Joint Entity and Event Extraction with Generative Adversarial Imitation Learning (Postprint)

Authors: Zhang, Tongtao, Ji, Heng, Avirup, Sil, Ji, Heng

Date: 2022-11-27T00:00:00+00:00

Abstract

We propose a new framework for entity and event extraction based on generative adversarial imitation learning—an inverse reinforcement learning method using a generative adversarial network (GAN). We assume that instances and labels yield to various extents of difficulty and the gains and penalties (rewards) are expected to be diverse. We utilize discriminators to estimate proper rewards according to the difference between the labels committed by the ground-truth (expert) and the extractor (agent). Our experiments demonstrate that the proposed framework outperforms state-of-the-art methods.

Full Text

Preamble

RESEARCH PAPER

Joint Entity and Event Extraction with Generative Adversarial Imitation Learning

Tongtao Zhang[1], Heng Ji[1]† & Avirup Sil[2]

[1]Computer Science Department, Rensselaer Polytechnic Institute, Troy, New York 12180-3590, USA

[2]IBM Research AI, Armonk, New York 10504-1722, USA

Keywords: Information extraction; Event extraction; Imitation learning; Generative adversarial network

Citation: T. Zhang, H. Ji & A. Sil. Joint entity and event extraction with generative adversarial imitation learning. *Data Intelligence* 1(2019), 99-120. doi: 10.1162/dint_a_{00014}

Received: December 24, 2018; **Revised:** February 11, 2019; **Accepted:** February 19, 2019

ABSTRACT

We propose a new framework for entity and event extraction based on generative adversarial imitation learning—an inverse reinforcement learning method using a generative adversarial network (GAN). We assume that instances and labels yield varying degrees of difficulty, and the gains and penalties (rewards) should be correspondingly diverse. We utilize discriminators to estimate proper rewards according to the difference between the labels committed by the ground-truth (expert) and the extractor (agent). Our experiments demonstrate that the proposed framework outperforms state-of-the-art methods.

1. INTRODUCTION

Event extraction (EE) is a crucial information extraction (IE) task that focuses on extracting structured information (i.e., a structure of event trigger and arguments, "what is happening," and "who or what is involved") from unstructured texts. For example, in the sentence "Masih's alleged comments of blasphemy are punishable by death under Pakistan Penal Code"shown in Figure 1 [Figure 1: see original paper], there is a Sentence event ("punishable") and an Execute event ("death") involving the person entity "Masih." Most event extraction research has been conducted in the context of the 2005 National Institute of Standards and Technology (NIST) Automatic Content Extraction (ACE) sentence-level event mention task [1], which also provides the standard corpus.

The annotation guidelines of the ACE program define an event as a specific occurrence involving participants, often described as a change of state [2]. More recently, the NIST Text Analysis Conference' s Knowledge Base Population (TAC-KBP) community has introduced document-level event argument extraction shared tasks for 2014 and 2015 (KBP EA).

In the last five years, many event extraction approaches have achieved encouraging results by retrieving additional related text documents [3], introducing rich features from multiple categories [4, 5], incorporating relevant information within or beyond context [6, 7, 8, 9], and adopting neural network frameworks [10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20].

However, challenging cases remain. For example, in the sentences "Masih' s alleged comments of blasphemy are punishable by death under Pakistan Penal Code" and "Scott is charged with first-degree homicide for the death of an infant," the word "death" can trigger an Execute event in the former and a Die event in the latter. With similar local information (word embeddings) or contextual features (both sentences include legal events), supervised models pursue a probability distribution resembling that in the training set (e.g., we have overwhelmingly more Die annotations on "death" than Execute) and will label both as a Die event, causing an error in the former instance.

Such mistakes arise from the lack of a mechanism that explicitly deals with wrong and confusing labels. Many multi-classification approaches utilize cross-

entropy loss, which aims to boost the probability of correct labels and usually treats wrong labels equally, merely inhibiting them indirectly. Models are trained to capture features and weights to pursue correct labels but become vulnerable and unable to avoid mistakes when facing ambiguous instances, where the probabilities of confusing and wrong labels are not sufficiently "suppressed." Therefore, exploring information from wrong labels is key to making models robust.

Figure 1 illustrates our framework, which includes a reward estimator based on a generative adversarial network (GAN) to issue dynamic rewards regarding the labels (actions) committed by the event extractor (agent). The reward estimator is trained on the difference between labels from ground truth (expert) and extractor (agent). If the extractor repeatedly misses the Execute label for "death," the penalty (negative reward values) is strengthened. If the extractor makes surprising mistakes—such as labeling "death" as Person or labeling Person "Masih" as a Place role in a Sentence event—the penalty is also strong. For cases where the extractor is correct, simpler cases such as Sentence on "death" will receive smaller gains, while difficult cases like Execute on "death" will be awarded larger reward values.

In this paper, to combat the problems of previous approaches, we propose a dynamic mechanism—inverse reinforcement learning—to directly assess correct and wrong labels in entity and event extraction. We assign explicit scores to cases, or rewards in reinforcement learning (RL) terms. We adopt discriminators from a generative adversarial network (GAN) to estimate reward values. Discriminators ensure the highest reward for ground truth (expert), and the extractor attempts to imitate the expert by pursuing the highest rewards. For challenging cases, if the extractor continues selecting wrong labels, the GAN keeps expanding the margins between rewards for ground-truth labels and (wrong) extractor labels, eventually deviating the extractor from wrong labels.

