond et al. [27] leverage the Monte Carlo tree search [25, 52] to select better tokens in a step from the sampled generation. These methods are different from ours since they need either heuristically defined score functions or parallel data, limiting their methods to certain tasks or to the supervised paradigm. However, given the success of these methods, we consider the search-based approach an interesting future extension of our work.

Imitation Learning. Our work is also related to imitation learning methods in general. Typically, these methods aim to obtain a good policy given a dataset containing state–action pairs. The easiest approach is behavior cloning [39], which greedily imitates the demonstration. Similar to the

exposure bias, behavior cloning also faces the problem of compounding errors [46]. SMILe [46] and DAgger [47] mitigate the problem by querying an expert. In text generation, Du and Ji [12] empirically verify that imitation learning methods are helpful. Recently, Pang and He [37] frame the text generation task as an offline reinforcement learning problem, which learns from a dataset containing tuples of state, action, and reward. Compared with our method, these approaches rely on parallel sentence pairs and cannot effectively make use of non-parallel datasets.

5 Conclusion

Summary. In this paper, we show that a reward function can be derived from a model trained with teacher forcing. The derivation does not rely on human heuristics for certain tasks. Additionally, the derived reward function assigns step-wise scores and makes the RL training easier. Our convenient approach leads to a training algorithm in a semi-supervised manner. We conduct experiments on the dialogue and paraphrase generation tasks. The empirical results show that the performance of our approach is better compared with the baselines: self-training and reward regression. We further analyze our reward function and show the benefits of our approach.

Limitation and Future Work. First, the scale of the experiments in this paper is restricted by computational resources. It is interesting to see if our approach could obtain better performance with large models [3, 41] and larger datasets.

We also notice that Assumption 1 has a deep connection with entropy-regularized RL [17, 19, 45]. Our approach can be easily extended to such cases in the future.

Another interesting direction would be using the reward as an interface between humans and the model to control the generation. Specifically, the current seq2seq models treat data as the ground truth. But the data may be contaminated with undesired or harmful information. We hope that our approach provides a way for humans to apply additional rules to the reward function to avoid the model generating harmful information.

Acknowledgments

We thank all reviewers for their valuable comments. The research is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant No. RGPIN2020-04465, the Amii Fellow Program, the Canada CIFAR AI Chair Program, a UAHJIC project, a donation from DeepMind, and the Digital Research Alliance of Canada (alliancecan.ca).

References

[1] Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. An actor-critic algorithm for sequence prediction. In *ICLR*, 2017. URL `https://openreview.net/forum?id=SJDaqqveg`.

[2] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, page 92–100, 1998. URL `https://doi.org/10.1145/279943.279962`.

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html`.

[4] Alex J Chan and Mihaela van der Schaar. Scalable bayesian inverse reinforcement learning. *ICLR*, 2021. URL `https://openreview.net/forum?id=4qR3coiNaIv`.

[5] Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daumé III, and John Langford. Learning to search better than your teacher. In *ICML*, pages 2058–2066, 2015. URL `http://proceedings.mlr.press/v37/changb15.html`.

[6] Ting-Rui Chiang and Yun-Nung Chen. Relating neural text degeneration to exposure bias. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 228–239, 2021. URL https://aclanthology.org/2021.blackboxnlp-1.16.

[7] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, 2014. URL https://aclanthology.org/D14-1179.

[8] Richard Csaky and Gábor Recski. The Gutenberg dialogue dataset. In *EACL*, pages 138–159, 2021. URL https://aclanthology.org/2021.eacl-main.11.

[9] Hal Daumé, John Langford, and Daniel Marcu. Search-based structured prediction. *Machine Learning*, 75(3):297–325, 2009.

[10] Thomas Degris, Martha White, and Richard S Sutton. Off-policy actor-critic. *ICML*, page 179–186, 2012. URL https://dl.acm.org/doi/abs/10.5555/3042573.3042600.

[11] Kaize Ding, Dingcheng Li, Alexander Hanbo Li, Xing Fan, Chenlei Guo, Yang Liu, and Huan Liu. Learning to selectively learn for weakly-supervised paraphrase generation. In *EMNLP*, pages 5930–5940, 2021. URL https://aclanthology.org/2021.emnlp-main.480.

