Teacher Forcing Recovers Reward Functions for Text Generation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Through the lens of inverse reinforcement learning, we propose a method to derive the reward function from models trained with the teacher-forcing objective. The method gives us a step-wise reward function for sequence generation and enables reinforcement learning for text generation. Compared to previous approaches, our method is task-agnostic and does not require any human heuristics. In addition, we develop a stable training strategy based on the policy gradient method, which leads to further improvement on non-parallel datasets when combined with our proposed reward. The empirical results show that our method outperforms several previous works, which confirms the effectiveness of our reward function. The code is publicly available [1].

1 Introduction

The common training method for text generation tasks is teacher-forcing training on parallel datasets [7]. Although this practice has been widely applied [7, 13, 56], there are mainly two drawbacks. 1) Teacher-forcing training is data-hungry because parallel datasets are usually expensive to obtain. Larger models [3, 42] further intensify this problem. Meanwhile, 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 ground-truth 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* [45]. While many discussions on exposure bias taking place [20, 64], abundant evidence [6, 28, 57] shows that the exposure bias cannot be overlooked.

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 [24], and back-translation [49]. However, the exposure bias remains in such cases.

To address the second problem, the model must be trained on the self-generated sentences, often based on reinforcement learning (RL). However, text generation does not have 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 different 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 within one framework with a learned dense reward function. Our approach is a two-step strategy: we first train a sequence-to-sequence (seq2seq)

[1] Anonymous URL

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

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.

Our method is task-agnostic and does not require handcrafted engineering or heuristics. Further, our reward function assigns dense (step-wise) values, 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 and paraphrase generation tasks to show the generality of this method. The empirical results suggest that our method leads to better performance than several baselines, including self-training and task-specific heuristic reward learning. This confirms the effectiveness of our framework.

2 Approach

Our approach resorts to reinforcement learning to perform exploration on non-parallel data, whose foundation is the Markov decision process (MDP). In this section, we first introduce the MDP formulation for text generation. Then we describe our approach that induces 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 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 dynamics $T(s'|s,a)$ 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 trained to maximize the expected total reward, given by 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 $\boldsymbol{y}_{<t} := (y_1, \ldots, y_{t-1})$, 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 a number of shortcomings (e.g., sparsity and task-specificity) as mentioned in Section 1. To address these problems, we propose to induce a reward function in a principled approach by inverse reinforcement learning for a text generation task at hand.

Maximum Entropy 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) satisfies $q(s,a) \geq q(s,a')$ whenever $a' \neq a$ for any sample (s,a) in a training set $\mathcal{D} := \{(s^{(i)}, a^{(i)})\}_i$. In other words, the decisions in $\mathcal{D}$ are made greedily by $\operatorname{argmax}_{\mathrm{a}} q(s,a)$ given any state s. Unfortunately, Ng and Russell show that this is an ill-posed problem, since the desirable reward function r is not unique. Therefore, we follow the assumption proposed previously [4, 44, 67] to resolve the ambiguities:

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

In traditional IRL [4, 44, 67], the 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 the action-value function q, and then a reward function r, from some learned policy π.

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 the teacher-forcing training, 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 for defining the reward. 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 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 then:

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

Teacher-Forcing Training. As 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 policy 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 [23] 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. Our theorem shows that seq2seq training with teacher forcing actually learns an IRL model. Thus, we may derive the 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.

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 training. In addition, we propose a variant of the off-policy policy gradient method [9] by using the periodically synchronized behavior policy.

Our RL training algorithm first adopts the off-policy variant of the REINFORCE [60], a policy gradient method, as the backbone of our algorithm. Specifically, we first parameterize the model policy π_φ with φ. Given a behavior policy π_b, the gradient of the expected reward with regard to the parameter φ can be obtained by the off-policy policy gradient theorem [9]

$$\nabla_\varphi \mathop{\mathbb{E}}_{\tau \sim \pi_\varphi} \left[\sum_{t=1}^{|\tau|} r(s_t, a_t) \right] = \mathop{\mathbb{E}}_{\tau \sim \pi_b} \left[\sum_{t=1}^{|\tau|} \rho_t \hat{q}_r(s_t, a_t) \nabla_\varphi \log \pi_\varphi(a_t|s_t) \right] \tag{8}$$

where $\rho_t := \pi_\varphi(a_t|s_t)/\pi_b(a_t|s_t)$ is the importance factor, $\hat{q}_r(s_t, a_t) := \sum_{t=1}^{|\tau|} r(s_t, a_t)$ is the estimated value function by accumulating the rewards.

