Learning Procedures from Text: Codifying How-to Procedures in Deep Neural Networks

Hogun Park*
Purdue University
West Lafayette, IN, USA
hogun@purdue.edu

Hamid Reza Motahari Nezhad
IBM Almaden Research Center
San Jose, CA, USA
hamid.motahari@ieee.org

ABSTRACT

A lot of knowledge about procedures and how-tos are described in text. Recently, extracting semantic relations from the procedural text has been actively explored. Prior work mostly has focused on finding relationships among verb-noun pairs or clustering of extracted pairs. In this paper, we investigate the problem of learning individual procedure-specific relationships (e.g. *is_method_of*, *is_alternative_of*, or *is_subtask_of*) among sentences. To identify the relationships, we propose an end-to-end neural network architecture, which can selectively learn important procedure-specific relationships. Using this approach, we could construct a how-to knowledge base from the largest procedure sharing-community, *wikihow.com*. The evaluation of our approach shows that it outperforms the existing entity relationship extraction algorithms.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Neural networks**;

KEYWORDS

Procedure Learning, How-to-Knowledge, Neural Network, Relationship Learning

ACM Reference Format:
Hogun Park and Hamid Reza Motahari Nezhad. 2018. Learning Procedures from Text: Codifying How-to Procedures in Deep Neural Networks. In *WWW '18 Companion: The 2018 Web Conference Companion, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/10.1145/3184558.3186347

1 INTRODUCTION

Procedural knowledge has been an important component for many semantic applications such as question answering [7], information retrieval [29], and conversational agents [12]. A lot of knowledge about procedures are described in text. A procedure is comprised of a nested set of inter-related tasks to achieve a goal. For a given procedure, there may be a few alternative methods of achieving the goal, each comprised of a set of tasks. For example, the goal "how to cook Kimchi" is composed of tasks such as "preparing ingredients" and "making the sauce." The task has a set of sub-tasks like "wash your hands" and "dissolve salt," and they are sequentially related to each other. In addition, a task or subtask could include conditional statements and be under specific temporal/spatial constraints. For the given sentence, "If the cabbage is not preserved, keep it in the brine at least 12 hours," we can represent it as <"keep it in the brine at least 12 hours">-*conditional_of*-<"the cabbage is not preserved">. Figure 1 shows an example of procedural text (left) and the corresponding procedure graph (right). The graph has many procedure-specific relationships (e.g. *is_method_of, is_alternative_of*, or *is_subtask_of*.) In this paper, we focus on such procedure-specific relationship extraction for learning a procedure knowledge base.

Learning procedures described in text is a challenging problem. This problem is investigated in prior work through approaches such as extracting or learning patterns [22, 23], learning probabilistic models [8], or hierarchical clustering [4]. Relationship extraction in these works has mostly been in the form of identifying verb/noun pairs in a sentence and their relationship. For example, a relation, *hasNextAction*, could be found among verb/noun pairs like (pump, brake pedal) and (take, spare tire). Thus, the extracted relationships are defined at term-level within a sentence, which are found based on syntactic patterns and rules. In this paper, we investigate ways to identify action-level relationships within and across sentences in text describing procedures. In addition, our proposed method allows finding richer procedural representation by allowing action arguments to model conditions or contexts associated with actions in the text. In addition, our method could prohibit the error propagation from preprocessing steps such as POS Tagging and Dependency Parsing, which are used in the above work.

In this paper, we propose an end-to-end neural relationship extraction model for procedural text, which is the core element of constructing a procedure knowledge base. Recently, neural network models have shown their effectiveness in many different kinds of relationship classification tasks [1, 11, 14, 16, 19, 32–34]. Most models are based on Convolutional Neural Network (CNN) or Recurrent Neural Network (RNN) for extracting either word- or sentence-level relationships. Different from the previously proposed neural network architectures, this paper proposes a hierarchical attention mechanism, which can select out the most important part at the action-level, and at cross-sentence level.

We also propose an alternative architecture based on MemoryNet-augmented neural network to enable task relationships learning utilizing high-level contextual information. Previously, MemoryNet [25][26] has been widely used in question-answering and reading comprehension. In this paper, we employ the concept of MemoryNet's associated memory to learn the underlying structure of

*The work had been done during an internship at IBM Almaden Research Center

WWW '18 Companion, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5640-4/18/04.
https://doi.org/10.1145/3184558.3186347

