An Advantage Actor-Critic Algorithm with Confidence Exploration for Open Information Extraction

Guiliang Liu, Xu Li, Miningming Sun, Ping Li
Cognitive Computing Lab
Baidu Research
No.10 Xibeiwang East Road, Beijing 10085, China
10900 NE 8th ST. Bellevue, WA 98004, USA
gla68@sfu.ca, {lixu13,sunmingming01,liping11}@baidu.com

Abstract

Open Information Extraction (OIE) is a task of generating the structured representations of information from natural language sentences. Recently years, many works have trained an End-to-End OIE extractor based on Sequence-to-Sequence (Seq2Seq) model and applied Reinforce Algorithm to update the model. However, the model performance often suffers from a large training variance and limited exploration. This paper introduces a reinforcement learning framework that enables an Advantage Actor-Critic (AAC) algorithm to update the Seq2Seq model with samples from a novel Confidence Exploration (CE). The AAC algorithm reduces the training variance with a fine-grained evaluation of each individual word. The confidence exploration provides effective training samples by exploring the word at key positions. Empirical evaluations demonstrate the leading performance of our Advantage Actor-Critic algorithm and Confidence Exploration over other comparison methods.

Keywords: *Open Information Extraction, Seq2Seq model, Actor-Critic, Confidence Exploration*

1 Introduction

Open Information Extraction (OIE) is an emerging task of extracting knowledge from natural language sentences. The OIE extractor generates a structured representation of information (e.g., n-ary tuples) from unstructured knowledge stored in source sentences. Unlike Closed Information Extraction (CIE) which applies a finite and predefined natural language corpus, OIE requires an unlexicalized, domain-independent extractor that scales to the entire corpus of a language. Figure 1 shows an example of OIE. The extracted facts are applied as input data for many data mining and natural language processing tasks and benefit lots of downstream applications [9] including event schema induction, text comprehension, and word embedding generation.

Traditional works for OIE mainly apply pattern matching methods [10] or the self-supervised training (with data generated by heuristic patterns) [3]. However, the pattern matching method has limited generalization ability across different domains, and the data generated by the self-supervised training often contain lots of noise.

Source Sentence:

黄石 国家 公园 是 第一个 国家公园， 位于 怀俄明州，
Yellowstone National Park is the first National Park, located in Wyoming;
以其 丰富的 野生动物 种类 和 地热 资源 闻名。
It is famous for its rich wildlife and geothermal resources.

Target Sequence (contains three facts):

(黄石国家公园 $ 是 $ 第一个 国家 公园) (黄石国家公园 $ LOCATE $ 怀俄明州)
(Yellowstone $ is $ the first National Park) (Yellowstone $ LOCATE $ Wyoming)
(黄石国家公园 $ 以 X 闻名 $ 丰富的野生动物 | 地热 资源)
(Yellowstone $ is famous for X $ rich wildlife | geothermal resources)

Figure 1: The task of OIE: Given a source sentence, a model is required to extract facts. Under our sequence prediction setting, the facts are stored in a target sequence, where we apply parenthesis to separate facts and '$' to mark different components of a fact. In this example, we also use symbol 'X' as a place-holder and '|' to divide different objects.

To tackle the problems, many works have applied the supervised learning method to train an encoder-decoder recurrent model that directly generates the predicted facts when given a source sentence. For example, a recent work [17] applied a Sequence to Sequence (Seq2Seq) model and built an End-to-End knowledge extractor named Logician. To improve model performance, many works have applied reinforced algorithms to update the Seq2Seq model. A common method [14] is optimizing the Seq2Seq extractor under the guidance of evaluation scores, but instead of evaluating individual words, the score function often computes expected rewards for the entire prediction sequence. This coarse-grained evaluation generates a large training variance. Furthermore, the limited exploration of their algorithm also undermines the

model's confidence to its predictions, and thus influences the quality of predicted facts.

In this paper, we apply the *Advantage Actor-Critic (AAC)* algorithm to optimize the Seq2Seq model. Unlike the reinforcement algorithm which evaluates the entire target sequence with only one sequence reward, the Critic provides a fine-grained evaluation to each item (e.g., word or symbol like '$') in a prediction sequence. Together with an advantage function from the Actor, our AAC algorithm updates the Seq2Seq model with a smooth gradient estimate and substantially reduces the training variance.

To overcome the problem of insufficient exploration, we propose a *Confidence Exploration (CE)* algorithm to explore candidate words at key positions of predicted sequences. Compared to previous works on OIE, CE generates sufficient training samples to improve both the confidence and the quality of predicted facts. Our empirical evaluation compares state-of-the-art baseline models and demonstrates a leading performance of our model by measuring 1) the quality of predicted facts, 2) the scale of approximation error and 3) the impact of our Confidence Exploration.

Our main contributions are summarized as follows:

- We design a reinforcement learning framework that formulates a Markov Decision Process (MDP) for OIE.
- We introduce an Advantage Actor-Critic (AAC) algorithm to provide a fine-grain evaluation over each predicted word. This algorithm significantly reduces the variances during training (with theoretical proof).
- We propose a Confidence Exploration algorithm that can be easily generalized to the sequence prediction tasks.

In the rest of the paper, Section 2 discusses the related works. We formulate an MDP for the OIE task in Section 3 and describe our AAC algorithm in Section 4. We introduce our Confidence Exploration in Section 5 and present our experiments in Section 6.

