Reasoning over Entity-Action-Location Graph for Procedural Text Understanding

Hao Huang[1*], **Xiubo Geng**[2†], **Pei Jian**[3], **Guodong Long**[1], **Daxin Jiang**[2†]
[1]Australian AI Institute, School of CS, FEIT, University of Technology Sydney
[2]STCA NLP Group, Microsoft
[3]School of Computing Science, Simon Fraser University
{hao.huang-4,guodong.long}@{student.uts,uts}.edu.au
{xigeng,djiang}@microsoft.com, jpei@cs.sfu.ca

Abstract

Procedural text understanding aims at tracking the states (e.g., create, move, destroy) and locations of the entities mentioned in a given paragraph. To effectively track the states and locations, it is essential to capture the rich semantic relations between entities, actions, and locations in the paragraph. Although recent works have achieved substantial progress, most of them focus on leveraging the inherent constraints or incorporating external knowledge for state prediction. The rich semantic relations in the given paragraph are largely overlooked. In this paper, we propose a novel approach (REAL) to procedural text understanding, where we build a general framework to systematically model the entity-entity, entity-action, and entity-location relations using a graph neural network. We further develop algorithms for graph construction, representation learning, and state and location tracking. We evaluate the proposed approach on two benchmark datasets, ProPara, and Recipes. The experimental results show that our method outperforms strong baselines by a large margin, i.e., 5.0% on ProPara and 3.2% on Recipes, illustrating the utility of semantic relations and the effectiveness of the graph-based reasoning model.

1 Introduction

Procedural text often consists of a sequence of sentences describing processes, such as a phenomenon in nature (e.g., how sedimentary rock forms) (Dalvi et al., 2018) or instructions to complete a task (e.g., the recipe of Mac and Cheese) (Bosselut et al., 2018). Given a paragraph and its participant entities, the task of procedural text understanding is to track the states (e.g., create, move, destroy) and locations (a span in the text) of the entities. Compared with traditional machine reading task, which mainly focuses on the static relations among entities, procedural text understanding is more challenging since it involves discovering complex temporal-spatial relations among various entities from the process dynamics.

To effectively track the states and locations of entities, it is crucial to systematically model rich relations among various concepts in the paragraph, including entities, actions, and locations. Three types of relations are of particular interest.

First, mentions of the same entity in different sentences are related. The inherent relation among these mentions may provide clues for a model to generate consistent predictions about the entity. For example, the entity *electrical pulses* are mentioned in two sentences "*The retina's rods and cones convert it to electrical pulses. The optic nerve carries electrical pulses through the optic canal.*". Connecting its two mentions in two sentences helps to infer its location in the first sentence using the second sentence's information.

Second, detecting connections between an entity and the corresponding actions helps to make state predictions more accurate. Take the sentence "*As the encased bones decay, minerals seep in replacing the organic material.*" as an example. The entity *bone* is related to *decay* which indicates the state *destroy*, while it is not connected to *seep* indicating the state *move*. Given the relation between *bone* and *decay*, it is easier for the model to predict the state of *bone* as *destroy*, instead of being misled by the action *seep*.

Last, when the state or location of one entity changes, it may impact all associated entities. For example, in sentence "*trashbags are thrown into trashcans.*", *trashbags* are associated with *trashcans*. Then, in the following sentence "*The trashcan is emptied by a large trash truck.*", although *trashbags* are not explicitly mentioned, their loca-

[*] Work is done during internship at Microsoft.
[†] Corresponding author.

tions are changed by the association with *trashcan*.

Recent works on procedural text understanding have achieved remarkable progress (Tandon et al., 2018; Bosselut et al., 2018; Gupta and Durrett, 2019b; Du et al., 2019; Das et al., 2019; Gupta and Durrett, 2019a). However, the existing methods do not systematically model the relations among entities, actions, and locations. Instead, most methods either leverage inherent constraints on entity states or exploit external knowledge to make predictions. For example, Gupta and Durrett (2019b) propose a structural neural network to track each entity's hidden state and summarize the global state transitions with a CRF model. Tandon et al. (2018) inject commonsense knowledge into a neural model with soft and hard constraints. Although Das et al. (2019) model the relation between entities and locations, there is no general framework to model the relations, and some important relations, such as entity-action and entity-entity relations, are ignored.

A general framework to systematically model the rich types of relations among entities, actions, and locations is essential to procedural text understanding. To the best of our knowledge, we are the first to explore comprehensive relation modeling, representation, and reasoning systematically. Specifically, we first construct an entity-action-location graph from a given paragraph, where three types of concepts (i.e., entities, locations, and actions) are identified and extracted as nodes. We then detect critical connections among those concepts and represent them as edges. Finally, we adopt a graph attention network to conduct **R**easoning over the **E**ntity-**A**ction-**L**ocation graph (REAL), which provides expressive representations for downstream state and location predictions.