The main contributions of this paper can be summarized as follows:

- We apply an RL framework to event extraction tasks, and the proposed framework is an end-to-end and pipelined approach that extracts entities and event triggers and determines argument roles for detected entities.
- With inverse RL propelled by the GAN, we demonstrate that a dynamic reward function ensures more optimal performance in a complicated RL task.

2. RELATED WORK

One recent event extraction approach mentioned in the introductory section [18] utilizes the GAN in event extraction. The GAN in that work outputs generated features to regulate the event model away from features leading to errors, while our approach directly assesses mistakes to explore levels of difficulty in labels. Moreover, our approach also covers argument role labeling, while the cited paper does not.

RL-based methods have recently been applied to a few information extraction tasks such as relation extraction, and both relation frameworks from [21, 22] apply RL to entity relation detection with a series of predefined rewards.

We acknowledge that the term "imitation learning" is slightly different from "inverse reinforcement learning." Techniques of imitation learning [23, 24, 25] attempt to map states to expert actions by following demonstrations, which resembles supervised learning, while inverse RL [26, 27, 28, 29, 30] estimates the rewards first and applies them to RL. Reference [31] is an imitation learning application on biomedical event extraction, and no reward estimator is used. We humbly recognize our work as an inverse reinforcement learning approach, although "Generative Adversarial Imitation Learning" (GAIL) is named after imitation learning.

3. TASK AND TERM PRELIMINARIES

In this paper, we follow the schema of Automatic Content Extraction (ACE) [1] to detect the following elements from unstructured natural language data:

Entity: A word or phrase that describes a real-world object such as a person ("Masih" as PER in Figure 1). The ACE schema defines seven types of entities.

Event trigger: The word that most clearly expresses an event (interaction or change of status). The ACE schema defines 33 types of events, such as Sentence ("punishable" in Figure 1) and Execute ("death").

Event argument: An entity that serves as a participant or attribute with a specific role in an event mention. In Figure 1, for example, a PER "Masih" serves as a Defendant in a Sentence event triggered by "punishable."

The ACE schema also includes a dataset—ACE2005—which has been used as a benchmark for information extraction frameworks and will be introduced in Section 6.

For readers unfamiliar with RL, we briefly introduce their counterparts or equivalent concepts in supervised models with RL terms in parentheses: our goal is to train an extractor (agent A) to label entities, event triggers, and argument roles (actions a) in text (environment e). To commit correct labels, the extractor consumes features (state s) and follows the ground truth (expert E). A reward R will be issued to the extractor according to whether it differs from the ground truth and how serious the difference is—as shown in Figure 1, a repeated mistake is definitely more serious. The extractor improves the extraction model (policy π) by pursuing maximized rewards.

Our framework can be briefly described as follows: given a sentence, our extractor scans the sentence and determines the boundaries and types of entities and event triggers using Q-Learning (Section 4.1). Meanwhile, the extractor determines the relations between triggers and entities—argument roles—with policy

gradient (Section 4.2). During training epochs, GANs estimate rewards that stimulate the extractor to pursue the most optimal joint model (Section 5).

4.1 Q-Learning for Entities and Triggers

Entity and trigger detection is often modeled as a sequence labeling problem, where long-term dependency is a core characteristic, and RL is a well-suited method [32].

From an RL perspective, our extractor (agent A) explores the environment—unstructured natural language sentences—when going through sequences and committing labels (actions a) for tokens. When the extractor arrives at the t-th token in the sentence, it observes information from the environment and its previous action a_{t-1} as its current state s_t. When the extractor commits a current action a_t and moves to the next token, it has a new state s_{t+1}. The information from the environment is the token' s context embedding v_t, usually acquired from Bidirectional Long Short-Term Memory (Bi-LSTM) [33] outputs. The previous action a_{t-1} may impose constraints on the current action a_t, e.g., I-ORG does not follow B-PER. With these notations, we have:

$$s_t = \langle v_t, a_{t-1} \rangle \tag{1}$$

To determine the current action a_t, we generate a series of Q-Tables with:

$$Q(s_t, a_t) = f_{sl}(s_t, s_{t-1}, a_{t-1}) \tag{2}$$

where f_{sl}($\cdot$) denotes a function that determines the Q-values using the current state as well as previous states and actions. Then we achieve:

$$a_t = \arg\max_{a_t} Q(s_t, a_t) \tag{3}$$

Equations (2) and (3) suggest that a Recurrent Neural Network (RNN)-based framework that consumes current input and previous inputs and outputs can be adopted, and we use a unidirectional LSTM as in [34]. We have a full pipeline as illustrated in Figure 2 [Figure 2: see original paper].

For each label (action a_t) with regard to s_t, a reward r_t = r(s_t, a_t) is assigned to the extractor (agent). We use Q-Learning to pursue the most optimal sequence labeling model (policy π) by maximizing the expected value of the sum of future rewards E(R_t), where R_t represents the sum of discounted future rewards r_t + cr_{t+1} + c^{2r}_{t+2} + ⋯with a discount factor c, which determines the influence between current and next states.

We utilize the Bellman Equation to update the Q-value with regard to the current assigned label to approximate an optimal model (policy π*):

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + c \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \quad (4)$$

As illustrated in Figure 3 [Figure 3: see original paper], when the extractor assigns a wrong label on the "death" token because the Q-value of Die ranks first, Equation (4) will penalize the Q-value with regard to the wrong label. In later epochs, if the extractor commits the correct label of Execute, the Q-value will be boosted, reinforcing the decision.

We minimize the loss in terms of mean squared error between the original and updated Q-values, denoted as Q(s_t, a_t), and apply backpropagation to optimize the parameters in the neural network.

