Nested-Wasserstein Self-Imitation Learning for Sequence Generation

Ruiyi Zhang[1] **Changyou Chen**[2] **Zhe Gan**[3] **Zheng Wen**[4] **Wenlin Wang**[1] **Lawrence Carin**[1]
[1]Duke University [2]University at Buffalo [3]Microsoft Dynamics 365 AI Research [4]DeepMind
ryzhang@cs.duke.edu

Abstract

Reinforcement learning (RL) has been widely studied for improving sequence-generation models. However, the conventional rewards used for RL training typically cannot capture sufficient semantic information and therefore manifest model bias. Further, the sparse and delayed rewards make RL exploration inefficient. To alleviate these issues, we propose the concept of nested-Wasserstein distance for distributional semantic matching. To further exploit it, a novel nested-Wasserstein self-imitation learning framework is developed, encouraging the model to exploit historical high-reward sequences for enhanced exploration and better semantic matching. Our solution can be understood as approximately executing proximal policy optimization with Wasserstein trust-regions. Experiments on a variety of unconditional and conditional sequence-generation tasks demonstrate the proposed approach consistently leads to improved performance.

1 Introduction

Sequence generation is an important research topic in machine learning, covering a wide range of applications, including machine translation (Bahdanau et al., 2015; Cho et al., 2014; Sutskever et al., 2014), image captioning (Anderson et al., 2017; Vinyals et al., 2015; Xu et al., 2015), and text summarization (Paulus et al., 2017; Rush et al., 2015). Standard sequence generation follows an auto-regressive model design under maximum likelihood estimation (MLE) learning (Huszár, 2015; Sutskever et al., 2014; Wiseman and Rush, 2016). That is, models are trained to maximize the expected log-likelihood of the next word conditioned on its preceding ground-truth partial sentence. However, when testing, the generated partial sequence is fed to the generator to draw the next token. Such a discrepancy between training and testing, commonly known as *exposure bias*, leads to accumulated approximation errors along the sequence-generation trajectory (Bengio et al., 2015; Ranzato et al., 2016).

Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108. Copyright 2020 by the author(s).

To address exposure bias, reinforcement learning (RL) techniques have been introduced (Ranzato et al., 2016). Unlike MLE, which only leverages training examples, RL can also exploit samples drawn from the current policy. Improvements are gained from reinforcing the training towards more-plausible generations, typically based on a user-specified reward function (Ranzato et al., 2016; Yu et al., 2017). However, the manually designed rewards often target specific desirable properties in sequence generation (*e.g.*, matching n-gram overlap between generated sequences and ground-truth references), which unintentionally induces extra bias and is often criticized as a bad proxy for human evaluation (Wang et al., 2018a; Hu et al., 2019). Concerns have also been raised w.r.t. efficient exploration in sequence generation. In existing RL-based methods for sequence generation (Bahdanau et al., 2017; Ranzato et al., 2016; Rennie et al., 2016), all experiences are treated as equivalent. However, merely relying on policy samples to explore often leads to forgetting a high-reward trajectory, unless it can be re-sampled frequently (Liang et al., 2018). This problem becomes severe in the sparse-reward setting in sequence generation, *i.e.*, the reward is only available after the whole sentence is generated.

Motivated by the above observations, we present a novel nested-Wasserstein Self-Imitation Learning (WSIL) framework for sequence generation. Specifically, we propose the nested-Wasserstein distance, a generalization of the Wasserstein distance, and exploit it to measure distance between the behavior policy and the artificial policy defined by the replay buffer, to encourage self-imitation. The nested-Wasserstein distance is well suited for distributional semantic matching between two (sequence) distributions whose samples are still discrete distributions, as in the case of sequence generation. The proposed WSIL is inspired by and derived from the policy optimization with Wasserstein trust-regions (Zhang et al., 2018b). It provides a novel reward function to match the generated sequences with the high-reward sequences in the replay buffer, encouraging distributional semantic matching rather than simple n-gram overlap.

The main contributions of this paper are summarized as follows. (*i*) A novel nested-Wasserstein self-imitation learning framework is developed for sequence generation, *exploiting* historical good explorations for better future *exploration.* (*ii*) A novel nested-Wasserstein distance is introduced for sequence generation via distributional semantic matching, effectively alleviating the model training bias imposed by conventional rewards. (*iii*) Extensive empirical evaluation is performed on both unconditional and conditional text generation tasks, demonstrating consistent performance improvement over existing state-of-the-art approaches.

2 Background

Sequence-generation model We consider the problem of discrete-sequence generation, which learns to generate a sequence $Y = (y_1, \ldots, y_T) \in \mathcal{Y}$ of length T, possibly conditioned on context X. Here each y_t is a token from vocabulary $\mathcal{A}$. Pairs (X, Y) are used for training a sequence-generation model. We are particularly interested in applications to text generation, where Y is a sentence and each y_t is a word. Starting from the initial state $\boldsymbol{s}_0$, a recurrent neural network (RNN) produces a sequence of states $(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_T)$ given an input sequence-feature representation $(e(y_1), \ldots, e(y_T))$, where $e(\cdot)$ denotes a word embedding that maps a token to its d-dimensional feature representation. The states are recursively updated with a function known as the *cell*: $\boldsymbol{s}_t = h_\theta(\boldsymbol{s}_{t-1}, e(y_t))$, where θ denotes the model parameters. Popular implementations include Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and the Gated Recurrent Unit (GRU) (Cho et al., 2014). In order to generate sequence Y^s from a (trained) model, one iteratively applies the following operations:

$$y^s_{t+1} \sim \text{Multi}(\text{softmax}(g(\boldsymbol{s}_t))) \,, \tag{1}$$
$$\boldsymbol{s}_t = h(\boldsymbol{s}_{t-1}, e(y^s_t)) \,, \tag{2}$$

where Multi$(\cdot)$ denotes a multinomial distribution. In conditional generation, $\boldsymbol{s}_0$ is initialized with Enc(X), where Enc$(\cdot)$ encodes the relevant information from the context (Bahdanau et al., 2017; Cho et al., 2014). For unconditional generation, one typically draws $\boldsymbol{s}_0$ from a standard Gaussian distribution.