We evaluate the proposed approach on two benchmark datasets for procedural text understanding, ProPara (Dalvi et al., 2018) and Recipes (Bosselut et al., 2018). Our approach outperforms the state-of-the-art strong baselines by a large marge, i.e., 5.0% on ProPara and 3.2% on Recipes. The ablation study and analysis show that the graph-based reasoning approach generates better representations for entities, locations, and actions. Thus, it is highly valuable for both state and location tracking of entities.

2 Related Work

REAL is closely related to two lines of works, i.e., procedural text understanding and graph reasoning in language understanding.

Procedural Text Understanding. Compared with early-stage models (Henaff et al., 2017; Seo et al., 2017), recent progress in the procedural text understanding task is mainly made on ensuring the prediction's consistency or injecting external knowledge. Various approaches (Dalvi et al., 2018; Gupta and Durrett, 2019b; Amini et al., 2020) have been proposed to predict consistent state sequence. For example, NCET (Gupta and Durrett, 2019b) tracks the entity in a continuous space and leverages a conditional random field (CRF) to keep a consistent prediction sequence. Other models inject knowledge from external data sources to complement missing knowledge. ProStruct (Tandon et al., 2018) introduces commonsense constraints to refine the probability space, while KOALA (Zhang et al., 2020) leverages Bert Encoder pre-trained on related corpus from Wiki, and injects the ConceptNet (Speer et al., 2017) knowledge. Besides, a few models (Das et al., 2019; Dalvi et al., 2019) are proposed to build graphs on the procedural text. For instance, KG-MRC (Das et al., 2019) constructs dynamic knowledge graphs between entities and locations. However, these methods can not systematically capture the relations among entities, actions, and locations, and entity-action and entity-entity relations are ignored.

Graph Reasoning in Language Understanding. Graph-based reasoning methods (Zeng et al., 2020; Zhong et al., 2020; Zheng and Kordjamshidi, 2020) are widely used in natural language understanding tasks to enhance performance. For example, Zeng et al. (2020) constructs a double graph design for the document-level Relation Extraction (RE) task, Zhong et al. (2020) constructs the retrieved evidence sentences as a graph for Fact-Checking task. Compared with these works, the entity-action-location graph in our approach copes better with procedural text understanding task since it precisely defines concepts we are concerned within the task and captures the rich and expressive relations among them.

3 Model

Task Definition. The procedural text understanding task is defined as follows. Given a paragraph P consists of T sentences $(S_1, S_2, ..., S_T)$, describing the process (e.g., photosynthesis, erosion) of a set of N pre-specified entities $\{e_1, e_2, ..., e_N\}$, we need to predict the state y_t^s and location y_t^l for each entity at each step t corresponding to sentence S_t[1]. Candidate states are pre-defined (e.g., $y_t^s \in$ {not_exist (O), exist (E), move (M), create (C), destroy (D)} in the ProPara dataset), and location y_t^l is usually a text span in the paragraph. Gold annotations for state and location at each step t are denoted as $\widetilde{y}_t^s$ and $\widetilde{y}_t^s$, respectively.

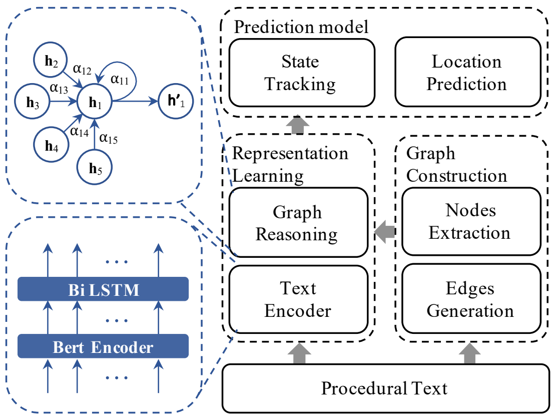


Figure 1: An overview of REAL.

Figure 1 shows the overview of our approach, which consists of three main components: graph construction, graph-based representation learning, and prediction module. The graph construction module extracts nodes and edges from the input procedural paragraph and constructs a graph. The graph reasoning module initializes nodes representations using contextual word representations and reasons over the built graph. Finally, the prediction module leverages the graph-based representations to predict the state and location.

3.1 Graph Construction

Figure 2 shows an example of the graph constructed for a paragraph which describes *how fossil forms*. A semantic graph is denoted as $G = (N, E)$, where $N = \{n_i\}_{i=1}^K$ denotes all the nodes, and $E = \{e_i\}_{i=1}^L$ denotes all the edges.