The off-policy REINFORCE [9] is more exploratory compared to the on-policy ($\pi_b = \pi_\varphi$) one because the samples from the behavior policy π_b are able to cover all possible trajectories, whereas the model policy π_φ may only generate fixed trajectories when it becomes more optimal.

However, the fixed behavior policy π_b has the exploitation problem and does not capture the current state of the model policy π_φ. This leads to less informative training because trajectories sampled from the behavior policy are unlikely to occur under the current model policy π_φ. To balance the exploration and exploitation, we aim to make the behavior policy close to the model policy but stay exploratory for a long time.

We propose a periodical updating schedule of the behavior policy by keeping the behavior policy at a proper distance from the current model policy. Particularly, the behavior policy keeps unchanged for a certain period until the model policy is sufficiently updated. In other words, the behavior policy synchronizes the weights with the model policy after k steps of gradient updates for some positive integer k.

Overall, our final algorithm is described in Algorithm 1. In practice, all loops in the algorithm are able to execute in parallel to speed up the training. Further, we distribute the behavior policy and the model policy to different processes and communicate the weights through the shared memory mechanism to reduce the waiting time of the sampling. Details are presented in Appendix B.

3 Experiments

We conduct experiments on two different tasks, namely, dialogue generation and paraphrase generation. In this section, we show the details of our experiments.

Algorithm 1: Overall Algorithm

Input: A non-parallel dataset $\mathcal{D}$, learned logit (value) function f_ω, policy π_φ with the initial parameter φ, total update steps U, 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 $\boldsymbol{x} \in \mathcal{D}$

    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

3.1 Settings

Our approach involves training a reward function and a policy network by teacher forcing, which requires (at least a small amount) of parallel data. In this paper, we mainly focus on semi-supervised learning, where we assume a small parallel corpus and a large non-parallel corpus are available. Nevertheless, our approach may be extended to other scenarios, such as generative adversarial networks (GANs) [18, 65], which may be addressed in the future work.

For each task, we first train a model on the parallel data by Eqn. (4). We initialize the parameters with the pre-trained T5-Base [43] model, and fine-tune all parameters by teacher forcing in a supervised manner. Then, we apply our proposed approach and further train the model by reinforcement learning on the non-parallel data.

We compare our approach with the following competing methods.

Self-Training. We use the supervised model to generate the corresponding pseudo-target sentences given the non-parallel datasets. This generated pseudo-parallel dataset is then applied for training on the same model. This is commonly used as a semi-supervised approach for text generation [22, 66].

R-Regression. We replicate the reward regression (R-Regression) method [63] by following the settings of [63] and train an additional token-level reward function through regression with BLEU scores. Similarly, we then apply the same REINFORCE [60] method to further train the supervised model. Compared to our method, the reward function of R-Regression directly predicts the metric of interest, whereas our method does not have the access to any defined metrics. Note that the only difference between this method and ours is the reward function.

The details of our experiments are included in Appendix A.

3.2 Datasets and Metrics

Dialogue Generation. For the dialogue generation task, we use DailyDialogue [30] as the parallel dataset and collect only source sentences in OpenSubtitle [55] as the non-parallel dataset. The DailyDialogue dataset is constructed from English dialogues crawled from the Internet, whereas the OpenSubtitle dataset is constructed from movie subtitles based on IMDB identifiers. For each piece

Table 1: Results on both the dialogue and paraphrase generation tasks. Numbers for models labeled with † are quoted from previous work. "[T5B]" indicates the parameter of the model is initialized with the T5-Base model trained on the parallel dataset. We use ↓ to represent that we want the metric as minimum as possible, whereas ↑ means the opposite.

(a) Results of the dialogue generation task evaluated on the cleaned DailyDialogue test set.