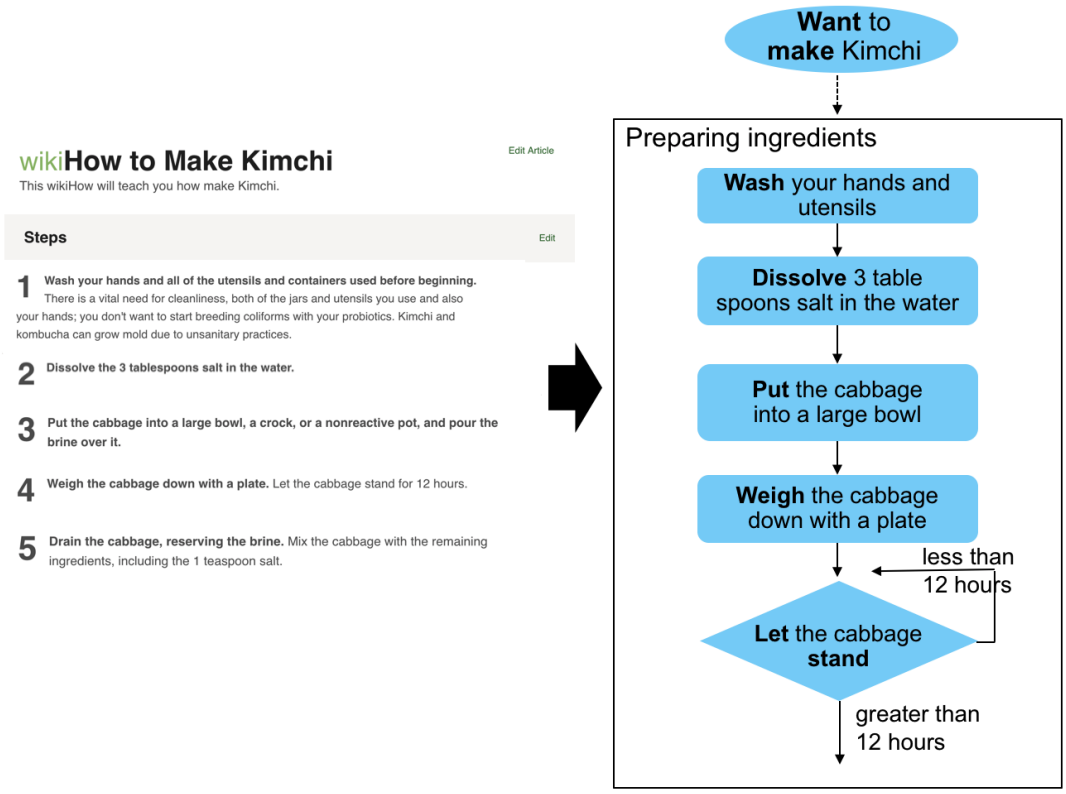

Figure 1: Example of procedural text and Extracted Procedure Graph from the Text

procedures and propose a new method to model and classify relationships. For this, our hierarchical attention encoder is incorporated into the MemoryNet in order to encode the input sentences and contextual information (e.g goal or previous task).

As the dataset to evaluate our proposed architecture, we obtained procedure description data from *wikihow.com* for procedures with relatively long and complete tasks/actions. The proposed methods and architectures are compared to a select set of baselines and prior art alternatives, which demonstrate superior performance of the proposed methods on identifying and classifying procedure-specific relationships.

The rest of paper is structured as follows. Related work is discussed in Section 2. The research problem is defined in Section 3, and the proposed architectures for procedural relationship classification is presented in Section 4. We explain experimental results in Section 5. Finally, we conclude and discuss future work in Section 6.

2 RELATED WORK

2.1 Relation Classification

For entity-relationship extraction and classification, many feature/kernel-based methods [2][31] have been proposed. Recently, [27] and [28] showed that the shortest dependency paths between relation arguments are still useful in neural network architectures like convolutional neural network (CNN) and long short-term memory (LSTM). In particular, it was shown that LSTM-based approach is worse than CNN-based model in their framework [28]. Later, many existing models have been tried to extract entity relationship in a sentence, and most of them rely on complex natural language processing pipelines. To reduce the burden, there were some attempts at extracting relations between entities (e.g., [14] [34]) by utilizing neural networks. However, [14] still uses part-of-speech tags, and [14] [34] do not take full sentences as their input.

Recently, more variants of CNN and RNN have been proposed for relation classification. For example, [1] uses CNN to get the vector representations of sentences by splitting them for in-sentence entity relationship detection. In the work, relationships are learned by three different CNNs,and their output vectors are concatenated before the softmax layer to classify relationships. [32] additionally uses positional vectors to represent the sentence through the convolutional neural network. The positional vectors are attached to the existing word embedding layers and used to indicate target entities for classification. For our objective, it can not be used directly, but, instead, we can still model each sentence using the convolutional neural network without using splitting or positional vector. In this paper, we implemented these architectures for comparison with our proposed methods. Similarly, the architecture proposed in [33], although much simpler in principle, was implemented and used for comparison, as discussed in Section 5.