[12] Wanyu Du and Yangfeng Ji. An empirical comparison on imitation learning and reinforcement learning for paraphrase generation. In *EMNLP*, pages 6012–6018, 2019. URL https://aclanthology.org/D19-1619.

[13] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. In *ICML*, pages 1243–1252, 2017. URL http://proceedings.mlr.press/v70/gehring17a.html.

[14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014. URL https://proceedings.neurips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html.

[15] Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. Non-autoregressive neural machine translation. In *ICLR*, 2018. URL https://openreview.net/forum?id=B1l8BtlCb.

[16] Xiaodong Gu, Kang Min Yoo, and Jung-Woo Ha. DialogBERT: Discourse-aware response generation via learning to recover and rank utterances. *AAAI*, pages 12911–12919, 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17527.

[17] Han Guo, Bowen Tan, Zhengzhong Liu, Eric P Xing, and Zhiting Hu. Text generation with efficient (soft) q-learning. *arXiv preprint arXiv:2106.07704*, 2021. URL https://arxiv.org/abs/2106.07704.

[18] Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long text generation via adversarial training with leaked information. In *AAAI*, pages 5141–5148, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16360.

[19] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *ICML*, pages 1352–1361, 2017. URL http://proceedings.mlr.press/v70/haarnoja17a.html.

[20] Chenyang Huang, Hao Zhou, Osmar R Zaïane, Lili Mou, and Lei Li. Non-autoregressive translation with layer-wise prediction and deep supervision. *AAAI*, pages 10776–10784, 2022. URL https://ojs.aaai.org/index.php/AAAI/article/view/21323.

[21] Wenxiang Jiao, Xing Wang, Zhaopeng Tu, Shuming Shi, Michael Lyu, and Irwin King. Self-training sampling with monolingual data uncertainty for neural machine translation. In *ACL*, pages 2840–2850, 2021. URL https://aclanthology.org/2021.acl-long.221.

[22] Olav Kallenberg. *Foundations of Modern Probability*. Springer, 1997.

[23] Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *EMNLP*, pages 1317–1327, 2016. URL https://aclanthology.org/D16-1139.

[24] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. URL https://arxiv.org/abs/1412.6980.

[25] Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *ECML*, pages 282–293, 2006. URL https://doi.org/10.1007/11871842_29.

[26] Julia Kreutzer, Artem Sokolov, and Stefan Riezler. Bandit structured prediction for neural sequence-to-sequence learning. In *ACL*, pages 1503–1513, 2017. URL https://aclanthology.org/P17-1138.

[27] Rémi Leblond, Jean-Baptiste Alayrac, Laurent Sifre, Miruna Pislar, Lespiau Jean-Baptiste, Ioannis Antonoglou, Karen Simonyan, and Oriol Vinyals. Machine translation decoding beyond beam search. In *EMNLP*, pages 8410–8434, 2021. URL https://aclanthology.org/2021.emnlp-main.662.

[28] Haoran Li and Wei Lu. Mixed cross entropy loss for neural machine translation. In *ICML*, pages 6425–6436, 2021. URL http://proceedings.mlr.press/v139/li21n.html.

[29] Jingjing Li, Zichao Li, Lili Mou, Xin Jiang, Michael R. Lyu, and Irwin King. Unsupervised text generation by learning from search. In *NeurIPS*, pages 10820–10831, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/7a677bb4477ae2dd371add568dd19e23-Abstract.html.

[30] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. DailyDialog: A manually labelled multi-turn dialogue dataset. In *IJCNLP*, pages 986–995, 2017. URL https://aclanthology.org/I17-1099.

[31] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, 2004. URL https://aclanthology.org/W04-1013.

[32] Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. Unsupervised paraphrasing by simulated annealing. In *ACL*, pages 302–312, 2020. URL https://aclanthology.org/2020.acl-main.28.

[33] Edward Loper and Steven Bird. NLTK: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pages 63–70, 2002. URL https://aclanthology.org/W02-0109.

[34] Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. CGMH: Constrained sentence generation by Metropolis-Hastings sampling. In *AAAI*, pages 6834–6842, 2019. URL https://doi.org/10.1609/aaai.v33i01.33016834.

[35] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. URL https://doi.org/10.1038/nature14236.

[36] Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *ICML*, pages 663–670, 2000. URL https://dl.acm.org/doi/10.5555/645529.657801.