Nodes Extraction. We first extract text spans as nodes from the given paragraph. The text spans in the extracted nodes should cover all essential concepts in the paragraph. Three types of concepts play an important role in the entity tracking task, i.e., actions, entity mentions, and location mentions. Therefore, we extract nodes for them and get all the nodes $N = \{N_a, N_e, N_l\}$ where N_a represents action nodes, N_e represents entity mention nodes, and N_l represents location mention nodes.

We first tag all the verbs by an off-the-shelf part-of-speech (POS) tagger[2] and construct a set of action nodes N_a with each node associated with a single verb or a phrase consisting of two consecutive verbs. For the entity mentions, we extract the explicit (exact matching or matching after lemmatization) or implicit (pronouns) mentions of all the entities. Coreference resolution is used to find pronoun mentions in data pre-processing. Besides, we utilize the POS tagger to extract location mentions. Each tagged noun or consecutive phrase of adjective + noun is identified as a location mention.

Edges Generation. Capturing the semantic relations between various nodes is critical for understanding the process dynamics in the procedural text. To this end, we first derive verb-centric semantic structures via semantic role labeling (SRL)[3] (Shi and Lin, 2019) for each sentence and then establish intra- and inter-semantic structure edges.

Given a verb-centric structure consisting of a central verb and corresponding arguments, we create two types of edges. (1) If an entity mention $n_e \in N_e$ or location mention $n_l \in N_l$ is a substring of an argument for verb $n_a \in N_a$, then we connect n_e/n_l to n_a. For example, for the sentence "*As the encased bones decay, minerals seep in replacing ...*", the verb *decay* has an argument *the encased bones* where *bones* is an entity mention, then we will connect the action node *decay* and entity mention node *bones*. (2) Two mentions in two arguments of the same verb are connected too. For example, for the sentence "*The trashbags are thrown into a large outdoor trashcan*", the verb *thrown* has two arguments, *the trashbags* and *into a large outdoor trashcan*, then we connect the two mention nodes *trashbags* and *trashcans*.

We also create edges between mentions of the same entity in different semantic structures. For example, in Figure 2, the entity *bones* are mentioned in two sentences, which correspond to two entity mention nodes. We connect these two nodes to propagate information from one to the other during graph-based reasoning.

3.2 Graph-based Representation Learning

Nodes Representation. We first feed the entire paragraph to the BERT (Devlin et al., 2019)

[1]We will use *step* and *sentence* interchangeably.

[2]https://github.com/flairNLP/flair

[3]https://github.com/allenai/allennlp.

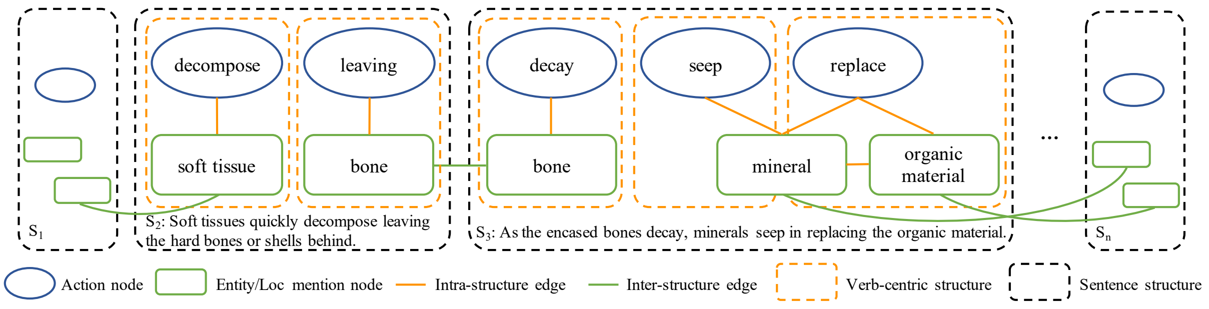


Figure 2: An example of entity-action-location graph, constructed for paragraph "*...Soft tissues quickly decompose leaving the hard bones or shells behind. As the encased bones decay, minerals seep in replacing the organic material...* "

model, which is then sent into a Bidirectional LSTM (Hochreiter and Schmidhuber, 1997) (BiLSTM) to obtain the contextual embedding for each token. Each node in our graph is associated with a text span in the paragraph. Therefore, the initial node representation is derived by mean pooling over all token embeddings in its corresponding text span. The contextual representation of node $n_i \in N$ is denoted as $\mathbf{h}_i$ $(i = 1, \dots, K)$ with $\mathbf{h}_i \in \mathbb{R}^d$.

