# From Credit Assignment to Entropy Regularization: Two New Algorithms for Neural Sequence Prediction

**Zihang Dai*, Qizhe Xie*, Eduard Hovy**
Language Technologies Institute
Carnegie Mellon University
{dzihang, qizhex, hovy}@cs.cmu.edu

## Abstract

In this work, we study the credit assignment problem in reward augmented maximum likelihood (RAML) learning, and establish a theoretical equivalence between the token-level counterpart of RAML and the entropy regularized reinforcement learning. Inspired by the connection, we propose two sequence prediction algorithms, one extending RAML with fine-grained credit assignment and the other improving Actor-Critic with a systematic entropy regularization. On two benchmark datasets, we show the proposed algorithms outperform RAML and Actor-Critic respectively, providing new alternatives to sequence prediction.

## 1 Introduction

Modeling and predicting discrete sequences is the central problem to many natural language processing tasks. In the last few years, the adaption of recurrent neural networks (RNNs) and the sequence-to-sequence model (seq2seq) (Sutskever et al., 2014; Bahdanau et al., 2014) has led to a wide range of successes in conditional sequence prediction, including machine translation (Sutskever et al., 2014; Bahdanau et al., 2014), automatic summarization (Rush et al., 2015), image captioning (Karpathy and Fei-Fei, 2015; Vinyals et al., 2015; Xu et al., 2015) and speech recognition (Chan et al., 2016).

Despite the distinct evaluation metrics for the aforementioned tasks, the standard training algorithm has been the same for all of them. Specifically, the algorithm is based on maximum likelihood estimation (MLE), which maximizes the log-likelihood of the "ground-truth" sequences empirically observed.[1]

While largely effective, the MLE algorithm has two obvious weaknesses. Firstly, the MLE training ignores the information of the task specific metric. As a result, the potentially large discrepancy between the log-likelihood during training and the task evaluation metric at test time can lead to a suboptimal solution. Secondly, MLE can suffer from the exposure bias, which refers to the phenomenon that the model is never exposed to its own failures during training, and thus cannot recover from an error at test time. Fundamentally, this issue roots from the difficulty in statistically modeling the exponentially large space of sequences, where most combinations cannot be covered by the observed data.

To tackle these two weaknesses, there have been various efforts recently, which we summarize into two broad categories:

- A widely explored idea is to directly optimize the task metric for sequences produced by the model, with the specific approaches ranging from minimum risk training (MRT) (Shen et al., 2015) and learning as search optimization (LaSO) (Daumé III and Marcu, 2005; Wiseman and Rush, 2016) to reinforcement learning (RL) (Ranzato et al., 2015; Bahdanau et al., 2016). In spite of the technical differences, the key component to make these training algorithms *practically efficient* is often a delicate credit assignment scheme, which transforms the sequence-level signal into dedicated smaller units (e.g., token-level or chunk-level), and allocates them to specific decisions, allowing for efficient optimization with a much lower variance. For instance, the beam search optimiza-

* Equal contribution.

[1] In this work, we use the terms "ground-truth" and "reference" to refer to the empirical observations interchangeably.



*Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers)*, pages 1672–1682
Melbourne, Australia, July 15 - 20, 2018. ©2018 Association for Computational Linguistics

tion (BSO) (Wiseman and Rush, 2016) utilizes the position of margin violations to produce signals to the specific chunks, while the actor-critic (AC) algorithm (Bahdanau et al., 2016) trains a critic to enable token-level signals.

- Another alternative idea is to construct a task metric dependent target distribution, and train the model to match this task-specific target instead of the empirical data distribution. As a typical example, the reward augmented maximum likelihood (RAML) (Norouzi et al., 2016) defines the target distribution as the exponentiated pay-off (sequence-level reward) distribution. This way, RAML not only can incorporate the task metric information into training, but it can also alleviate the exposure bias by exposing imperfect outputs to the model. However, RAML only works on the sequence-level training signal.

In this work, we are intrigued by the question whether it is possible to incorporate the idea of fine-grained credit assignment into RAML. More specifically, inspired by the token-level signal used in AC, we aim to find the token-level counterpart of the sequence-level RAML, i.e., defining a token-level target distribution for each auto-regressive conditional factor to match. Motived by the question, we first formally define the desiderata the token-level counterpart needs to satisfy and derive the corresponding solution (§2). Then, we establish a theoretical connection between the derived token-level RAML and entropy regularized RL (§3). Motivated by this connection, we propose two algorithms for neural sequence prediction, where one is the token-level extension to RAML, and the other a RAML-inspired improvement to the AC (§4). We empirically evaluate the two proposed algorithms, and show different levels of improvement over the corresponding baseline. We further study the importance of various techniques used in our experiments, providing practical suggestions to readers (§6).

## 2 Token-level Equivalence of RAML

We first introduce the notations used throughout the paper. Firstly, capital letters will denote random variables and lower-case letters are the values to take. As we mainly focus on conditional sequence prediction, we use $\mathbf{x}$ for the conditional input, and $\mathbf{y}$ for the target sequence. With $\mathbf{y}$ denoting a sequence, $\mathbf{y}_i^j$ then denotes the subsequence from position $i$ to $j$ inclusively, while $y_t$ denotes the single value at position $t$. Also, we use $|\mathbf{y}|$ to indicate the length of the sequence. To emphasize the ground-truth data used for training, we add superscript $^*$ to the input and target, i.e., $\mathbf{x}^*$ and $\mathbf{y}^*$. In addition, we use $\mathcal{Y}$ to denote the set of all possible sequences with one and only one `eos` symbol at the end, and $\mathcal{W}$ to denote the set of all possible symbols in a position. Finally, we assume length of sequences in $\mathcal{Y}$ is bounded by $T$.