2 Related Works

2.1 Open Information Extraction: OIE is a task of directly extracting entity and relation level intermediate structures from Natural Language Sentences, in contrast with the Close Information Extraction (CIE) which identifies instances from a fixed and finite set of natural language corpus. Traditional works applied handcrafted heuristics to extract the required structures rather than a commonly applicable NLP component. The pattern matching method [10] also suffers from the limitation of generalization ability. To tackle these problems, a recent work [15] trained an end-to-end recurrent model and directly transferred source sentences to the entity-relation facts.

2.2 Sequence to Sequence Model: Seq2Seq [18] model applies an encoder-decoder structure. The encoder is a recurrent neural network that takes source sentences as input and produces context features in the form of hidden states. The decoder is another recurrent neural network that takes hidden states from the encoder and generates predicted sequences word-by-word. The Seq2Seq model has been applied to many promising applications [2] including machine translation, image captioning and, parse tree generation.

Many recent works [14, 22, 16] applied the reinforce algorithm to improve the Seq2Seq models. Their models applied the sequence-level update, which produces a high training variance. To tackle the problem, [1] applied the Actor-Critic algorithm to improve the Seq2Seq model. Its critic network evaluates every word with word-level rewards. The issue of insufficient exploration, however, undermines the performance of their Actor-Critic algorithm.

2.3 Exploration for Reinforcement Learning: As an attempt to maximize knowledge gain, an effective exploration significantly improves both training efficiency and efficacy. Apart from the traditional method like the Boltzmann Exploration [6] and Epsilon-Greedy, many recent works have introduced more advanced exploration methods. For example, [13] applied a bootstrapped method to explore the Deep Q network. Previous works are mainly implemented under the game environments with which the agent can interact consistently. For a sequence prediction task (e.g., OIE in this work) , we do not have a concrete environment and reward is only computed after predicting the entire sequence, whose length is rather varied during training. There is a more recent work [8] that built an Monte-Carlo Tree Search (MCTS) to explore the candidate words. Instead of applying external exploration algorithm, their model merges the generation and the exploration process by performing multiple MCTS, which significantly increases the time complexity.

3 Task Formulation

This section introduces our approach to constructing a Markov Decision Process (MDP) for the OIE task and applying the Actor-Critic algorithm to improve a Seq2Seq model.

3.1 Construct MDP for Sequence Prediction: We accomplish the OIE task in the form of sequence prediction [15, 17]. Given a source sentence $X_{1...M} = \{x_1, x_2, \cdots, x_M\}$ containing M words, our model generates a predicted sequence $\hat{Y}_{1...T} = \{\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_T\}$ of T words. The predicted sequence $\hat{Y}_{1...T}$ records a sequence of extracted facts $\{F_1...F_n\}$. In constructing the MDP, we define the source-predicted sequence pair $(\langle X, \hat{Y}_{1..t-1}\rangle)$ as the **state** at predicting step t. **Action** is an item (a word or a symbol such as '$' or '|') to be predicted ($\hat{y}_t$). After determining the action, our agent reaches the **next states** $(X, \hat{Y}_{1..t})$. The Markov

Property is strictly preserved as $(X, \hat{Y}_{1..t})$ models previous knowledge with hidden states from a recurrent network [2].

3.2 Potential-Based Reward Shaping: At predicting step t, our reward function measures the prediction $\hat{Y}_{1..t}$ by how much it resembles the target sequence Y. We introduce our approach to computing such a similarity score. Given a predicted sequence $\hat{Y}_{1..t}$ (contains a set of predicted facts $\{\hat{F}_i\}_1^{n_P}$) and a target sequence Y (records a set of ground truth facts $\{F_j\}_1^{n_G}$), we match each predicted fact to a ground truth fact by finding an optimal assignment to maximize matching similarity. We obtain a set of matched fact pairs $\{(\hat{F}_i, F_j)_l\}_1^{\min(n_P,n_G)}$ and compute a similarity score between the predicted sequence and target sequence:

$$Sim(\hat{Y}_{1..t}, Y) = \sum_{l=1}^{\min(n_P,n_G)} g((\hat{F}_i, F_j)_l) \tag{3.1}$$

where g denotes the Gestalt Pattern Matching function. In this work, similarity score (Sim) is used as both our sequence reward and our evaluation metric, which effectively avoids the mismatch between optimization and evaluation [14]. To project the sequence rewards to word rewards, we apply a potential-based reward shaping [1] with potentials $\Phi(\hat{Y}_{1..t}) = Sim(\hat{Y}_{1..t}, Y)$. This method enables our model to evaluate each predicted word. When generating a predicted sequence, we compute the similarity score for all prefixes and obtain a sequence of similarity scores $[Sim(\hat{Y}_{1..1}, Y), Sim(\hat{Y}_{1..2}, Y), ..., Sim(\hat{Y}_{1..T}, Y)]$. At predicting step t, an **Actor-Critic reward** $r_{ac}(\hat{y}_t, Y)$ is the difference of similarity scores between two consecutive predicting step : $r_{ac}(\hat{y}_t, Y) = Sim(\hat{Y}_{1..t}, Y) - Sim(\hat{Y}_{1..t-1}, Y)$. Replacing a sequence reward at the end of prediction with the shaping (Actor-Critic) rewards $r_{ac}(\hat{y}_t, Y)$ does not influence the optimal policy [11].