[37] Richard Yuanzhe Pang and He He. Text generation by learning from demonstrations. In *ICLR*, 2021. URL https://openreview.net/forum?id=RovX-uQ1Hua.

[38] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A method for automatic evaluation of machine translation. In *ACL*, pages 311–318, 2002. URL https://aclanthology.org/P02-1040.

[39] Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991.

[40] Lihua Qian, Lin Qiu, Weinan Zhang, Xin Jiang, and Yong Yu. Exploring diverse expressions for paraphrase generation. In *EMNLP*, pages 3173–3182, 2019. URL https://aclanthology.org/D19-1313.

[41] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training Gopher. *arXiv preprint arXiv:2112.11446*, 2021. URL https://arxiv.org/abs/2112.11446.

[42] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text Transformer. *arXiv preprint arXiv:1910.10683*, 2019. URL https://arxiv.org/abs/1910.10683.

[43] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *IJCAI*, pages 2586–2591, 2007. URL https://dl.acm.org/doi/10.5555/1625275.1625692.

[44] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *ICLR*, 2016. URL http://arxiv.org/abs/1511.06732.

[45] Siddharth Reddy, Anca D. Dragan, and Sergey Levine. SQIL: Imitation learning via reinforcement learning with sparse rewards. In *ICLR*, 2020. URL https://openreview.net/forum?id=S1xKd24twB.

[46] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *AISTATS*, pages 661–668, 2010. URL http://proceedings.mlr.press/v9/ross10a.html.

[47] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, pages 627–635, 2011. URL http://proceedings.mlr.press/v15/ross11a/ross11a.pdf.

[48] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

[49] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *ACL*, pages 86–96, 2016. URL https://aclanthology.org/P16-1009.

[50] Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. Minimum risk training for neural machine translation. In *ACL*, pages 1683–1692, 2016. URL https://aclanthology.org/P16-1159.

[51] Zhan Shi, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. Toward diverse text generation with inverse reinforcement learning. In *IJCAI*, pages 4361–4367, 2018. URL https://doi.org/10.24963/ijcai.2018/606.

[52] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. URL https://doi.org/10.1038/nature16961.

[53] Artem Sokolov, Julia Kreutzer, Stefan Riezler, and Christopher Lo. Stochastic structured prediction under bandit feedback. In *NIPS*, pages 1489–1497, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/795c7a7a5ec6b460ec00c5841019b9e9-Abstract.html.

[54] Hong Sun and Ming Zhou. Joint learning of a dual SMT system for paraphrase generation. In *ACL*, pages 38–42, 2012. URL https://aclanthology.org/P12-2008.

[55] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[56] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pages 2818–2826, 2016. URL https://doi.org/10.1109/CVPR.2016.308.

[57] Jörg Tiedemann. News from OPUS-A collection of multilingual parallel corpora with tools and interfaces. In *Recent Advances in Natural Language Processing*, pages 237—-248, 2009. URL http://dx.doi.org/10.1075/cilt.309.19tie.

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[59] Elena Voita, Rico Sennrich, and Ivan Titov. Analyzing the source and target contributions to predictions in neural machine translation. In *ACL*, pages 1126–1140, 2021. URL https://aclanthology.org/2021.acl-long.91.

[60] Yida Wang, Yinhe Zheng, Yong Jiang, and Minlie Huang. Diversifying dialog generation via adaptive label smoothing. In *ACL*, pages 3507–3520, 2021. URL https://aclanthology.org/2021.acl-long.272.

[61] Yuqiao Wen, Yongchang Hao, Yanshuai Cao, and Lili Mou. An equal-size hard EM algorithm for diverse dialogue generation. *arXiv preprint arXiv:2209.14627*, 2022. URL https://arxiv.org/abs/2209.14627.

[62] Yuqiao Wen, Guoqing Luo, and Lili Mou. An empirical study on the overlapping problem of open-domain dialogue datasets. In *LREC*, pages 146–153, 2022. URL https://aclanthology.org/2022.lrec-1.16.

[63] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, 1992.

[64] Sam Wiseman and Alexander M. Rush. Sequence-to-sequence learning as beam-search optimization. In *EMNLP*, pages 1296–1306, 2016. URL https://aclanthology.org/D16-1137.

