Improving Information Extraction by Acquiring External Evidence with Reinforcement Learning

Karthik Narasimhan
CSAIL, MIT
karthikn@mit.edu

Adam Yala
CSAIL, MIT
adamyala@mit.edu

Regina Barzilay
CSAIL, MIT
regina@csail.mit.edu

Abstract

Most successful information extraction systems operate with access to a large collection of documents. In this work, we explore the task of acquiring and incorporating external evidence to improve extraction accuracy in domains where the amount of training data is scarce. This process entails issuing search queries, extraction from new sources and reconciliation of extracted values, which are repeated until sufficient evidence is collected. We approach the problem using a reinforcement learning framework where our model learns to select optimal actions based on contextual information. We employ a deep Q-network, trained to optimize a reward function that reflects extraction accuracy while penalizing extra effort. Our experiments on two databases – of shooting incidents, and food adulteration cases – demonstrate that our system significantly outperforms traditional extractors and a competitive meta-classifier baseline.[1]

1 Introduction

In many realistic domains, information extraction (IE) systems require exceedingly large amounts of annotated data to deliver high performance. Increases in training data size enable models to handle robustly the multitude of linguistic expressions that convey the same semantic relation. Consider, for instance, an IE system that aims to identify entities such as the perpetrator and the number of victims in a shooting incident (Figure 1). The document does not explicitly mention the shooter (*Scott Westerhuis*), but instead refers to him as a suicide victim. Extraction of the number of fatally shot victims is similarly difficult, as the system needs to infer that *"A couple and four children"* means *six people*. Even a large annotated training set may not provide sufficient coverage to capture such challenging cases.

ShooterName: Scott Westerhuis
NumKilled: 6

A couple and four children found dead in their burning South Dakota home had been shot in an apparent murder-suicide, officials said Monday.
...
Scott Westerhuis's cause of death was "shotgun wound with manner of death as suspected suicide," it added in a statement.

Figure 1: Sample news article on a shooting case. Note how the article contains both the name of the shooter and the number of people killed but both pieces of information require complex extraction schemes.

In this paper, we explore an alternative approach for boosting extraction accuracy, when a large training corpus is not available. Instead, the proposed method utilizes external information sources to resolve ambiguities inherent in text interpretation. Specifically, our strategy is to find other documents that contain the information sought, expressed in a form that a basic extractor can "understand". For instance, Figure 2 shows two other articles describing the same event, wherein the entities of interest

[1]Code is available at http://people.csail.mit.edu/karthikn/rl-ie/

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 2355–2365,
Austin, Texas, November 1-5, 2016.

The **six** members of a South Dakota family found dead in the ruins of their burned home were fatally shot, with one death believed to be a suicide, authorities said Monday.

AG Jackley says all evidence supports the story he told based on preliminary findings back in September: **Scott Westerhuis** shot his wife and children with a shotgun, lit his house on fire with an accelerant, then shot himself with his shotgun.

Figure 2: Two other articles on the same shooting case. The first article clearly mentions that six people were killed. The second one portrays the shooter in an easily extractable form.

– the number of people killed and the name of the shooter – are expressed explicitly. Processing such stereotypical phrasing is easier for most extraction systems, compared to analyzing the original source document. This approach is particularly suitable for extracting information from news where a typical event is covered by multiple news outlets.

The challenges, however, lie in (1) performing *event coreference* (i.e. retrieving suitable articles describing the same incident) and (2) reconciling the entities extracted from these different documents. Querying the web (using the source article's title for instance) often retrieves documents about other incidents with a tangential relation to the original story. For example, the query "*4 adults, 1 teenager shot in west Baltimore 3 april 2015*" yields only two relevant articles among the top twenty results on Bing search, while returning other shooting events at the same location. Moreover, the values extracted from these different sources require resolution since some of them might be inaccurate.

One solution to this problem would be to perform a single search to retrieve articles on the same event and then reconcile values extracted from them (say, using a *meta-classifier*). However, if the confidence of the new set of values is still low, we might wish to perform further queries. Thus, the problem is inherently sequential, requiring alternating phases of querying to retrieve articles and integrating the extracted values.

We address these challenges using a Reinforcement Learning (RL) approach that combines query formulation, extraction from new sources, and value reconciliation. To effectively select among possible actions, our state representation encodes information about the current and new entity values along with the similarity between the source article and the newly retrieved document. The model learns to select good actions for both article retrieval and value reconciliation in order to optimize the reward function, which reflects extraction accuracy and includes penalties for extra moves. We train the RL agent using a Deep Q-Network (DQN) (Mnih et al., 2015) that is used to predict both querying and reconciliation choices simultaneously. While we use a maximum entropy model as the base extractor, this framework can be inherently applied to other extraction algorithms.