### 2.1 Background: RAML

As discussed in §1, given a ground-truth pair $(\mathbf{x}^*, \mathbf{y}^*)$, RAML defines the target distribution using the exponentiated pay-off of sequences, i.e.,

$$P_R(\mathbf{y} \mid \mathbf{x}^*, \mathbf{y}^*) = \frac{\exp\left(R(\mathbf{y}; \mathbf{y}^*)/\tau\right)}{\sum_{\mathbf{y}' \in \mathcal{Y}} \exp\left(R(\mathbf{y}'; \mathbf{y}^*)/\tau\right)}, \quad (1)$$

where $R(\mathbf{y}; \mathbf{y}^*)$ is the sequence-level reward, such as BLEU score, and $\tau$ is the temperature hyper-parameter controlling the sharpness. With the definition, the RAML algorithm simply minimizes the cross entropy (CE) between the target distribution and the model distribution $P_\theta(\mathbf{Y} \mid \mathbf{x}^*)$, i.e.,

$$\min_\theta \mathrm{CE}\left(P_R(\mathbf{Y} \mid \mathbf{x}^*, \mathbf{y}^*) \| P_\theta(\mathbf{Y} \mid \mathbf{x}^*)\right). \quad (2)$$

Note that, this is quite similar to the MLE training, except that the target distribution is different. With the particular choice of target distribution, RAML not only makes sure the ground-truth reference remains the mode, but also allows the model to explore sequences that are not exactly the same as the reference but have relatively high rewards.

Compared to algorithms trying to directly optimize task metric, RAML avoids the difficulty of tracking and sampling from the model distribution that is consistently changing. Hence, RAML enjoys a much more stable optimization without the need of pretraining. However, in order to optimize the RAML objective (Eqn. (2)), one needs to sample from the exponentiated pay-off distribution, which is quite challenging in practice. Thus, importance sampling is often used (Norouzi et al., 2016; Ma et al., 2017). We leave the details of the practical implementation to Appendix B.1.

### 2.2 Token-level Target Distribution

Despite the appealing properties, RAML only operates on the sequence-level reward. As a result, the reward gap between any two sequences cannot be attributed to the responsible decisions precisely,

which often leads to a low sample efficiency. Ideally, since we rely on the auto-regressive factorization $P_\theta(\mathbf{y} \mid \mathbf{x}^*) = \prod_{t=1}^{|\mathbf{y}|} P_\theta(y_t \mid \mathbf{y}_1^{t-1}, \mathbf{x}^*)$, the optimization would be much more efficient if we have the target distribution for each token-level factor $P_\theta(Y_t \mid \mathbf{y}_1^{t-1}, \mathbf{x}^*)$ to match. Conceptually, this is exactly how the AC algorithm improves upon the vanilla sequence-level REINFORCE algorithm (Ranzato et al., 2015).

With this idea in mind, we set out to find such a token-level target. Firstly, we assume the token-level target shares the form of a Boltzmann distribution but parameterized by some unknown negative energy function $Q_R$, i.e.,[2]

$$P_{Q_R}(y_t \mid \mathbf{y}_1^{t-1}, \mathbf{y}^*) = \frac{\exp\left(Q_R(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*)/\tau\right)}{\sum_{w \in \mathcal{W}} \exp\left(Q_R(\mathbf{y}_1^{t-1}, w; \mathbf{y}^*)/\tau\right)}. \tag{3}$$

Intuitively, $Q_R(\mathbf{y}_1^{t-1}, w; \mathbf{y}^*)$ measures how much *future* pay-off one can expect if $w$ is generated, given the current status $\mathbf{y}_1^{t-1}$ and the reference $\mathbf{y}^*$. This quantity highly resembles the action-value function ($Q$-function) in reinforcement learning. As we will show later, it is indeed the case.

Before we state the desiderata for $Q_R$, we need to extend the definition of $R$ in order to evaluate the goodness of an unfinished partial prediction, i.e., sequences without an `eos` suffix. Let $\mathcal{Y}^-$ be the set of unfinished sequences, following Bahdanau et al. (2016), we define the pay-off function $R$ for a partial sequence $\hat{\mathbf{y}} \in \mathcal{Y}^-, |\hat{\mathbf{y}}| < T$ as

$$R(\hat{\mathbf{y}}; \mathbf{y}^*) = R(\hat{\mathbf{y}} + \texttt{eos}; \mathbf{y}^*), \tag{4}$$

where the $+$ indicates string concatenation.

With the extension, we are ready to state two requirements for $Q_R$:

1. **Marginal match**: For $P_{Q_R}$ to be the token-level equivalence of $P_R$, the sequence-level marginal distribution induced by $P_{Q_R}$ must match $P_R$, i.e., for any $\mathbf{y} \in \mathcal{Y}$,

$$\prod_{t=1}^{|\mathbf{y}|} P_{Q_R}(y_t \mid \mathbf{y}_1^{t-1}) = P_R(\mathbf{y}). \tag{5}$$

   Note that there are infinitely many $Q_R$'s satisfying Eqn. (5), because adding any constant value to $Q_R$ does not change the Boltzmann distribution, known as shift-invariance w.r.t. the energy.