Sequence generation as an RL problem Sequence generation can be considered as an RL problem with deterministic state transition and sparse reward. It can be formulated as a Markov decision process (MDP) $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, r \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, P is the deterministic environment dynamics and $r(\boldsymbol{s}, y)$ is a reward function. The policy π_θ, parameterized by θ, maps each state $\boldsymbol{s} \in \mathcal{S}$ to a probability distribution over $\mathcal{A}$. The objective is to maximize the expected reward, defined as:

$$J(\pi_\theta) = \mathbb{E}_{Y \sim \pi_\theta}\left[r(Y)\right] = \sum_{t=1}^{T} \mathbb{E}_{(\boldsymbol{s}_t, y_t) \sim \pi_\theta}\left[r(\boldsymbol{s}_t, y_t)\right] \,, \tag{3}$$

where $Y \triangleq (\boldsymbol{s}_1, y_1, \cdots, \boldsymbol{s}_T, y_T)$ is a trajectory from policy π_θ with $y_t \in \mathcal{A}$, and $r(Y)$ represents the reward for a sentence Y, and $r(\boldsymbol{s}_t, y_t)$ is the step-wise reward. RL seeks to learn an optimal policy, that maximizes the expected total reward $J(\pi_\theta)$.

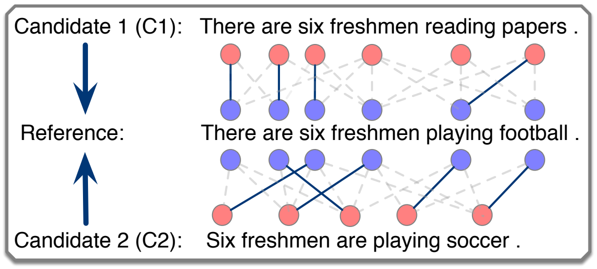
Optimal transport on discrete domains The optimal transport (OT) distance $W_c(\boldsymbol{\mu}, \boldsymbol{\nu})$ is a discrepancy score that measures the distance between two probability distributions $\boldsymbol{\mu}(\cdot)$ and $\boldsymbol{\nu}(\cdot)$ w.r.t. a cost function $c(\cdot, \cdot)$. Specifically, we consider two discrete distributions $\boldsymbol{\mu} \triangleq \sum_{i=1}^{n} u_i \delta_{\boldsymbol{z}_i}$ and $\boldsymbol{\nu} \triangleq \sum_{j=1}^{m} v_j \delta_{\boldsymbol{z}'_j}$ with $\delta_{\boldsymbol{z}}$ the Dirac delta function centered on $\boldsymbol{z}$. The weight vectors $\boldsymbol{u} = \{u_i\}_{i=1}^{n} \in \Delta_n$ and $\boldsymbol{v} = \{v_j\}_{j=1}^{m} \in \Delta_m$ respectively belong to the n and m-dimensional simplex, *i.e.*, $\sum_{i=1}^{n} u_i = \sum_{j=1}^{m} v_j = 1$. Accordingly, Wasserstein distance is equivalent to solving the following minimization problem:

$$\begin{aligned} W_c(\boldsymbol{\mu}, \boldsymbol{\nu}) &= \min_{\mathbf{T} \in \Gamma(\boldsymbol{\mu}, \boldsymbol{\nu})} \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{T}_{ij} \cdot c(\boldsymbol{z}_i, \boldsymbol{z}'_j) \\ &= \min_{\mathbf{T} \in \Gamma(\boldsymbol{\mu}, \boldsymbol{\nu})} \langle \mathbf{T}, \mathbf{C} \rangle \,, \end{aligned} \tag{4}$$

where $\sum_{j=1}^{n} \mathbf{T}_{ij} = \frac{1}{m}$ and $\sum_{i=1}^{m} \mathbf{T}_{ij} = \frac{1}{n}$ are the constraints, $\langle \cdot, \cdot \rangle$ represents the Frobenius dot-product, and $\mathbf{C}$ is the cost matrix defined by $\mathbf{C}_{ij} = c(\boldsymbol{z}_i, \boldsymbol{z}'_j)$. Intuitively, the OT distance is the minimal cost of transporting mass from $\boldsymbol{\mu}$ to $\boldsymbol{\nu}$.

3 Distributional Semantic Matching

We first consider evaluating the sentence from syntactic and semantic perspectives. Conventional metric rewards (*e.g.*, BLEU) can capture the syntactic structure better, where the exact matching of words (or short phases) to the reference sequences is encouraged, which induces strong bias in many cases. As such, we focus on the semantic matching and propose the nested-Wasserstein distance, that defines the distance between two sequence distributions. Nested-Wasserstein distance provides a natural way to manifest semantic matching compared with the conventional rewards used in existing RL-based sequence models. Alternatively, we can train a discriminator to learn the reward model, but empirically it only rewards high-quality generations, even though they may be characterized by mode


Candidate 1 (C1): There are six freshmen reading papers .
Reference: There are six freshmen playing football .
Candidate 2 (C2): Six freshmen are playing soccer .

	BLEU	ROUGE-L	CIDEr	Naive	Wasserstein
C1	**36.8**	**50.0**	**163.7**	**84.1**	76.3
C2	0.0	35.8	55.9	42.5	**80.1**

Table 1: Comparison of different rewards in terms of the sequence-level (higher is better). The top figure illustrates the Wasserstein reward of comparing two candidate sentences with a reference sentence, which will automatically match semantically similar words. Dominant edges are shown in dark blue, determined by the optimal transport matrix $\mathbf{T}$.

collapse (He et al., 2019). This undermines diversity, an important aspect in evaluation.

To better understand the issue, consider the example on sequence matching in Table 1. One can also use a naive way of semantic matching, *i.e.*, measuring a distance between average word embeddings. It is clear that while the first candidate sentence has a similar syntactic structure to the reference, the second candidate sentence is more semantically consistent with the reference. However, popular hard-matching metrics (Papineni et al., 2002; Vedantam et al., 2015) and the naive method consistently indicate the first candidate is a better match to the reference. The above contradiction can be alleviated if the reward metric is more semantic-aware. So motivated, the remainder of this section is devoted to a discussion of design and implementation of Wasserstein rewards. The general idea is to match the semantic features via minimizing the Wasserstein distance between hypothesis sentences and their references in the semantic space. A nested version of the Wasserstein distance arises when integrating the distributional semantic matching into the objective of sequence distribution matching.