We evaluate our system on two datasets where available training data is inherently limited. The first dataset is constructed from a publicly available database of mass shootings in the United States. The database is populated by volunteers and includes the source articles. The second dataset is derived from a FoodShield database of illegal food adulterations. Our experiments demonstrate that the final RL model outperforms basic extractors as well as a meta-classifier baseline in both domains. For instance, in the *Shootings* domain, the average accuracy improvement over the meta-classifier is 7%.

2 Related Work

Open Information Extraction Existing work in open IE has used external sources from the web to improve extraction accuracy and coverage (Agichtein and Gravano, 2000; Etzioni et al., 2011; Fader et al., 2011; Wu and Weld, 2010). Such research has focused on identifying multiple instances of the same relation, independent of the context in which this information appears. In contrast, our goal is to extract information from additional sources about a specific event described in a source article. Therefore, the novel challenge of our task resides in performing event coreference (Lee et al., 2012; Bejan and Harabagiu, 2014) (i.e identifying other sources describing the same event) while simultaneously reconciling extracted information. Moreover, relations typically considered by open IE systems have significantly higher coverage in online documents than a specific incident described in

a few news sources. Hence, we require a different mechanism for finding and reconciling online information.

Entity linking, multi-document extraction and event coreference Our work also relates to the task of multi-document information extraction, where the goal is to connect different mentions of the same entity across input documents (Mann and Yarowsky, 2005; Han et al., 2011; Durrett and Klein, 2014). Since this setup already includes multiple input documents, the model is not required to look for additional sources or decide on their relevance. Also, while the set of input documents overlap in terms of entities mentioned, they do not necessarily describe the same event. Given these differences in setup, the challenges and opportunities of the two tasks are distinct.

Knowledge Base Completion and Online Search Recent work has explored several techniques to perform Knowledge Base Completion (KBC) such as vector space models and graph traversal (Socher et al., 2013; Yang et al., 2014; Gardner et al., 2014; Neelakantan et al., 2015; Guu et al., 2015). Though our work also aims at increasing extraction recall for a database, traditional KBC approaches do not require searching for additional sources of information. West et al. (2014) explore query reformulation in the context of KBC. Using existing search logs, they learn how to formulate effective queries for different types of database entries. Once query learning is completed, the model employs several selected queries, and then aggregates the results based on retrieval ranking. This approach is complementary to the proposed method, and can be combined with our approach if search logs are available.

Kanani and McCallum (2012) also combine search and information extraction. In their task of faculty directory completion, the system has to find documents from which to extract desired information. They employ reinforcement learning to address computational bottlenecks, by minimizing the number of queries, document downloads and extraction action. The extraction accuracy is not part of this optimization, since the baseline IE system achieves high performance on the relations of interest. Hence, given different design goals, the two RL formulations are very different. Our approach is also close in spirit to the AskMSR system (Banko et al., 2002) which aims at using information redundancy on the web to better answer questions. Though our goal is similar, we learn to query and consolidate the different sources of information instead of using predefined rules. Several slot-filling methods have experimented with query formulation over web-based corpora to populate knowledge bases (Surdeanu et al., 2010; Ji and Grishman, 2011).

3 Framework

We model the information extraction task as a markov decision process (MDP), where the model learns to utilize external sources to improve upon extractions from a source article (see Figure 3). The MDP framework allows us to dynamically incorporate entity predictions while also providing flexibility to choose the type of articles to extract from. At each step, the system has to reconcile extracted values from a related article (e_{new}) with the current set of values (e_{cur}), and decide on the next query for retrieving more articles.

We represent the MDP as a tuple $\langle S, A, T, R \rangle$, where $S = \{s\}$ is the space of all possible states, $A = \{a = (d, q)\}$ is the set of all actions, $R(s, a)$ is the reward function, and $T(s'|s, a)$ is the transition function. We describe these in detail below.

States The state s in our MDP consists of the extractor's confidence in predicted entity values, the context from which the values are extracted and the similarity between the new document and the original one. We represent the state as a continuous real-valued vector (Figure 3) incorporating these pieces of information:

1. Confidence scores of current and newly extracted entity values.
2. One-hot encoding of matches between current and new values.
3. Unigram/tf-idf counts[2] of context words. These are words that occur in the neighborhood of the entity values in a document (e.g. the words *which*, *left*, *people* and *wounded* in the phrase "*which left 5 people wounded*").
4. *tf-idf* similarity between the original article and the new article.