2. **Terminal condition**: Secondly, let's consider the value of $Q_R$ when emitting an `eos` symbol to immediately terminate the generation. As mentioned earlier, $Q_R$ measures the expected future pay-off. Since the emission of `eos` ends the generation, the future pay-off can only come from the immediate increase of the pay-off. Thus, we require $Q_R$ to be the incremental pay-off when producing `eos`, i.e.

$$Q_R(\hat{\mathbf{y}}, \texttt{eos}; \mathbf{y}^*) = R(\hat{\mathbf{y}} + \texttt{eos}; \mathbf{y}^*) - R(\hat{\mathbf{y}}; \mathbf{y}^*), \tag{6}$$

   for any $\hat{\mathbf{y}} \in \mathcal{Y}^-$. Since Eqn. (6) enforces the absolute of $Q_R$ at a point, it also solves the ambiguity caused by the shift-invariance property.

Based on the two requirements, we can derive the form $Q_R$, which is summarized by Proposition 1.

**Proposition 1.** *$P_{Q_R}$ and $Q_R$ satisfy requirements (5) and (6) if and only if for any ground-truth pair $(\mathbf{x}^*, \mathbf{y}^*)$ and any sequence prediction $\mathbf{y} \in \mathcal{Y}$,*

$$Q_R(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) = R(\mathbf{y}_1^t; \mathbf{y}^*) - R(\mathbf{y}_1^{t-1}; \mathbf{y}^*) + \tau \log \sum_{w \in \mathcal{W}} \exp\left(Q_R(\mathbf{y}_1^t, w; \mathbf{y}^*)/\tau\right), \tag{7}$$

*when $t < |\mathbf{y}|$, and otherwise, i.e., when $t = |\mathbf{y}|$*

$$Q_R(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) = R(\mathbf{y}_1^t; \mathbf{y}^*) - R(\mathbf{y}_1^{t-1}; \mathbf{y}^*). \tag{8}$$

*Proof.* See Appendix A.1. □

Note that, instead of giving an explicit form for the token-level target distribution, Proposition 1 only provides an equivalent condition in the form of an implicit recursion. Thus, we haven't obtained a practical algorithm yet. However, as we will discuss next, the recursion has a deep connection to entropy regularized RL, which ultimately inspires our proposed algorithms.

## 3 Connection to Entropy-regularized RL

Before we dive into the connection, we first give a brief review of the entropy-regularized RL. For an in-depth treatment, we refer readers to (Ziebart, 2010; Schulman et al., 2017).

### 3.1 Background: Entropy-regularized RL

Following the standard convention of RL, we denote a Markov decision process (MDP) by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p_s, r, \gamma)$, where $\mathcal{S}, \mathcal{A}, p_s, r, \gamma$ are the state space, action space, transition probability, reward function and discounting factor respectively.[3]

[2] To avoid clutter, the conditioning on $\mathbf{x}^*$ will be omitted in the sequel, assuming it's clear from the context.

[3] In sequence prediction, we are only interested in the periodic (finite horizon) case.

Based on the notation, the goal of entropy-regularized RL augments is to learn a policy $\pi(a_t \mid s_t)$ which maximizes the discounted expected future return and causal entropy (Ziebart, 2010), i.e.,

$$\max_{\pi} \sum_t \mathop{\mathbb{E}}_{s_t \sim \rho_s, a_t \sim \pi(\cdot|s_t)} \gamma^{t-1}[r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot \mid s_t))],$$

where $\mathcal{H}$ denotes the entropy and $\alpha$ is a hyper-parameter controlling the relative importance between the reward and the entropy. Intuitively, compared to standard RL, the extra entropy term encourages exploration and promotes multi-modal behaviors. Such properties are highly favorable in a complex environment.

Given an entropy-regularized MDP, for any fixed policy $\pi$, the state-value function $V^\pi(s)$ and the action-value function $Q^\pi$ can be defined as

$$\begin{aligned} V^\pi(s) &= \mathop{\mathbb{E}}_{a \sim \pi(\cdot|s)}[Q^\pi(s,a)] + \alpha \mathcal{H}(\pi(\cdot \mid s)), \\ Q^\pi(s,a) &= r(s,a) + \mathop{\mathbb{E}}_{s' \sim \rho_s}[\gamma V^\pi(s')]. \end{aligned} \tag{9}$$

With the definitions above, it can further be proved (Ziebart, 2010; Schulman et al., 2017) that the optimal state-value function $V^*$, the action-value function $Q^*$ and the corresponding optimal policy $\pi^*$ satisfy the following equations

$$V^*(s) = \alpha \log \sum_{a \in \mathcal{A}} \exp\left(Q^*(s,a)/\alpha\right), \tag{10}$$

$$Q^*(s,a) = r(s,a) + \gamma \mathop{\mathbb{E}}_{s' \sim \rho_s}[V^*(s')], \tag{11}$$

$$\pi^*(a \mid s) = \frac{\exp\left(Q^*(s,a)/\alpha\right)}{\sum_{a' \in \mathcal{A}} \exp\left(Q^*(s,a')/\alpha\right)}. \tag{12}$$

Here, Eqn. (10) and (11) are essentially the entropy-regularized counterparts of the optimal Bellman equations in standard RL. Following previous literature, we will refer to Eqn. (10) and (11) as the optimal *soft* Bellman equations, and the $V^*$ and $Q^*$ as optimal *soft* value functions.

### 3.2 An RL Equivalence of the Token-level RAML

To reveal the connection, it is convenient to define the incremental pay-off

$$r(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) = R(\mathbf{y}_1^t; \mathbf{y}^*) - R(\mathbf{y}_1^{t-1}; \mathbf{y}^*), \tag{13}$$

and the last term of Eqn. (7) as

$$V_R(\mathbf{y}_1^t; \mathbf{y}^*) = \tau \log \sum_{w \in \mathcal{W}} \exp\left(Q_R(\mathbf{y}_1^t, w; \mathbf{y}^*)/\tau\right) \tag{14}$$

Substituting the two definitions into Eqn. (7), the recursion simplifies as

$$Q_R(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) = r(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) + V_R(\mathbf{y}_1^t; \mathbf{y}^*). \tag{15}$$

Now, it is easy to see that the Eqn. (14) and (15), which are derived from the token-level RAML, highly resemble the optimal soft Bellman equations (10) and (11) in entropy-regularized RL. The following Corollary formalizes the connection.