4.2 Policy Gradient for Argument Roles

After the extractor determines entities and triggers, it takes pairs of one trigger and one entity (argument candidate) to determine whether the latter serves a role in the event triggered by the former.

In this task, for each trigger-argument candidate pair, our extractor observes the context embeddings of the trigger and argument candidate—v_t^{tr} and v_t^{ar}, respectively—as well as the output of another Bi-LSTM consuming the sequence of context embeddings between trigger and argument candidates in the state. The state also includes a representation (one-hot vector) of the entity type of the argument candidate a_t^{ar}, and the event type of the trigger a_t^{tr} also determines the available argument role labels, e.g., an Attack event never has Adjudicator arguments as Sentence events do. With these notations, we have:

$$s_{tr,ar} = \langle v_t^{tr}, v_t^{ar}, f_{ss}(v_{t_{tr}} \dots v_{t_{ar}}), a_t^{ar}, a_t^{tr} \rangle \quad (6)$$

where the superscript tr denotes the trigger, ar denotes the argument candidate, and f_{ss} denotes the sub-sentence Bi-LSTM for the context embeddings between trigger and argument.

We have another ranking table for argument roles:

$$Q(s_{tr,ar}, a_{tr,ar}) = f_{tr,ar}(s_{tr,ar}) \quad (7)$$

where f_{tr,ar} represents a mapping function whose output sizes are determined by the trigger event type a_t^{tr}, e.g., an Attack event has five labels—Attacker, Target, Instrument, Place, and Not-a-role, and the mapping function for the Attack event contains a fully-connected layer with output size of five.

We determine the role with Equation (8):

$$a_{tr,ar} = \arg\max_{a_{tr,ar}} Q(s_{tr,ar}, a_{tr,ar}) \tag{8}$$

We assign a reward r(s_{tr,ar}, a_{tr,ar}) to the extractor. Since there is one step in determining the argument role label, the expected value of R = r(s_{tr,ar}, a_{tr,ar}).

We utilize another RL algorithm—Policy Gradient [36]—to pursue the most optimal argument role labeling performance. We have a probability distribution of argument role labels from the softmax output of Q-values:

$$P(a_{tr,ar}|s_{tr,ar}) = \text{softmax}(Q(s_{tr,ar}, a_{tr,ar})) \tag{9}$$

To update the parameters, we minimize the loss function with Equation (10):

$$\mathcal{L} = -\log P(a_{tr,ar}|s_{tr,ar}) \cdot r(s_{tr,ar}, a_{tr,ar}) \tag{10}$$

From Equation (10) and Figure 4 [Figure 4: see original paper], we observe that when the extractor commits a correct label (Agent for the GPE entity "Pakistan"), the reward encourages P(a_{tr,ar}|s_{tr,ar}) to increase. When the extractor is wrong (e.g., Place for "Pakistan"), the reward will be negative, leading to a decreased P(a_{tr,ar}|s_{tr,ar}).

4.3 Choice of Algorithms

Here we provide a brief clarification on different choices of RL algorithms for the two tasks.

In the sequence labeling task, we do not adopt the policy gradient approach due to the high variance of E(R_t). Specifically, the sum of future rewards R_t should be negative when the extractor chooses a wrong label, but an ill-set reward and discount factor c assignment or estimation may give a positive R_t (often with a small value) and still push up the probability of the wrong action, which is undesirable. While there are variance reduction approaches to constrain R_t, they still require additional estimation, and poor estimation introduces new risks. Q-Learning only requires rewards on current actions r_t, which are relatively easy to constrain.

In the argument role labeling task, determination for each trigger-entity pair consists of only a single step, and R_t is exactly the current reward r. Consequently, the policy gradient approach performs correctly if we ensure negative rewards for wrong actions and positive rewards for correct ones. However, this one-step property impacts the Q-Learning approach: without new positive values from further steps, a small positive reward on current correct labels may make the updated Q-value smaller than those for wrong labels.

5. GENERATIVE ADVERSARIAL IMITATION LEARNING

So far in our paper, the reward values demonstrated in the examples are fixed:

$$r = \begin{cases} c_1 & \text{when } a \text{ is correct} \\ -c_2 & \text{otherwise} \end{cases} \tag{11}$$

and typically we have c_1 > c_2.

This strategy makes the RL-based approach no different from classification approaches with cross-entropy loss in terms of "treating wrong labels equally," as discussed in the introductory section. Moreover, recent RL approaches on relation extraction [21, 22] adopt a fixed setting of reward values with regard to different phases of entity and relation detection based on empirical tuning, which requires additional tuning work when switching to another dataset or schema.

In the event extraction task, entity, event, and argument role labels yield a complex structure with varying difficulties. Errors should be evaluated case by case and from epoch to epoch. In earlier epochs, when parameters in the neural networks are slightly optimized, all errors are tolerable; for example, in sequence labeling, the extractor within the first two or three iterations usually labels most tokens with O labels. As the epoch number increases, the extractor is expected to output more correct labels. However, if the extractor makes repeated mistakes —e.g., persistently labeling "death"as O in the example sentence "···are punishable by death ···" during multiple epochs—or is stuck in difficult cases—e.g., whether the FAC (facility) token "bridges" serves as a Place or Target role in an Attack event triggered by "bombed" in the sentence "US aircraft bombed Iraqi tanks holding bridges···"—a mechanism is required to assess these challenges and correct them with salient and dynamic rewards.