3.3 The Framework of Proposed Solution: Many recent works [22, 7] have applied the Seq2Seq model to generate predicted sequences. In improving the model performance, traditional Reinforce Algorithm suffers from a large training variance and limited exploration. To resolve the issues, we introduce an *Advantage Actor-Critic (ACC)* algorithm with *Confidence Exploration*, whose framework is illustrated in Figure 2. Algorithm 1 concludes the training process. The details for our AAC algorithm and our Confidence Exploration are introduced in Section 4 and Section 5, respectively.

4 Advantage Actor-Critic for Sequence Prediction

This section introduces the Advantage Actor-Critic (AAC) algorithm. We also describe a novel Local Vocabulary technique to reduce the action space.

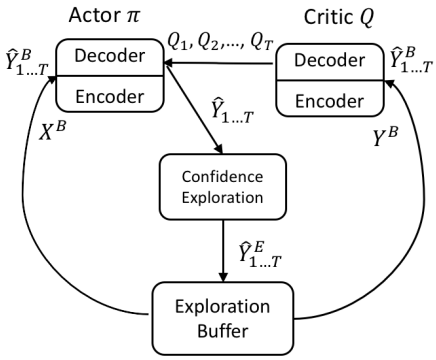

Figure 2: The framework of our Actor-Critic algorithm with Confidence Exploration. Predictions ($\hat{Y}_{1...T}$) from the Actor are sent to our Confidence Exploration algorithm, which produces exploration sequences ($\hat{Y}^e$). We store the sequences in an exploration buffer, where we can sample batches of training data $\langle X^B, \hat{Y}^B_{1...T}, Y^B \rangle$.

Algorithm 1: Advantage Actor-Critic with Confidence Exploration

Require: A Critic $\hat{Q}(a; \hat{Y}_{1...t}, Y)$,
an Actor $\pi(a|\hat{Y}_{1...t}, X)$,
an exploration depth D, and
an exploration width W.
Pre-train the Actor with cross entropy loss.
Pre-train the Critic with gradient computed in Eqn. (4.2).
Initialize the Exploration Buffer B.
while not converge **do**
  Generate $W * D$ exploration sequences $\{\hat{Y}^E_{1...T}\}_1^{W*D}$ with our Confidence-Exploration algorithm.
  Update buffer by $B = B \cup \{\hat{Y}^E_{1...T}\}_1^{W*D}$.
  for inner iteration t **do**
    Randomly select N training samples $\{X^B, Y^B, \hat{Y}^B\}_1^N$ from the buffer B.
    Update $\hat{Q}$ with TD gradient from Eqn. (4.3).
    Update π with gradient from Eqn. (4.4).
  end for
end while

4.1 Model Structure: We introduce the network structure of both our Actor model and our Critic Model.

4.1.1 Critic Model: Similar to [1], we implement the Critic as an attention-based encoder-decoder structure. It takes target sequences Y and predicted sequences $\hat{Y}$ as the input of encoder and decoder, respectively, and produces Q values for the predicted words at every predicting step.

4.1.2 Actor Model: As Figure 2 shows, the Actor takes a source sentence X as input and generates a predicted

sequence $\hat{Y}$ that records a sequence of predicted facts. We implement the Actor with an attention-based Seq2Seq model [18]. The model embeds the source sentences and encodes the embeddings into a N_h dimensional hidden states $H_E = [h_1, \ldots, h_{N_h}]$ by a bi-directional GRU (Gated Recurrent Units) network. We input the H_E to a decoder, which applies another bi-directional GRU network to generate the predicted words. The decoder computes the probability of generating y_t by $\Pr(y_t|y_1, \cdots, y_{t-1}; c_t) = g(h_{t-1}, s_t, c_t)$ where h_{t-1} and s_t are the hidden states from the GRU encoder and the GRU decoder respectively, g is the word generation model, and c_t is the dynamic context vector. We implement the generation model g with the *copy mechanism* [4] which generates a prediction $\hat{y}_t$ by either copying words from the input sentence X or selecting symbols from a set of pre-defined symbols (e.g., '$' or '('). This design allows OIE extractor to select meaningfully information from source sentences (by copying words) and reconstituting the words with a pre-defined format (marked by the symbols). The dynamic context vector c_t applies the *coverage mechanism* [21], which introduces an extra coverage vector for every word in source sentences, in order to remember their individual attention history. This mechanism significantly prevents information loss or redundancy in prediction sequences.

4.2 Training Actor and Critic: We introduce the training details of our Actor-Critic algorithm.

4.2.1 Training the Critic: The pre-training of our Critic applies the Monte Carlo method [19] which uses the cumulative reward $\sum_{\gamma=t}^{T} r_{ac}(\hat{y}_\gamma^n, Y^n)$ to supervise training. The gradient is computed by:

$$\frac{d}{d\theta^{\hat{Q}}} \sum_{n=1}^{N} \sum_{t=1}^{t} \left[\hat{Q}(\hat{y_t^n}; \hat{Y}_{1\ldots t-1}^n, Y^n) - \sum_{\gamma=t}^{T} r_{ac}(\hat{y}_\gamma^n, Y^n) \right]^2 \tag{4.2}$$