**Corollary 1.** *For any ground-truth pair $(\mathbf{x}^*, \mathbf{y}^*)$, the recursion specified by Eqn. (13), (14) and (15) is equivalent to the optimal soft Bellman equation of a "deterministic" MDP in entropy-regularized reinforcement learning, denoted as $\mathcal{M}_R$, where*

- *the state space $\mathcal{S}$ corresponds to $\mathcal{Y}^-$,*
- *the action space $\mathcal{A}$ corresponds to $\mathcal{W}$,*
- *the transition probability $\rho_s$ is a deterministic process defined by string concatenation*
- *the reward function $r$ corresponds to the incremental pay-off defined in Eqn. (13),*
- *the discounting factor $\gamma = 1$,*
- *the entropy hyper-parameter $\alpha = \tau$,*
- *and a period terminates either when `eos` is emitted or when its length reaches $T$ and we enforce the generation of `eos`.*

*Moreover, the optimal soft value functions $V^*$ and $Q^*$ of the MDP exactly match the $V_R$ and $Q_R$ defined by Eqn. (14) and (15) respectively. The optimal policy $\pi^*$ is hence equivalent to the token-level target distribution $P_{Q_R}$.*

*Proof.* See Appendix A.1. □

The connection established by Corollary 1 is quite inspiring:

- Firstly, it provides a rigorous and generalized view of the connection between RAML and entropy-regularized RL. In the original work, Norouzi et al. (2016) point out RAML can be seen as reversing the direction of $\mathrm{KL}(P_\theta \| P_R)$, which is a sequence-level view of the connection. Now, with the equivalence between the token-level target $P_{Q_R}$ and the optimal $Q^*$, it generalizes to matching the future action values consisting of both the reward and the entropy.
- Secondly, due to the equivalence, if we solve the optimal soft $Q$-function of the corresponding MDP, we directly obtain the token-level target distribution. This hints at a practical algorithm with token-level credit assignment.

- Moreover, since RAML is able to improve upon MLE by injecting entropy, the entropy-regularized RL counterpart of the standard AC algorithm should also lead to an improvement in a similar manner.

## 4 Proposed Algorithms

In this section, we explore the insights gained from Corollary 1 and present two new algorithms for sequence prediction.

### 4.1 Value Augmented Maximum Likelihood

The first algorithm we consider is the token-level extension of RAML, which we have been discussing since §2. As mentioned at the end of §2.2, Proposition 1 only gives an implicit form of $Q_R$, and so is the token-level target distribution $P_{Q_R}$ (Eqn. (3)). However, thanks to Corollary 1, we now know that $Q_R$ is the same as the optimal soft action-value function $Q^*$ of the entropy-regularized MDP $\mathcal{M}_R$. Hence, by finding the $Q^*$, we will have access to $P_{Q_R}$.

At the first sight, it seems recovering $Q^*$ is as difficult as solving the original sequence prediction problem, because solving $Q^*$ from the MDP is essentially the same as learning the optimal policy for sequence prediction. However, it is not true because $Q_R$ (i.e., $P_{Q_R}$) can condition on the correct reference $\mathbf{y}^*$. In contrast, the model distribution $P_\theta$ can only depend on $\mathbf{x}^*$. Therefore, the function approximator trained to recover $Q^*$ can take $\mathbf{y}^*$ as input, making the estimation task much easier. Intuitively, when recovering $Q^*$, we are trying to train an ideal "oracle", which has access to the ground-truth reference output, to decide the best behavior (policy) given any arbitrary (good or not) state.

Thus, following the reasoning above, we first train a parametric function approximator $Q_\phi$ to search the optimal soft action value. In this work, for simplicity, we employ the Soft Q-learning algorithm (Schulman et al., 2017) to perform the policy *optimization*. In a nutshell, Soft Q-Learning is the entropy-regularized version of Q-Learning, an off-policy algorithm which minimizes the mean squared soft Bellman residual according to Eqn. (11). Specifically, given ground-truth pair $(\mathbf{x}^*, \mathbf{y}^*)$, for any trajectory $\mathbf{y} \in \mathcal{Y}$, the training objective is

$$\min_\phi \sum_{t=1}^{|\mathbf{y}|} \left[ Q_\phi(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) - \hat{Q}_\phi(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) \right]^2, \quad (16)$$

where $\hat{Q}_\phi(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) = r(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) + V_\phi(\mathbf{y}_1^t; \mathbf{y}^*)$ is the one-step look-ahead target Q-value, and $V_\phi(\mathbf{y}_1^t; \mathbf{y}^*) = \tau \log \sum_{w \in \mathcal{W}} \exp\left(Q_\phi(\mathbf{y}_1^t, w; \mathbf{y}^*)/\tau\right)$ as defined in Eqn. (10). In the recent instantiation of Q-Learning (Mnih et al., 2015), to stabilize training, the target Q-value is often estimated by a separate slowly updated target network. In our case, as we have access to a significant amount of reference sequences, we find the target network not necessary. Thus, we directly optimize Eqn. (16) using gradient descent, and let the gradient flow through both $Q_\phi(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*)$ and $V_\phi(\mathbf{y}_1^t; \mathbf{y}^*)$ (Baird, 1995).