We describe the training approach as a process of the extractor (agent A) imitating the ground truth (expert E). During this process, a mechanism ensures that the highest reward values are issued to correct labels (actions a), including those from both expert E and agent A:

$$\pi_E = \arg\max_{\pi} \mathbb{E}_{a\sim\pi}[R(s,a)] \tag{12}$$

This mechanism is Inverse Reinforcement Learning [26], which estimates the reward first in an RL framework. Equation (12) reveals an adversarial scenario between ground truth and extractor, and GAIL [29], which is based on GAN [37], fits such an adversarial nature.

In the original GAN, a generator produces (fake) data and attempts to confuse a discriminator D, which is trained to distinguish fake data from real data. In our

proposed GAIL framework, the extractor (agent A) substitutes the generator and commits labels to the discriminator D. The discriminator D, now serving as a reward estimator, aims to issue the largest rewards to labels (actions) from the ground truth (expert E) or identical ones from the extractor but provides lower rewards for other/wrong labels.

Rewards R(s,a) and the output of D are now equivalent, and we ensure:

$$D(s, a_E) > D(s, a_A) \tag{13}$$

where s, a_E, and a_A are inputs of the discriminator. In the sequence labeling task, s consists of the context embedding of the current token v_t and a one-hot vector representing the previous action a_{t-1} according to Equation (1). In the argument role labeling task, s comes from the representations of all elements mentioned in Equation (6). a_E is a one-hot vector of the ground-truth label (expert, or "real data"), while a_A denotes the counterpart from the extractor (agent, or "generator"). The concatenated s and a_E form the input for the "real data" channel, while s and a_A build the input for the "generator" channel of the discriminator.

In our framework, due to the different dimensions in the two tasks and event types, we have 34 discriminators (one for sequence labeling and 33 for event argument role labeling with regard to 33 event types). Every discriminator consists of two fully-connected layers with a sigmoid output. The original output of D denotes a probability bounded in [0,1], and we use linear transformation to shift and expand it:

$$R(s, a) = a \cdot D(s, a) - b \tag{14}$$

For example, in our experiments, we set a = 20 and b = 0.5, making R(s,a) [-10,10].

To pursue Equation (13), we minimize the loss function and optimize the parameters in the neural network:

$$\mathcal{L}_D = -\log D(s, a_E) - \log(1 - D(s, a_A)) \tag{15}$$

During training, after feeding the neural networks mentioned in Sections 4.1 and 4.2 with a mini-batch of data, we collect the features (or states s), corresponding extractor labels (agent actions a_A), and ground truth (expert actions a_E) to update the discriminators according to Equation (15). Then we feed features and extractor labels into the discriminators to acquire reward values and train the extractor—or the generator from the GAN's perspective.

Since the discriminators are continuously optimized, if the extractor (generator) makes repeated mistakes or surprising ones (e.g., considering a PER as a Place),

the margin of rewards between correct and wrong labels expands, outputting rewards with larger absolute values. Hence, in the sequence labeling task, the updated Q-values are updated with a more discriminative difference. Similarly, in the argument role labeling task, the P(a|s) also increases or decreases more significantly with larger absolute reward values.

Figure 5 [Figure 5: see original paper] illustrates how we utilize a GAN for reward estimation.

In cases where discriminators are not sufficiently optimized (e.g., in early epochs) and may output undesired values—e.g., negative values for correct actions—we impose a hard margin:

$$R(s,a) = \begin{cases} \max(0.1, R(s,a)) & \text{when } a \text{ is correct} \\ \min(-0.1, R(s,a)) & \text{otherwise} \end{cases} \tag{16}$$

to ensure that correct actions always receive positive reward values and wrong ones receive negative values.

6. EXPLORATION

In the training phase, the extractor selects labels according to the rankings of Q-values in Equations (3) and (8), and GANs issue rewards to update the Q-Tables and policy probabilities. We also adopt an -greedy strategy: we set a probability threshold [0,1) before the extractor commits a label for an instance:

$$\text{action} = \begin{cases} \arg\max_a Q(s,a) & \text{if rand}[0,1] > \epsilon \\ \text{Randomly pick an action} & \text{otherwise} \end{cases} \tag{17}$$

With this strategy, the extractor can explore all possible labels (including correct and wrong ones) and acquire rewards regarding all labels to update the neural networks with richer information.

Moreover, after one step of -greedy exploration, we also force the extractor to commit ground-truth labels and issue it with expert (highest) rewards, updating the parameters accordingly. This additional step is inspired by [38, 39], which combines cross-entropy loss from supervised models with RL loss functions. Such a combination can simultaneously and explicitly encourage correct labels and penalize wrong labels, greatly improving the efficiency of pursuing optimal models.

7.1 Experiment Setup

To evaluate our proposed approach, we utilize ACE2005 documents. To align with state-of-the-art frameworks such as [13, 16], we exclude informal docu-

ments from Conversational Telephone Speech (cts) and UseNet (un). The remaining documents include newswire (nw), weblogs (wl), broadcast news (bn), and broadcast conversations (bc) crawled between 2003 and 2005, fully annotated with 5,272 triggers and 9,612 arguments. To ensure fair comparison with state-of-the-art methods, we follow the splits of training (529 documents with 14,180 sentences), validation (30 documents with 863 sentences), and test (40 documents with 672 sentences) data and adopt the same evaluation criteria:

- An entity (named entities and nominals) is correct if its entity type and offsets find a match in the ground truth.
- A trigger is correct if its event type and offsets find a match in the ground truth.
- An argument is correctly labeled if its event type, offsets, and role find a match in the ground truth.