The pre-training provides an unbiased evaluation for each action but generates high variance [19], so during updating, we follow [1] and apply Temporal Difference (TD) gradient to smooth the variance by:

$$\begin{aligned} &\frac{d}{d\theta^{\hat{Q}}} \sum_{n=1}^{N} \sum_{t=1}^{T} \left[\hat{Q}(\hat{y_t^n}; \hat{Y}_{1\ldots t-1}^n, Y^n) - q_t^n \right]^2 \\ &\text{where } q_t^n = r_{ac}(\hat{y_t}^n, Y^n) \\ &\qquad + \sum_{a \in \mathcal{A}} \pi(a|\hat{Y}_{1\ldots t}^n, X^n) \hat{Q}(a; \hat{Y}_{1\ldots t}^n, Y^n) \end{aligned} \tag{4.3}$$

4.2.2 Training the Actor: For our Actor, we first pre-train it by minimizing the cross-entropy loss computed with target sequences and predicted sequences. Similar to MIXER [14], the pre-training applies the teacher-forcing technique which directly feeds the target sequences to the decoder. This technique exploits the availability of target sequences but has limited generalization ability to the unseen sequences.

To tackle this issue, in this work, we continuously update our Actor using an advantages function $\mathbb{A}^\pi$ and compute the training gradient by:

$$\sum_{n=1}^{N} \sum_{t=1}^{T} \sum_{a \in \mathcal{A}} \frac{d\pi(a|\hat{Y}_{1\ldots t-1}^n, X^n)}{d\theta^\pi} \mathbb{A}^\pi(a; \hat{Y}_{1\ldots t-1}^n, Y^n) \tag{4.4}$$

where the advantage function of the action a is

$$\begin{aligned} \mathbb{A}^\pi(a; \hat{Y}_{1\ldots t-1}^n, Y^n) =& \hat{Q}(a; \hat{Y}_{1\ldots t-1}^n, Y^n) - \\ & \sum_{b \in \mathcal{A}} \pi(b|\hat{Y}_{1\ldots t-1}^n, Y^n) \hat{Q}(b; \hat{Y}_{1\ldots t-1}^n, Y^n) \end{aligned} \tag{4.5}$$

Compared to the traditional Actor-Critic algorithm which evaluates only on the selected action (1-sample estimate), our Critic provides a fine-grained evaluation for all the candidate actions (words and symbols).

A key benefit of applying advantage function is reducing the training variance. Apart from it, advantage function also provides an unbiased gradient estimate. We formally describe the benefit of the AAC algorithm through the following theorem (see proof in the Supplementary Materials[1]):

THEOREM 4.1. *Let $\mathcal{A}$ be a finite and discrete action space, T be the prediction length and $T \in [1, \inf]$, our gradient function $\sum_{t=1}^{T} \sum_{a \in \mathcal{A}} \pi(a|\hat{Y}_{1\ldots t-1}, X) \mathbb{A}^\pi(a; \hat{Y}_{1\ldots t-1}^n, Y^n)$ provides a unbiased gradient estimate, and the advantage $\mathbb{A}^\pi$ reduces the training variance.*

4.3 Reduce the Action Space by Local Vocabulary Representation: One major issue during implementation is the massive action spaces (vocabularies) at every predicting step t. It is a common challenge for many sequence prediction works [1, 14]. To tackle this problem, we propose a novel local vocabulary representation for the OIE task.

Local vocabulary representation maintains only a local dictionary *for each source sentence*. Applying the copy mechanism [21], our model predicts a word by either copying the words from the input source sentence X or selecting a symbol from a set of keywords (containing symbols like '(', or '$') that mark the structure of predicted facts. A local dictionary stores only $length(sequence) + length(keywords)$ words and *it is guaranteed that our local dictionary contains the word to be extracted*. When global operations (e.g., Word Embedding) are required, we use another "local to global" dictionary to transfer the local words back to global words.

Applying local vocabulary representation, our actor produces only a probability distribution for less than 100 words, and thus the exploration scans a significant smaller action

[1] https://github.com/Guiliang/homepage/blob/master/external-materials/SDM20/SDM2020-Supplementary-Materials.pdf

space at each predicting step, as opposed to 363,851 actions (words) in the original global dictionary. Time and space consumed by transferring local vocabulary dictionaries to the global ones are essentially trivial, compared to the vocabulary size that this technique manages to reduces (more than 300 times). Significant computation and space complexity are also saved for the Critic as the evaluated actions are less than 100 (see Eqn. (4.3)). This local vocabulary representation can be generalized to other information extraction tasks when applying the Seq2Seq model.

5 Confidence Exploration

The trade-off between exploration and exploitation is a key problem for reinforcement learning models. While many models [22, 1] exploit the predicted sequences, their exploration is not sufficient. In this work, we propose a Confidence Exploration to expend the training samples. This section provides two major motivations and introduces the implementation details of our exploration.

5.1 Motivations: We explain how the confidence of predictions influences the quality of predicted facts and introduce two major motivations for our Confidence Exploration.

Score is correlated with Confidence. We observe that a Seq2Seq model tends to extract correct facts when it has large confidence (probability) for its predictions. Figure 3 shows this phenomenon by illustrating the scatter plot of the similarity score (between a predicted fact and a target fact) and the average prediction confidence (for every word in predicted facts). For the sequences with above zero scores, the correlation between similarity scores and prediction confidences is over 0.61. It indicates that to generate the predicted facts that are more similar to the target facts, a model should become more confident in its predictions.