Discourse relationship classification [11, 16, 19] is also relevant work to our paper. The task of implicit discourse relationship classification is to recognize how two adjacent sentences are associated without explicit discourse markers (e.g. "because"). The paper [19] proposed a recurrent neural model for identifying implicit discourse relations in Chinese text. Their model is based on the single-level bi-directional LSTM model with attention. Our hierarchical attention model is a more generalized neural network by allowing multiple bi-directional LSTM layers. We have compared our models and theirs in our experiments for procedural text. Paper [16] utilizes document graph [18] and learns a Graph LSTM to classify the relationships. However, it is required to have the document graph for the input and use distant supervision from the existing knowledge base, which is not appropriate for our purpose and quite heavy for learning both neural networks. Paper [11] also proposed interesting attention-based models for the implicit discourse relationship, but they were also needed to learn a neural model using the external knowledge base to overcome the lack of training data.

2.2 Procedure Knowledge Mining

Constructing procedural knowledge base has been investigated in [8] and [4]. [8] extracts procedural knowledge from *eHow.com* and *wikiHow.com*. They proposed a framework to detect verbs/actions and objects, and their relationships to build a situation ontology. However, it is required to go through many pre-processing steps like normalization and pattern-matching. Their relationship extraction is also limited to simpler types of relationships such as "hasTopic" and "hasAction". Paper [4] uses Open-IE 4.2 software, which is the successor to ReVerb[5] and Ollie[21] to extract noisy candidates and accomplish hierarchical clustering to find similar tasks. However, they also rely on preprocessing for extracting subject-predicate-object tuples and time/location information. Therefore, its constructed knowledge base is still prone to inaccuracies and rigidity of the information extraction pipeline.

Workflow detection is also related to the procedure knowledge mining. Paper [23] attempted to detect tasks, products, task facets, and control flows based on information extraction pipeline and rules. They also suggested both term- and frame-based approaches that deal with workflow elements. The term-based approach uses Java Pattern Annotation Engine to utilize hand-crafted and domain-specific rules. The frame-based approach is based on predefined patterns to detect a new workflow. Paper [22] also proposed to detect sequences among tasks and control flow. Both are much depending on hand-written rules and patterns to detect workflow and could be limited to previously seen patterns.

2.3 MemoryNet-Augmented Neural Network

MemoryNet was proposed by [25] and [26]. They designed new types of memory representation for modeling and storing a set of sentences which are sequentially related to each other. The earliest recent work proposed an external memory component [26], and they has been applied to many language processing tasks. Their input module computes sentence vectors independently and store/load when it is needed. However, it has a limitation that strong supervision for training stage is needed. To overcome the problem, authors in [26] proposed an end-to-end MemoryNet that

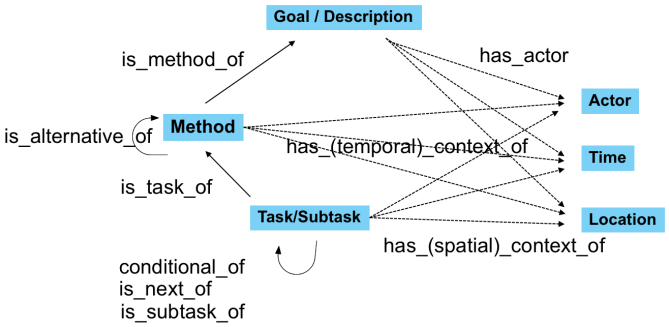

Figure 2: Process-Goal Meta Model

is composed of two main components: supporting memories and predicting answers. Supporting memories get a set of inputs and then generate an output with memory cells, which is stored and updated using external embedding matrices. In this paper, we define a new approach to modeling the underlying structure of procedures using the architecture of MemoryNet. The memory representation is used for augmenting our proposed hierarchical attention neural network.

3 PROBLEM DEFINITION

Let us assume that target sentences, $\boldsymbol{S}$, which contain actions, are identified before relationship classification. In this paper, we have leveraged our earlier work on action identification [15] for identification of target sentences, and the corresponding method is described in Section 5.1. To represent the identified relationships and sentences, we propose a process-goal meta model.

3.1 Process-Goal Meta Model

Procedure [1] text describes different alternative methods, each consisting of their own tasks/actions, arbitrary levels of nesting/hierarchy of tasks, and may describe various context when each method or action is applicable. For modeling a procedure, we propose a process-goal meta model as depicted in Figure 2. Each procedure document can be converted into the connected graph representation. Each procedure has a goal, which is represented as the head of the graph. This is the highest level element in the procedure. It is composed of (alternative) methods, tasks, and sub-tasks in turn. Each goal, method, task, and subtask can have contexts such as time, location, and actor. The contextual information can be extracted by many information extraction tools like named-entity recognition modules and semantic role labeling (out of scope in this paper). In this paper, we focus on the relations in the process-goal meta model. The relations of interest include: *is_method_of, is_alternative_of, is_task_of, is_subtask_of, conditional_of,* and *is_next_of* (Total 6 types.) Even though many previous semantic knowledge bases like ConceptNet [24] include some similar relationships such as "hassubevent" and "hasprerequisite," our process-goal meta model can represent more process-specific, sequential, or hierarchical relationships among tasks.