[2]Counts are computed on the documents used to train the basic extraction system.

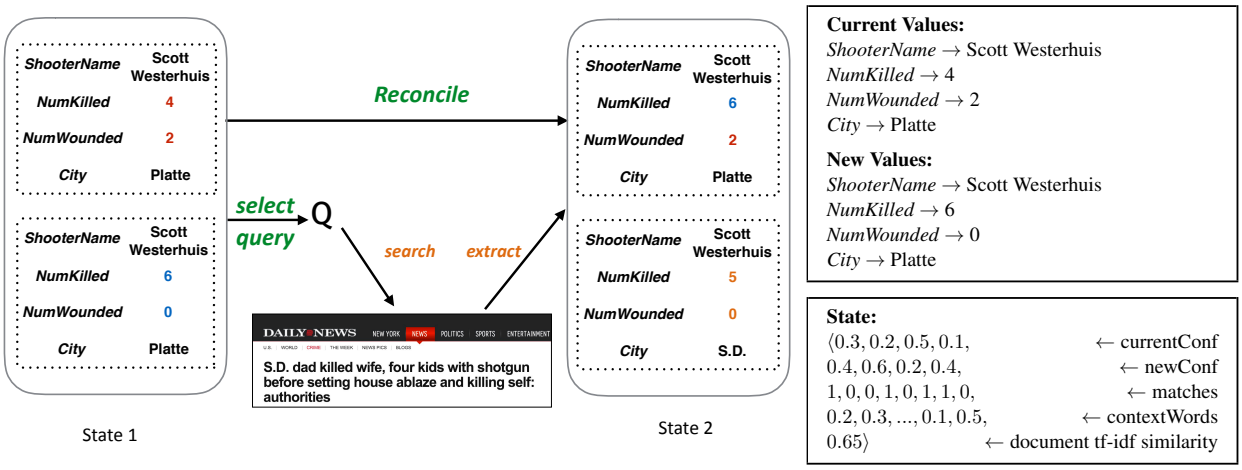

Figure 3: Left: Illustration of a transition in the MDP – the top box in each state shows the current entities and the bottom one consists of the new entities extracted from a downloaded article on the same event. **Right:** Sample state representation (bottom) in the MDP based on current and new values of entities (top). *currentConf*: confidence scores of current entities, *newConf*: confidence scores of new entities, *contextWords*: tf-idf counts of context words.

Actions At each step, the agent is required to take two actions - a reconciliation decision d and a query choice q. The decision d on the newly extracted values can be one of the following types: (1) accept a specific entity's value (one action per entity)[3], (2) accept all entity values, (3) reject all values or (4) stop. In cases 1-3, the agent continues to inspect more articles, while the episode ends if a stop action (4) is chosen. The current values and confidence scores are simply updated with the accepted values and the corresponding confidences.[4] The choice q is used to choose the next query from a set of automatically generated alternatives (details below) in order to retrieve the next article.

Rewards The reward function is chosen to maximize the final *extraction accuracy* while minimizing the number of queries. The accuracy component is calculated using the difference between the accuracy of the current and the previous set of entity values:

$$R(s,a) = \sum_{\text{entity } j} \text{Acc}(e^j_{cur}) - \text{Acc}(e^j_{prev})$$

There is a negative reward per step to penalize the agent for longer episodes.

[3]No entity-specific features are used for action selection.

[4]We also experiment with other forms of value reconciliation. See Section 5 for details.

Queries The queries are based on automatically generated templates, created using the title of an article along with words[5] most likely to co-occur with each entity type in the training data. Table 1 provides some examples – for instance, the second template contains words such as *arrested* and *identified* which often appear around the name of the shooter.

⟨*title*⟩
⟨*title*⟩ + (police | identified | arrested | charged)
⟨*title*⟩ + (killed | shooting | injured | dead | people)
⟨*title*⟩ + (injured | wounded | victim)
⟨*title*⟩ + (city | county | area)

Table 1: Examples of different query templates for web search for articles on mass shootings. The \| symbol represents logical OR. The last 4 queries contain context words around values for entity types ShooterName, NumKilled, NumWounded and City, respectively. At query time, ⟨*title*⟩ is replaced by the source article's title.

We use a search engine to query the web for articles on the same event as the source article and retrieve the top k links per query.[6] Documents that are more than a month older than the original article are filtered out of the search results.

Transitions Each episode starts off with a single source article x_i from which an initial set of entity

[5]Stop words, numeric terms and proper nouns are filtered.