All aforementioned elements are evaluated using precision (denoted as P in the tables, the ratio of correct instances in system results), recall (denoted as R, the ratio of correct system results in ground-truth annotation), and F1 scores (denoted as F1, the harmonic average of precision and recall).

We use ELMo embeddings [35]. Because ELMo is delivered with built-in Bi-LSTMs, we treat ELMo embedding as context embeddings in Figures 2 and 4. We denote this setting as GAIL-ELMo in the tables.

Moreover, to disentangle the contribution from ELMo embeddings, we also present performance in a non-ELMo setting (denoted as GAIL-W2V), which utilizes the following embedding techniques to represent tokens in the input sentence:

- **Token surface embeddings:** For each unique token in the training set, we maintain a look-up dictionary for embeddings that is randomly initialized and updated during training.
- **Character-based embeddings:** Each character also has a randomly initialized embedding, which is fed into a token-level Bi-LSTM network, and the final output enriches token information.
- **POS embeddings:** We apply Part-of-Speech (POS) tagging on sentences using the Stanford CoreNLP tool [40]. POS tags of tokens also have a trainable look-up dictionary (embeddings).
- **Pre-trained embeddings:** We also acquire embeddings trained from a large, publicly available corpus. These embeddings preserve semantic information of tokens and are not updated during training.

We concatenate these embeddings and feed them into Bi-LSTM networks as demonstrated in Figures 2 and 4. To relieve overfitting issues, we utilize a dropout strategy on input data during training. We intentionally set "UNK" (unknown) masks, which hold entries in the look-up dictionaries of tokens, POS tags, and characters. We randomly mask known tokens, POS tags, and characters in training sentences with the "UNK" mask. We also set an all-zero vector on Word2Vec embeddings of randomly selected tokens.

We tune parameters according to the F1 score of argument role labeling. For Q-Learning, we set a discount factor c = 0.01. For all RL tasks, we set exploration threshold = 0.1. We set all hidden layer sizes (including those in discriminators) and LSTM cell memory sizes (for subsentence Bi-LSTM) as 128. The dropout rate is 0.2. When optimizing parameters in the neural networks, we use stochastic gradient descent (SGD) with momentum, with learning rates starting from 0.02 (sequence labeling), 0.005 (argument labeling), and 0.001 (discriminators). The learning rate decays every 5 epochs with an exponential factor of 0.9; all momentum values are set to 0.9.

For the non-ELMo setting, we set 100 dimensions for token embeddings, 20 for POS embeddings, and 20 for character embeddings. For pre-trained embeddings, we train a 100-dimension Word2Vec [41] model from English Wikipedia articles (January 1, 2017), preserving all tokens with a context window of five from both left and right.

We also implement an RL framework with fixed rewards of $±$5 as a baseline with identical parameters as above. For sequence labeling (entity and event trigger detection), we also set an additional reward value of -50 for B-I errors, namely, an I-label that does not follow a B-label with the same tag name (e.g., I-GPE follows B-PER). We denote these fixed-reward settings as RL-W2V and RL-ELMo.

7.2.1 Entity Extraction Performance

We compare entity extraction performance (including named entities and nominal mentions) with the following state-of-the-art and high-performing approaches:

- **JointIE [42]:** A joint approach that extracts entities, relations, events, and argument roles using structured prediction with rich local and global linguistic features.
- **JointEntityEvent [6]:** An approach that simultaneously extracts entities and arguments with document context.
- **Tree-LSTM [43]:** A Tree-LSTM based approach that extracts entities and relations.
- **KBLSTM [8]:** An LSTM-CRF (conditional random field) hybrid model that applies knowledge base information to sequence labeling.

From Table 1 , we conclude that our proposed method outperforms other approaches, especially with impressively high recall. CRF-based models are applied to sequence labeling tasks because CRF can consider the label on the previous token to avoid mistakes such as appending an I-GPE to a B-PER, but it neglects information from later tokens. Our proposed approach avoids such mistakes by issuing strong penalties (negative rewards with large absolute values), and the Q-values in our sequence labeling sub-framework also consider rewards for later tokens, which significantly enhances prediction performance.

7.2.2 Event Extraction Performance

For event extraction performance with system-predicted entities as argument candidates, besides [42] and [6], we compare our performance with:

- **dbRNN [16]:** An LSTM framework incorporating dependency graph (dependency-bridge) information to detect event triggers and argument roles.

Table 2 demonstrates that our proposed framework performs better than state-of-the-art approaches except for a lower F1 score on argument identification compared to [16]. Reference [16] utilizes Stanford CoreNLP to detect noun phrases and takes the detected phrases as argument candidates, while our argument candidates come from system-predicted entities, and some entities may be missed. However, [16]' s approach misses entity type information, causing many errors in argument role labeling, whereas our argument candidates hold entity types, and our final role labeling performance is better than [16].

Our framework is also flexible enough to consume ground-truth (gold) annotation of entities as argument candidates. We demonstrate performance comparison with the following state-of-the-art approaches on this same setting, besides [16]:

- **JointIE-GT [4]:** Similar to [42], the only difference is that this approach detects arguments based on ground-truth entities.
- **JRNN [13]:** An RNN-based approach that integrates local lexical features.

For this setting, we keep identical parameters (including both trained and preset ones) and network structures used to report our performance in Tables 1 and 2, substituting system-predicted entity types and offsets with ground-truth counterparts.

Table 3 demonstrates that, without any further deliberate tuning, our proposed approach still provides better performance.