Name	BLEU2↑	BLEU4↑
Parallel DailyDialogue		
AdaLabel† [58]	6.72	2.29
DialogBERT† [16]	5.42	2.16
T5-Base [43]	**8.96**	**3.69**
+ Parallel OpenSubtitle		
[T5B] Fully-Supervised	8.75	3.06
+ Non-Parallel OpenSubtitle		
[T5B] Self-Training [24]	9.10	3.73
[T5B] R-Regression [63]	10.34	4.18
[T5B] Ours	**11.02**	**4.30**

(b) Results of the paraphrase generation task evaluated on the Quora Paraphrase Generation test set. The result of "Copy" uses the source sentences as their generation.

Name	BLEU4↑	SBLEU4↓	iBLEU4↑
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 [43]	**30.83**	44.77	**23.27**
+ Non-Parallel Quora Generatoin			
LTSL† [10]	29.25	71.25	19.20
[T5B] Self-Training [24]	31.39	48.02	23.44
[T5B] R-Regression [63]	30.77	**44.23**	23.27
[T5B] Ours	**31.47**	45.43	**23.78**

of the dialogue in both datasets, there are only two speakers and they speak in an in-turn manner. For each turn, we view it as a context-response paradigm and our model learns to generate the response given the context. We only consider the single-turn dialogue in this experiment.

However, there are duplicated sentences across training and test sets in both two datasets which would result in an unfair comparison [59]. To address this problem, we turn to using the cleaned version [59]. In this version, DailyDialogue contains 60K/6.5K/7K samples for training/validation/test and OpenSubtitle contains 1M/12K/12K samples. Additionally, for the cleaned OpenSubtitle dataset we only use the source sentences of the training set to serve as a non-parallel dataset. Note that parallel sentences in the OpenSubtitle dataset are noisy and often result in lower performance [59].

We use the BLEU score [38] as the major evaluation metric, which is widely used in the dialogue generation tasks [59]. BLEU-n evaluates the geometric average of i-gram precision scores where i is from 1 up to n. Following the previous work [59], we lowercase all sentences and tokenize them with the NLTK [33] library.

Paraphrase Generation. For the paraphrase generation task, the dataset is constructed from the Quora Question Pair classification dataset [41] following the common practice [10, 32, 34]. This classification dataset contains question pairs labeled as either paraphrase or non-paraphrase. We collect all paraphrase pairs as the parallel dataset and non-paraphrase sentences as the non-parallel dataset. Following the previous work [10, 32, 34], we split the parallel dataset into 124k/4k/20k as training/validation/test; the resulting non-parallel dataset contains 510k sentences.

We use the iBLEU [53] score as the main evaluation metrics. The calculation of iBLEU involves a penalty of the Self-BLEU (the BLEU score between the generation and the source corpus) to avoid directly copying the source sentences. Specifically, iBLEU = α BLEU $-(1-\alpha)$ SBLEU. Following the previous work, we use iBLEU with $\alpha = 0.1$. We also report the BLEU and Self-BLEU (SBLEU) scores for clarity.

3.3 Main Results

Our main results are shown in Table 1.

Results of Dialogue Generation. For the dialogue generation task shown in Table 1a. The self-training method only provides a very slight improvement in BLEU2 and BLEU4 and stays roughly close to the supervised model. The reward regression (R-Regression) model shows a moderate

improvement over the supervised model. However, it is still obviously lower than the performance of our model. Specifically, our method significantly improves the supervised model by +2.06 (+23.0%) in BLEU2 and +0.61 (+16.5%) in BLEU4. It also outperforms the R-Regression by 0.68 (+6.6%) in BLEU2 and 0.12 (+2.9%) in BLEU4. Note that throughout the process, our model does not have the access to the evaluation metric (i.e., BLEU score) of its generation, whereas the R-Regression directly optimizes towards higher BLEU score predictions. Even with this obstacle, our proposed reward achieves higher final performance, suggesting the generality of our method.

We notice that RL training with either reward functions shows superior performance compared to self-training, which indicates the importance of explorations on the dialogue task. This is within our expectation because tasks like dialogue usually require the model to capture the multi-modality of responses. RL training naturally possesses the advantage of exploring different possible hypotheses of the response given the context and thus generally tend to have higher performance.