Graph Reasoning. We leverage a graph attention network (GAT) (Velickovic et al., 2018) for reasoning over the built graph. The network performs masked attention over neighbor nodes (i.e., connected with an edge) instead of all the nodes in the graph. We apply a two-layer GAT, which means each node can aggregate information from their two-hop neighbor nodes (nodes that can be reached within two edges).

In each GAT layer, we first extract a set of neighbor nodes $\mathcal{N}_i$ for each node n_i. The attention coefficients between node n_i and its neighbour n_j can be computed through a shared attention mechanism,

$$e_{ij} = \mathbf{a}^T[\mathbf{W}\mathbf{h}_i \| \mathbf{W}\mathbf{h}_j], \tag{1}$$

where $\mathbf{a} \in \mathbb{R}^{2d}$ and $\mathbf{W} \in \mathbb{R}^{d\times d}$ are learnable parameters, and $\|$ is the concatenation operation. We apply a LeakyReLU activate function and normalize the attention coefficients,

$$\alpha_{ij} = \operatorname*{softmax}_j \left(\text{LeakyReLU}\left(e_{ij}\right)\right). \tag{2}$$

Then, we aggregate the information from the neighbor nodes with multi-head attention to enhance the stability and efficiency. The aggregated feature for n_i with a K-head attention can be represented as

$$\mathbf{h}'_i = \Big\|_{k=1}^{K} \sigma\left(\sum_{n_j\in\mathcal{N}_i} \alpha^k_{ij}\mathbf{W}^k\mathbf{h}_j\right) \tag{3}$$

in the first layer, and

$$\mathbf{h}''_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{n_j\in\mathcal{N}_i} \alpha'^k_{ij}\mathbf{W}'^k\mathbf{h}'_j\right) \tag{4}$$

in the second layer, where $\|$ is the concatenation operation, σ is the sigmoid activate function, $\mathbf{W}^k \in \mathbb{R}^{d\times d}$ is learnable matrix for kth head in first layer, and $\mathbf{W}'^k \in \mathbb{R}^{Kd\times d}$ is learnable matrix for kth head in second layer. α^k_{ij} and α'^k_{ij} are calculated with the corresponding $\mathbf{W}^k$ and $\mathbf{W}'^k$, respectively.

3.3 Prediction Model

Inspired by NCET (Gupta and Durrett, 2019b), we track the state and location separately, by a state tracking and a location prediction module. Each module takes the representations of concerned nodes as input and outputs the prediction (i.e., state or location of an entity) at each time step.

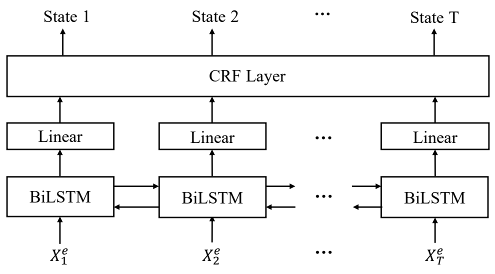


Figure 3: Overview of state tracking model, which predicts states of the entity in every sentence S_t given entity e and paragraph P.

State Tracking. Given a paragraph P and an entity e, the state tracking module tracks the state of the entity for each sentence. We first generate the representations of all sentences for the entity. Considering that actions are good state-changing signals, we concatenate the embeddings of entity mention node and action node in the sentence as representation at step t. That is,

$$\mathbf{x}_t^e = \begin{cases} [\mathbf{h}_t^e \| \mathbf{h}_t^v], & \text{if } S_t \text{ contains } n_e \\ \mathbf{0}, & \text{otherwise} \end{cases} \quad (5)$$

where $\mathbf{x}_t^e$ denotes the representation of entity e in sentence S_t , $\mathbf{h}_t^e$ denotes the representation of the entity mention node n_e in sentence S_t, $\mathbf{h}_t^v$ denotes the representation of the action node n_a connected with n_e in sentence S_t. If entity e is not mentioned in sentence S_t, we use zero vector as representation of S_t for e. Note if there are multiple mention nodes for the entity e in sentence S_t, we take the mean pooling over all mention nodes as $\mathbf{h}_t^e$. And we take similar approach for multiple actions.

We utilize a BiLSTM layer on the sequence of sentence embeddings. And a conditional random field (CRF) (Durrett and Klein, 2015) is applied on the top of the BiLSTM to make the final prediction. The loss function for the state tracking module is defined as

$$L_{state} = -\sum_{(e,P)\in\mathcal{D}} \frac{1}{T}\sum_{t=1}^{T} \log\mathcal{P}\left(\widetilde{y}_t^s | P, e; \theta^G, \theta^{st}\right), \quad (6)$$

where $\mathcal{D}$ is the training collection containing entity-paragraph pairs, $\mathcal{P}\left(\widetilde{y}_t^s | P, e; \theta^G, \theta^{st}\right)$ represents the predicted probability of gold state $\widetilde{y}_t^s$ in sentence S_t given the entity e and paragraph P, θ^G are parameters for graph reasoning and the text encoder, and θ^{st} are parameters in state tracking module.