After the training of $Q_\phi$ converges, we fix the parameters of $Q_\phi$, and optimize the cross entropy $\mathrm{CE}\left(P_{Q_\phi} \| P_\theta\right)$ w.r.t. the model parameters $\theta$, which is equivalent to[4]

$$\min_\theta \mathop{\mathbb{E}}_{\mathbf{y} \sim P_{Q_\phi}} \left[ \sum_{t=1}^{|\mathbf{y}|} \mathrm{CE}\left(P_{Q_\phi}(Y_t \mid \mathbf{y}_1^{t-1}) \| P_\theta(Y_t \mid \mathbf{y}_1^{t-1})\right) \right]. \quad (17)$$

Compared to the of objective of RAML in Eqn. (2), having access to $P_{Q_\phi}(Y_t \mid \mathbf{y}_1^{t-1})$ allows us to provide a distinct token-level target for each conditional factor $P_\theta(Y_t \mid \mathbf{y}_1^{t-1})$ of the model. While directly sampling from $P_R$ is practically infeasible (§2.1), having a parametric target distribution $P_{Q_\phi}$ makes it theoretically possible to sample from $P_{Q_\phi}$ and perform the optimization. However, empirically, we find the samples from $P_{Q_\phi}$ are not diverse enough (§6). Hence, we fall back to the same importance sampling approach (see Appendix B.2) as used in RAML.

Finally, since the algorithm utilizes the optimal soft action-value function to construct the token-level target, we will refer to it as value augmented maximum likelihood (VAML) in the sequel.

### 4.2 Entropy-regularized Actor Critic

The second algorithm follows the discussion at the end of §3.2, which is essentially an actor-critic algorithm based on the entropy-regularized MDP in Corollary 1. For this reason, we name the algorithm entropy-regularized actor critic (ERAC). As with standard AC algorithm, the training process interleaves the evaluation of current policy using the parametric critic $Q_\phi$ and the optimization of the actor policy $\pi_\theta$ given the current critic.

**Critic Training.** The critic is trained to perform policy *evaluation* using the temporal difference

[4] See Appendix A.2 for a detailed derivation.

learning (TD), which minimizes the TD error

$$\min_{\phi} \mathop{\mathbb{E}}_{\mathbf{y}\sim\pi_\theta} \sum_{t=1}^{|\mathbf{y}|} \left[ Q_\phi(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) - \hat{Q}_{\bar{\phi}}(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) \right]^2 \tag{18}$$

where the TD target $\hat{Q}_{\bar{\phi}}$ is constructed based on fixed policy iteration in Eqn. (9), i.e.,

$$\hat{Q}_{\bar{\phi}}(\mathbf{y}_1^{t-1}, y_t; \mathbf{y}^*) = r(\mathbf{y}_1^{t-1}, y_t) + \tau\, \mathcal{H}(\pi_\theta(\cdot \mid \mathbf{y}_1^t)) + \sum_{w\in\mathcal{W}} \pi_\theta(w \mid \mathbf{y}_1^t) Q_{\bar{\phi}}(\mathbf{y}_1^t, w; \mathbf{y}^*). \tag{19}$$

It is worthwhile to emphasize that the objective (18) trains the critic $Q_\phi$ to evaluate the current policy. Hence, it is entirely different from the objective (16), which is performing policy optimization by Soft Q-Learning. Also, the trajectories $\mathbf{y}$ used in (18) are sequences drawn from the actor policy $\pi_\theta$, while objective (16) theoretically accepts any trajectory since Soft Q-Learning can be fully off-policy.[5] Finally, following Bahdanau et al. (2016), the TD target $\hat{Q}_{\bar{\phi}}$ in Eqn. (9) is evaluated using a target network, which is indicated by the bar sign above the parameters, i.e., $\bar{\phi}$. The target network is slowly updated by linearly interpolating with the up-to-date network, i.e., the update is $\bar{\phi} \leftarrow \beta\phi + (1-\beta)\bar{\phi}$ for $\beta$ in $(0, 1)$ (Lillicrap et al., 2015).

We also adapt another technique proposed by Bahdanau et al. (2016), which smooths the critic by minimizing the "variance" of Q-values, i.e.,

$$\min_{\phi} \lambda_{\text{var}} \mathop{\mathbb{E}}_{\mathbf{y}\sim\pi_\theta} \sum_{t=1}^{|\mathbf{y}|} \sum_{w\in\mathcal{W}} \left[ Q_\phi(\mathbf{y}_1^t, w; \mathbf{y}^*) - \bar{Q}_\phi(\mathbf{y}_1^t; \mathbf{y}^*) \right]^2$$

where $\bar{Q}_\phi(\mathbf{y}_1^t; \mathbf{y}^*) = \frac{1}{|\mathcal{W}|} \sum_{w'\in\mathcal{W}} Q_\phi(\mathbf{y}_1^t, w'; \mathbf{y}^*)$ is the mean Q-value, and $\lambda_{\text{var}}$ is a hyper-parameter controlling the relative weight between the TD loss and the smooth loss.

**Actor Training.** Given the critic $Q_\phi$, the actor gradient (to maximize the expected return) is given by the policy gradient theorem of the entropy-regularized RL (Schulman et al., 2017), which has the form

$$\mathop{\mathbb{E}}_{\mathbf{y}\sim\pi_\theta} \sum_{t=1}^{|\mathbf{y}|} \sum_{w\in\mathcal{W}} \nabla_\theta \pi_\theta(w \mid \mathbf{y}_1^{t-1}) Q_\phi(\mathbf{y}_1^{t-1}, w; \mathbf{y}^*) + \tau \nabla_\theta \mathcal{H}(\pi_\theta(\cdot \mid \mathbf{y}_1^{t-1})). \tag{20}$$

Here, for each step $t$, we follow Bahdanau et al. (2016) to sum over the entire symbol set $\mathcal{W}$, instead of using the single sample estimation often seen in RL. Hence, no baseline is employed. It is worth mentioning that Eqn. (20) is *not* simply adding an entropy term to the standard policy gradient as in A3C (Mnih et al., 2016). The difference lies in that the critic $Q_\phi$ trained by Eqn. (18) additionally captures the *entropy from future steps*, while the $\nabla_\theta \mathcal{H}$ term only captures the entropy of the current step.