[6]We use k=20 in our experiments.

values are extracted. The subsequent steps in the episode involve the extra articles, downloaded using different types of query formulations based on the source article. A single transition in the episode consists of the agent being given the state s containing information about the current and new set of values (extracted from a single article) using which the next action $a = (d, q)$ is chosen. The transition function $T(s'|s, a)$ incorporates the reconciliation decision d from the agent in state s along with the values from the next article retrieved using query q and produces the next state s'. The episode stops whenever d is a stop decision.

Algorithm 1 details the entire MDP framework for the training phase. During the test phase, each source article is handled only once in a single episode (lines 8-23).

Algorithm 1 MDP framework for Information Extraction (Training Phase)

1: Initialize set of original articles X
2: **for** $x_i \in X$ **do**
3: **for** each query template T^q **do**
4: Download articles with query $T^q(x_i)$
5: Queue retrieved articles in Y_i^q
6: **for** $epoch = 1, M$ **do**
7: **for** $i = 1, |X|$ **do** //episode
8: Extract entities e_0 from x_i
9: $e_{cur} \leftarrow e_0$
10: q $\leftarrow 0$, r $\leftarrow 0$ //query type, reward
11: **while** Y_i^q not empty **do**
12: Pop next article y from Y_i^q
13: Extract entities e_{new} from y
14: Compute tf-idf similarity $\mathcal{Z}(x_i, y)$
15: Compute context vector $\mathcal{C}(y)$
16: Form state s using e_{cur}, e_{new}, $\mathcal{Z}(x_i, y)$ and $\mathcal{C}(y)$
17: Send (s, r) to agent
18: Get decision d, query q from agent
19: **if** q == "end_episode" **then break**
20: $e_{prev} \leftarrow e_{cur}$
21: $e_{cur} \leftarrow$ *Reconcile*(e_{cur}, e_{new}, d)
22: $r \leftarrow \sum_{\text{entity } j} \text{Acc}(e_{cur}^j) - \text{Acc}(e_{prev}^j)$
23: Send (s_{end}, r) to agent

4 Reinforcement Learning for Information Extraction

In order to learn a good policy for an agent, we utilize the paradigm of reinforcement learning (RL). The MDP described in the previous section can be viewed in terms of a sequence of transitions (s, a, r, s'). The agent typically utilizes a state-action value function $Q(s, a)$ to determine which action a to perform in state s. A commonly used technique for learning an optimal value function is Q-learning (Watkins and Dayan, 1992), in which the agent iteratively updates $Q(s, a)$ using the rewards obtained from episodes. The updates are derived from the recursive Bellman equation (Sutton and Barto, 1998) for the optimal Q:

$$Q_{i+1}(s, a) = \mathrm{E}[r + \gamma \max_{a'} Q_i(s', a') \mid s, a]$$

Here, $r = R(s, a)$ is the reward and γ is a factor discounting the value of future rewards and the expectation is taken over all transitions involving state s and action a.

Since our problem involves a continuous state space S, we use a deep Q-network (DQN) (Mnih et al., 2015) as a function approximator $Q(s, a) \approx Q(s, a; \theta)$. The DQN, in which the Q-function is approximated using a deep neural network, has been shown to learn better value functions than linear approximators (Narasimhan et al., 2015; He et al., 2015) and can capture non-linear interactions between the different pieces of information in our state.

We use a DQN consisting of two linear layers (20 hidden units each) followed by rectified linear units (ReLU), along with two separate output layers.[7] The network takes the continuous state vector s as input and predicts $Q(s, d)$ and $Q(s, q)$ for reconciliation decisions d and query choices q simultaneously using the different output layers (see Supplementary material for the model architecture).

Parameter Learning The parameters θ of the DQN are learnt using stochastic gradient descent with RMSprop (Tieleman and Hinton, 2012). Each parameter update aims to close the gap between the $Q(s_t, a_t; \theta)$ predicted by the DQN and the expected Q-value from the Bellman equation, $r_t + \gamma \max_a Q(s_{t+1}, a; \theta)$. Following Mnih et al. (2015), we make use of a (separate) target Q-network to calculate the expected Q-value, in order

[7] We did not observe significant differences with additional linear layers or the choice of non-linearity (Sigmoid/ReLU).