[1] we use the pair of "process" and "procedure", and the pair of "action" and "task" interachangeably

In the process-goal-meta model, a node is presented by a sentence [2] except actor, time, and location. The reason why the relationship among them is important is that a sentence in procedural text usually describes the step(s) with its pre-condition and post-condition. Therefore, it is reasonable to get sentence representations for nodes. In particular, business procedure (process) for accounting, recruitment, call center, and technical support are often described by domain experts to document the procedures for other workers or for automation. The main challenge is learning the structure of a process, which is expressed in terms of relationships among different tasks/subtasks of the process, as described in the text as a series of steps using sentences. In this paper, we assume that each sentence represents a (sub)task or a method. Moreover, a goal is to understand the procedure including all (alternative) methods, their (sub)tasks, and the relationships among them in the document describing the procedure.

4 NEURAL NETWORK ARCHITECTURES FOR TASK RELATION CLASSIFICATION

4.1 Hierarchical Attention Encoder (HAE)

LSTM [6] is a neural network architecture which can model the state of sequences. The LSTM has a way of "remembering" important information while disregarding the superfluous information through its gating mechanisms. LSTMs have the advantage that it can model variable-sized inputs, while traditional MLPs and other forms of neural networks are only designed for fixed-length inputs.

In this paper, we propose a hierarchical attention neural network using LSTM as in Figure 3. Assume that w_{it} with $t \in [1, T]$ represents the t_{th} word in the i_{th} sentence, s_i. In this architecture, we always have two sentences for relations. The proposed model encodes the raw sentences into a vector representation, which will be used for relationship classification. In the following subsections, we will describe how we encode each word and relation in detail.

Word Encoder. Firstly, the architecture gets words to have word vectors through an embedding matrix, W_e. For the embedding, we use the pre-trained Glove embedding matrix [17]. To get the representation from the word vectors, a bi-directional LSTM layer is used. The bi-directional LSTM contains the forward LSTM that reads from the first word, w_{i1} to the last word, w_{iT}. The backward LSTM reads the word in another direction.

$$
\begin{aligned}
x_{it} &= W_e w_{it}, t \in [1, T], \\
\vec{h}_{it} &= \overrightarrow{LSTM}(x_{it}), t \in [1, T], \\
\overleftarrow{h}_{it} &= \overleftarrow{LSTM}(x_{it}), t \in [T, 1]
\end{aligned}
$$

It concatenates the forward hidden state $\vec{h}_{it}$ and the backward hidden state $\overleftarrow{h}_{it}$. The concatenated vectors, $h_{it} = [\vec{h}_{it}, \overleftarrow{h}_{it}]$ where $t \in [1, T]$, summarize the meaning of a sentence, s_i. This will be summed using our proposed attention mechanism as the following subsection.

[2] In case of "conditional_of", the node could be a phrase.

Word Attention. Attention mechanism is to automatically focus on the words that have a decisive effect on classification. In other words, it is to capture the most important semantic information in a sentence, without using extra knowledge and NLP systems. In this attention layer, it produces a weight vector, u_w, and merge word-level features from each time step into a sentence-level feature vector, by multiplying the weight vector.

$$
\begin{aligned}
u_{it} &= t(W_w h_{it} + b_w), \\
\alpha_{it} &= \frac{exp(u_{it}^T u_w)}{\sum_t exp(u_{it}^T u_w)} \\
k_i &= \sum_t \alpha_{it} h_{it}
\end{aligned}
$$

As in the above equation, we measure the importance of the word as the similarity of u_{it} with a word level context vector u_w and get the normalized weight α_{it} through a softmax function. After that, the weighted sum of words is computed to have the representation for k_i. The word context vector u_w is initialized randomly. This attention mechanism is similar to [30], but we use Glove embedding for handling sparse terminologies and applied the attention mechanism only to the word encoder.

Sentence Encoder. Sentence encoder is to learn the representation of relationship using the sentence vectors in a similar way.

$$
\begin{aligned}
\vec{h}_i &= \overrightarrow{LSTM}(k_i), i \in [1, 2], \\
\overleftarrow{h}_i &= \overleftarrow{LSTM}(k_i), i \in [2, 1] \\
v &= \sum_i^2 h_i \\
y &= \text{softmax}(W_c v + b_c)
\end{aligned}
$$

h_i is the concatenated variable as in $[\vec{h}_i, \overleftarrow{h}_i]$, and then summed to feed to a softmax layer. The output dimension of W_c is the number of relationships.

$$
L = -\sum_r y_r \log \hat{y}_r
$$

Its loss function is defined as above using the output of softmax layer. We use the categorical cross-entropy as the loss function. The loss function will be calculated upon the relation, r. The r is one of relationship labels. Its optimization is accomplished through Adam optimizer [10].