Finally, similar to (Bahdanau et al., 2016), we find it necessary to first pretrain the actor using MLE and then pretrain the critic before the actor-critic training. Also, to prevent divergence during actor-critic training, it is helpful to continue performing MLE training along with Eqn. (20), though using a smaller weight $\lambda_{\text{mle}}$.

## 5 Related Work

**Task Loss Optimization and Exposure Bias** Apart from the previously introduced RAML, BSO, Actor-Critic (§1), MIXER (Ranzato et al., 2015) also utilizes chunk-level signals where the length of chunk grows as training proceeds. In contrast, minimum risk training (Shen et al., 2015) directly optimizes sentence-level BLEU. As a result, it requires a large number (100) of samples per data to work well. To solve the exposure bias, scheduled sampling (Bengio et al., 2015) adopts a curriculum learning strategy to bridge the training and the inference. Professor forcing (Lamb et al., 2016) introduces an adversarial training mechanism to encourage the dynamics of the model to be the same at training time and inference time. For image caption, self-critic sequence training (SCST) (Rennie et al., 2016) extends the MIXER algorithm with an improved baseline based on the current model performance.

**Entropy-regularized RL** Entropy regularization been explored by early work in RL and inverse RL (Williams and Peng, 1991; Ziebart et al., 2008). Lately, Schulman et al. (2017) establish the equivalence between policy gradients and Soft Q-Learning under entropy-regularized RL. Motivated by the multi-modal behavior induced by entropy-regularized RL, Haarnoja et al. (2017) apply energy-based policy and Soft Q-Learning to continuous domain. Later, Nachum et al. (2017) proposes path consistency learning, which can be seen as a multi-step extension to Soft Q-Learning. More recently, in the domain of simulated control, Haarnoja et al. (2018) also consider the actor critic algorithm under the framework of en-

[5] Different from Bahdanau et al. (2016), we don't use a delayed actor network to collect trajectories for critic training.

tropy regularized reinforcement learning. Despite the conceptual similarity to ERAC presented here, Haarnoja et al. (2018) focuses on continuous control and employs the advantage actor critic variant as in (Mnih et al., 2016), while ERAC follows the Q actor critic as in (Bahdanau et al., 2016).

## 6 Experiments

### 6.1 Experiment Settings

In this work, we focus on two sequence prediction tasks: machine translation and image captioning. Due to the space limit, we only present the information necessary to compare the empirical results at this moment. For a more detailed description, we refer readers to Appendix B and the code[6].

**Machine Translation** Following Ranzato et al. (2015), we evaluate on IWSLT 2014 German-to-English dataset (Mauro et al., 2012). The corpus contains approximately $153K$ sentence pairs in the training set. We follow the pre-processing procedure used in (Ranzato et al., 2015).

Architecture wise, we employ a seq2seq model with dot-product attention (Bahdanau et al., 2014; Luong et al., 2015), where the encoder is a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) with each direction being size 128, and the decoder is another LSTM of size 256. Moreover, we consider two variants of the decoder, one using the input feeding technique (Luong et al., 2015) and the other not.

For all algorithms, the sequence-level BLEU score is employed as the pay-off function $R$, while the corpus-level BLEU score (Papineni et al., 2002) is used for the final evaluation. The sequence-level BLEU score is scaled up by the sentence length so that the scale of the immediate reward at each step is invariant to the length.

**Image Captioning** For image captioning, we consider the MSCOCO dataset (Lin et al., 2014). We adapt the same preprocessing procedure and the train/dev/test split used by Karpathy and Fei-Fei (2015).

The NIC (Vinyals et al., 2015) is employed as the baseline model, where a feature vector of the image is extracted by a pre-trained CNN and then used to initialize the LSTM decoder. Different from the original NIC model, we employ a pre-trained 101-layer ResNet (He et al., 2016) rather than a GoogLeNet as the CNN encoder.

[6] https://github.com/zihangdai/ERAC-VAML

For training, each image-caption pair is treated as an i.i.d. sample, and sequence-level BLEU score is used as the pay-off. For testing, the standard multi-reference BLEU4 is used.

### 6.2 Comparison with the Direct Baseline

Firstly, we compare ERAC and VAML with their corresponding direct baselines, namely AC (Bahdanau et al., 2016) and RAML (Norouzi et al., 2016) respectively. As a reference, the performance of MLE is also provided.

Due to non-neglected performance variance observed across different runs, we run each algorithm for 9 times with different random seeds,[7] and report the average performance, the standard deviation and the performance range (min, max).

**Machine Translation** The results on MT are summarized in the left half of Tab. 1. Firstly, all four advanced algorithms significantly outperform the MLE baseline. More importantly, both VAML and ERAC improve upon their direct baselines, RAML and AC, by a clear margin on average. The result suggests the two proposed algorithms both well combine the benefits of a delicate credit assignment scheme and the entropy regularization, achieving improved performance.