Another important observation is that our model is confident in most of its predicted words ($\Pr(\hat{y}) \geq 0.8$) in general. The small average confidence for some predicted sequences is caused by several words with very low prediction probabilities. By substituting the unconfident predictions with other candidate words and evaluating the new sequences with Critic values $\hat{Q}$, our model learns the consequence of selecting a different combination of words. After updating the predicting policy with the Actor-Critic algorithm, our model will become more confident in its predictions.

Explore to improve the predictions. For a Seq2Seq model, a prediction made at the predicting step t depends on the predictions from previous $t-1$ steps [2]. A formerly predicted word has significant influences on the following predictions. By selecting an optimal word at a key step t', the quality of a predicted sequence will be improved [14].

Figure 4 shows an example of Confidence Exploration. Through replacing the word or symbol having small confidence (e.g., '_' with probability 0.6) with other candidate words (e.g., 'Yellowstone' with probability 0.3), we obtain an exploration sequence that is more close to the target sequence (see Figure 1). Guided by rewards and critic values computed with the exploration sequences, our model learns how to reach the target sequence from the predicted sequence by selecting words at key positions.

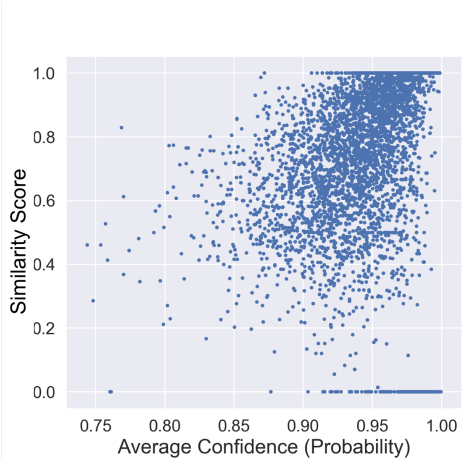

Figure 3: The scatter plot of similarity score v.s. confidence. Each point represents a predicted fact. For a predicted fact, its similarity scores with the target fact are positively correlated to its prediction confidence.

5.2 Implementation: We introduce the approach of implementing our Confidence Exploration for the task of sequence prediction. The inference process of a sequence prediction model (e.g., Seq2Seq model) is iteratively generating a probability distribution for all candidate words and selecting a word $\hat{y}_t$ along the predicting step from 1 to T. We explore a predicted sequence by:

1) Locating the word $\hat{y}_e$ with the smallest probability. This process fits well with MinMax representation: $\min_t \left\{ \max_{\hat{y}_1} \left[\Pr(\hat{y}_1 \in \mathcal{A}|X, \hat{Y}_0) \right], ..., \max_{\hat{y}_T} \left[\Pr(\hat{y}_T \in \mathcal{A}|X, \hat{Y}_{1...T-1}) \right] \right\}$, where at every predicting step t, we select a word $\hat{y}$ that maximizes the prediction probability distribution. After generating the predicted sequence $\hat{y}_1, ..., \hat{y}_T$, we find the word $\hat{y}_e$ with the smallest prediction probability among them. $\hat{y}_e$ is the word to be explored.

2) Substituting the word to be explored ($\hat{y}_e$) with other candidate words. For the depth $d \in 1, 2, .., D$, we repeatedly replace the $\hat{y}_e$ with the candidate word having the d^{th} prediction probability. By freezing the words before and at step e with the teacher forcing technique and dynamically generating the rest of words, we generate D exploration sequences $\{Y^d_{1..T}\}^D_1$. Figure 4 shows an example.

After generating an exploration sequence $Y^d_{1..T}$, we evaluate it by computing either a sequence reward:

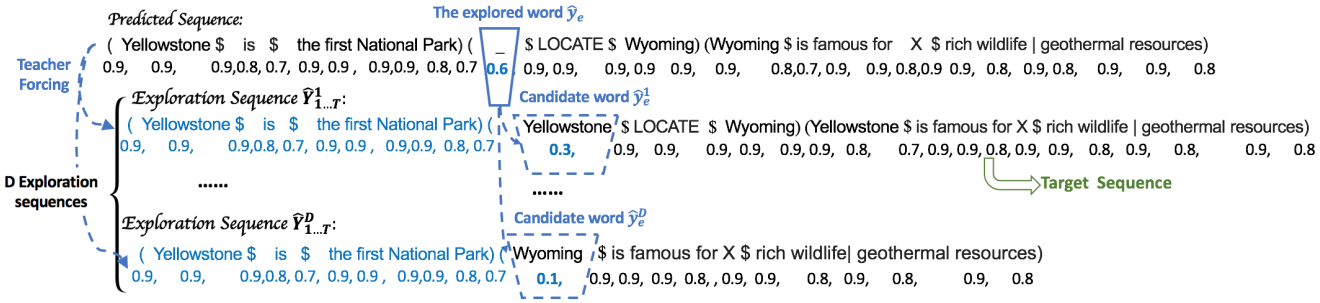

Figure 4: An example of Confidence Exploration. By replacing the word $\hat{y}_e$ with other candidate words and calculating the sequence rewards, we learn how to improve predicted sequence by selecting words at the key positions.

$r(Y^d_{1..T}, Y) = Sim(Y^d_{1..T}, Y)$ or the word-level rewards (Actor-Critic rewards) $\{r_{ac}(y^d_t, Y) = Sim(Y^d_{1..t}, Y) - Sim(Y^d_{1..t-1}, Y)\}^T_{t=1}$. The rewards and the exploration sequences can be applied to update the Seq2Seq Model with the Reinforced or Actor-Critic algorithm.