Algorithm 2 Training Procedure for DQN agent with ϵ-greedy exploration

Initialize experience memory $\mathcal{D}$
Initialize parameters θ randomly
for $episode = 1, M$ **do**
  Initialize environment and get start state s_1
  for $t = 1, N$ **do**
    if $random() < \epsilon$ **then**
      Select a random action a_t
    else
      Compute $Q(s_t, a)$ for all actions a
      Select $a_t = \text{argmax}\ Q(s_t, a)$
    Execute action a_t and observe reward r_t and new state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$
    Sample random mini batch of transitions (s_j, a_j, r_j, s_{j+1}) from $\mathcal{D}$
    $y_j = \begin{cases} r_j, & \text{if } s_{j+1} \text{ is terminal} \\ r_j + \gamma \max_{a'} Q(s_{j+1}, a'; \theta_t), & \text{else} \end{cases}$
    Perform gradient descent step on the loss $\mathcal{L}(\theta) = (y_j - Q(s_j, a_j; \theta))^2$
    if $s_{t+1} == s_{end}$ **then break**

to have 'stable updates'. The target Q-network is periodically updated with the current parameters θ. We also make use of an experience replay memory $\mathcal{D}$ to store transitions. To perform updates, we sample a batch of transitions $(\hat{s}, \hat{a}, \hat{s}', r)$ at random from $\mathcal{D}$ and minimize the loss function[8]:

$$\mathcal{L}(\theta) = \mathrm{E}_{\hat{s},\hat{a}}[(y - Q(\hat{s}, \hat{a}; \theta))^2]$$

where $y = r + \gamma \max_{a'} Q(\hat{s}', a'; \theta_t)$ is the target Q-value. The learning updates are made every training step using the following gradients:

$$\nabla_\theta \mathcal{L}(\theta) = \mathrm{E}_{\hat{s},\hat{a}}[2(y - Q(\hat{s}, \hat{a}; \theta)) \nabla_\theta Q(\hat{s}, \hat{a}; \theta)]$$

Algorithm 2 details the DQN training procedure.

5 Experimental Setup

Data We perform experiments on two different datasets. For the first set, we collected data from the Gun Violence archive,[9] a website tracking shootings in the United States. The data contains a news article on each shooting and annotations for (1) the name of the shooter, (2) the number of people killed, (3) the number of people wounded, and (4) the city where the incident took place. We consider these as the entities of interest, to be extracted from the articles. The second dataset we use is the Foodshield EMA database[10] documenting adulteration incidents since 1980. This data contains annotations for (1) the affected food product, (2) the adulterant and (3) the location of the incident. Both datasets are classic examples where the number of recorded incidents is insufficient for large-scale IE systems to leverage.

Number	Shootings			Adulteration		
	Train	**Test**	**Dev**	**Train**	**Test**	**Dev**
Source articles	306	292	66	292	148	42
Downloaded articles	8201	7904	1628	7686	5333	1537

Table 2: Stats for *Shootings* and *Adulteration* datasets

For each source article in the above databases, we download extra articles (top 20 links) using the Bing Search API[11] with different automatically generated queries. We use only the source articles from the *train* portion to learn the parameters of the base extractor. The entire *train* set with downloaded articles is used to train the DQN agent and the meta-classifier baseline (described below). All parameters are tuned on the *dev* set. For the final results, we train the models on the combined train and dev sets and use the entire *test* set (source + downloaded articles) to evaluate. Table 2 provides data statistics.

Extraction model We use a maximum entropy classifier as the base extraction system, since it provides flexibility to capture various local context features and has been shown to perform well for information extraction (Chieu and Ng, 2002). The classifier is used to tag each word in a document as one of the entity types or not (e.g. {*ShooterName, NumKilled, NumWounded, City, Other*} in the *Shootings* domain). Then, for each tag except *Other*, we choose the mode of the values to obtain the set of entity extractions from the article.[12] Features used in the classifier are provided in the Supplementary material.

The features and context window $c = 4$ of neighboring words are tuned to maximize performance on a dev set. We also experimented with a conditional random field (CRF) (with the same features) for the sequence tagging (Culotta and McCallum, 2004)

[8] The expectation is over the transitions sampled uniformly at random from $\mathcal{D}$.

[9] www.shootingtracker.com/Main_Page

[10] www.foodshield.org/member/login/

[11] www.bing.com/toolbox/bingsearchapi

[12] We normalize numerical words (e.g. "one" to "1") before taking the mode.

but obtained worse empirical performance (see Section 6). The parameters of the base extraction model are not changed during training of the RL model.

Evaluation We evaluate the extracted entity values against the gold annotations and report the corpus-level average accuracy on each entity type. For entities like ShooterName, the annotations (and the news articles) often contain multiple names (first and last) in various combinations, so we consider retrieving either name as a successful extraction. For all other entities, we look for exact matches.