**Image Captioning** The results on image captioning are shown in the right half of Tab. 1. Despite the similar overall trend, the improvement of VAML over RAML is smaller compared to that in MT. Meanwhile, the improvement from AC to ERAC becomes larger in comparison. We suspect this is due to the multi-reference nature of the MSCOCO dataset, where a larger entropy is preferred. As a result, the explicit entropy regularization in ERAC becomes immediately fruitful. On the other hand, with multiple references, it can be more difficult to learn a good oracle $Q^*$ (Eqn. (15)). Hence, the token-level target can be less accurate, resulting in smaller improvement.

### 6.3 Comparison with Existing Work

To further evaluate the proposed algorithms, we compare ERAC and VAML with the large body of existing algorithms evaluated on IWSTL 2014. As a note of caution, previous works don't employ the exactly same architectures (e.g. number of layers, hidden size, attention type, etc.). Despite that,

[7] For AC, ERAC and VAML, 3 different critics are trained first, and each critic is then used to train 3 actors.

| Algorithm | MT (w/o input feeding) | | | MT (w/ input feeding) | | | Image Captioning | | |
|---|---|---|---|---|---|---|---|---|---|
| | Mean | Min | Max | Mean | Min | Max | Mean | Min | Max |
| MLE | $27.01 \pm 0.20$ | 26.72 | 27.27 | $28.06 \pm 0.15$ | 27.84 | 28.22 | $29.54 \pm 0.21$ | 29.27 | 29.89 |
| RAML | $27.74 \pm 0.15$ | 27.47 | 27.93 | $28.56 \pm 0.15$ | 28.35 | 28.80 | $29.84 \pm 0.21$ | 29.50 | 30.17 |
| VAML | $\mathbf{28.16 \pm 0.11}$ | **28.00** | **28.26** | $\mathbf{28.84 \pm 0.10}$ | **28.62** | **28.94** | $\mathbf{29.93 \pm 0.22}$ | **29.51** | **30.24** |
| AC | $28.04 \pm 0.05$ | 27.97 | 28.10 | $29.05 \pm 0.06$ | 28.95 | 29.16 | $30.90 \pm 0.20$ | 30.49 | 31.16 |
| ERAC | $\mathbf{28.30 \pm 0.06}$ | **28.25** | **28.42** | $\mathbf{29.31 \pm 0.04}$ | **29.26** | **29.36** | $\mathbf{31.44 \pm 0.22}$ | **31.07** | **31.82** |

Table 1: Test results on two benchmark tasks. Bold faces highlight the best in the corresponding category.

for VAML and ERAC, we use an architecture that is most similar to the majority of previous works, which is the one described in §6.1 with input feeding.

Based on the setting, the comparison is summarized in Table 2.[8] As we can see, both VAML and ERAC outperform previous methods, with ERAC leading the comparison with a significant margin. This further verifies the effectiveness of the two proposed algorithms.

| Algorithm | BLEU |
|---|---|
| MIXER (Ranzato et al., 2015) | 20.73 |
| BSO (Wiseman and Rush, 2016) | 27.9 |
| Q(BLEU) (Li et al., 2017) | 28.3 |
| AC (Bahdanau et al., 2016) | 28.53 |
| RAML (Ma et al., 2017) | 28.77 |
| VAML | 28.94 |
| ERAC | **29.36** |

Table 2: Comparison with existing algorithms on IWSTL 2014 dataset for MT. All numbers of previous algorithms are from the original work.

### 6.4 Ablation Study

Due to the overall excellence of ERAC, we study the importance of various components of it, hopefully offering a practical guide for readers. As the input feeding technique largely slows down the training, we conduct the ablation based on the model variant *without* input feeding.

Firstly, we study the importance of two techniques aimed for training stability, namely the target network and the smoothing technique (§4.2). Based on the MT task, we vary the update speed $\beta$ of the target critic, and the $\lambda_{\text{var}}$, which controls the strength of the smoothness regularization. The average *validation* performances of different hyper-parameter values are summarized in Tab. 3.

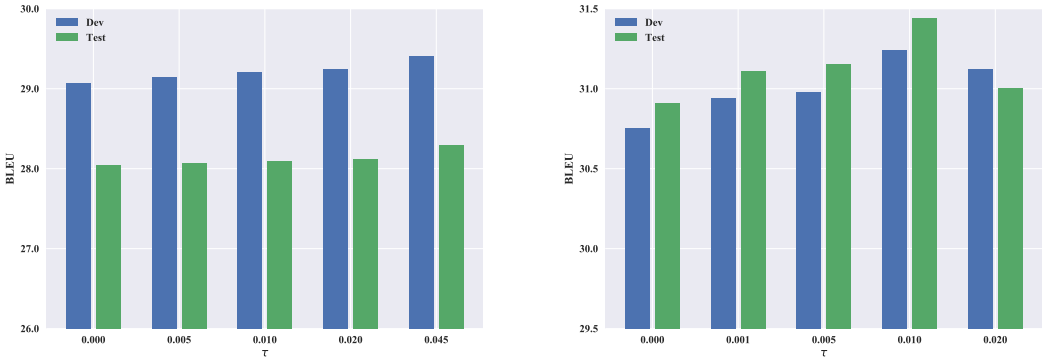
| $\lambda_{\text{var}}$ \ $\beta$ | 0.001 | 0.01 | 0.1 | 1 |
|---|---|---|---|---|
| 0 | 27.91 | $26.27^{\dagger}$ | 28.88 | $27.38^{\dagger}$ |
| 0.001 | **29.41** | 29.26 | 29.32 | 27.44 |

Table 3: Average validation BLEU of ERAC. As a reference, the average BLEU is 28.1 for MLE. $\lambda_{\text{var}} = 0$ means not using the smoothing technique. $\beta = 1$ means not using a target network. $^{\dagger}$ indicates excluding extreme values due to divergence.