4.2 MemoryNet-augmented Relation Classifier (MARC)

The introduced Hierarchical Attention Network in Section 4.1 models each relation. To encode procedure information, though, we need to model not only the sentence-level relations but also higher-level relation information such as context or hierarchical action information (e.g. Parent Task or Previous Task). In order to do this, we use MemoryNet [25] to augment the contextual, hierarchical

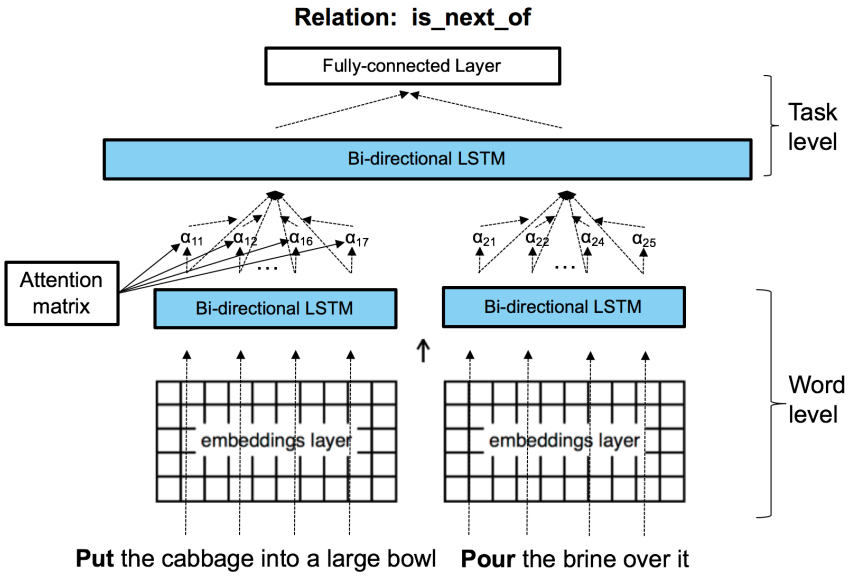

Figure 3: The proposed hierarchical attention neural network

and underlying intrinsic information for relations. The underlying intrinsic information could be domain knowledge or shared common sense. For example, the intrinsic information for recipe could be basic ingredient preparation in commonsense and apply different importance to each words depending on the domain. In this paper, we use the architecture of end-to-end Hierarchical Attention Network [25] as an encoder (HAE) and utilize it to store the encoded vectors in the associated memory in MemoryNet.

$$u = \text{HAE}_1(s_i, s_j)$$
$$m_i = \text{HAE}_2(f_i)$$
$$c_i = \text{HAE}_3(f_i)$$
$$p_i = \text{softmax}(u^T m_i)$$
$$o = \sum_{i=1}^{k} p_i c_i$$
$$\hat{u} = o + u$$
$$\hat{y} = \text{softmax}(W_m \hat{u} + b_m)$$

Assume that we are given input sentences, $S = \{s_i, s_j\}$, and the contextual information of the sentences, s_i, is $f_i = \{f_{i1}, f_{i2}, ..., f_{in}\}$ that is composed of contextual sentences. Again, the contextual information could be goal for method or goal + method for task, for example. Figure 4 shows the architecture of MemoryNet-augmented Relation Classifier (MARC). The above equations also show the internal computation.

The input sentences, $S = \{s_i, s_j\}$, are converted into memory vectors, u. The contextual sentences, f_i are also converted into m_i. We compute the match between u and the corresponding memory m_i by taking the inner product followed by a softmax. HAE_3 returns output memory representation of f_i. The response vector, c_i will be weighted by the probability vector, p_i.

The weighted output, o will be added to the input vector, u, and the softmax activation of $W_m\hat{u} + b_m$ will be followed. The optimization and loss function are same as those in Hierarchical Attention Encoder (HAE). As in [25], it could have more layers for MemoryNet. In other words, the HAE_2 and HAE_3 can be stacked up multiple times. In the figure 4, [1, ..., N] indicates that the two layers could be used multiple times. In this paper, we use a 2-layer memory network architecture. As the architecture has multiple layers, each HAN is shared at each layer.

5 EXPERIMENTS

5.1 Data

In the experiment, we use data from an online community, *wikihow.com* which has been contributed by thousands of users. Wikihow includes abundant informative contents about procedures in many domains such as recipes, finance, cars, education, health, hobbies, personal care, and etc. In total, 198,163 articles were downloaded using the wikiTeam [3] data processing interface. We only used articles which have at least one of each of these entities of interest: goal, method, task, and subtask. "Goal" is from the first sentence of the main description in a procedural description in an article. Similarly, sentences for method, task, and subtask were extracted. The identification of method, task, subtask could be solved using inherent mediaWiki structural conventions (e.g. starting of new child section).