For comparison with RL training, we also show the fully-supervised result by using the ground truth references of OpenSubtitle started with the same model. Interestingly, due to the low quality of the OpenSubtitle dataset, the resulting model performs even worse compared to the one trained on the parallel DailyDialogue only. This emphasizes the need for semi-supervised approaches when the quality of the parallel dataset is not desirable.

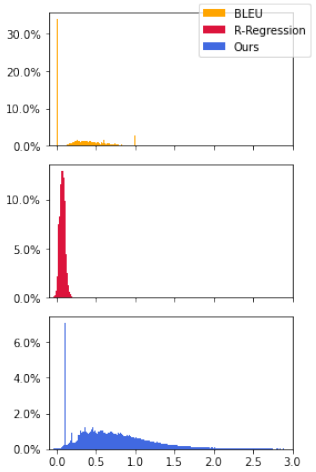

Figure 1: The distributions of token-level estimation of future rewards ($\hat{q}_t^r$ in Algorithm 1) on the Quora Generation test set. The BLEU score of a sentence is shared among tokens in the same sentence.

Results of Paraphrase Generation. For the paraphrase generation task in Table 1b, our approach consistently achieves the best performance in iBLEU among all baselines. We report Self-BLEU (SBLEU) scores to directly display the similarity between the model generation and the source sentence. As shown, several previous methods [10, 24, 40, 63] cannot maintain a good balance between BLEU and SBLEU, some even achieve SBLEU scores over 60, indicating a severe problem of the overlapping between the source sentences and model generation. On the contrary, our method shows a total advantage by having the highest BLEU4 and a lower SBLEU4 at the same time, resulting in a much more desired iBLEU score. This further implies that our reward function implicitly controls the similarity between the source sentence and generation without any human heuristics injected, which again confirms the generality of our method.

3.4 Analysis

Step-Wise Reward. In Figure 1, we show that our derived reward function solves the sparsity problem as the rewards are mostly distributed around 0.5. However, The ground-truth BLEU scores largely concentrate on 0, providing little information for the training. For the regression with the ground-truth BLEU scores (R-regression), although the sparsity problem is mitigated, R-Regression

Table 2: Results the sparse reward function compare to the dense ones.

(a) The results are evaluated on the DailyDialogue test set.

Dense	Name	BLEU2↑	BLEU4↑
-	Self-Training [24]	9.10	3.73
No	R-Regression [63]	9.45	3.73
	Ours	**9.75**	**3.99**
Yes	R-Regression [63]	10.34	4.18
	Ours	**11.02**	**4.30**

(b) The results are evaluated on the Quora Paraphrase Generation test set.

Dense	Name	BLEU4↑	SBLEU↓	iBLEU4↑
-	Self-Training [24]	31.39	48.11	23.44
No	R-Regression [63]	30.78	**44.32**	23.27
	Ours	**31.28**	45.22	**23.63**
Yes	R-Regression [63]	30.77	**44.23**	23.27
	Ours	**31.47**	45.43	**23.78**

still assigns near 0 rewards extensively. This results in a more sparse distribution with a higher peak compared to our method.

The density of our reward is beneficial for training. In Table 2, we show the results when we reduce the step-wise (dense) reward to the sentence-level (sparse) one by only using the summation of rewards at the end of the sentence. As shown, when the reward is sparse, the final performance would uniformly drop to a lower level. This verifies that the step-wise reward is essential to RL training in text generation.

The Effect of the Synchronizing Period. The synchronizing period k we introduced in Section 2.3 plays an important role in the RL training. In Figure 2 we show that if $k = 1$ (on-policy), then the training will be unstable and drastically drop into a degeneration model, where the generation will be uninformative. We provide examples of the generation in Appendix C. The training is more stable when $k = 1000$. It quickly goes up to a more optimal point but slowly decreases with more update steps. This is due to the lack of exploration as we mentioned in Section 2.3. When k is infinitely large, the behavior policy is then fixed as in the conventional off-policy case throughout the training. In this case, the training will be inefficient and stop improving after a certain step within each period. As a result, we choose $k = 5000$ in all experiments to balance the exploitation and exploration.