Baselines We explore 4 types of baselines:

Basic extractors: We use the CRF and the Maxent classifier mentioned previously.

Aggregation systems: We examine two systems that perform different types of value reconciliation. The first model (*Confidence*) chooses entity values with the highest confidence score assigned by the base extractor. The second system (*Majority*) takes a majority vote over all values extracted from these articles. Both methods filter new entity values using a threshold τ on the cosine similarity over the tf-idf representations of the source and new articles.

Meta-classifer: To demonstrate the importance of modeling the problem in the RL framework, we consider a meta-classifier baseline. The classifier operates over the same input state space and produces the same set of reconciliation decisions $\{d\}$ as the DQN. For training, we use the original source article for each event along with a related downloaded article to compute the state. If the downloaded article has the correct value and the original one does not, we label it as a positive example for that entity class. If multiple such entity classes exist, we create several training instances with appropriate labels, and if none exist, we use the label corresponding to the *reject all* action. For each *test* event, the classifier is used to provide decisions for all the downloaded articles and the final extraction is performed by aggregating the value predictions using the *Confidence*-based scheme described above.

Oracle: Finally, we also have an ORACLE score which is computed assuming perfect reconciliation and querying decisions on top of the Maxent base extractor. This helps us analyze the contribution of the RL system in isolation of the inherent limitations of the base extractor.

RL models We perform experiments using three variants of RL agents – (1) *RL-Basic*, which performs only reconciliation decisions[13], (2) *RL-Query*, which takes only query decisions with the reconciliation strategy fixed (similar to Kanani and McCallum (2012)), and (3) *RL-Extract*, our full system incorporating both reconciliation and query decisions.

We train the models for 10000 steps every epoch using the Maxent classifier as the base extractor, and evaluate on the entire *test* set every epoch. The final accuracies reported are averaged over 3 independent runs; each run's score is averaged over 20 epochs after 100 epochs of training. The penalty per step is set to -0.001. For the DQN, we use the dev set to tune all parameters. We used a replay memory $\mathcal{D}$ of size 500k, and a discount (γ) of 0.8. We set the learning rate to 2.5E^{-5}. The ϵ in ϵ-greedy exploration is annealed from 1 to 0.1 over 500k transitions. The target-Q network is updated every 5k steps.

6 Results

Performance Table 3 demonstrates that our system (RL-Extract) obtains a substantial gain in accuracy over the basic extractors on all entity types over both domains. For instance, RL-Extract is 11.4% more accurate than the basic Maxent extractor on City and 7.1% better on NumKilled, while also achieving gains of more than 5% on the other entities on the *Shootings* domain. The gains on the *Adulteration* dataset are also significant, up to a 11.5% increase on the Location entity.

We can also observe that simple aggregation schemes like the *Confidence* and *Majority* baselines don't handle the complexity of the task well. RL-Extract outperforms these by 7.2% on *Shootings* and 5% on *Adulteration* averaged over all entities. Further, the importance of sequential decision-making is established by RL-Extract performing significantly better than the meta-classifier (7.0% on *Shootings* over all entities). This is also due to the fact that the meta-classifier aggregates over the entire set of extra documents, including the long tail of noisy, irrelevant documents. Finally, we see the advantage of enabling the RL system to select queries as our full model RL-Extract obtains significant im-

[13] Articles are presented to the agent in a round-robin fashion from the different query lists.

System	Shootings				Adulteration		
	ShooterName	**NumKilled**	**NumWounded**	**City**	**Food**	**Adulterant**	**Location**
CRF extractor	9.5	65.4	64.5	47.9	41.2	28.3	51.7
Maxent extractor	45.2	69.7	68.6	53.7	56.0	52.7	67.8
Confidence Agg. (τ)	45.2 (0.6)	70.3 (0.6)	72.3 (0.6)	55.8 (0.6)	56.0 (0.8)	54.0 (0.8)	69.2 (0.6)
Majority Agg. (τ)	47.6 (0.6)	69.1 (0.9)	68.6 (0.9)	54.7 (0.7)	56.7 (0.5)	50.6 (0.95)	72.0 (0.4)
Meta-classifier	45.2	70.7	68.4	55.3	55.4	52.7	72.0
RL-Basic	45.2	71.2	70.1	54.0	57.0	55.1	76.1
RL-Query (conf)	39.6	66.6	69.4	44.4	39.4	35.9	66.4
RL-Extract	**50.0**	**77.6***	**74.6***	**65.6***	**59.6***	**58.9***	**79.3***
ORACLE	57.1	86.4	83.3	71.8	64.8	60.8	83.9

Table 3: Accuracy of various baselines (italics), our system (DQN) and the Oracle on *Shootings* and *Adulteration* datasets. **Agg.** refers to aggregation baselines. Bold indicates best system scores. *statistical significance of $p < 0.0005$ vs basic Maxent extractor using the Student-t test. Numbers in parentheses indicate the optimal threshold (τ) for the aggregation baselines. Confidence-based reconciliation was used for RL-Query.