The first sentences of methods and tasks are presented as "verb object (*)" in general. Subtasks may contain many conditional statements or details about the task. Table 1 shows the number of identified relation triples with their underlying statistics. As in the first row of the table, the number of extracted relation triples is

[3] https://github.com/WikiTeam/wikiteam

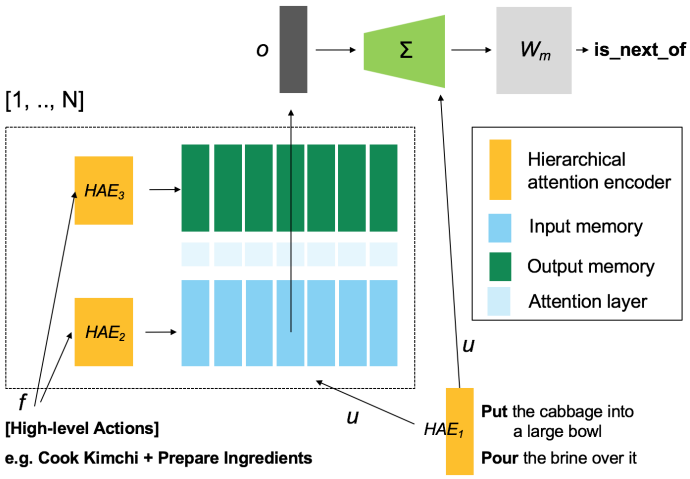

Figure 4: MemoryNet-augmented Relation Classifier

Table 1: Data Statistics: Extracted Relations

Data Statistics	Counts
Number of Extracted Relation Triples	78,217
Avg Number of Methods per Goal	2.15
Avg Number of Tasks per Method	7.57
Avg Number of sub-Tasks per Tasks	3.43

78,217. Here, we counted every triple of sentences and relationships, which is extracted from our Hierarchical Attention Neural Network. For example, <"Dissolve 3 table spoons salt in the water">-*is_next_of*-<"Wash your hands and utensils"> is a triple. To compute the averages in the table 1, each score is summed and averaged over the number of WikiHow's articles. For extracting the *conditional_of*, Stanford dependency parser [3] is used for detecting conditional and consequent clauses. From the parsed tree, sub-nodes of SBAR are utilized to detect the clauses, and sentences were considered if they have the following connectives: IF, UNLESS, EVEN IF, AS (SO) LONG AS, ASSUMING/SUPPOSING, IN CASE, and ONLY IF.

5.2 Methods

To evaluate our method, alternative methods below are considered. As an evaluation metric, we use accuracy, which is calculated by counting how many correct labels are predicted over the total number of possible relation triples. For training and testing below models, we use 70%, 10%, and 20% of procedural articles for training, validation, and testing, respectively. For all neural network models, max epochs are set to 100, and if accuracy on validation dataset does not increase during 15 epochs, it stops training. We also use dropout regularization (0.2) and rectified linear units (ReLU) for activation functions. For optimization, *Adam* optimizer[10] is used to update variables. In the rest of this subsection, models that we used for the experiment are described.

Convolutional Neural Network. The approach in [1] takes a sentence as an input and transforms it into the vector representation through Convolutional Neural Network (CNN) by splitting the sentence into left, middle, and right phrases. They learn three different CNNs and concatenate their outputs at the end to classify entity-to-entity relationship. Authors in [32] additionally employ the concept of positional vector to represent the relationship in the sentence through CNN. In our case, we would like to find the relations among sentences, so we could not use the same approach. Instead, we model the whole sentence using the convolutional neural network without using splitting and positional vector. This approach is well-studied and has shown good performance for sentence classification [9]. We also use max pooling and concatenate in output. The concatenated output feeds fully-connected layer with softmax activation. For parameters for CNN, the number of feature maps is searched over [8, 16, 32, 64], and the size of feature maps is chosen from [3, 5, 7]. Max pooling size is selected from [4, 8, 16].

Bi-directional LSTM. We also use LSTM for the purpose of comparison. As in [33], we use the bi-directional LSTM for modeling sentence. Again, because the method in [33] was also designed for detecting relationship among entities in a sentence, we use natural sentences as inputs of LSTM. The number of hidden nodes is chosen by grid search. The grid search is selected from [50, 100, 200].

Hierarchical Bi-directional LSTM. This is to verify how the attention matrix works for relation classification. This is the architecture which does not have attention-based activation. To search its appropriate parameters, the number of hidden nodes is searched from [50, 100, 200].

Hierarchical Bi-directional LSTM with Attention. This is one of our proposed models, which is labeled as (Our Model1) in the result section below. This model uses the word attention, and its performance is related and compared for effectiveness. Here, to search appropriate parameters, the number of hidden nodes for LSTM is also searched from [50, 100, 200].