7.2.3 Merit of Dynamic Rewards

The statistical results in Tables 1, 2, and 3 demonstrate that dynamic rewards outperform fixed reward settings. As presented in Section 5, fixed reward settings resemble classification methods with cross-entropy loss, which treat errors equally and do not incorporate much information from errors. Consequently, performance is similar to some earlier approaches but does not outperform state-of-the-art methods.

For instances with ambiguity, our dynamic reward function can provide more salient margins between correct and wrong labels. With the identical parameter set as aforementioned, the reward for the wrong Die label is as low as -8.27, while the correct Execute label gains as high as 9.35. Figure 6 [Figure 6: see original paper] illustrates reward curves with regard to epoch numbers. For

easier instances, e.g., the trigger word "arrested" in "···police have arrested ···" have flatter reward values such as 1.24 for Arrest-Jail, -1.53 for None, or -1.37 for Attack, which are sufficient for correct labels.

7.2.4 Impact from Pretrained Embeddings

Scores in Tables 1, 2, and 3 prove that non-ELMo settings already outperform state-of-the-art methods, confirming the advantage and contribution of our GAIL framework. Moreover, despite an insignificant drop in fixed reward settings, we agree that ELMo is a good replacement for a combination of word, character, and POS embeddings. The only shortcoming, according to our empirical practice, is that ELMo requires a huge amount of GPU memory, and the training procedure is slow (even though we do not update the pre-trained parameters during our training phase).

7.3 Remaining Errors

Score losses mainly come from missed trigger words and arguments. For instance, the End-Position trigger "sack" is missed because it is considered informal for expressing an End-Position event, and there are no similar tokens in the training data or pre-trained embeddings. Another error example is due to informal expression, e.g., "I miss him to death," where "death" does not trigger any event, while our system mistakenly detects it as a Die event. Since most training sentences are formal writing, expressions from oral speeches that are usually informal may cause errors.

We also notice some special errors caused by biased annotation. In the sentence "Bush invites his 'good friend' Putin to his weekend 'retreat' outside Camp David in Washington in September," the FAC (facility) entity "retreat" is mistakenly labeled as a trigger word of Transport. In the training data, all "retreat" tokens (or "retreated," "retreating," "retreats") are labeled as Transport; however, this "retreat" means a facility which is "a quiet or secluded place for relaxation." We also notice that the reward value for FAC (correct label) is -2.73 and for Transport (wrong label) is 3.13, which contradicts the expectation that correct labels come with higher reward values. This error implies that our approach still requires exposure to all possible labels to acknowledge and explore ambiguous and possible actions.

8. CONCLUSIONS AND FUTURE WORK

In this paper, we propose an end-to-end entity and event extraction framework based on inverse reinforcement learning. Experiments demonstrate that performance benefits from dynamic reward values estimated from discriminators in a GAN, and we also demonstrate the performance of recent embedding work. In the future, besides releasing the source code, we will attempt to interpret the dynamics of these rewards with regard to instances so that researchers and event extraction system developers can better understand and explore the algorithm

and remaining challenges. Our future work also includes using cutting-edge approaches such as BERT [44] and exploring joint models to alleviate the impact of upstream errors in the current pipelined framework.

AUTHOR CONTRIBUTIONS

T. Zhang (zhangt13@rpi.edu) contributed to the design and implementation of the research. All authors—T. Zhang, H. Ji (jih@rpi.edu), and Avirup Sil (avi@us.ibm.com)—contributed to the analysis of results and writing of the manuscript.

ACKNOWLEDGEMENTS

This work was supported by the US Air Force (No. FA8650-17-C-7715), DARPA AIDA Program (No. FA8750-18-2-0014), and US ARL NS-CTA (No. W911NF-09-2-0053). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Government. The US Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

REFERENCES

[1] C. Walker, S. Strassel, J. Medero, & K. Maeda. ACE 2005 multilingual training corpus. Philadelphia: Linguistic Data Consortium, 2006. isbn: 1-58563-376-3.

[2] Linguistic Data Consortium. ACE (Automatic Content Extraction) English Annotation Guidelines for events. Available at: https://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/eng events-guidelines-v5.4.3.pdf.

[3] Z. Song, A. Bies, S. Strassel, T. Riese, J. Mott, J. Ellis, J. Wright, ···& X. Ma. From light to rich ERE: Annotation of entities, relations, and events. In: *Proceedings of the 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, 2015, pp. 89–98. doi: 10.3115/v1/W15-0812.

[4] Q. Li, H. Ji, & L. Huang. Joint event extraction via structured prediction with global features. In: *Proceedings of 2013 Annual Meeting of the Association for Computational Linguistics*, 2013, pp. 73–82. Available at: http://aclweb.org/anthology/P/P13/P13-1008.pdf.

[5] T. Zhang, S. Whitehead, H. Zhang, H. Li, J. Ellis, L. Huang, W. Liu, ···& S.-F. Chang. Improving event extraction via multimodal integration. In: *Proceedings of the 2017 ACM on Multimedia Conference*, 2017, pp. 270–278. doi: 10.1145/3123266.3123294.

[6] B. Yang, & T. Mitchell. Joint extraction of events and entities within a document context. In: *Proceedings of 2016 Annual Conference of the North Amer-*

ican Chapter of the Association for Computational Linguistics, 2016, pp. 289–299. doi: 10.18653/v1/N16-1033.

[7] A. Judea, & M. Strube. Incremental global event extraction. In: *The 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 2279–2289. Available at: http://aclweb.org/anthology/C/C16/C16-1215.pdf.