Entity	System: Value	Example
ShooterName	**Basic**: Stewart	A source tells Channel 2 Action News that Thomas Lee has been arrested in Mississippi ... Sgt . Stewart Smith, with the Troup County Sheriff's office, said.
	RL-Extract: Lee	Lee is accused of killing his wife, Christie; ...
NumKilled	**Basic**: 0	Shooting leaves 25 year old Pittsfield man dead , 4 injured
	RL-Extract: 1	One man is dead after a shooting Saturday night at the intersection of Dewey Avenue and Linden Street.
NumWounded	**Basic**: 0	Three people are dead and a fourth is in the hospital after a murder suicide
	RL-Extract: 1	3 dead, 1 injured in possible Fla. murder-suicide
City	**Basic**: Englewood	A 2 year old girl and four other people were wounded in a shooting in West Englewood Thursday night, police said
	RL-Extract: Chicago	At least 14 people were shot across Chicago between noon and 10:30 p.m. Thursday. The last shooting left five people wounded.

Table 4: Sample outputs (along with corresponding article snippets) on the *Shootings* domain showing correct predictions from RL-Extract where the basic extractor (Maxent) fails.

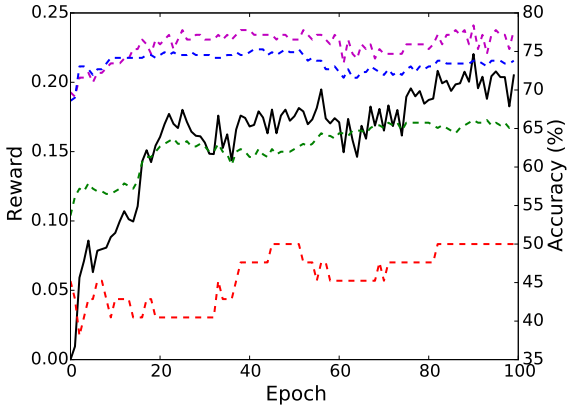

Figure 4: Evolution of average reward (solid black) and accuracy on various entities (dashed lines; red=*ShooterName*, magenta=*NumKilled*, blue=*NumWounded*, green=*City*) on the *test* set of the *Shootings* domain.

provements over RL-Basic on both domains. The full model also outperforms RL-Query, demonstrating the importance of performing both query selection and reconciliation in a joint fashion.

Figure 4 shows the learning curve of the agent by measuring reward on the test set after each training epoch. The reward improves gradually and the accuracy on each entity increases simultaneously. Table 4 provides some examples where our model is able to extract the right values when the baseline fails. One can see that in most cases this is due to the model making use of articles with prototypical language or articles containing the entities in readily extractable form.

Analysis We also analyze the importance of different reconciliation schemes, rewards and context-vectors in RL-Extract on the Shootings domain (Table 5). In addition to simple replacement (Re-

Reconciliation (RL-Extract)	Context	Reward	Accuracy				Steps
			S	K	W	C	
Confidence	tf-idf	Step	47.5	71.5	70.4	60.1	8.4
Majority	tf-idf	Step	43.6	71.8	69.0	59.2	9.9
Replace	*No context*	Step	44.4	77.1	72.5	63.4	8.0
Replace	*Unigram*	Step	48.9	76.8	74.0	63.2	10.0
Replace	tf-idf	*Episode*	42.6	62.3	68.9	52.7	6.8
Replace	tf-idf	Step	**50.0**	**77.6**	**74.6**	**65.6**	9.4

Table 5: Effect of using different reconciliation schemes, context-vectors, and rewards in our RL framework (*Shootings* domain). The last row is the overall best scheme (deviations from this are in *italics*). Context refers to the type of word counts used in the state vector to represent entity context. Rewards are either per step or per episode. (**S**: ShooterName, **K**: NumKilled, **W**: NumWounded, **C**: City, **Steps**: Average number of steps per episode)

place), we also experiment with using Confidence and Majority-based reconciliation schemes for RL-Extract. We observe that the Replace scheme performs much better than the others (2-6% on all entities) and believe this is because it provides the agent with more flexibility in choosing the final values.