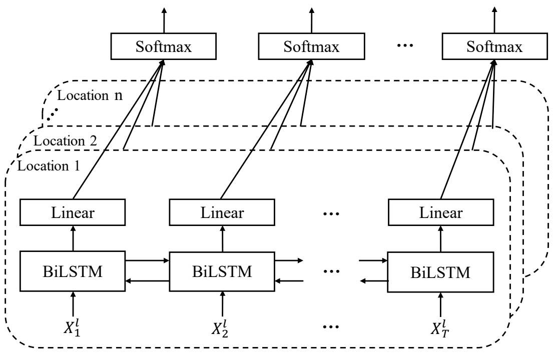


Figure 4: Overview of location prediction model, which predicts locations of the entity in every sentence S_t given entity e and paragraph P.

Location Prediction. For the location prediction module, we first collect all the location mention nodes as location candidates set $\mathcal{C}$. We add an isolated location node to represent the special location candidate '?', which means the location cannot be found in the paragraph. The representation of this node is randomly initialized and learnable during the training process.

Given an entity e and location candidate $l \in \mathcal{C}$, we represent the sentence S_t as

$$\mathbf{x}_t^l = [\mathbf{h}_t^e \| \mathbf{h}_t^l], \quad (7)$$

where $\mathbf{h}_t^e$ and $\mathbf{h}_t^l$ denotes the representation of the entity mention node and location mention node in sentence S_t. If the entity or location candidate is not mentioned in sentence S_t, we use a zero vector replacing $\mathbf{h}_t^e$ or $\mathbf{h}_t^l$.

We use a BiLSTM followed by a linear layer for the location predictor. The model outputs a score for each candidate at each step t. Then, we apply a softmax layer over all the location candidates' scores at the same step, resulting in a normalized probabilistic distribution. The location loss is defined as

$$L_{loc} = -\sum_{(e,P)\in\mathcal{D}} \frac{1}{T}\sum_{t=1}^{T} \log\mathcal{P}\left(\widetilde{y}_t^l | P, e; \theta^G, \theta^{loc}\right), \quad (8)$$

where $\mathcal{P}\left(\widetilde{y}_t^l | P, e; \theta^G, \theta^{loc}\right)$ represents the predicted probability of gold location $\widetilde{y}_t^l$ for entity e in sentence S_t, and θ^{loc} are parameters for location prediction module.

3.4 Learning and Inference

We create a single graph for each paragraph, which stays unchanged once created. Then the graph reasoning module and state/location prediction module are jointly trained in an end-to-end manner. The overall loss is defined as

$$L_{total} = L_{state} + \lambda_{loc} L_{loc}, \quad (9)$$

where λ_{loc} is the hyper-parameter to balance the state tracking and the location prediction loss.

We perform inference in pipeline mode. Specifically, for each entity, we first apply the state tracking module to infer its state at each time step. Then we only predict its location at steps when its state is changed (i.e., the predicted state is *create* or *move*[4]). And the locations of an entity with unchanged states can be inferred according to its locations in previous steps. Such pipeline fashion

[4]The location of an entity will be *None* if its state is *destroy*. Therefore, we do not need to predict its location when an entity is *destroyed*.

can increase consistency between states and locations of an entity than inferring location and state simultaneously.

4 Experiments

This section describes the evaluation results of REAL on two datasets (ProPara (Dalvi et al., 2018) and Recipes (Bosselut et al., 2018)). We also provide ablation study and case analysis to illustrate the effectiveness of graph-based reasoning.

4.1 Datasets and Evaluation Metrics

Statistics	**ProPara**	**Recipes**
#sentences	3.3K	7.6K
#para	488	866
#train/#dev/#test	391/43/54	693/86/87
avg. #entities per para	4.17	8.57
avg. #sentences per para	6.7	8.8

Table 1: Statistics of ProPara and Recipes dataset.

ProPara contains procedural texts about scientific processes, e.g., photosynthesis, fossil formulation. It contains about 1.9k instances (one entity-paragraph pair as an instance) written and annotated by human crowd workers. We follow the official split (Dalvi et al., 2018) for train/dev/test set. The Recipes dataset consists of paragraphs describing cooking procedures and their ingredients as entities. We only use the human-labeled data in our experiment, with 80%/10%/10% of the data for train/dev/test, respectively. Detail statistics for the two datasets can be found in Table 1.