[8] B. Yang, & T. Mitchell. Leveraging knowledge bases in LSTMs for improving machine reading. In: *Proceedings of 2017 Annual Meeting of the Association for Computational Linguistics*, 2017, pp. 1436–1446. doi: 10.18653/v1/P17-1132.

[9] S. Duan, R. He, & W. Zhao. Exploiting document level information to improve event detection via recurrent neural networks. In: *Proceedings of 2017 International Joint Conference on Natural Language Processing (Poster and Demo Session)*, 2017. Available at: http://ijcnlp2017.org/site/page.aspx?pid=172&sid=1133&lang=en.

[10] Y. Chen, L. Xu, K. Liu, D. Zeng, & J. Zhao. Event extraction via dynamic multi-pooling convolutional neural networks. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, 2015, pp. 167–176. doi: 10.3115/v1/P15-1017.

[11] T.H. Nguyen, & R. Grishman. Event detection and domain adaptation with convolutional neural networks. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers)*, 2015, pp. 365–371. Available at: http://www.anthology.aclweb.org/P/P15/P15-2060.pdf.

[12] X. Feng, L. Huang, D. Tang, B. Qin, H. Ji, & T. Liu. A language independent neural network for event detection. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 66–71. Available at: http://wing.comp.nus.edu.sg/~antho/P/P16/P16-2011.pdf.

[13] T.H. Nguyen, K. Cho, & R. Grishman. Joint event extraction via recurrent neural networks. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 300–309. doi: 10.18653/v1/N16-1034.

[14] L. Huang, T. Cassidy, X. Feng, H. Ji, C.R. Voss, J. Han, & A. Sil. Liberal event extraction and event schema induction. In: *Proceedings of 2016 Annual Meeting of the Association for Computational Linguistics*, 2016, doi: 10.18653/v1/P16-1025.

[15] T.H. Nguyen, & R. Grishman. Graph convolutional networks with argument-aware pooling for event detection. In: *The 32nd AAAI Conference on Artificial Intelligence (AAAI-18)*, 2018, pp. 1–8. Available at: http://ix.cs.uoregon.edu/~thien/pubs/graphConv.pdf.

[16] L. Sha, F. Qian, S. Li, B. Chang, & Z. Sui. Jointly extracting event triggers and arguments by dependency-bridge RNN and tensor-based argument inter-

action. In: *The 32nd AAAI Conference on Artificial Intelligence (AAAI-18)*, 2018, pp. 1–8. Available at: http://shalei120.github.io/docs/sha2018Joint.pdf.

[17] L. Huang, H. Ji, K. Cho, I. Dagan, S. Riedel, & C. Voss. Zero-shot transfer learning for event extraction. In: *Proceedings of 2018 Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 1–10. Available at: http://nlp.cs.rpi.edu/paper/zeroshot2017.pdf.

[18] Y. Hong, W. Zhou, J. Zhang, Q. Zhu, & G. Zhou. Self-regulation: Employing a generative adversarial network to improve event detection. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers)*, 2018, pp. 515–526. Available at: http://aclweb.org/anthology/P18-1048.

[19] Y. Zhao, X. Jin, Y. Wang, & X. Cheng. Document embedding enhanced event detection with hierarchical and supervised attention. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 414–419. Available at: http://aclweb.org/anthology/P18-2066.

[20] T.M. Nguyen, & T.H. Nguyen. One for all: Neural joint modeling of entities and events. arXiv preprint. arXiv:1812.00195, 2018.

[21] Y. Feng, H. Zhang, W. Hao, & G. Chen. Joint extraction of entities and relations using reinforcement learning and deep learning. *Computational Intelligence and Neuroscience* (2017), Article ID 7643065. doi: 10.1155/2017/7643065.

[22] H. Zhang, Y. Feng, W. Hao, G. Chen, & D. Jin. Relation extraction with deep reinforcement learning. *IEICE Transactions on Information and Systems* E100.D(8)(2017), 1893–1902. doi: 10.1587/transinf.2016EDP7450.

[23] H. Daumé, J. Langford, & D. Marcu. Search-based structured prediction. *Machine Learning* 75(3) 2009, 297–325. doi: 10.1007/s10994-009-5106-x.

[24] S. Ross, G. Gordon, & D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In: *Proceedings of 2011 International Conference on Artificial Intelligence and Statistics*. arXiv preprint. arXiv:1011.0686, 2011.

[25] K.-W. Chang, A. Krishnamurthy, A. Agarwal, H. Daumé III, & J. Langford. Learning to search better than your teacher. In: *Proceedings of the 32nd International Conference on Machine Learning*, 2015, pp. 2058–2066. Available at: https://dl.acm.org/citation.cfm?id=3045337.

[26] P. Abbeel, & A.Y. Ng. Apprenticeship learning via inverse reinforcement learning. In: *Proceedings of the 21st International Conference on Machine Learning*, 2004, pp. 1–8. doi: 10.1145/1015330.1015430.

[27] U. Syed, M. Bowling, & R.E. Schapire. Apprenticeship learning using linear programming. *Proceedings of the 25th International Conference on Machine Learning*, 2008, pp. 1032–1039. doi: 10.1145/1390156.1390286.

[28] B.D. Ziebart, A.L. Maas, J.A. Bagnell, & A.K. Dey. Maximum entropy inverse reinforcement learning. In: *Proceedings of the 23rd National Conference on Artificial Intelligence, AAAI*, 2008, pp. 1433–1438. Available at: https://dl.acm.org/citation.cfm?id=1620297.

[29] J. Ho, & S. Ermon. Generative adversarial imitation learning. In: *The 30th Conference on Neural Information Processing Systems (NIPS 2016)*, 2016, pp. 1–9. Available at: http://papers.nips.cc/paper/6391-generative-adversarial-imitation-learning.pdf.