End-to-end MemoryNet. This model is adapted to our procedure learning task to evaluate the performance of the end-to-end MemoryNet [25]-like neural model. The original end-to-end MemoryNet

Table 2: Result of Relation Classification

Approaches	Accuracy
Convolutional Neural Network	0.73
Bi-directional LSTM	0.75
Hierarchical Bi-directional LSTM	0.869
Hierarchical Bi-directional LSTM with Attention (Our Model 1)	0.878
MemoryNet-augmented Relation Classifier (Our Model 2)	0.874

Table 3: Result of Relation Classification on Smaller Procedures for Training and Testing (10 Percent of Randomly Sampled Procedural Articles)

Approaches	Accuracy
Hierarchical Bi-directional LSTM with Attention (Our Model 1)	0.829
End-to-end Neural Network	0.74
MemoryNet-augmented Relation Classifier (Our Model 2)	0.849

is designed for reading comprehension. Instead of the given background sentences in the task, we use all the high-level information, f_i, and the first sentence, s_i, to train embedding and attention variables. In addition, the second sentence, s_j, is used for input query, instead of the question in the reading comprehension task. For a fair comparison, we used HAE encoders for embedding matrices and trained using same parameters with our MemoryNet-augmented Relational Classifier's ones.

MemoryNet-augmented Relational Classifier. This classifier is the other proposed model in this paper (Our Model 2). This model employs the encoder, which is proposed as *Hierarchical Bi-directional LSTM with Attention*, to model each sentence for feeding inputs or embedding vectors in MemoryNet. The three different encoders (HAE_1, HAE_2, and HAE_3) are trained separately. For their parameter search, the number of hidden nodes for LSTM is also selected from [50, 100, 200].

5.3 Results

Table 2 shows results of experiments. In the experiment, CNN-based relation classifier showed the worst result. When we used the Bi-directional LSTM, it was able to get a better result than the CNN's one. In addition to the Bi-directional LSTM, a layer of word-level LSTM was additionally considered, the neural network got the additional significant gain. This means that the idea of hierarchy is crucial to have better classification results.

In the experiment, our proposed model (Hierarchical Bi-directional LSTM with Attention) showed the best result for the classification. It indicates that the attention model is useful to detect the important words for modeling each relation triple. We also tested whether the additional attention matrix for the second Bi-directional LSTM layer is needed or not, but it did not show any improvement. MemoryNet-augmented Relation Classifier shows 0.874 in accuracy in the dataset, but slightly less than Hierarchical Bi-directional LSTM with Attention (Our model 1). This implies that it is enough to model the procedural text using one HAN encoder when we have enough training data.

However, when we used 10 percent of total procedural articles, which are randomly chosen, MemoryNet-augmented Relation Classifier got better than the model 1 (Hierarchical Bi-directional LSTM with Attention) as shown in Table 3. This means MemoryNet-augmented Relation Classifier has an advantage when it is trained by smaller training data. This result aligns with the previous observation, i.e [13] and [20]. Authors in [20] reported that the MemoryNet-Driven Neural Network is effective at one-shot learning, and multiple MemoryNets [13], which maximize the effect of single MemoryNets, also showed a better performance on Question/Answering Tasks. When our Model 2 was compared to End-to-end Neural Network, it was also significantly better in accuracy. This shows that our HAE encodes the relation well and is more effective only if the contextual information is stored in the memory.

6 CONCLUSION AND FUTURE WORK

In this paper, we presented a neural network architecture for extracting procedure-specific relationships. Unlike the previous work, our methodology does not rely on information extraction pipeline for identifying task relationships. Rather, it learns rich procedure-specific relations like the sequential, conditional and hierarchical relationship among goals, methods, and tasks by directly feeding sentences in the text of procedure descriptions (also in a hybrid, attention-aware manner) into a novel neural network architecture designed for this purpose. In experiments, the proposed methods yielded superior results compared to the state of the art methods for relationship extraction. In particular, MemoryNet-augmented Classifier demonstrates great performance using a relatively small size of training data.

As future work, we plan to investigate methods that use the proposed architectures for learning procedures knowledge bases in question-answering scenarios. Moreover, our work can be extended to learn more complex and chained action relationships from process and procedure descriptions.

REFERENCES

[1] Heike Adel, Benjamin Roth, and Hinrich Schütze. 2016. Comparing convolutional neural networks to traditional models for slot filling. *arXiv preprint arXiv:1603.05157* (2016).

[2] Razvan C. Bunescu and Raymond J. Mooney. 2005. A Shortest Path Dependency Kernel for Relation Extraction. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT '05)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 724–731. https://doi.org/10.3115/1220575.1220666

[3] Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 740–750.

[4] Cuong Xuan Chu, Niket Tandon, and Gerhard Weikum. 2017. Distilling Task Knowledge from How-To Communities. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 805–814. https://doi.org/10.1145/3038912.3052715

[5] Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. 2011. Open Information Extraction: The Second Generation.. In *IJCAI*, Vol. 11. 3–10.

