IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received June 29, 2020, accepted July 22, 2020, date of publication July 31, 2020, date of current version August 17, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3013237

Learning a Planning Domain Model From Natural Language Process Manuals

YONGFENG HUO[1,2], JING TANG[3], YINGHUI PAN[4], YIFENG ZENG[5], AND LANGCAI CAO[1,2]

[1]Department of Automation, Xiamen University, Xiamen 361005, China
[2]Xiamen Key Laboratory of Big Data Intelligent Analysis and Decision-Making, Xiamen 361005, China
[3]Newcastle Business School, Northumbria University, Newcastle upon Tyne NE1 8QH, U.K.
[4]College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China
[5]Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne NE1 8QH, U.K.

Corresponding authors: Yinghui Pan (yinghuipan.uk@gmail.com) and Langcai Cao (langcai@xmu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772442, Grant 61836005, Grant 11671335, Grant 61562033, and Grant 61806089.

ABSTRACT Artificial intelligence planning techniques have been widely used in many applications. A big challenge is to automate a planning model, especially for planning applications based on natural language (NL) input. This requires the analysis and understanding of NL text and a general learning technique does not exist in real-world applications. In this article, we investigate an intelligent planning technique for natural disaster management, e.g. typhoon contingency plan generation, through natural language process manuals. A planning model is to optimise management operations when a disaster occurs in a short time. Instead of manually building the planning model, we aim to automate the planning model generation by extracting disaster management-related content through NL processing (NLP) techniques. The learning input comes from the published documents that describe the operational process of preventing potential loss in the typhoon management. We adopt a classical planning model, namely planning domain definition language (PDDL), in the typhoon contingency plan generation. We propose a novel framework of FPTCP, which stands for a *Framework of Planning Typhoon Contingency Plan*, for learning a domain model of PDDL from NL text. We adapt NLP techniques to construct a ternary template of sentences of NL inputs from which actions and their objects are extracted to build a domain model. We also develop a comprehensive suite of user interaction components and facilitate the involvement of users in order to improve the learned domain models. The user interaction is to remove semantic duplicates of NL objects such that the users can select model-generated actions and predicates to better fit the PDDL domain model. We detail the implementation steps of the proposed FPTCP and evaluate its performance on real-world typhoon datasets. In addition, we compare FPTCP with two state-of-the-art approaches in applications of narrative generation, and discuss its capability and limitations.

INDEX TERMS Domain learning, PDDL, typhoon contingency plan.

I. INTRODUCTION

Artificial intelligence (AI) planning techniques are increasingly playing a supporting role in the society from our daily life to public operations where decisions or policies need to be made in a complex environment. For example, people use intelligent dialogue robot assistants, e.g. *Cortana* to facilitate their daily needs. and *DuerOS*, A wildlife security system is deployed for predicting poaching threats and planning ranger patrols to combat poaching in a forest [1]. The planning techniques analyse a surrounding environment to reasoning among a number of alternative actions, given resource constraints and related constraints according to an intended goal, and to comprehensively optimise an action sequence to achieve a goal [2]. AI planning provides considerable flexibility and potential explanatory power thereby having been applied in many types of real-time systems [3]–[6].

A number of AI planning techniques have been developed in the past of several decades since they are the core where AI origins. They are ranged from a classical

The associate editor coordinating the review of this manuscript and approving it for publication was Victor Hugo Albuquerque.

https://www.microsoft.com/en-us/cortana
https://dueros.baidu.com/en/index.html

logic-based AI planner e.g. STanford Research Institute Problem Solver (STRIPS) [7], belief models for dealing with observational uncertainty e.g. partially observable Markov decision process [8], to more explicit representations of probabilistic graphical models e.g. influence diagrams, interactive dynamic influence diagrams and so on [9].

Most the planning techniques still demand significant inputs from domain experts so that the planning models can be built in a problem domain. The tedious model construction is often unavailable and tends to be inaccurate in encoding the domain knowledge therefore compromising the planning quality. In this paper, we choose the planning domain definition language (PDDL) [10] - a classical planning model that is well developed in the AI community, and study the domain learning that automates the PDDL model construction. PDDL becomes a reliable tool for many applications and is continuously supported by academics and practitioners [11]. There has seen a line of research on the PDDL domain learning most of which still considers a structured input of action sequences in a problem domain [12]. In contrast, we investigate the domain learning from natural language inputs e.g. official documents, public news and statements, and focus on its applications in generating nature disaster contingency plans. As the document often describes a sensible process that deals with the corresponding planning problem, we refer the document full of natural language text as a natural language process manual.