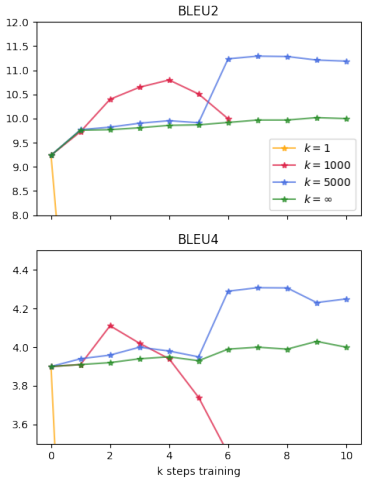

Figure 2: The learning curves by choosing different k. Scores are measured on the validation set of DailyDialogue. Training is terminated when the BLEU4 score drops below 3.5.

Data Efficiency. Our method also demonstrates the ability to efficiently utilize the data. Specifically, we sample different subsets of various sizes from the non-parallel OpenSubtitle training set and compare the performance of our method and self-training. This is shown in Figure 3. The trend shows that our method achieves a higher performance even when there is only 0.1% of the training set available. Additionally, the performance of our method increases to an even higher level when more data is available, whereas self-training can not effectively utilize the data and grows slowly.

4 Related Work

Semi-Supervised Learning for Text Generation. In text generation, the most popular ways to utilize both the parallel and non-parallel data are by self-training [22, 24, 66] or back-translation [49]. Specifically, they are first trained on the parallel sentences and then generate the pseudo-parallel pairs for the non-parallel sentences. These approaches are widely applied to different paradigms. For example, Gu et al. propose to train the non-autoregressive models [11, 15, 21] on the pseudo-parallel pairs. We mainly compare with self-training as we do not train the reversed model. In this work, we show that such a practice (self-training) cannot efficiently exploit the data because of the lack of exploration. Additionally, since they are trained on the given sentence, the exposure bias remains.

Text Generation beyond Teacher Forcing. Many researchers try to alleviate the exposure bias by training the model on self-generated sentences. These methods usually involve generative adversarial networks (GAN) [14] training or RL. For example, Yu et al. and Guo et al. propose to use GAN-style training to generate text similar to the training set. Shi et al. further formulate the adversarial training using the IRL interpretation. The GAN-style methods are different from ours in mainly two ways. First, GAN-style training requires the parallel corpora and thus can not be applied in the case of semi-supervised learning to utilize non-parallel datasets. Second, GAN-style training involves optimizing a counterpart component during the training, making it unstable (e.g. mode collapse [14]). In another line of work, Bahdanau et al. and Shen et al. heuristically define BLEU as the reward and conduct RL training with parallel datasets. These methods are task-specific and suffer from the sparse reward as we mentioned in Section 3.4. More importantly, these approaches require the ground-truth reference in parallel data to calculate the reward.

In an effort to remove the reliance on the ground-truth references for calculating the reward, the most straightforward way is learning to predict the reward function by regression on the parallel dataset [63]. As we discussed in Section 1, this method is still task-specific because it requires human heuristics of the task to define the proper reward function. Additionally, the reward regression also suffers from the sparsity of the metric in Section 3.4 and hence reduces the quality of the generated sentences.

Searching is also a popular way to replace teacher forcing. The Learning to Search (L2S) framework [5, 8] enables the model to search for a better score during learning and is widely applied in text generation. For example, Wiseman and Rush propose to optimize the beam-search results through training. Li et al. develop an unsupervised learning approach for text generation based on local search. In addition to the L2S framework, Leblond et al. leverage Monte Carlo tree search [26, 52] to unify the training and inference in text 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.

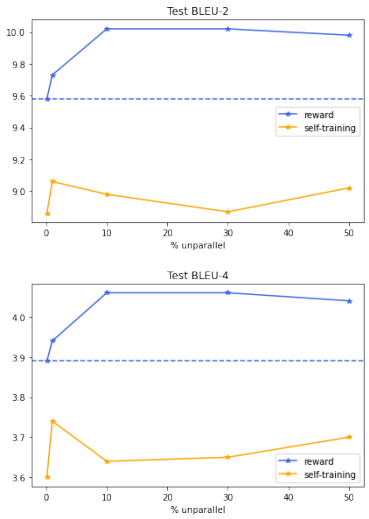

Figure 3: Trends of self-training and our method given different sizes of the non-parallel data. Scores are measured on the Daily-Dialogue test set.