[6] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[7] Konrad Höffner, Sebastian Walter, Edgard Marx, Ricardo Usbeck, Jens Lehmann, and Axel-Cyrille Ngonga Ngomo. 2017. Survey on challenges of question answering in the semantic web. *Semantic Web* 8, 6 (2017), 895–920.

[8] Yuchul Jung, Jihee Ryu, Kyung-min Kim, and Sung-Hyon Myaeng. 2010. Automatic Construction of a Large-scale Situation Ontology by Mining How-to Instructions from the Web. *Web Semant.* 8, 2-3 (July 2010), 110–124. https://doi.org/10.1016/j.websem.2010.04.006

[9] Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP '14)*. 1746–1751.

[10] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[11] Man Lan, Jianxiang Wang, Yuanbin Wu, Zheng-Yu Niu, and Haifeng Wang. 2017. Multi-task Attention-based Neural Networks for Implicit Discourse Relationship Representation and Identification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 1299–1308.

[12] Lijun Mei, Qicheng Li, Rangachari Anand, Juhnyoung Lee, Feng Li, and Shaochun Li. 2014. Enabling customizable service-based procedural knowledge applicationsâĂŤAn service-orientation for Dialog Manager. In *Service Operations and Logistics, and Informatics (SOLI), 2014 IEEE International Conference on*. IEEE, 412–417.

[13] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126* (2016).

[14] Makoto Miwa and Mohit Bansal. 2016. End-to-end Relation Extraction using LSTMs on Sequences and Tree Structures. *Proceedings of ACL* (2016).

[15] Hamid R. Motahari Nezhad, Kalpa Gunaratna, and Juan Cappi. 2017. eAssistant: Cognitive Assistance for Identification and Auto-Triage of Actionable Conversations. In *Proceedings of the 26th International Conference on World Wide Web Companion (WWW '17 Companion)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 89–98. https://doi.org/10.1145/3041021.3054147

[16] Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-Sentence N-ary Relation Extraction with Graph LSTMs. *Transactions of the Association for Computational Linguistics* 5 (2017), 101–115.

[17] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[18] Chris Quirk and Hoifung Poon. 2017. Distant supervision for relation extraction beyond the sentence boundary. *Proceedings of ACL* (2017), 1171–1182.

[19] Samuel Rönnqvist, Niko Schenk, and Christian Chiarcos. 2017. A Recurrent Neural Model with Attention for the Recognition of Chinese Implicit Discourse Relations. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 256–262. https://doi.org/10.18653/v1/P17-2040

[20] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. One-shot learning with memory-augmented neural networks. *arXiv preprint arXiv:1605.06065* (2016).

[21] Michael Schmitz, Robert Bart, Stephen Soderland, Oren Etzioni, et al. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 523–534.

[22] Pol Schumacher and Mirjam Minor. 2014. Extracting control-flow from text. In *Information Reuse and Integration (IRI), 2014 IEEE 15th International Conference on*. IEEE, 203–210.

[23] Pol Schumacher, Mirjam Minor, Kirstin Walter, and Ralph Bergmann. 2012. Extraction of Procedural Knowledge from the Web: A Comparison of Two Workflow Extraction Approaches. In *Proceedings of the 21st International Conference on World Wide Web (WWW '12 Companion)*. ACM, New York, NY, USA, 739–747. https://doi.org/10.1145/2187980.2188194

[24] Robert Speer and Catherine Havasi. 2013. ConceptNet 5: A large semantic network for relational knowledge. In *The PeopleâĂŹs Web Meets NLP*. Springer, 161–176.

[25] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*. 2440–2448.

[26] Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory networks. *arXiv preprint arXiv:1410.3916* (2014).

[27] Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015. Semantic relation classification via convolutional neural networks with simple negative sampling. *arXiv preprint arXiv:1506.07650* (2015).

[28] Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015. Classifying Relations via Long Short Term Memory Networks along Shortest Dependency Paths.. In *EMNLP*. 1785–1794.

[29] Zi Yang and Eric Nyberg. 2015. Leveraging Procedural Knowledge for Task-oriented Search. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15)*. ACM, New York, NY, USA, 513–522. https://doi.org/10.1145/2766462.2767744

[30] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J Smola, and Eduard H Hovy. 2016. Hierarchical Attention Networks for Document Classification. In *HLT-NAACL*. 1480–1489.

[31] Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel Methods for Relation Extraction. *J. Mach. Learn. Res.* 3 (March 2003), 1083–1106. http://dl.acm.org/citation.cfm?id=944919.944964

[32] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. 2014. Relation Classification via Convolutional Deep Neural Network.. In *COLING*. 2335–2344.

[33] Shu Zhang, Dequan Zheng, Xinchen Hu, and Ming Yang. 2015. Bidirectional Long Short-Term Memory Networks for Relation Classification.. In *PACLIC*.

[34] Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint Extraction of Entities and Relations Based on a Novel Tagging Scheme. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1227–1236.