[65] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019. URL https://arxiv.org/abs/1910.03771.

[66] Lijun Wu, Li Zhao, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. Sequence prediction with unlabeled data by reward function learning. In *IJCAI*, pages 3098–3104, 2017. URL https://doi.org/10.24963/ijcai.2017/432.

[67] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. SeqGAN: Sequence generative adversarial nets with policy gradient. In *AAAI*, pages 2852–2858, 2017. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14344.

[68] Jiajun Zhang and Chengqing Zong. Exploiting source-side monolingual data in neural machine translation. In *EMNLP*, pages 1535–1545, 2016. URL https://aclanthology.org/D16-1160.

[69] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *AAAI*, pages 1433–1438, 2008. URL http://www.aaai.org/Library/AAAI/2008/aaai08-227.php.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See Section 2-4.
   (b) Did you describe the limitations of your work? [Yes] See Section 5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 5.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 2.1.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Section 2.2.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 3.

(b) Did you mention the license of the assets? [Yes] See Appendix.

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We make our code public.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See Appendix.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] It is discussed in the papers of the datasets.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Proof of Theorem 2

Theorem 2. *Let r^* be an underlying true reward function and q^* be the corresponding optimal value function. Given an approximate value function q, we denote by r the reward function derived from Eqn. (7). Then, we must have $\|r - r^*\|_\infty$ bounded by $O(\|q - q^*\|_\infty)$. Here, $\|\cdot\|_\infty$ takes the maximum absolute value over all $s \in \mathcal{S}$ and $a \in \mathcal{A}$.*

Proof. For any $s \in \mathcal{S}$ and $a \in \mathcal{A}$, we have

$$
\begin{aligned}
&|r(s,a) - r^*(s,a)| \\
=&|q(s,a) - q^*(s,a) + \max_{a'} q^*(s+[a],a') - \max_{a'} q(s+[a],a')| && (9)\\
\leq&|q(s,a) - q^*(s,a)| + |\max_{a'} q^*(s+[a],a') - \max_{a'} q(s+[a],a')| && (10)\\
\leq&\max_{s',a'} |q(s',a') - q^*(s',a')| + \max_{s'} |\max_{a'} q^*(s',a') - \max_{a'} q(s',a')| && (11)\\
\leq&\max_{s',a'} |q(s',a') - q^*(s',a')| \\
&+ \max_{s'} \max\{\max_{a'} q^*(s',a') - \max_{a'} q(s',a')), \max_{a'} q(s',a') - \max_{a'} q^*(s',a))\} && (12)\\
\leq&\max_{s',a'} |q(s',a') - q^*(s',a')| \\
&+ \max_{s'} \max\{\max_{a'}(q^*(s',a') - q(s',a')), \max_{a'}(q(s',a') - q^*(s',a'))\} && (13)\\
\leq&2\max_{s',a'} |q(s',a') - q^*(s',a')| && (14)\\
=&2\|q - q^*\|_\infty && (15)
\end{aligned}
$$

Here, Eqn. (9) is from the Bellman equation; Eqn. (10) follows the triangle inequality; and Eqn. (11) generalizes certain s and a to all possible $s' \in \mathcal{S}, a' \in \mathcal{A}$. Eqn. (12) discusses two possible cases: whether $\max_{a'} q(s',a') \geq \max_{a'} q^*(s',a')$ or not. Eqn. (13) is because $-\max_{a'} q(s',a') \leq -q(s',a'')$ for any $a'' \in \mathcal{A}$. Eqn. (14) merges all the maximum operation, and Eqn. (15) is the definition of the infinity norm.

Since the last equation does not depend on s, a, we conclude $\|r - r^*\|_\infty$ is bounded by $O(\|q - q^*\|_\infty)$. □

B Experiments Details

For all experiments, we initialize the model with T5-Base [42] provided by HuggingFace [65]. We use the label smoothing [56] with a coefficient of 0.1. We use the Adam [24] optimizer with $(\beta_1, \beta_2) = (0.9, 0.999)$. Each batch contains around 32K tokens.

Table 3: Two cases where our model generates better responses.