Learning from Demonstration. Our work is also related to learning from demonstration in general. Typically, these methods aim to return a good policy given the demonstration 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 compounding errors problem [47], which gives rise to many solutions [47, 48]. Du and Ji empirically show that these methods are also helpful in text generation. Recently, Pang and He frame text generation tasks as an offline reinforcement learning problem given the demonstrations from the experts. Compared to our method, these approaches rely on parallel sentence pairs and cannot effectively make use of non-parallel datasets.

5 Conclusion

Summary. We show that models trained with the teacher-forcing objective can be viewed as reward functions. The derivation of the reward function does not rely on human heuristics for certain tasks. Additionally, the 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. The empirical results show that the performance of our approach is strong compared to several other methods. Specifically, it outperforms self-training and reward regression as two popular semi-supervised baselines. We further analyze our reward function and show the benefits of our approach.

Limitations and future works. In Figure 3, we show that our method greatly boosts the data efficiency when the non-parallel data is insufficient. However, we notice that if the dataset grows larger, although we still maintain the edge compared to other methods, the help of the exploration does not effectively scale with the dataset. We leave the solution to this as a direction for future work. We also notice that Assumption 1 has a deep connection with entropy regularized RL [17, 19, 46]. Our approach can be easily extended to the 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 human to apply additional rules to the reward function to avoid the model generating harmful information.

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 *Proc. of ICLR*, 2017.

[2] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. of COLT*, 1998.

[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 *Proc. of NeurIPS*, 2020.

[4] Alex J Chan and Mihaela van der Schaar. Scalable bayesian inverse reinforcement learning. *Proc. of ICLR*, 2021.

[5] Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daumé III, and John Langford. Learning to search better than your teacher. In *Proc. of ICML*, pages 2058–2066, 2015.

[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*, 2021.

[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 *Proc. of EMNLP*, pages 1724–1734, 2014.

[8] Hal Daumé, John Langford, and Daniel Marcu. Search-based structured prediction. *Machine learning*, 2009.

[9] Thomas Degris, Martha White, and Richard S Sutton. Off-policy actor-critic. *Proc. of ICML*, 2012.

[10] 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 *Proc. of EMNLP*, 2021.

[11] Liang Ding, Longyue Wang, Xuebo Liu, Derek F Wong, Dacheng Tao, and Zhaopeng Tu. Understanding and improving lexical choice in non-autoregressive translation. *Proc. of ICLR*, 2021.

[12] Wanyu Du and Yangfeng Ji. An empirical comparison on imitation learning and reinforcement learning for paraphrase generation. In *Proc. of EMNLP*, pages 6012–6018, 2019.

[13] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. In *Proc. of ICML*, pages 1243–1252, 2017.

[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 *Proc. of NeurIPS*, pages 2672–2680, 2014.

[15] Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. Non-autoregressive neural machine translation. In *Proc. of ICLR*, 2018.

[16] Xiaodong Gu, Kang Min Yoo, and Jung-Woo Ha. Dialogbert: Discourse-aware response generation via learning to recover and rank utterances. *Proc. of AAAI*, 2021.

[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.

[18] Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long text generation via adversarial training with leaked information. In *Proc. of AAAI*, pages 5141–5148, 2018.

[19] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *Proc. of ICML*, pages 1352–1361, 2017.

[20] Tianxing He, Jingzhao Zhang, Zhiming Zhou, and James Glass. Exposure bias versus self-recovery: Are distortions really incremental for autoregressive text generation? In *Proc. of EMNLP*, 2021.

[21] Chenyang Huang, Hao Zhou, Osmar R Zaïane, Lili Mou, and Lei Li. Non-autoregressive translation with layer-wise prediction and deep supervision. *Proc. of AAAI*, 2022.

[22] 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 *Proc. of ACL*, pages 2840–2850, 2021.

[23] Olav Kallenberg. *Foundations of Modern Probability*, pages 80–95. Springer, 1997.

[24] Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proc. of EMNLP*, pages 1317–1327, 2016.

[25] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of ICLR*, 2015.

[26] Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European conference on machine learning*, pages 282–293, 2006.

[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 *Proc. of EMNLP*, pages 8410–8434, 2021.