We follow previous work's setting (Dalvi et al., 2018) and evaluate the proposed approach on two types of tasks on the ProPara dataset, document-level task and sentence-level task. Document-level task focuses on figuring out input entities, output entities, entity conversions, and entity movements by answering corresponding questions. More details can be found in the official script[5]. Following the official script, we evaluate models with averaged precision, recall, and F1 scores. In sentence-level task, we need to answer three categories of questions: (Cat-1) Is entity e created (destroyed, moved) in the process? (Cat-2) When is e created (destroyed, moved)? (Cat-3) Where is e created (destroyed, moved from/to)? For this task, we take macro-average and micro-average of the score for three sets of questions as evaluation metrics[6].

For the Recipes dataset, we take the same setting as (Zhang et al., 2020), where the goal is to predict the ingredients' location changes during the process. We take precision, recall, and F1 scores to evaluate models[7].

4.2 Implementation Details

We use Bert base (Devlin et al., 2019) as encoder and reason with 3-heads GAT. Batch size is set to 16, and embedding size is set to 256. The learning rate r, location loss coefficient λ_{loc} and dropout rate d are derived by grid searching with in 9 trials in $r \in \{2.5 \times 10^{-5}, 3 \times 10^{-5}, 3.5 \times 10^{-5}\}$, $\lambda_{loc} \in \{0.2, 0.3, 0.4\}$, and $d \in \{0.3, 0.4, 0.5\}$. The implementation is based on Python and trained on a Tesla P40 GPU with Adam optimizer for approximately one hour (with approximately 112M parameters). We choose the best model with highest prediction accuracy on development set.

4.3 Main Results

Table 2 compares REAL with previous work on the ProPara data for both document-level and sentence-level tasks. Our proposed approach consistently outperforms all previous models, which do not utilize external knowledge on all metrics. In particular, compared to DYNAPRO, it increases the document-level F1 score by 5.3%, and sentence-level macro averaged accuracy from 55.4% to 58.2%. Without any external data, our approach achieves comparable results to KOALA, which extensively leverages rich external knowledge in ConceptNet and Wikipedia pages, demonstrating the effectiveness of exploiting the entity-action-location graph. We also compare REAL with the re-implemented NCET[8] on the Recipes dataset. As shown in 3, REAL also surpass the strong baseline by 3.2%. All these results verify the effectiveness of the proposed graph-based reasoning approach.

4.4 Ablations

We conduct an ablation study to testify the effectiveness of multiple components in our approach. Table 4 and Table 3 list the results on ProPara and

[5]https://github.com/allenai/aristo-leaderboard/tree/master/propara

[6]https://github.com/allenai/propara/tree/master/propara/evaluation

[7]https://github.com/ytyz1307zzh/Recipes

[8]The re-implemented NCET achieves comparable accuracy with the previous state-of-the-art algorithm, DYNAPRO, i.e., 65.2% F1 score for NCET v.s. 65.5% for DYNAPRO.

Models	Document-level task			Sentence-level task				
	Precsion	Recall	F1	Cat-1	Cat-2	Cat-3	Macro-Avg	Micro-Avg
EntNet (Henaff et al., 2017)	54.7	30.7	39.4	51.6	18.8	7.8	26.1	26.0
QRN (Seo et al., 2017)	60.9	31.1	41.4	52.4	15.5	10.9	26.3	26.5
ProLocal (Dalvi et al., 2018)	81.7	36.8	50.7	62.7	30.5	10.4	34.5	34.0
ProGlobal (Dalvi et al., 2018)	48.8	61.7	51.9	63.0	36.4	35.9	45.1	45.4
ProStruct (Tandon et al., 2018)	74.3	43.0	54.5	-	-	-	-	-
XPAD (Dalvi et al., 2019)	70.5	45.3	55.2	-	-	-	-	-
KG-MRC (Das et al., 2019)	69.3	49.3	57.6	62.9	40.0	38.2	47.0	46.6
NCET (Gupta and Durrett, 2019b)	67.1	58.5	62.5	73.7	47.1	41.0	53.9	54.0
DYNAPRO (Amini et al., 2020)	75.2	58.0	65.5	72.4	49.3	**44.5**	55.4	55.5
KOALA (Zhang et al., 2020)	77.7	**64.4**	70.4	**78.5**	53.3	41.3	57.7	57.5
REAL (our approach)	**81.9**	61.9	**70.5**	78.4	**53.7**	42.4	**58.2**	**57.9**

Table 2: Experiment results on ProPara document-level task and sentence-level task. KOALA uses rich external data from Wikipedia and ConceptNet. Our approach achieves comparable performance to KOALA without any external knowledge.

Models	Precsion	Recall	F1
NCET re-implementation	**56.5**	46.4	50.9
REAL	55.2	**52.9**	**54.1**
-Location	54.9	51.7	53.3
-State	54.9	52.0	53.4
-Graph	57.2	47.9	52.1

Table 3: Comparison on Recipes dataset.