One can also define an exploration width W. That is, at step 1), instead of finding only the word with the smallest probability, we select the words with the minimum W probabilities and continue the following operations. The exploration sequence sets will be $\{Y^{dw}_{1..T}\}^{W*D}_1$.

6 Empirical Evaluation

In this section, we evaluate the Actor-Critic algorithm and analyze the impact of our Confidence Exploration.

6.1 Experiment Settings

Dataset: A recent work [12] provided a detailed survey of OIE and investigated the available datasets. The most commonly applied datasets are WEB, WIKI, NYT, and PENN which contain sequences from web text, Wikipedia, New York Times Corpus and Penn Treebank respectively. However, they are proposed to evaluate pattern matching methods and record only less than 500 source sentences. Our experiment shows the lack of training data leads to over-fitting in the model.

To address this problem, we apply a recently proposed SAOKE dataset[2] , which contains over 47,000 source-target sequences pairs $\langle X, Y\rangle$ (Y records the facts to be extracted, check Figure 1 for an example). SAOKE applies a unified n-ary tuple ($subject$, $predicate$, $object_1$, $\cdots$, $object_N$) and formulates facts to four categories: 1) Relation: verb/preposition-based n-ary relations between entity mentions; 2) Attribute: nominal attributes for entity mentions; 3) Description: descriptive phrases of entity mentions; 4) Concept: hyponymy and synonymy relations among concepts and instances. Unlike many other system-labeling datasets (e.g., OIE 2016 [15]), [17] manually labeled all the fact with the Crowdsourcing technique. The labeling procedure is under the supervision of the "Completeness" criterion, so the facts intend to cover all the information from the extracted sequences. A previous work [17] has provided a detailed study over the SAOKE dataset and ensured the data validity.

To the best of our knowledge, the SAOKE dataset is the few *valid open-domain source-target formatted dataset*, which has enough training samples and fulfills the requirement of the OIE task and our Seq2Seq extractor.

Training Settings: Following the prior work [17], we split the SAOKE dataset into training data, validation data, and testing data which take up 80%, 10%, and 10% of the entire dataset respectively. To prepare training samples, the facts to be extracted are represented as target sequences Y. Both the Actor and the Critic are trained and updated with the Stochastic Gradient Descent with RMSPROP optimizer, implemented with PaddlePaddle[3] deep learning platform.

Comparison Methods: We compare four state-of-the-art methods that apply a deep model for the task of OIE. The first baseline model Logician [17] is an end-to-end Seq2Seq model trained by the supervised learning method. We also experiment with updating the Seq2Seq model with the Reinforce (RF) algorithm [14]. Another recent work [1] applied the Actor-Critic (AC) algorithm (without advantage function or exploration) to improve the performance of a Seq2Seq model for sequence prediction problem. To compare exploration methods, we follow [22] and use Monte-Carlo Exploration (ME) to generate sequences with a Generative Adversarial Nets. We modify their model for the OIE task and combine it with the Actor-Critic algorithm (ME+AC).

We also experiment with two pattern matching methods. Chinese Open Relation Extraction (CORE) [20] is a system designed for extracting entity-relation triples from text sequences based on a series of NLP techniques, including

[2]http://ai.baidu.com/broad/subordinate?dataset=saoke

[3]https://www.paddlepaddle.org.cn

word segmentation, POS tagging, syntactic parsing, and rules extraction. Another method is an unsupervised OIE model based on Dependency Semantic Normal Forms (DSNF) [5] .

To study the impact of exploration, we experiment different exploration widths $W \in \{2, 4, 6\}$ for our Confidence Exploration (CE+AAC($W2$), CE+AAC($W4$), CE+AAC($W6$)).

6.2 Actor Evaluation: Are we predicting the correct facts? We feed source sentences (from the hold-out testing dataset) to the models and evaluate their performance by measuring the quality of predicted facts. The evaluation metric is the similarity score Sim between predicted sequences and target sequences. Following [17], a predicted fact is labeled as True if the score exceeds the correctness threshold (0.85) and False otherwise, with which we compute the **precision (P)**, **recall (R)** and, **F1-score (F1)** for all the predicted facts. To assess whether the scores of predicted facts from our CE+AAC model have a significant difference with that from the comparison methods, we perform a paired t-tests over all the scores of predicted facts. The null hypothesis is rejected with respective p-values: 1.08E-09, 2.55E-12, 5.15E-11 and 1.43E-11 for Logician, RF, AC, and ME+AC.

Table 1: The Evaluation Results for the Actor.