[28] Haoran Li and Wei Lu. Mixed cross entropy loss for neural machine translation. In *Proc. of ICML*, 2021.

[29] Jingjing Li, Zichao Li, Lili Mou, Xin Jiang, Michael R. Lyu, and Irwin King. Unsupervised text generation by learning from search. In *Proc. of NeurIPS*, 2020.

[30] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. DailyDialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, 2017.

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

[32] Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. Unsupervised paraphrasing by simulated annealing. In *Proc. of ACL*, pages 302–312, 2020.

[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.

[34] Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. CGMH: constrained sentence generation by metropolis-hastings sampling. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6834–6842, 2019.

[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*, 2015.

[36] Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *Proc. of ICML*, pages 663–670, 2000.

[37] Richard Yuanzhe Pang and He He. Text generation by learning from demonstrations. In *Proc. of ICLR*, 2021.

[38] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL*, pages 311–318, 2002.

[39] Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 1991.

[40] Lihua Qian, Lin Qiu, Weinan Zhang, Xin Jiang, and Yong Yu. Exploring diverse expressions for paraphrase generation. In *Proc. of EMNLP*, pages 3173–3182, 2019.

[41] Quora. Quora question pairs, 2017.

[42] 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.

[43] 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.

[44] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *Proc. of IJCAI*, 2007.

[45] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *Proc. of ICLR*, 2016.

[46] Siddharth Reddy, Anca D. Dragan, and Sergey Levine. SQIL: imitation learning via reinforcement learning with sparse rewards. In *Proc. of ICLR*, 2020.

[47] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 661–668, 2010.

[48] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proc. of AISTATS*, 2011.

[49] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *Proc. of ACL*, pages 86–96, 2016.

[50] Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. Minimum risk training for neural machine translation. In *Proc. of ACL*, pages 1683–1692, 2016.

[51] Zhan Shi, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. Toward diverse text generation with inverse reinforcement learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4361–4367, 2018.

[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*, pages 484–489, 2016.

[53] Hong Sun and Ming Zhou. Joint learning of a dual SMT system for paraphrase generation. In *Proc. of ACL*, pages 38–42, 2012.

[54] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2818–2826, 2016.

[55] Jörg Tiedemann. News from opus-a collection of multilingual parallel corpora with tools and interfaces. In *Recent advances in natural language processing*, 2009.

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proc. of NeurIPS*, pages 5998–6008, 2017.

[57] Elena Voita, Rico Sennrich, and Ivan Titov. Analyzing the source and target contributions to predictions in neural machine translation. In *Proc. of ACL*, pages 1126–1140, 2021.

[58] Yida Wang, Yinhe Zheng, Yong Jiang, and Minlie Huang. Diversifying dialog generation via adaptive label smoothing. In *Proc. of ACL*, pages 3507–3520, 2021.

[59] Yuqiao Wen, Guoqing Luo, and Lili Mou. An empirical study on the overlapping problem of open-domain dialogue datasets. *Proc. of LREC*, 2022.

[60] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 1992.

[61] Sam Wiseman and Alexander M. Rush. Sequence-to-sequence learning as beam-search optimization. In *Proc. of EMNLP*, pages 1296–1306, 2016.

[62] 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.

[63] Lijun Wu, Li Zhao, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. Sequence prediction with unlabeled data by reward function learning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 3098–3104, 2017.

[64] Yifan Xu, Kening Zhang, Haoyu Dong, Yuezhou Sun, Wenlong Zhao, and Zhuowen Tu. Rethinking exposure bias in language modeling. *arXiv preprint arXiv:1910.11235*, 2019.

[65] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proc. of AAAI*, pages 2852–2858, 2017.

[66] Jiajun Zhang and Chengqing Zong. Exploiting source-side monolingual data in neural machine translation. In *Proc. of EMNLP*, pages 1535–1545, 2016.

[67] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Proc. of AAAI*, 2008.

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]

Additional Appendix: Theoretical Results

In this section, we will provide an additional theorem that links an approximate q-value function with an approximate reward. Should the paper be accepted, we would merge this theorem into our appendix.

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, $\|\cdot\|_\infty$ essentially 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 $O(\|q - q^*\|_\infty)$. □