Considering the application of generating a typhoon contingency plan, we notice that the planning requires a significant amount of cooperation among various departments. A manual generation of the planning model becomes rather difficult and tedious particularly when a typhoon often arrives in a fast pace. Hence, automating the planning model generation is extremely important in a time-critical situation. The PDDL model can fully represent a typhoon planning problem and has defined clear semantic components to be learned in the model generation.

One important component in the PDDL domain learning is to extract a sequence of actions from the text input. For example, given a paragraph of a typhoon control news -

"The municipal flood control office promptly forwarded the typhoon news to all districts and member units, and reported to the municipal party committee, municipal government and the leadership of the command department.",

we may expect to extract a sequence of actions of *forward* (the municipal flood control office, typhoon news), *reported* to (municipal party committee), *reported* to (municipal government), *reported* to (the leadership of the command department), which are key in a typhoon planning model. Nevertheless, this expectation may be too glorious to be easily and automatically achieved. There are generally two ways to perform automatic action extractions. One is based on restricted action templates and using natural language processing (NLP) tools directly. For example, Hayton *et al.* [13] used Learning Object Centered Models (LOCM) [14] to complete the story reconstruction via extracting action sequence from natural language (NL) stories through Stanford CoreNLP tools. However, the extraction is not accurate as shown in our experiments. The other method is to extract actions in a free NL text. For example, Feng *et al.* [15] proposed to extract action sequences from texts based on a deep reinforcement learning framework for several public datasets, which needs much labor cost to label a dataset. This is not applicable to a typhoon planning document due to the lacking of labels for sentences in the document.

In this article, we deal with an official typhoon plan document and assume that each action in the inputs is meaningful and correct. Hence, the extracted actions are essential in a PDDL domain model. Subsequently, we can transfer the problem of extracting action sequences into the problem of sequence labelling and dependency parsing for an application consideration. Under this setting, we resort to NLP approaches to handle the above sequence labelling problem, and propose an intelligent planning solution based on learning domain from NL inputs. We adopt a bidirectional long-short term memory-conditional random field (BiLSTM-CRF) model [16] to address the sequence labeling problem since it performs extremely well in NL part-of-speech (POS)-tagging and dependency parsing fields. We also make a further step to apply bidirectional encoder representations from transformers (BERT) model [17] to extract feature factors for each word of which the output can be directly fed into the BiLSTM-CRF model. By doing this, we can build a structured representation of a document and keep a user in an interaction to optimise the final output of the PDDL model.

The main contributions of this work are summarized as follows.

- We implement a prototype system of FPTCP (Framework of Planning Typhoon Contingency Plan) to automate the learning of a PDDL domain model from a NL process manual.
- We design and develop an interaction component so that users can provide useful knowledge to refine the PDDL model for a practical application.
- We conduct comprehensive experiments on two official datasets and show that the new techniques are superior to the state-of-the-arts on the PDDL domain learning.
- We investigate how the domain learning can be transferred between different cities in the typhoon planning development. It leads to the sharing of valuable experience in avoiding significant loss in the face of a natural disaster.

The rest of the article is organised as follows. Section II reviews related works on PDDL domain learning and discuss their difference. Section III elaborates background knowledge of a PDDL model through a toy example. Section IV presents the novel FPTCP that develops the domain learning in a typhoon plan application. Extensive experiments are conducted in Section V to demonstrate the performance of

FPTCP. Section VI concludes the work and discusses directions of future research.

II. RELATED WORKS

An automatic generation of domain models is a popular topic in the field of automated AI planning [18]–[22]. Extracting actions from the NL text input is undoubtedly a major challenge in acquiring domains [23]. The mapping of SAIL route descriptions to action sequences has generated a great interest in the NLP community [24]. Most of the previous work [25]–[29] relies heavily on manually given rules such as learned dictionaries. Recently, Mei *et al.* [30] used the LSTM-based auto-encoder model to automate the learning; however, the model does not process multiple sentences simultaneously. It can only extract action sequences from a single sentence. A series of research has been conducted on learning a domain model through reinforcement learning techniques that are explored in the NLP research [31]. This line of work belongs to a supervised learning paradigm and demands inputs on labelling sentences, which is however not available in our problem domain.