Definition 1 ***(Wasserstein Distance between Sequence Pairs)*** *Consider sequence $Y = (y_1, \dots, y_T)$ as a discrete distribution $p_Y = \frac{1}{T}\sum_t \delta_{e(y_t)}$ in the semantic space, with the length-normalized point mass placed at the word embedding, i.e., $\boldsymbol{z}_t = e(y_t)$ of each token y_t from the sequence Y. Given a hypothesis sequence Y w.r.t. a reference sequence Y', we define the Wasserstein distance as $W_c(p_Y, p_{Y'}) \triangleq \min_{\mathbf{T}}\langle \mathbf{T}, \mathbf{C}\rangle$ between p_Y and $p_{Y'}$ with cost $c(\boldsymbol{z}, \boldsymbol{z}')$. When the cosine distance $c_{\cos}(\boldsymbol{z}, \boldsymbol{z}') = 1 - \frac{\boldsymbol{z}^\intercal \boldsymbol{z}'}{\|\boldsymbol{z}\|_2 \|\boldsymbol{z}'\|_2}$ is used as our cost, we define the Wasserstein reward as $r_s(Y, Y') \triangleq \langle \mathbf{T}^*, 1 - \mathbf{C}\rangle$, where $\mathbf{T}^*$ is the optimal transport matrix.*

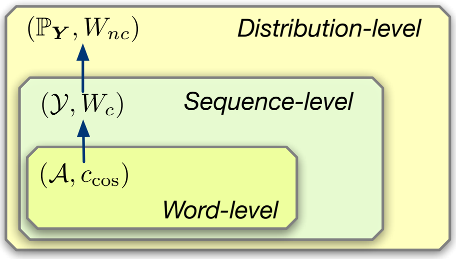

$(\mathbb{P}_{\boldsymbol{Y}}, W_{nc})$
Distribution-level
$(\mathcal{Y}, W_c)$
Sequence-level
$(\mathcal{A}, c_{\cos})$
Word-level

Figure 1: Illustration of nested-Wasserstein distance (W_{nc}) over distributions of sequences ($\mathbb{P}_{\boldsymbol{Y}}$), showing how the distance is defined in a nested manner to measure distance of sequence distributions. $c_{\cos}$ is the word ground metric; W_c is the sequence ground metric.

Nested-Wasserstein distance Our ultimate goal is to measure distance between two policy distributions instead of sequence pairs. Given two sets of sequences from two policies, one aims to incorporate the semantic information between sequences into the distance measure. To this end, we propose the nested-Wasserstein distance in Definition 2. Figure 1 illustrates the nested-Wasserstein distance, considering both word- and sequence-level matching with Wasserstein distance.

Definition 2 (Nested-Wasserstein Distance) *Consider two sets of sequences $\boldsymbol{Y} = \{Y_i\}_{i=1}^K$ and $\boldsymbol{Y}' = \{Y_j'\}_{j=1}^{K'}$ drawn from two sequence distributions $\mathbb{P}_{\boldsymbol{Y}}$ and $\mathbb{P}_{\boldsymbol{Y}'}$, where K and K' are the number of sequences in $\boldsymbol{Y}$ and $\boldsymbol{Y}'$. The nested-Wasserstein distance, denoted as $\mathcal{W}_{nc}(\mathbb{P}_{\boldsymbol{Y}}, \mathbb{P}_{\boldsymbol{Y}'})$, is a metric measuring the distance between $\mathbb{P}_{\boldsymbol{Y}}$ and $\mathbb{P}_{\boldsymbol{Y}'}$, defined in a nested manner:*

$$\mathcal{W}_{nc}(\mathbb{P}_{\boldsymbol{Y}}, \mathbb{P}_{\boldsymbol{Y}'}) \triangleq \min_{T^s} \sum_{i=1}^{K} \sum_{j=1}^{K'} T^s_{ij} W_c(p_{Y_i}, p_{Y_j'}) \ , \quad (5)$$

where $T'_{ij} \geq 0$ satisfies $\sum_i T^s_{ij} = \frac{1}{K}$ and $\sum_j T^s_{ij} = \frac{1}{K'}$; $W_c(\cdot,\cdot)$ denotes the c-Wasserstein distance defined in (4).

Remark 1 *The word "nested" comes from the definition in* (5)*, that essentially consists of two nested levels of Wasserstein distance. The proposed nested-Wasserstein distance brings in the semantic information via the distance measure W_c in the first-level distance. Note that we have omitted the expectation over samples in* (5) *for simplicity, as we essentially use a single set of samples to approximate $\mathcal{W}_{nc}(\cdot,\cdot)$ in algorithms.*

Sample-based estimation of nested-Wasserstein distance Computing the exact Wasserstein distance is computationally intractable (Arjovsky et al., 2017; Genevay et al., 2018; Salimans et al., 2018), let alone the proposed nested-Wasserstein distance. Fortunately, we can employ the recently proposed IPOT algorithm (Xie et al., 2018) to obtain an efficient approximation. Specifically, IPOT considers the following proximal gradient descent to solve the optimal transport matrix $\mathbf{T}$ via iterative optimization, *i.e.*, $\mathbf{T}^{(t+1)} =$

$\arg\min_{\mathbf{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\nu})}\left\{\langle\mathbf{T},\mathbf{C}\rangle+\gamma\cdot\mathbb{D}_{\mathrm{KL}}(\mathbf{T},\mathbf{T}^{(t)})\right\}$, where $1/\gamma > 0$ is the generalized step size and the generalized KL-divergence $\mathbb{D}_{\mathrm{KL}}(\mathbf{T},\mathbf{T}^{(t)}) = \sum_{i,j}\mathbf{T}_{ij}\log\frac{\mathbf{T}_{ij}}{\mathbf{T}^{(t)}_{ij}} - \sum_{i,j}\mathbf{T}_{ij} + \sum_{i,j}\mathbf{T}^{(t)}_{ij}$ is used as the proximity metric. Standard Sinkhorn iterations (Cuturi, 2013) are used to solve the above sub-problem. The IPOT was designed to approximately calculate the standard Wasserstein distance. Here we extend it to calculate the nested-Wasserstein distance by applying IPOT twice in a nested manner, *i.e.*, in the sequence and distribution levels. The full IPOT approach is summarized as Algorithm 2 in Appendix B.

4 Nested-Wasserstein Self-Imitation Learning

Purely adopting the nested-Wasserstein distance as the reward in a standard policy-gradient method is not effective, because the syntactic information is missing. Specifically, we consider sequences generated from a conditional behavior policy $\pi_{\theta,X}$, parameterized by θ with the conditional variable X. For example, in image captioning, each sequence is generated conditioned on a given image. For unconditional generation, the conditional variable is empty. Instead of combining the rewards with different weights (Liu et al., 2017; Pasunuru et al., 2017), we present the nested-Wasserstein Self-Imitation Learning (WSIL) framework, providing a novel way of leveraging both syntactic (metric) and semantic (Wasserstein) information.