Context		We can make shipment within one month from receipt of order.
Response	Self-Training	I see.
	R-Regression	I see.
	Ours	I see. I'll have to discuss it with my manager.
Context		Where's your girlfriend? I thought you were going out with her today.
Response	Self-Training	I got engaged. We broke up last week.
	R-Regression	I got engaged. She told me she's just married.
	Ours	She came back from Australia last week. She is a nice girl but there's nothing I can do about her.

Table 4: Failure cases of on-policy training ($k = 1$).

Context	We can make shipment within one month from receipt of order.
Response	I see. I'll have to think about it.
Context	Where's your girlfriend? I thought you were going out with her today.
Response	I'm sorry, but I'm not sure I'll be able to make it. I'll have to think about it.

For all conventional seq2seq training, the learning rate is scheduled according to the original Transformer [58] with the warm-up steps set as 4000. For all RL training, we drop the warm-up phase and set the maximum learning rate as $1e-5$. We set the synchronizing period $k = 5000$. The reward of our method is scaled down by 100 times. We apply the reward clipping trick [35] to bound the reward within $[-1, 1]$ to stabilize the training.

For inference, we follow previous work and use greedy decoding in the dialogue generation task and use beam search with a beam size of 5 in the paraphrase generation task.

All the experiments are done on either $4\times$NVIDIA A100 or $4\times$NVIDIA V100.

C Case Study

We demonstrate several cases from the generation of different models. These cases come from the DailyDialog validation set.

Dialogue. In the first case of Table 3, we show a phenomenon that previous methods tend to generate short and meaningless responses. On the other hand, our method usually generates more informative sentences and makes the conversation more natural and human-like.

The previous methods usually have the problem that the semantics of the generated sentences are inconsistent or even conflicting. In the second case in Table 3, both Self-Training and R-Regression reply "I got engaged" but the next sentences are illogical. This implies that previous methods may generate low-quality sentences even if they have seemingly decent BLEU scores. By contrast, our model generates a more proper response.

On-Policy Degeneration. In Section 2.3, we mention that if $k = 1$ (on-policy), the generation will become deterministic and uninformative. We show such cases in Table 4. The responses are generated by the first save (1000 updates) of the model in the experiment.

For both cases, the model replies "I'll have to think about it" at the end of the sentences. In fact, most of the generated responses end with this phrase, which is redundant and meaningless. This phenomenon is likely to be a result of over-deterministic and insufficient exploration of the on-policy update. If the behavior policy becomes slightly more deterministic of a certain phrase, it will have a smaller chance to explore other hypotheses. Hence it will enhance the preferred responses and become even more deterministic. On the contrary, our periodically synchronized behavior policy keeps to be exploratory and does not have the degeneration problem as shown in Table 1 and Figure 2.

D Additional Results

We analyze the effect of the sizes of parallel data in Figure 4. Our method consistently outperforms competing methods in all settings. Notably, our method drops by 6.8% when having 10% of the parallel data, whereas R-Regression drops by 10.6%. This show that our reward induction approach utilizes the parallel data more effectively.

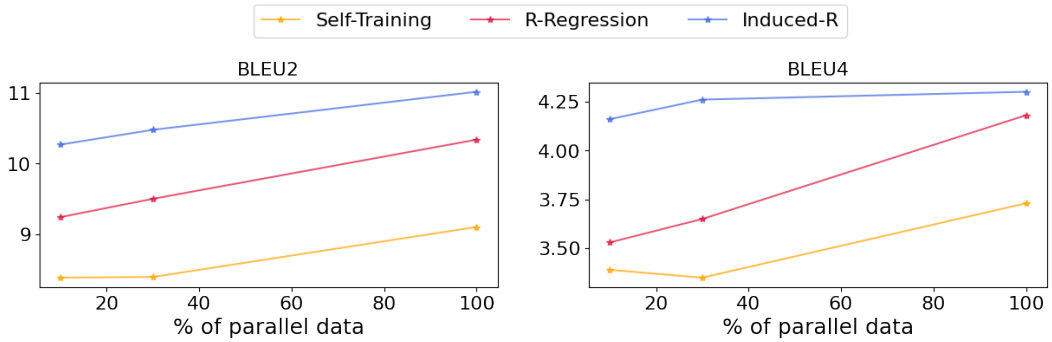

Figure 4: Results of different methdos given different sizes of the parallel data. Scores are measured on the DailyDialog test set.