A lot of research has been done on learning to represent actions [7], [32], [33]. For example, Sil *et al.* [34], Sil and Yates [35] used text correlation techniques to identify text containing words that represent the target action and then applied inductive learning techniques to identify appropriate pre-and-post-conditions for the actions. The approach is able to learn action representations although facing deficiencies such as the unidentified number of predicate parameters. Branavan *et al.* [36] introduced an action learning model based on reinforcement learning and learned cues from text about conditional relationship pairs based on surface linguistics. However, the performance depends on feedback rewards that are automatically obtained from attempts in a plan execution.

Another line of related work is in the study of narrative generation and various methods have tended to use (semi-)automated techniques to collect story content such as crowdsourcing, weblogs and story libraries. For example, Li *et al.* [37] acquired typical story elements that can be assembled into a plot diagram for a story generation. Sina *et al.* [38] built a database of everyday activity scenarios and possible replacement scenarios for use in fixed game contexts. Nazar and Janssen [39] hand-drew annotated logs for user conversations in restaurant games to facilitate an automatic generation of character interactions with human participants in a voice-based narrative environment. There is also a way to get the content of the narrative by mining weblogs and story libraries. For example, Swanson and Gordon [40] selected random stories from weblogs to perform text-based interactions with users and obtained satisfied content. Sil *et al.* [34] analysed a syntactic structure of narrative sentences from web contents by using off-the-shelf toolkits, e.g., OpenNLP and Stanford CoreNLP. Hayton *et al.* [13] used LOCM to complete the story reconstruction after extracting an action sequence through NLP tools. They also provided a specific method to domain model acquisition for domains that require non-technical experts to create content to populate their models [41]. Janghorbani *et al.* [23] developed virtual agents that can obtain preconditions and effects from a compound of NL sentences.

In this article, instead of extracting actions from NL sentences directly, e.g., [13] and [42], we convert the extraction into a sequence labelling problem, i.e., POS tagging, for generating a typhoon contingency plan. This is partially because the document we encountered is a public official one in which all sentences have a similar semi-regular patterns, which leads to the novel design of action representations we proposed in Section IV-C. To the best of our knowledge, this is the first domain learning technique that turns the action extraction problem into a POS tagging problem. The new technique also performs well in other applications such as a narrative generation.

III. BACKGROUND KNOWLEDGE OF PDDL

A problem-solving system usually consists of three parts: knowledge representation, a data structure that stores knowledge and knowledge reasoning. Knowledge representation is the premise of knowledge reasoning. A planner can be seen as a problem-solving system, and naturally it also consists of three parts: definition of a planning problem, a data structure that stores the plan (such as a plan map, state space map, etc.) and search of a plan. The definition of a planning problem is the premise of solving a planning problem. If a planning problem cannot be defined through a planning language, no planner would solve it. Hence, PDDL was proposed to define a planning problem in a rigorous way [10]. The PDDL model not only gives the syntax of the planning problem definition, but also gives the definition of the planning from a semantic perspective. It has a strong expression ability and can describe time and numerical attributes in the planning problem.

A complete PDDL program is composed of a domain file and a problem file. Before introducing them in detail, we describe some common data members in PDDL below.

- **Objects.** Things we are interested in during the planning process.
- **Initial State.** The state of the world that we begin with.
- **Goal Specification.** The status of each object we aim to reach.
- **Predicates.** Descriptions of the properties of objects we are interested in and can be either *True* or *False*.
- **Actions/Operators.** For the operation of objects, the action should contain three descriptions, i.e., the object involved, the state of the object before the action and the state of the object after the action.