The overall idea of the proposed nested-Wasserstein self-imitation learning is to define a Wasserstein trust-region between the current policy (a.k.a. behavior policy) and the artificial policy defined by the replay buffer. Intuitively, the Wasserstein trust-region encourages the self-imitation of historical high-reward sequences, which provides semantic signals to guide training, in addition to the stabilizing effect from trust-region optimization. Furthermore, a replay buffer is used to store high-reward historical sequences, whose induced conditional policy is denoted $\pi_{\mathcal{B},X}$. Our new objective function with a Wasserstein trust-region is defined as:

$$J(\pi_\theta) = \mathbb{E}_{X\sim p_d}\{\mathbb{E}_{Y^s\sim\pi_{\theta,X}}[r(Y^s)] - \lambda\cdot\mathcal{W}_{nc}(\pi_{\theta,X},\pi_{\mathcal{B},X})\}\ , \qquad (6)$$

where $\mathcal{W}_{nc}$ is the nested-Wasserstein distance defined in Definition 2, and $r(\cdot)$ can be a metric reward between Y^s and the ground-truth references $\boldsymbol{Y}$. With a little abuse of notation, but for conciseness, we use π_θ to denote both the policy and the distribution over the sequences. Distinct from classic trust-region policy optimization, which defines the trust region based on KL-divergence (Schulman et al., 2015), WSIL defines the trust region based on the nested-Wasserstein distance between the behavior policy $\pi_{\theta,X}$ and the artificial policy $\pi_{\mathcal{B},X}$. Note when $K = K' = 1$, the nested Wasserstein distance degenerates to the definition of Wasserstein distance between two sequences.

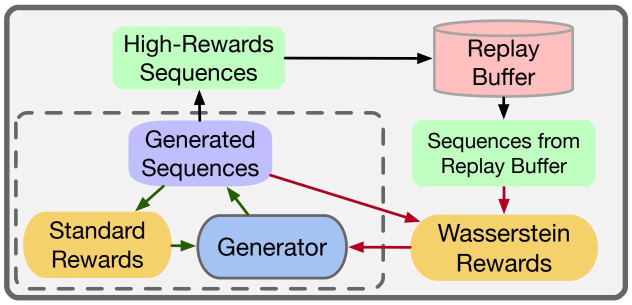


Figure 2: Illustration of the proposed nested-Wasserstein Self-Imitation Learning (WSIL) framework, where Wasserstein self-imitation rewards are defined to encourage the generator to imitate samples from the replay buffer. The standard RL framework is given in the gray dotted box.

Remark 2 ***Unconditional Generation:*** *By considering samples (features) themselves as discrete distributions, we replace the mean square difference over features of sequence pairs, i.e., Euclidean norm, with the Wasserstein distance. Then for the distributions of sequences, we again adopt the Wasserstein distance as in WGAN (Arjovsky et al., 2017) but in the discrete domain. Thus, the Wasserstein distance is defined in a nested manner.*

Remark 3 ***Conditional Generation:*** *We replace the exact matching of sequence pairs with metric rewards in RL training, with the Wasserstein distance. In this case, we are matching two conditional distributions with Wasserstein distance, instead of matching the generated sentence with all reference sentences by average. This is deemed to be a more suitable procedure, as a generated sentence does not necessarily need to match all the references.*

For simplicity, we sometimes omit the first expectation $\mathbb{E}_{X\sim p_d}$. With the proposed nested Wasserstein distance, we propose the Wasserstein self-imitation scheme in (6), as illustrated in Figure 2. We seek to use historical high-reward sequences to define a "*self-imitation*" reward function, which is then combined with the original reward function to update the generator with policy gradient methods. Intuitively, higher self-imitation rewards are achieved when the generated sequences are close to historical high-reward sequences. Thus the generator is guided to perform self imitation and we call this method indirect nested-Wasserstein self-imitation learning (WSIL-I). The word "*indirect*" comes from the mechanism that historical sequences interact with the policy indirectly via the *self-imitation* reward.

WSIL-I incorporates a self-imitation reward, denoted

$r_s(Y^s, Y^b)$, into the objective function. Here Y^b denotes a sample from the replay buffer and Y^s denotes a sample from the current policy. To this end, we replace the Wasserstein distance W_c in the nested-Wasserstein distance with $r_s(Y^s, Y^b)$ in the general objective (6). Specifically, we define the two sets of sample sequences from $\pi_{\theta,X}$ and $\pi_{\mathcal{B},X}$ to be $\boldsymbol{Y}^s \triangleq \{Y_i^s\}_{i=1}^K$ and $\boldsymbol{Y}^b \triangleq \{Y_j^b\}_{j=1}^{K'}$, with sizes of K and K', respectively. Here $Y_i^s \sim \pi_{\theta,X}$ and $Y_j^b \sim \pi_{\mathcal{B},X}$, $\forall j$. $\{Y_i^s\}_{i=1}^K$ and $\boldsymbol{Y}^b$ will be used in calculating the nested-Wasserstein distance. Let $r_{ns}(Y_i^s, \boldsymbol{Y}^b) \triangleq \sum_j T_{ij}^s r_s(Y_i^s, Y_j^b)$ be the nested-Wasserstein reward, with $\mathbf{T}^s = \{T_{ij}^s\}$ the optimal weights in distribution-level. Based on (6), the objective of WSIL-I is adapted to be:

$$J_I(\pi_\theta) \triangleq \mathbb{E}_{X\sim p_d}\mathbb{E}_{Y^s\sim\pi_{\theta,X}}\left[r(Y^s) + \lambda r_{ns}(Y^s, \boldsymbol{Y}^b)\right] , \quad (7)$$

where r is the original RL reward; r_{ns} is the nested-Wasserstein reward. Since not all historically explored samples are helpful for updating the current policy, we only consider a subset of the high-reward sequences when performing self-imitation. Using K trajectories sampled *i.i.d.* from π_θ and introducing a baseline b, the gradient estimate of WSIL-I is expressed as:

$$\begin{aligned}\nabla_\theta J_I(\pi_\theta) \approx -\sum_{k=1}^{K}[(r(Y_k^s) - b)\nabla_\theta \log \pi_\theta(Y_k^s) \\ + \lambda r_{ns}(Y_k^s, \boldsymbol{Y}^b)\nabla_\theta \log \pi_\theta(Y_k^s)] .\end{aligned} \quad (8)$$

In practice, $\mathcal{I}\left[r(Y^b) > r(Y^s)\right]$ will be combined with the nested-Wasserstein rewards, where $\mathcal{I}(\cdot) = 1$ if the condition is satisfied, and it is zero otherwise; b is the baseline to stabilize training. If the reward of a historical high-reward sequence is greater than the current one (*i.e.*, $r(Y^b) > r(Y^s)$), the generator learns to imitate this high-reward sequence. Otherwise, the update based on the historical sequence is not performed due to the $\mathcal{I}(\cdot)$ operator. This encourages the agent to only imitate its good historical explorations. We have also developed an alternative way of implementing (direct) WSIL (WSIL-D), as discussed in the Appendix A. Algorithm 1 describes the general implementation procedure of the WSIL.