Model	Search Method	P	R	F1
DSNF	N/A	0.220	0.112	0.148
CORE		0.400	0.1760	0.232
Logician	Greedy Search	0.3803	0.3885	0.3844
RF		0.4015	0.4005	0.4010
AC		0.3834	0.3919	0.3876
ME+AC		0.3812	0.3890	0.3851
CE+AAC(W2)		0.4327	0.4062	0.4190
CE+AAC(W4)		0.4394	**0.4159**	0.4273
CE+AAC(W6)		0.4382	0.4148	0.4262
Logician	Beam Search	0.4699	0.4004	0.4324
RF		0.4873	0.4099	0.4453
AC		0.4702	0.3998	0.4322
ME+AC		0.4705	0.4027	0.4340
CE+AAC(W2)		0.5089	0.4047	0.4509
CE+AAC(W4)		**0.5164**	0.4086	**0.4562**
CE+AAC(W6)		0.5123	0.4063	0.4532

Table 1 presents the evaluation results with both beam search (beam size = 3, determined experimentally) and greedy search. Without exploration, the improvement made by AC itself is very limited, which is smaller than that from RF, because it is difficult for AC to learn a complex word-level Critic without exploration. The result is generally consistent with [1]. After adding exploration, the performance of the Actor is improved. However, compared with our Confidence Exploration, the improvement from Monte-Carlo Exploration is still limited for both beam search and greedy search. In ME, we sample sequences by the probability distribution over every candidate words. Sampling for each sequence is independent. It's highly possible to get the same exploration sequences across different sampling. Whereas our Confidence Exploration forces the model to explore different words at key positions and improves the efficiency and efficacy of exploration. We experiment with different exploration widths and obtain the best result when $W = 4$. It indicates expending exploration width does not always improve the performance. The number of keywords in a predicted sequence is limited and exploring too many irrelevant words will generate noise.

The SAOKE dataset contains some noise from human labeling (sub-optimal or incomplete facts). We find our model manages to overcome the noise and *generates the facts that substantially exceed human labeling.* For example,

- Source: 李白的诗歌对后代产生了极为深远的影响。(Li Bai's poetry has a profound impact on future generations.).
- Human labeled sequence: (李白$ 诗歌$ 对后代产生了极为深远的影响)(LiBai $ poetry $ has a profound impact on future generations).
- Our model prediction: (李白的诗歌$ 对X产生了Y $ 后代$ 极为深远的影响)(Li Bai 's poetry $ has a X on Y $ profound impact $ future generations).

6.3 Critic Evaluation: Are the Q values smooth and accurate? We evaluate the Critic by measuring 1) if it *reduces the Temporal Difference (TD) error*. 2) if it *approximates the expected cumulative rewards* and thus provides accurate evaluations. Following [19], we measure the TD error with the Expected TD Update and compute both the Absolute and the Norm of the Expected TD Update (AEU, NEU). For the approximation performance, a common approach is to compute the difference between Q values from the Critic and real cumulative rewards from input sentences: $\hat{Q} - \sum_t r_{ac}(t)$. We compute the Mean Absolute Difference (MAD) and the Mean Square Difference (MSD). The input sentences are sampled from the hold-out testing dataset.

Table 2: Evaluation Results for the Critic.

Model	AEU	NEU	MAD	MSD
AC	0.0467	0.0084	0.4620	0.7343
ME+AC	0.0437	0.0072	0.4483	0.6618
CE+AAC($W2$)	0.0374	0.0060	**0.3789**	**0.3716**
CE+AAC($W4$)	0.0325	0.0052	0.4352	0.4397
CE+AAC($W6$)	**0.0321**	**0.0050**	0.4416	0.4338

Table 2 presents the evaluation results. Both the TD errors (AEU and NEU) and approximation difference (MAD

and MSD) are significantly reduced by our model (CE+AAC). Compared to Monte-Carlo Exploration, our Confidence Exploration manages to reduce both kinds of errors more effectively. When the exploration width (W) expands, we find the TD errors become smaller while the approximation difference increases. It is because as the exploration scale becomes larger, the model has to spend more steps to finish the same epochs of training, which facilitates the TD learning in reducing the training variance but adds some noise to the Critic. It also explains why our model does not achieve the best performance with the largest exploration width.

6.4 Drill-down Analysis: Has the exploration benefited our model? We investigate the influences of our Confidence Exploration by analyzing 1) for all the predicted facts, if the algorithm manages to increase their scores and confidence. 2) for the predicted facts whose scores exceed over or drop from the correctness threshold (0.85) after adding the exploration, how their confidence fluctuates. This experiment compares the model trained by only the Actor-Critic algorithm (exploration width = 0) and that updated with our Confidence Exploration (width $= 2, 4, 6$).

Table 3: Average scores and confidence for all the predicted facts. AC does not explore, so width $= 0$.

	Beam Search		Greedy Search	
Width	Score	Confidence	Score	Confidence
0	0.6993	0.8581	0.6846	0.8558
2	0.7558	**0.9387**	0.7320	**0.9219**
4	**0.7567**	0.9338	**0.7364**	0.9154
6	0.7566	0.9331	0.7354	0.9156

Table 3 shows the scores and confidence for all predicted facts. We find our Confidence Exploration increases both scores and confidence for the predicted facts, compared to the model without exploration (width $= 0$). The results from beam searches are better than from greedy search. Our model obtains the highest average score when exploration width $= 4$, which is consistent with the results in Table 1.

Table 4: Number of scores exceeding or dropping from the threshold (0.85) after adding our Confidence Exploration.

	Beam Search		Greedy Search	
Width	Exceed	Drop	Exceed	Drop
2	**1301**	**1231**	**1462**	**1300**
4	1129	1030	1323	1078
6	1121	1041	1318	1063

The following experiments study the impact of Confidence Exploration on improving the scores of predicted facts over the correctness threshold. After adding our Confidence Exploration, the scores of many predicted facts are improved over the correctness threshold while some scores (less than the number of improved scores) also drop from this threshold (Table 4). We want to know whether their confidence increases during this process and report the percentage of increasing confidence in Table 5.