[30] N. Baram, O. Anschel, I. Caspi, & S. Mannor. End-to-end differentiable adversarial imitation learning. In: *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 390–399. Available at: http://proceedings.mlr.press/v70/baram17a.html.

[31] A. Vlachos, & M. Craven. Search-based structured prediction applied to biomedical event extraction. In: *Proceedings of 2011 Conference on Computational Natural Language Learning*, 2011, pp. 49–57. Available at: https://dl.acm.org/citation.cfm?id=2018943.

[32] F. Maes, L. Denoyer, & P. Gallinari. Sequence labeling with reinforcement learning and ranking algorithms. In: *Proceedings of the 18th European Conference on Machine Learning*, 2007, pp. 648–657. doi: 10.1007/978-3-540-74958-5_{64}.

[33] S. Hochreiter, & J. Schmidhuber. Long short-term memory. *Neural Computation* 9(8)(1997), pp. 1735–1780. doi: 10.1162/neco.1997.9.8.1735.

[34] B. Bakker. Reinforcement learning with long short-term memory. In: *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, pp. 1475–1482. Available at: https://dl.acm.org/citation.cfm?id=2980731.

[35] M.E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, & L. Zettlemoyer. Deep contextualized word representations. In: *Proceedings of 2018 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2018, pp. 2227–2237. doi: 10.18653/v1/N18-1202.

[36] R.S. Sutton, D.A. McAllester, S.P. Singh, & Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In: S.A. Solla, T.K. Leen, & K.R. Müller (eds.) *Advances in Neural Information Processing Systems 12 (NIPS 1999)*. Cambridge, MA: The MIT Press, 2000, pp. 1057–1063. Available at: http://papers.nips.cc/paper/1713-policy-gradient-methods-for-reinforcement-learning-with-function-approximation.pdf.

[37] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, & Y. Bengio. Generative adversarial nets. In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, 2014, pp. 2672–2680. Available at: https://dl.acm.org/citation.cfm?id=2969125.

[38] R. Pasunuru, & M. Bansal. Reinforced video captioning with entailment rewards. arXiv preprint. arXiv:1708.02300, 2017.

[39] R. Pasunuru, & M. Bansal. Multi-reward reinforced summarization with saliency and entailment. arXiv preprint. arXiv:1804.06451, 2018.

[40] K. Toutanova, D. Klein, C.D. Manning, & Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, 2003, pp. 173–180. doi: 10.3115/1073445.1073478.

[41] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, & J. Dean. Distributed representations of words and phrases and their compositionality. In: *The 27th Conference on Neural Information Processing Systems (NIPS 2013)*, 2013, pp. 1–9. Available at: http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

[42] Q. Li, H. Ji, Y. Hong, & S. Li. Constructing information networks using one single model. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1846–1851. Available at: http://aclweb.org/anthology/D14-1198.

[43] M. Miwa, & M. Bansal. End-to-end relation extraction using LSTMs on sequences and tree structures. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016, pp. 1105–1116. Available at: http://aclweb.org/anthology/P16-1105.

[44] H.J. Devlin, M.-W. Chang, K. Lee, & K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint. arXiv:1810.04805, 2018.

AUTHOR BIOGRAPHY

Tongtao Zhang is a PhD candidate at the Computer Science Department, Rensselaer Polytechnic Institute and a member of BLENDER. He focuses on event extraction with multi-modal approaches, encompassing domains such as natural language processing, computer vision, and machine learning. He attempts to fuse techniques from these domains to pursue a more comprehensive knowledge base. Tongtao received his Master of Science degree from the Department of Electrical Engineering, Columbia University, Bachelor of Science degree from the Department of Applied Physics, Donghua University with "City' s Graduate of Excellence," and Bachelor of Arts degree from the Department of German, Shanghai International Studies University.

Heng Ji is Edward P. Hamilton Development Chair Professor at the Computer Science Department of Rensselaer Polytechnic Institute. She received her Bachelor of Arts and Master of Arts degrees in Computational Linguistics from Tsinghua University and her Master' s and PhD degrees in Computer Science

from New York University. Her research interests focus on natural language processing and its connections with data mining, Social Sciences, and vision. She was selected as "Young Scientist" and a member of the Global Future Council on the Future of Computing by the World Economic Forum in 2016 and 2017. She received the "AI' s 10 to Watch" Award by IEEE Intelligent Systems in 2013, Google Research Awards in 2009 and 2014, Sloan Junior Faculty Award in 2012, IBM Watson Faculty Award in 2012 and 2014, and Bosch Research Awards in 2015, 2016, and 2017.

Avirup Sil is a Research Scientist in the Information Extraction and Natural Language Processing (NLP) Group at IBM Research AI. He is also the Chair of the NLP professional community of IBM. Currently, he works on industry-scale NLP and deep learning algorithms. His work focuses on information extraction: entity recognition and linking and relation extraction. He is currently working on Question Answering algorithms by attaching world knowledge to systems. He is a senior program committee member for major Computational Linguistics conferences, including serving as Area Chair multiple times. Avirup finished his PhD in Computer Science under the supervision of his thesis advisor Alexander Yates. He also worked on temporal information extraction in the Machine Learning Group at Microsoft Research, Redmond, managed by Chris Burges and John Platt. His mentor was Silviu Cucerzan. He has filed more than 12 US patents in the area of artificial intelligence and its applications in various spheres.

Note: Figure translations are in progress. See original paper for figures.