Exploration Efficiency The exploration space of MLE is the examples in the training set (Tan et al., 2018), *i.e.,* no exploration is performed in supervised training. In contrast, standard policy optimization (Ranzato et al., 2016) allows the whole exploration space. However, the exploration may become inefficient since it may be too flexible, and some good historical sequences tend to be less explored and imitated due to the sparse rewards. Our proposed WSIL aims to provide more efficient and systematic exploration. It allows whole-space exploration, but re-weights the exploration space to focus more on the exploration that may provide better performance with the Wasserstein trust-region.

Algorithm 1 Nested-Wasserstein Self-Imitation.

```
Require: Generator policy $\pi_\theta$; a sequence dataset $\mathcal{D} = \{Y_{1\dots T}\}_1^N$; a possibly empty condition $\mathcal{X} = \{X\}_1^N$.
  Initialize $\pi_\theta$ and replay buffer $\mathcal{B}$.
  Pretrain generator $\pi_\theta$ with MLE.
  repeat
    Generate $K$ sequences $\boldsymbol{Y}^s = \{Y_k^s\}_{k=1}^K$, where $Y_k^s \sim \pi_\theta$.
    Update replay buffer $\mathcal{B}$ using $\boldsymbol{Y}^s$.
    if Self-Imitation then
      Sample $K'$ sequences $\boldsymbol{Y}^b = \{Y_j^b\}_{j=1}^{K'}$, where $Y_j^b \sim \pi_{\mathcal{B}}$.
      Estimate the OT matrix $\mathbf{T}$ and $\mathbf{T}^s$ via IPOT
      Compute $r_{ns}(Y_k^s, \boldsymbol{Y}^b)$ and update $\pi_\theta$ with (8).
    else
      Update the generator $\pi_\theta$ with (3) using $\boldsymbol{Y}^s$.
    end if
  until Algorithm converges
```

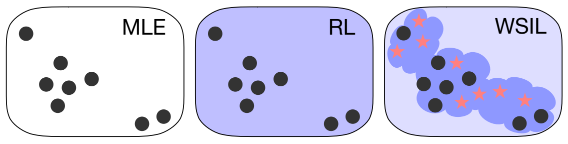


Figure 3: Exploration space of different methods. Circle: ground truth; Star: high-reward sequences.

Increasing Self-Imitation According to the theory of Wasserstein gradient flows (Villani, 2008), $1/\lambda$ can be interpreted as a generalized decaying learning rate. With more explorations, λ becomes larger, and the algorithm should focus more on the self-imitation learning, providing a guideline to balance standard RL training and self-imitation learning; more details are provided in Appendix B. Practically, nested-Wasserstein provides weak supervision focusing on semantic matching, which is reasonable since the historical high-reward sequences contain some noise.

5 Related Work

Optimal transport Kusner et al. (2015) proposed the *word mover's distance* (WMD) and first applied optimal transport (OT) to NLP; OT has also been employed to improve topic modeling (Huang et al., 2016). The transportation cost is usually defined as Euclidean distance, and OT distance is approximated by solving a Kantorovich-Rubinstein dual (Gulrajani et al., 2017) or a less-accurate lower bound (Kusner et al., 2015). Yurochkin et al. (2019) proposed a hierarchical OT representation (Dukler et al., 2019) for documents, but the hierarchy was at the word- and topic-level based on the WMD. Our work considers nested-Wasserstein distance, presenting an efficient IPOT-based implementation for OT distance approximation (Xie et al., 2018), successfully using it to guide sequence generation.

Self-Imitation Learning Experience replay has been widely considered in RL. Deterministic policy gradient (Silver et al., 2014; Lillicrap et al., 2016) performs experience replay, but is limited to continuous control. Actor-critic approaches (Konda and Tsitsiklis, 2000) can also utilize a replay buffer to improve performance. Prioritized experience replay (Schaul et al., 2015) samples trajectories based on the time-difference error, and we adopt it in our implementation. These approaches indiscriminately buffer *all* experiences, while the approach proposed here only buffers high-reward experience. Further, episodic control (Lengyel and Dayan, 2008) can be regarded as an extreme way of exploiting past experience, trying to reproduce its best past decisions, but retrieving states leads to poor efficiency and generalization in testing. Self-imitation learning was first applied in Atari games and Mujoco (Oh et al., 2018; Gangwani et al., 2018), reporting performance improvement w.r.t. sparse rewards. Compared with that work, our solution considers a novel self-imitation learning scheme in the context of sequence generation.

RL for Sequence Generation RL techniques have been explored in detail for sequence generation. For example, a Seq2Seq model can be trained by directly optimizing BLEU/ROUGE scores via policy gradient (Ranzato et al., 2016; Bahdanau et al., 2017). Furthermore, Rennie et al. (2016) baselines the actor with the reward of a greedy-decoding sequence for the REINFORCE method. Model-based RL and hierarchical RL have also been studied for sequence generation (Zhang et al., 2018a; Huang et al., 2019). Further, a learned discriminator (or, critic) can also be used to provide sequence-level guidance. By constructing different objectives, previous work (Yu et al., 2017; Lin et al., 2017; Guo et al., 2017; Fedus et al., 2018) combines the policy-gradient algorithm with the original GAN training procedure. However, mode-collapse problems make the training of these methods challenging. By contrast, we propose the use of self-imitation learning, and maintain a replay buffer to exploit past good explorations.

6 Experiments

We evaluate the proposed method on both unconditional and conditional text-generation tasks, considering standard benchmark datasets. Our approach achieves state-of-the-art results on unconditional text generation and video captioning. We also observed improved performance on image captioning, though relying on much simpler features compared to prior state-of-the-art methods. We also perform ablation studies to understand the improvements brought by self-imitation and Wasserstein rewards individually. Details of the datasets, experimental setup and model architectures are provided in Appendix C.

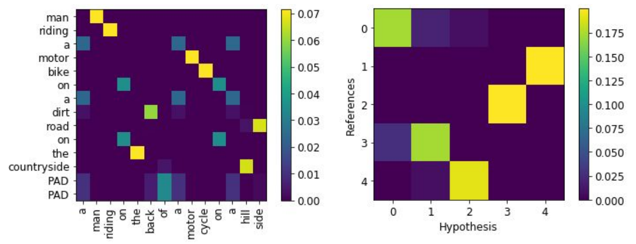
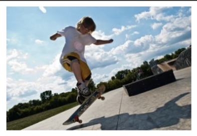


Figure 4: Demonstration of nested-Wasserstein distance at the word-level (left) and sentence-level (right).