From the same table, we see that using the tf-idf counts of context words as part of the state provides better performance than using no context or using simple unigram counts. In terms of reward structure, providing rewards after each step is empirically found to be significantly better (>10% on average) compared to a single delayed reward per episode. The last column shows the average number of steps per episode – the values range from 6.8 to 10.0 steps for the different schemes. The best system (RL-Extract with Replace, tf-idf and step-based rewards) uses 9.4 steps per episode.

7 Conclusions

In this paper, we explore the task of acquiring and incorporating external evidence to improve information extraction accuracy for domains with limited access to training data. This process comprises issuing search queries, extraction from new sources and reconciliation of extracted values, repeated until sufficient evidence is obtained. We use a reinforcement learning framework and learn optimal action sequences to maximize extraction accuracy while penalizing extra effort. We show that our model, trained as a deep Q-network, outperforms traditional extractors by 7.2% and 5% on average on two different domains, respectively. We also demonstrate the importance of sequential decision-making by comparing our model to a meta-classifier operating on the same space, obtaining up to a 7% gain.

Acknowledgements

We thank David Alvarez, Tao Lei and Ramya Ramakrishnan for helpful discussions and feedback, and the members of the MIT NLP group and the anonymous reviewers for their insightful comments. We also gratefully acknowledge support from a Google faculty award.

References

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 85–94. ACM.

Michele Banko, Eric Brill, Susan Dumais, and Jimmy Lin. 2002. Askmsr: Question answering using the worldwide web. In *Proceedings of 2002 AAAI Spring Symposium on Mining Answers from Texts and Knowledge Bases*, pages 7–9.

Cosmin Adrian Bejan and Sanda Harabagiu. 2014. Unsupervised event coreference resolution. *Computational Linguistics*, 40(2):311–347.

Hai Leong Chieu and Hwee Tou Ng. 2002. A maximum entropy approach to information extraction from semi-structured and free text. In *Proceedings of AAAI*.

Aron Culotta and Andrew McCallum. 2004. Confidence estimation for information extraction. In *Proceedings of HLT-NAACL 2004: Short Papers*, pages 109–112. Association for Computational Linguistics.

Greg Durrett and Dan Klein. 2014. A joint model for entity analysis: Coreference, typing, and linking. *Trans-*

actions of the Association for Computational Linguistics, 2:477–490.

Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. 2011. Open information extraction: The second generation. In *IJCAI*, volume 11, pages 3–10.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545. Association for Computational Linguistics.

Matt Gardner, Partha Talukdar, Jayant Krishnamurthy, and Tom Mitchell. 2014. Incorporating vector space similarity in random walk inference over knowledge bases. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 397–406, Doha, Qatar, October. Association for Computational Linguistics.

Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 318–327, Lisbon, Portugal, September. Association for Computational Linguistics.

Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM.

Ji He, Jianshu Chen, Xiaodong He, Jianfeng Gao, Lihong Li, Li Deng, and Mari Ostendorf. 2015. Deep reinforcement learning with an action space defined by natural language. *arXiv preprint arXiv:1511.04636*.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1148–1158, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pallika H Kanani and Andrew K McCallum. 2012. Selecting actions for resource-bounded information extraction using reinforcement learning. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 253–262. ACM.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 489–500, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gideon S Mann and David Yarowsky. 2005. Multi-field information extraction and cross-document fusion. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 483–490. Association for Computational Linguistics.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 02.

Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. 2015. Language understanding for text-based games using deep reinforcement learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. 2015. Compositional vector space models for knowledge base completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 156–166, Beijing, China, July. Association for Computational Linguistics.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pages 926–934.

Mihai Surdeanu, David McClosky, Julie Tibshirani, John Bauer, Angel X Chang, Valentin I Spitkovsky, and Christopher D Manning. 2010. A simple distant supervision approach for the tac-kbp slot filling task. In *Proceedings of Text Analysis Conference 2010 Workshop*.

Richard S Sutton and Andrew G Barto. 1998. *Introduction to reinforcement learning*. MIT Press.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4.

Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning*, 8(3-4):279–292.

Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. 2014. Knowledge base completion via search-based question answering. In *Proceedings of the 23rd international conference on World wide web*, pages 515–526. ACM.

Fei Wu and Daniel S Weld. 2010. Open information extraction using wikipedia. In *Proceedings of the 48th*

Annual Meeting of the Association for Computational Linguistics, pages 118–127. Association for Computational Linguistics.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*.