- Comparing the two rows of Tab. 3, the smoothing technique consistently leads to performance improvement across all values of $\tau$. In fact, removing the smoothing objective often causes the training to diverge, especially when $\beta = 0.01$ and 1. But interestingly, we find the divergence does not happen if we update the target network a little bit faster ($\beta = 0.1$) or quite slowly ($\beta = 0.001$).
- In addition, even with the smoothing technique, the target network is still necessary. When the target network is not used ($\beta = 1$), the performance drops below the MLE baseline. However, as long as a target network is employed to ensure the training stability, the specific choice of target network update rate does not matter as much. Empirically, it seems using a slower ($\beta = 0.001$) update rate yields the best result.

Next, we investigate the effect of enforcing different levels of entropy by varying the entropy hyper-parameter $\tau$. As shown in Fig. 1, it seems there is always a sweet spot for the level of entropy. On the one hand, posing an over strong en-

[8] For a more detailed comparison of performance together with the model architectures, see Table 7 in Appendix C.




(a) Machine translation

(b) Image captioning

Figure 1: ERAC's average performance over multiple runs on two tasks when varying $\tau$.

tropy regularization can easily cause the actor to diverge. Specifically, the model diverges when $\tau$ reaches $0.03$ on the image captioning task or $0.06$ on the machine translation task. On the other hand, as we decrease $\tau$ from the best value to 0, the performance monotonically decreases as well. This observation further verifies the effectiveness of entropy regularization in ERAC, which well matches our theoretical analysis.

Finally, as discussed in §4.2, ERAC takes the effect of future entropy into consideration, and thus is different from simply adding an entropy term to the standard policy gradient as in A3C (Mnih et al., 2016). To verify the importance of explicitly modeling the entropy from future steps, we compared ERAC with the variant that only applies the entropy regularization to the actor but not to the critic. In other words, the $\tau$ is set to 0 when performing policy evaluating according to Eqn. (4.2), while the $\tau$ for the entropy gradient in Eqn. (20) remains. The comparison result based on 9 runs on test set of IWSTL 2014 is shown in Table 4. As we can see, simply adding a local entropy gradient does not even improve upon the AC. This further verifies the difference between ERAC and A3C, and shows the importance of taking future entropy into consideration.

| **Algorithm** | **Mean** | **Max** |
|---|---|---|
| ERAC | **28.30** $\pm$ **0.06** | **28.42** |
| ERAC w/o Future Ent. | 28.06 $\pm$ 0.05 | 28.11 |
| AC | 28.04 $\pm$ 0.05 | 28.10 |

Table 4: Comparing ERAC with the variant without considering future entropy.

## 7 Discussion

In this work, motivated by the intriguing connection between the token-level RAML and the entropy-regularized RL, we propose two algorithms for neural sequence prediction. Despite the distinct training procedures, both algorithms combine the idea of fine-grained credit assignment and the entropy regularization, leading to positive empirical results.

However, many problems remain widely open. In particular, the oracle Q-function $Q_\phi$ we obtain is far from perfect. We believe the ground-truth reference contains sufficient information for such an oracle, and the current bottleneck lies in the RL algorithm. Given the numerous potential applications of such an oracle, we believe improving its accuracy will be a promising future direction.

## References


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

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

Leemon Baird. 1995. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, Elsevier, pages 30–37.

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

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. 2016. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, pages 4960–4964.

Hal Daumé III and Daniel Marcu. 2005. Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of the 22nd international conference on Machine learning*. ACM, pages 169–176.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. 2017. Reinforcement learning with deep energy-based policies. *arXiv preprint arXiv:1702.08165* .

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290* .

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. pages 770–778.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Po-Sen Huang, Chong Wang, Dengyong Zhou, and Li Deng. 2017. Toward neural phrasebased machine translation .

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. pages 3128–3137.

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

Jiwei Li, Will Monroe, and Dan Jurafsky. 2017. Learning to decode for future success. *arXiv preprint arXiv:1701.06549* .

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* .

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, pages 740–755.

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

Xuezhe Ma, Pengcheng Yin, Jingzhou Liu, Graham Neubig, and Eduard Hovy. 2017. Softmax q-distribution estimation for structured prediction: A theoretical interpretation for raml. *arXiv preprint arXiv:1705.07136* .

Cettolo Mauro, Girardi Christian, and Federico Marcello. 2012. Wit3: Web inventory of transcribed and translated talks. In *Conference of European Association for Machine Translation*. pages 261–268.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*. pages 1928–1937.

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. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529.

Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. 2017. Bridging the gap between value and policy based reinforcement learning. In *Advances in Neural Information Processing Systems*. pages 2772–2782.

Mohammad Norouzi, Samy Bengio, Navdeep Jaitly, Mike Schuster, Yonghui Wu, Dale Schuurmans, et al. 2016. Reward augmented maximum likelihood for neural structured prediction. In *Advances In Neural Information Processing Systems*. pages 1723–1731.

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

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

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. 2016. Self-critical sequence training for image captioning. *arXiv preprint arXiv:1612.00563* .

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685* .

John Schulman, Pieter Abbeel, and Xi Chen. 2017. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440* .

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

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

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, pages 3156–3164.

Ronald J Williams and Jing Peng. 1991. Function optimization using connectionist reinforcement learning algorithms. *Connection Science* 3(3):241–268.

Sam Wiseman and Alexander M Rush. 2016. Sequence-to-sequence learning as beam-search optimization. *arXiv preprint arXiv:1606.02960* .

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*. pages 2048–2057.

Brian D Ziebart. 2010. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. 2008. Maximum entropy inverse reinforcement learning. In *AAAI*. Chicago, IL, USA, volume 8, pages 1433–1438.