	References: 1. youngster on a skateboard trying simple tip up stunt 2. young skateboarder displaying skills on sidewalk near field 3. a person on a skateboard on a street 4. a young boy is performing tricks on a skateboard 5. a little boy that is jumping a skateboard
a man riding a skateboard on a ramp **CIDEr**: 123.86 **Nested-Wasserstein**: 64.57	a man doing a trick on a skateboard on a ramp **CIDEr**: 101.38 **Nested-Wasserstein**: 65.74

Figure 5: An example of image captioning. The right generated sentence is better but given a lower CIDEr.

Implementation Details A few key techniques are required for successful model training. (*i*) The reward from a greedy-decoding sentence is used as the baseline (Rennie et al., 2016) in conditional text generation; in unconditional text generation, a constant baseline is used. (*ii*) A single large replay buffer is maintained for unconditional generation, and multiple replay buffers are maintained for different conditions in conditional generation. (*iii*) For each pair of sentences, the shorter one should be padded to the same length as the longer one for a balanced optimal transport, which is a key implementation technique.

Demonstration of nested-Wasserstein Figure 4 shows the optimal matching in sentence-level ($\mathbf{T}$) and distribution-level ($\mathbf{T}^s$). It is interesting to see that all similar words (*e.g.*, bike and cycle) are matched with each other (higher weights), which cannot be achieved via exact hard-matching metrics. At the distribution-level, we show an example in captioning tasks, where we have five reference and hypothesis sentences. Traditional methods will match a hypothesis sentence to each of the references and average over them; our method performs *distributional semantic matching*, *i.e.,* only matching similar references instead of all of them. For example, the third hypothesis is almost matched with the fifth reference, because they are more similar. This is reasonable, because the references are usually very different, and equivalently matching with all of them is confusing for the generator. As shown in Figure 5, CIDEr focuses more on the locality fluency and equivalent matching with all references, while nested-Wasserstein performs distributional semantic matching. More examples are provided in the Appendix.

Method	Test-BLEU-2	3	4	5	Self-BLEU-2	3	4
MLE (Caccia et al., 2018)	0.902	0.706	0.470	0.392	0.787	0.646	0.485
SeqGAN (Yu et al., 2017)	0.820	0.604	0.361	0.211	0.807	0.577	0.278
RankGAN (Lin et al., 2017)	0.852	0.637	0.389	0.248	0.822	0.592	0.230
TextGAN (Zhang et al., 2017)	0.910	0.728	0.484	0.306	0.806	0.548	0.217
FMGAN (Chen et al., 2018)	0.911	0.782	0.584	0.382	0.834	0.643	0.405
LeakGAN (Guo et al., 2017)	0.922	0.797	0.602	0.416	0.912	0.825	0.689
WSIL-D (ours)	0.917	0.774	0.576	0.393	0.797	0.569	0.284
WSIL-I (ours)	0.922	0.778	0.576	0.396	0.813	0.600	0.326

Table 2: Test-BLEU (↑) and Self-BLEU (↓) scores on Image COCO.

Method	Test-BLEU-2	3	4	5	Self-BLEU-2	3	4
MLE (Caccia et al., 2018)	0.905	0.701	0.464	0.278	0.764	0.522	0.295
SeqGAN (Yu et al., 2017)	0.630	0.354	0.164	0.087	0.728	0.411	0.139
RankGAN (Lin et al., 2017)	0.723	0.440	0.210	0.107	0.672	0.346	0.119
TextGAN (Zhang et al., 2017)	0.777	0.529	0.305	0.161	0.806	0.662	0.448
FMGAN (Chen et al., 2018)	0.913	0.751	0.512	0.315	0.830	0.682	0.427
LeakGAN (Guo et al., 2017)	0.923	0.757	0.546	0.335	0.837	0.683	0.513
SIL-D (ours)	0.875	0.634	0.401	0.243	0.724	0.466	0.256
SIL-I (ours)	0.869	0.633	0.399	0.242	0.710	0.455	0.263
WSIL-D (ours)	0.931	0.736	0.503	0.317	0.795	0.553	0.299
WSIL-I (ours)	0.926	0.726	0.492	0.307	0.815	0.595	0.380

Table 3: Test-BLEU (↑) and Self-BLEU (↓) scores on EMNLP2017 WMT News.

6.1 Unconditional Text Generation

We compare our approach with a number of related RL-based GAN models for unconditional text generation (Guo et al., 2017; Lin et al., 2017; Yu et al., 2017; Zhang et al., 2017). Our implementation is developed based on the LeakGAN model, by incorporating Wasserstein self-imitation learning. All baseline experiments are performed on the texygen platform (Zhu et al., 2018). The corpus-level BLEU score is employed to evaluate the generated sentences. Specifically, we follow the strategy in Yu et al. (2017); Guo et al. (2017) and adopt the BLEU score, referenced by test set (test-BLEU) and themselves (self-BLEU) to evaluate the quality of generated samples. Test-BLEU evaluates the goodness of generated samples, and self-BLEU measures their diversity. The BLEU scores for 1000 generated sentences are averaged to obtain the final score for each model. A good generator should achieve both a high test-BLEU score and a low self-BLEU score. Following previous work (Guo et al., 2017), we test the proposed method on the short and long text generation on Image COCO and EMNLP2017 WMT News datasets. The BLEU scores with different methods are provided in Tables 2 and 3.

Analysis Compared with other methods, LeakGAN, WSIL-D and WSIL-I achieve comparable test-BLEU scores, demonstrating high-quality generated sentences. However, LeakGAN tends to over-fit on training data, leading to much higher (worse) self-BLEU scores. Our proposed methods, by contrast, show good diversity of the generated text with lower self-BLEU scores. Other baselines obtain both low self-BLEU and test-BLEU scores, leading to more random generations.

Ablation Study We conduct ablation studies on EMNLP2017 WMT news to investigate the improvements brought by each part of WSIL. We first test the benefits of using two types of self-imitation schemes. We compare RL training with (*i*) self-imitation (SIL-D and SIL-I), where only a replay buffer and conventional matching (features extracted from a neural network) are employed; and (*ii*) Wasserstein self-imitation (WSIL-D and WSIL-I). Results are shown in Table 3. We observe that the self-imitation strategy, with specific replay buffer construction, can alleviate the discrepancies between reward model bias and conventional rewards (*e.g.*, self-BLEU). Without Wasserstein rewards, we achieve lower self-BLEU at the sacrifice of test-BLEU. When combining with Wasserstein rewards, WSIL-D and WSIL-I show superior performance relative to the baselines. The random generated samples in Appendix D and human evaluations further validate this.