Models	Precsion	Recall	F1
REAL	81.9	61.9	70.5
-Location	81.0 (-0.9)	57.7 (-4.2)	67.4 (-3.1)
-State	73.7 (-8.2)	61.2 (-0.7)	66.9 (-3.6)
-Graph	72.0 (-9.9)	61.2 (-0.7)	66.1 (-4.4)

Table 4: Ablation study on ProPara dataset.

Recipes, respectively. As shown in Table 4, removing the graph-based representation learning for location/state prediction decreases the F1 score by 3.1%/3.6%, the gap becomes 4.4% without any graph-based reasoning. We can get similar observations on the Recipes dataset, indicating that exploiting the paragraph's rich relations is critical for both state tracking and location prediction.

4.5 Analyses of Different Relations

To further illustrate the effectiveness of different types of relations, we conduct below analyses and present three cases with predictions of REAL with and without graph reasoning in Figure 5.

First, to verify the effectiveness of action-entity relations in multi-verb sentences, we compare REAL of with and without graph reasoning on sentences containing multiple (i.e., more than 2) verbs in Table 5. We figure out that graph-based reasoning increases the performance by 5.7%, indicating that accurately connecting entities and corresponding actions improves the prediction accuracy. For case 1 shown in Figure 5, the relation between the entity *bone* the action *decay* helps the model to correctly predict the state of *bone* as *destroy* since the action *decay* indicates *destroy*. However, without such accurate connection between *bone* and *decay*, the prediction model is very likely to be misled by other actions such as *seep* or *replace*.

Segments	Models	Precision	Recall	F1
muli-verb	w/o graph	73.0	58.2	64.8
	w/ graph	**82.5**	**61.0**	**70.1**
implicit	w/o graph	74.9	57.9	65.3
	w/ graph	**83.7**	**60.3**	**70.1**

Table 5: Analyses of impact of entity-action and entity-entity relations on ProPara.

Second, we illustrate the impact of entity-entity relations by comparing our approach and baseline where the entity is not explicitly mentioned[9]. As shown in Table 5, REAL increase the accuracy by 4.8%, which indicates the effectiveness of our approach by modeling cross-entity relations. The second case in Figure 5 illustrates the effectiveness of using entity-entity relations. The entity *bags* is not explicitly mentioned in the sentence "*Trashcan gets emptied into trash truck*", and thus the baseline model cannot correctly predict its state and

[9]We only compare performance for those entity-sentence pairs with gold state as *Move*, *Create* and *Destroy*.

Case 1 Entity: bone

Text Paragraph (extract)	**State**	**Location**
As the encased **bones decay** , minerals seep in replacing the organic material cell by cell in a process called petrification.	E → D	-

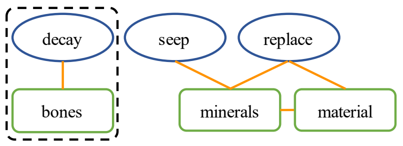
Case 2 Entity: bags

Text Paragraph (extract)	**State**	**Location**
1. **Bags** get carried out to the **trashcan**.	M	trashcan
2. Trashcan gets emptied into trash truck.	E → M	trashcan → trash truck

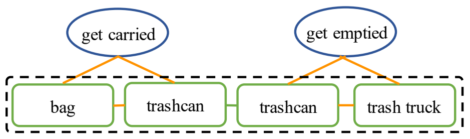
Case 3 Entity: small image

Text paragraph (extract)	**State**	**Location**
1. The cornea and lens refract light into a **small image**.	C	cornea and lens → retina
2. Shine **it** on the retina.	E	retina

Case 1 sub-graph



Case 2 sub-graph



Case 3 sub-graph

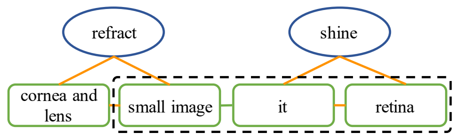


Figure 5: Examples of model predictions of our approach w/ (black) and w/o (red) graph reasoning. Corresponding sub-graph is plot on the right of the paragraph. Dotted rectangles in the sub-graph highlight key connections for correct prediction in graph-based reasoning.

location. However, connecting it to the entity *trashcan* which is derived in the first sentence, helps the model infer its state and location correctly.

Third, as discussed in section 1, mention-mention connections might improve accuracy when there are multiple mentions for the same entity. The third case in Figure 5 shows how REAL utilizes relations between different mentions for the same entity. In the first sentence, the location of entity *small image* is not mentioned, which results in wrong location prediction when no graph reasoning is used. In contrast, the built graph connects this mention with preposition *it* in the second sentence where its location is revealed as *retina*. Therefore, our model correctly predicts *small image*'s location by graph-based representation learning.