Table 5: Percentage of *increasing* confidence when the scores exceed or drop from the threshold (0.85) after adding the Confidence Exploration.

	Beam Search		Greedy Search	
Width	Exceed	Drop	Exceed	Drop
2	**63.10%**	**34.52%**	**58.34%**	**38.92%**
4	60.19%	32.45%	56.89%	33.58%
6	59.68%	31.89%	57.74%	33.99%

Table 5 shows when the Confidence Exploration manages to improve the scores over the threshold, nearly 60% of the confidence increases, but when the scores drop from the threshold, only around 30% of the confidence increases. It indicates a positive relationship between the confidence and scores, which is consistent with the motivation in Section 5. We observe when exploration width $W = 2$, the extractor has larger confidence (Table 3) and improves more facts (Table 4), but it also reduces the scores of many facts and thus, fails to have the leading performance. It indicates that the insufficient exploration (when $W = 2$) leads to many unstable predictions.

6.5 Error Analysis and Limitations This section provides an error analysis over the incorrect predictions from our method ($CE + ACC(W4)$) and introduces the related limitations. The error analysis is implemented by randomly selecting 100 samples among the incorrect predictions and manually summarizing the error in Table 6.

Table 6: Summary of the prediction error.

Error Type	Percentage
Correct relation, *incorrect subjects*	29%
Correct relation, *incorrect objects*	14%
Incomplete/over-extracted relation	22%
Incorrect relation category	9%
Wrong relation	17%
Other, include incomplete sequence	9%

We find more than 40% of the predictions have extracted correct relations but generate wrong subjects or objects. A typical error is the misuse of the placeholder (for some incomplete source sentences, the correct facts use a placeholder ('_') to represent the missing elements). Our model must determine if there is a missing element and fills it with a

placeholder at correct positions. This task is difficult for general OIE extractors and our confidence exploration fails to overcome this difficulty. It requires a more robust exploration method that can run during both training and predicting to evaluate more candidates words or symbols.

7 Conclusion and Future Work

This paper applies the Advantage Actor-Critic algorithm with Confidence Exploration on the OIE task. The AAC algorithm achieves a fine-grained optimization over each individual word and reduces the training variance, and the Confidence Exploration obtains more effective training samples by exploring predicted sequences at key positions. Empirical evaluation demonstrates the performance of our approach. A promising direction of future work is conducting detailed case studies over the wrong predictions and designing a more robust exploration algorithm to handle the difficult samples in error analysis.

Acknowledgement

This paper was initially submitted in December 2018 to NAACL-HLT'19 Conference. The authors sincerely thank all reviewers for their constructive comments.

References

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

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

[3] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open Question Answering over Curated and Extracted Knowledge Bases. In *KDD*, pages 1156–1165, 2014.

[4] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. In *ACL*, pages 1631–1640, 2016.

[5] Shengbin Jia, Shijia E, Maozhen Li, and Yang Xiang. Chinese open relation extraction and knowledge base establishment. *TALLIP*, 17(3):15:1–15:22, 2018.

[6] Levente Kocsis and Csaba Szepesvári. Bandit Based Monte-Carlo Planning. In *ECML*, pages 282–293, 2006.

[7] Xu Li, Mingming Sun, and Ping Li. Multi-agent discussion mechanism for natural language generation. In *AAAI*, pages 6096–6103, 2019.

[8] Guiliang Liu, Xu Li, Jiakang Wang, Sun Mingming, and Ping Li. Extracting Knowledge from Web Text with Monte Carlo Tree Search. In *WWW*, 2020.

[9] Mausam. Open Information Extraction Systems and Downstream Applications. In *IJCAI*, pages 4074–4077, 2016.

[10] Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. Open Language Learning for Information Extraction. In *EMNLP-CoNLL*, pages 523–534, 2012.

[11] Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In *ICML*, pages 278–287, 1999.

[12] Christina Niklaus, Matthias Cetto, André Freitas, and Siegfried Handschuh. A survey on open information extraction. In *COLING*, pages 3866–3878, 2018.

[13] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep Exploration via Bootstrapped DQN. In *NIPS*, pages 4026–4034, 2016.

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

[15] Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. Supervised open information extraction. In *NAACL-HLT*, pages 885–895, 2018.

[16] Mingming Sun, Xu Li, and Ping Li. Logician and Orator: Learning from the Duality between Language and Knowledge in Open Domain. In *EMNLP*, pages 2119–2130, 2018.

[17] Mingming Sun, Xu Li, Xin Wang, Miao Fan, Yue Feng, and Ping Li. Logician: A Unified End-to-End Neural Approach for Open-Domain Information Extraction. In *WSDM*, pages 556–564, 2018.

[18] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. In *NIPS*, pages 3104–3112, 2014.

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

[20] Yuen-Hsien Tseng, Lung-Hao Lee, Shu-Yen Lin, Bo-Shun Liao, Mei-Jun Liu, Hsin-Hsi Chen, Oren Etzioni, and Anthony Fader. Chinese open relation extraction for knowledge acquisition. In *EACL*, pages 12–16, 2014.

[21] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Modeling Coverage for Neural Machine Translation. In *ACL*, pages 76–85, 2016.

[22] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *AAAI*, pages 2852–2858, 2017.