Sweep the Temperature To better evaluate the proposed method, we follow Caccia et al. (2018) to evaluate the trade-off between the quality and diversity. We use the F1-BLEU score as a metric, which considers both quality and diversity, and is defined as the geometry average of BLEU score and 1− Self-BLEU:

$$\text{F1-BLEU} = \frac{2 \times \text{BLEU} \times \text{(1-Self-BLEU)}}{\text{BLEU} + \text{(1-Self-BLEU)}} \,. \quad (9)$$

Figure 6 indicates that WSIL is consistently better than the MLE model on the F1-BLEU-4 score.

Method	BLEU-4	METEOR	ROUGE-L	CIDEr
ED-LG (Yao et al., 2015)	35.2	25.2	-	-
SA-LSTM (Xu et al., 2016)	36.6	25.9	-	-
SCST (Pasunuru et al., 2017)	40.5	28.4	61.4	51.7
MBP (Wang et al., 2018b)	41.3	28.7	61.7	48.0
Our Implementations				
MLE	39.2	27.8	59.8	46.6
MIXER (Ranzato et al., 2016)	40.2	27.9	60.8	50.3
SCST (Rennie et al., 2016)	40.7	27.9	61.6	51.3
WSIL-D	**42.5**	**29.0**	**62.4**	52.1
WSIL-I	41.6	28.4	62.0	**52.2**

Table 4: Video captioning results on MSR-VTT.

Method	BLEU-4	METEOR	ROUGE-L	CIDEr
S & T (Vinyals et al., 2015)	27.7	23.7	-	85.5
OT (Chen et al., 2019)	31.0	24.6	-	94.7
Adaptive (Lu et al., 2017)	33.2	26.6	-	108.5
TD (Anderson et al., 2017)	33.3	26.3	55.3	111.4
Our Implementations				
MLE	28.8	24.4	52.0	91.3
MIXER (Ranzato et al., 2016)	30.8	24.7	52.9	101.2
SCST (Rennie et al., 2016)	**32.1**	25.4	53.9	105.5
WSIL-D	31.8	**25.7**	**54.0**	107.4
WSIL-I	32.0	25.6	53.9	**107.6**

Table 5: Image captioning results on COCO.

Methods	**MLE**	**LeakGAN**	**SIL-D**	**SIL-I**
Human scores	2.97±0.05	2.63±0.05	2.54±0.05	2.55±0.05
Methods	**Real**	**WSIL-D**	**WSIL-I**	-
Human scores	4.11±0.04	**3.49±0.05**	3.41±0.05	-

Table 6: Results of human evaluation.

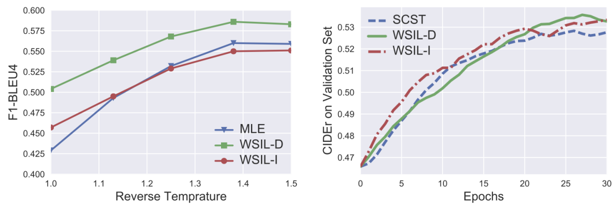


Figure 6: F1-BLEU-4 on sweeping temperature on unconditional generation; CIDEr scores of Video Captioning on validation set.

Human Evaluation Simply relying on the above metrics is not sufficient to evaluate the proposed method (Caccia et al., 2018). Following previous work (Guo et al., 2017), we performed additional human evaluation on the EMNNLP2017 WMT News dataset using Amazon Mechnical Turk. We require all the workers to be native English speakers, with approval rate higher than 95% and at least 100 assignments completed. Previous work has shown higher scores of LeakGAN compared with other baselines (Guo et al., 2017), therefore we mainly focus on the comparison of our methods with LeakGAN. We randomly sampled 200 sentences from each model, and asked 5 different workers to score each sentence on a scale of 1 to 5, considering its readability and meaning. Results are shown in Table 6, which indicates better performance of the proposed WSIL.

6.2 Conditional Text Generation

Video Captioning We conduct experiments on the MSR-VTT dataset (Xu et al., 2016) for video captioning. The MSR-VTT is a large-scale video dataset, consisting of 20 video categories. The dataset was split into 6513 and 3487 clips in the training and testing sets, respectively. Each video is annotated with about 20 captions. For each video, we sample at 3 fps and extract Inception-v4 (Szegedy et al., 2017) features from these sampled frames. We report BLEU-4 (Papineni et al., 2002), CIDEr (Vedantam et al., 2015), and METEOR (Banerjee and Lavie, 2005) scores. Results are summarized in Table 4. Consistent improvements are observed with the WSIL framework. WSIL-D performs slightly better than WSIL-I, both yielding much higher optimized CIDEr and METEOR scores than SCST. This indicates that Wasserstein self-imitation can improve the semantic matching between generated sentences and their references, while achieving reasonable exact-matching-based metric scores.

Image Captioning We consider image captioning using the COCO dataset (Lin et al., 2014), which contains 123,287 images in total, each of which is annotated with at least 5 captions. Following with Karpathy's split (Karpathy and Fei-Fei, 2015), 113,287 images are used for training and 5,000 images are used for validation and testing. We follow the implementation of the SCST approach (Rennie et al., 2016), and use extracted image tags (Gan et al., 2017) as image features (encoder). We report BLEU-k (k from 1 to 4) (Papineni et al., 2002), CIDEr (Vedantam et al., 2015), and METEOR (Banerjee and Lavie, 2005) scores. Results are summarized in Table 5. Compared with the MLE baseline, RL-based methods significantly increase the overall performance under all evaluation metrics. We choose CIDEr as the optimizing metric, since it performs best (Rennie et al., 2016). Our proposed WSIL shows improvement on most metrics compared with the SCST baseline. Examples of generated captions are provided in Appendix E.

7 Conclusions

We have proposed a novel Wasserstein self-imitation learning framework for sequence generation, to alleviate the sparse-reward problem of RL methods, and model-training bias imposed by conventional rewards. This is done by encouraging self imitation and semantic matching in policy learning. Further, our method can be approximately interpreted as policy optimization with Wasserstein trust-regions. Experiments on unconditional and conditional text generation demonstrate consistent performance improvement over strong baselines. For future work, the proposed method has the potential to be applied on other interesting sequence-generation tasks such as program synthesis (Liang et al., 2018).

Acknowledge The authors would like to thank the anonymous reviewers for their insightful comments. The research was supported in part by DARPA, DOE, NIH, NSF and ONR.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and vqa. In *CVPR*, 2017.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, 2017.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. An actor-critic algorithm for sequence prediction. In *ICLR*, 2017.

Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *ACL Workshop*, 2005.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *NeurIPS*, 2015.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. Language gans falling short. *arXiv:1811.02549*, 2018.

Liqun Chen, Shuyang Dai, Chenyang Tao, Haichao Zhang, Zhe Gan, Dinghan Shen, Yizhe Zhang, Guoyin Wang, Ruiyi Zhang, and Lawrence Carin. Adversarial text generation via feature-mover's distance. In *NeurIPS*, 2018.

Liqun Chen, Yizhe Zhang, Ruiyi Zhang, Chenyang Tao, Zhe Gan, Haichao Zhang, Bai Li, Dinghan Shen, Changyou Chen, and Lawrence Carin. Improving sequence-to-sequence learning via optimal transport. In *ICLR*, 2019.

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

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, 2013.

Yonatan Dukler, Wuchen Li, Alex Tong Lin, and Guido Montúfar. Wasserstein of wasserstein loss for learning generative models. In *ICML*, 2019.

William Fedus, Ian Goodfellow, and Andrew M Dai. Maskgan: Better text generation via filling in the _. *ICLR*, 2018.

Zhe Gan, Chuang Gan, Xiaodong He, Yunchen Pu, Kenneth Tran, Jianfeng Gao, Lawrence Carin, and Li Deng. Semantic compositional networks for visual captioning. In *CVPR*, 2017.

Tanmay Gangwani, Qiang Liu, and Jian Peng. Learning self-imitating diverse policies. *arXiv:1805.10309*, 2018.

Aude Genevay, Gabriel Peyré, and Marco Cuturi. Learning generative models with sinkhorn divergences. In *AISTATS*, 2018.

Xiaodong Gu, Kyunghyun Cho, Jungwoo Ha, and Sunghun Kim. Dialogwae: Multimodal response generation with conditional wasserstein auto-encoder. In *ICLR*, 2019.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of Wasserstein GANs. In *NeurIPS*, 2017.

Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long text generation via adversarial training with leaked information. In *AAAI*, 2017.

Junxian He, Daniel Spokoyny, Graham Neubig, and Taylor Berg-Kirkpatrick. Lagging inference networks and posterior collapse in variational autoencoders. In *ICLR*, 2019.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

Junjie Hu, Yu Cheng, Zhe Gan, Jingjing Liu, Jianfeng Gao, and Graham Neubig. What makes a good story? designing composite rewards for visual storytelling. *arXiv preprint arXiv:1909.05316*, 2019.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Controllable text generation. In *ICML*, 2017.

Gao Huang, Chuan Guo, Matt J Kusner, Yu Sun, Fei Sha, and Kilian Q Weinberger. Supervised word mover's distance. In *NeurIPS*, 2016.

Qiuyuan Huang, Zhe Gan, Asli Celikyilmaz, Dapeng Wu, Jianfeng Wang, and Xiaodong He. Hierarchically structured reinforcement learning for topically coherent visual story generation. In *AAAI*, 2019.

Ferenc Huszár. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101*, 2015.

Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015.

Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *NeurIPS*, 2000.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In *ICML*, 2015.

Máté Lengyel and Peter Dayan. Hippocampal contributions to control: the third way. In *NeurIPS*, 2008.

Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc Le, and Ni Lao. Memory augmented policy optimization for program synthesis with generalization. In *NeurIPS*, 2018.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, et al. Continuous control with deep reinforcement learning. In *ICLR*, 2016.

Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, and Ming-Ting Sun. Adversarial ranking for language generation. In *NeurIPS*, 2017.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

Siqi Liu, Zhenhai Zhu, Ning Ye, Sergio Guadarrama, and Kevin Murphy. Improved image captioning via policy gradient optimization of spider. In *ICCV*, 2017.

Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *CVPR*, 2017.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *LREC*, 2018.

Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. In *ICML*, 2018.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2002.

Ramakanth Pasunuru, Mohit Bansal, and Mohit Bansal. Reinforced video captioning with entailment rewards. In *NAACL*, 2017.

Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. In *ICLR*, 2017.

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

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

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

Tim Salimans, Han Zhang, Alec Radford, and Dimitris Metaxas. Improving GANs using optimal transport. In *ICLR*, 2018.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *ICLR*, 2015.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *ICML*, 2015.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *ICML*, 2014.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NeurIPS*, 2014.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2017.

Bowen Tan, Zhiting Hu, Zichao Yang, Ruslan Salakhutdinov, and Eric Xing. Connecting the dots between mle and rl for sequence generation. *arXiv:1811.09740*, 2018.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *CVPR*, 2015.

Cédric Villani. *Optimal transport: old and new*. Springer Science & Business Media, 2008.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015.

Wenlin Wang, Zhe Gan, Hongteng Xu, Ruiyi Zhang, Guoyin Wang, Dinghan Shen, Changyou Chen, and Lawrence Carin. Topic-guided variational autoencoders for text generation. In *NAACL*, 2019.

Xin Wang, Wenhu Chen, Yuan-Fang Wang, and William Yang Wang. No metrics are perfect: Adversarial reward learning for visual storytelling. In *ACL*, 2018a.

Xin Wang, Wenhu Chen, Jiawei Wu, Yuan-Fang Wang, and William Yang Wang. Video captioning via hierarchical reinforcement learning. In *CVPR*, 2018b.

Sam Wiseman and Alexander M Rush. Sequence-to-sequence learning as beam-search optimization. In *EMNLP*, 2016.

Yujia Xie, Xiangfeng Wang, Ruijia Wang, and Hongyuan Zha. A fast proximal point method for Wasserstein distance. In *arXiv:1802.04307*, 2018.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *CVPR*, 2016.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015.

Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *CVPR*, 2015.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, 2017.

Mikhail Yurochkin, Sebastian Claici, Edward Chien, Farzaneh Mirzazadeh, and Justin Solomon. Hierarchical optimal transport for document representation. In *NeurIPS*, 2019.

Ruiyi Zhang, Changyou Chen, Zhe Gan, Wenlin Wang, Liqun Chen, Dinghan Shen, Guoyin Wang, and Lawrence Carin. Sequence generation with guider network. *arXiv preprint arXiv:1811.00696*, 2018a.

Ruiyi Zhang, Changyou Chen, Chunyuan Li, and Lawrence Carin. Policy optimization as wasserstein gradient flows. In *ICML*, 2018b.

Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. Adversarial feature matching for text generation. In *ICML*, 2017.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. Texygen: A benchmarking platform for text generation models. In *SIGIR*, 2018.