4.6 Error Analyses

We randomly sample 100 wrongly predicted examples and summarize them into the following types.

First, the ambiguity between similar entities makes it difficult to derive accurate representations for them. For instance, *fixed nitrogen* and *gas-based nitrogen* are two different entities related to nitrogen in the paragraph "*Nitrogen exists naturally in the atmosphere. Bacteria in soil fix the nitrogen. Nitrogen is now usable by living things.*". It is difficult for a model to distinguish which entity the mention *nitrogen* refers to.

Second, commonsense knowledge is required. For example, it is difficult to infer the location of the entity *bone* in the sentence "*An animal dies. It is buried in a watery environment.*" without the knowledge "*bone is part of animal*". Therefore, injecting appropriate external knowledge while avoiding noise may improve the model.

Third, similar actions indicate different states in different contexts. For instance, in sentence "*the tree eventually dies.*", the state of *tree* is labeled as *destroy*, while in sentence "*most fossils formed when animals or plants die in wet environment.*", the state of *animals* and *plants* are all annotated as *exist*, which may confuse the model.

5 Conclusion and Future Work

In this work, we propose a novel approach REAL for procedural text understanding. Unlike all previous works, we systematically exploit the rich semantic relations between entities, location, and actions. We design an entity-action-location graph to systematically model various types of concepts and their relations and develop the algorithms for graph construction, representation, and reasoning. We comprehensively conduct a quantitative and qualitative comparison of the proposed approach with strong baselines on two popular benchmark datasets for procedural text understanding and demonstrate the effectiveness of our approach. In the future, we will investigate approaches to further advance the procedural text understanding task, such as incorporating entity disambiguation and external knowledge in our approach.

References

Aida Amini, Antoine Bosselut, Bhavana Dalvi Mishra, Yejin Choi, and Hannaneh Hajishirzi. 2020. Procedural reading comprehension with attribute-aware context flow. In *Conference on Automated Knowledge Base Construction, AKBC 2020, Virtual, June 22-24, 2020*.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1595–1604. Association for Computational Linguistics.

Bhavana Dalvi, Niket Tandon, Antoine Bosselut, Wen-tau Yih, and Peter Clark. 2019. Everything happens for a reason: Discovering the purpose of actions in procedural text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4495–4504. Association for Computational Linguistics.

Rajarshi Das, Tsendsuren Munkhdalai, Xingdi Yuan, Adam Trischler, and Andrew McCallum. 2019. Building dynamic knowledge graphs from text using machine reading comprehension. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Xinya Du, Bhavana Dalvi Mishra, Niket Tandon, Antoine Bosselut, Wen-tau Yih, Peter Clark, and Claire Cardie. 2019. Be consistent! improving procedural text comprehension using label consistency. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2347–2356. Association for Computational Linguistics.

Greg Durrett and Dan Klein. 2015. Neural CRF parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 302–312. The Association for Computer Linguistics.

Aditya Gupta and Greg Durrett. 2019a. Effective use of transformer networks for entity tracking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 759–769. Association for Computational Linguistics.

Aditya Gupta and Greg Durrett. 2019b. Tracking discrete and continuous entity state for process understanding. In *Proceedings of the Third Workshop on Structured Prediction for NLP@NAACL-HLT 2019, Minneapolis, Minnesota, Jun 7, 2019*, pages 7–12. Association for Computational Linguistics.

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. 2017. Tracking the world state with recurrent entity networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

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

Min Joon Seo, Sewon Min, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Query-reduction networks for question answering. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Peng Shi and Jimmy Lin. 2019. Simple BERT models for relation extraction and semantic role labeling. *CoRR*, abs/1904.05255.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

Niket Tandon, Bhavana Dalvi, Joel Grus, Wen-tau Yih, Antoine Bosselut, and Peter Clark. 2018. Reasoning about actions and state changes by injecting commonsense knowledge. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31*

- *November 4, 2018*, pages 57–66. Association for Computational Linguistics.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Shuang Zeng, Runxin Xu, Baobao Chang, and Lei Li. 2020. Double graph based reasoning for document-level relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1630–1640. Association for Computational Linguistics.

Zhihan Zhang, Xiubo Geng, Tao Qin, Yunfang Wu, and Daxin Jiang. 2020. Knowledge-aware procedural text understanding with multi-stage training. *CoRR*, abs/2009.13199.

Chen Zheng and Parisa Kordjamshidi. 2020. SRLGRN: semantic role labeling graph reasoning network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8881–8891. Association for Computational Linguistics.

Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. Reasoning over semantic-level graph for fact checking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6170–6180. Association for Computational Linguistics.