Note that an initial state, a goal specification and objects are three parts of a problem file, and others form a domain file. We will illustrate them through a toy example. Suppose that there are two rooms R_a and R_b and there are two balls B_1 and B_2. Both balls are in room R_a. We have a robot Anna in

```
(define (domain gripper-strips)
    (:predicates (room ?r)
                 (ball ?b)
                 (at ?b ?r)
                 (at-robby ?r)
                 (gripper ?g)
                 (free ?g)
                 (carry ?b ?g))
    (:action pick
        :parameters(?obj ?room ?gripper)
        :precondition
            (and
                (ball ?obj) (room ?room)
                (gripper ?gripper) (free ?gripper)
                (at ?obj ?room) (at-robby ?room)
            )
        :effect
            (and
                (not
                    (free ?gripper)
                )
                (carry ?obj ?gripper)
                (not (at ?obj ?room))
            )
        )
```

FIGURE 1. An example for a domain file in a PDDL model.

```
(define (problem solve)
    (:domain gripper-strips)
    (:objects
        rooma roomb ball1 ball2 left right
    )
    (:init
        (room rooma)
        (room roomb)
        (ball ball1)
        (ball ball2)
        (gripper left)
        (gripper right)
        (free left)
        (free right)
        (at ball1 rooma)
        (at ball2 rooma)
        (at-robby rooma)
    )
    (:goal
        (and
            (at ball1 roomb)
            (at ball2 roomb)
        )
    )
)
```

FIGURE 2. An example for a problem file in a PDDL model.

R_a and Anna has two arms A_l and A_r. The planning problem is on how to let Anna move the two balls from R_a to R_b.

A. DOMAIN FILE

A domain file consists of *Predicates* and *Actions*. As shown in Figure 1, there are seven properties of objects we are interested in, e.g., determining if it is a room, determining which room the balls are in and determining if the Anna's arms are free. Besides, there are three actions - "pick","move" and "drop" - for the operation of given objects. After setting up thee actions and predicates, we can obtain a suitable domain file for a PDDL model.

B. PROBLEM FILE

A problem file contains objects, an initial state and a goal specification. As illustrated in Figure 2, there are six objects we are interested in, e.g., room R_a, ball B_1 and left arm A_l. The initial state has been described in the previous hypothesis and so has the goal specification. Compared to a domain file, a problem file specification is more straightforward since it directly comes from requirements of users in a problem domain.

Once we have both the domain and problem files, we can build and run the PDDL model, and find a plan for the robot Anna. We notice that a domain file is the key to a PDDL model since it encodes action specification in a problem domain and requires the understanding of a problem domain. Instead of manually crafting the domain file, we aim to automate the domain file specification given the input of natural language (NL) text, which is coined as the domain learning in a PDDL model. Specifically, we expect to obtain predicates and actions through NLP tools as much as possible so as to automate the PDDL model development.

IV. FPTCP: A FRAMEWORK OF PLANNING TYPHOON CONTINGENCY PLAN

Currently, the transformation from NL input to a domain model is a semi-automated process, in which a user is constantly tweaking the learning over multiple stages in order to prevent ambiguity. To reduce the manual effort, we develop a more automatic framework for learning a PDDL domain model particularly in the development of typhoon contingency plan. In this section, we will first present this framework and then discuss technical details in this novel framework.

A. THE FPTCP OVERVIEW

We show our FPTCP in Figure 3. The flow of the FPTCP can be divided into five parts.

① First, the input of NL text (in terms of a set of sentences) are fed into a BERT model for segmentation and are mapped to latent feature vectors (i.e., BERT for word2vec). The obtained word vectors are subsequently fed into a BiLSTM-CFR model for POS recognition.

② The input NL sentences are turned to action ternaries in which only main elements of original sentences remain based on the outputs of the POS-tagging results. Subsequently, the action representations and their pre- and post-conditions can be obtained through the methods in Section IV-C.

③ Next is a user interaction stage. Users are asked to eliminate semantic ambiguities and remove duplicates. They can help to pick out appropriate predicates for both pre- and post-conditions if they would like to and finally provide a formal problem file containing a goal specification.

④ After the user interaction, both the domain and problem files are ready to complete the inputs to the PDDL model.

⑤ The user can solve the PDDL model and retrieve an optimal plan for the problem of interest. Many well-developed tools can be used to solve the PDDL model, which is not the scope of our work. We can represent the output of the optimal plan as one domain model, which provides solutions to the planning problem of interest.

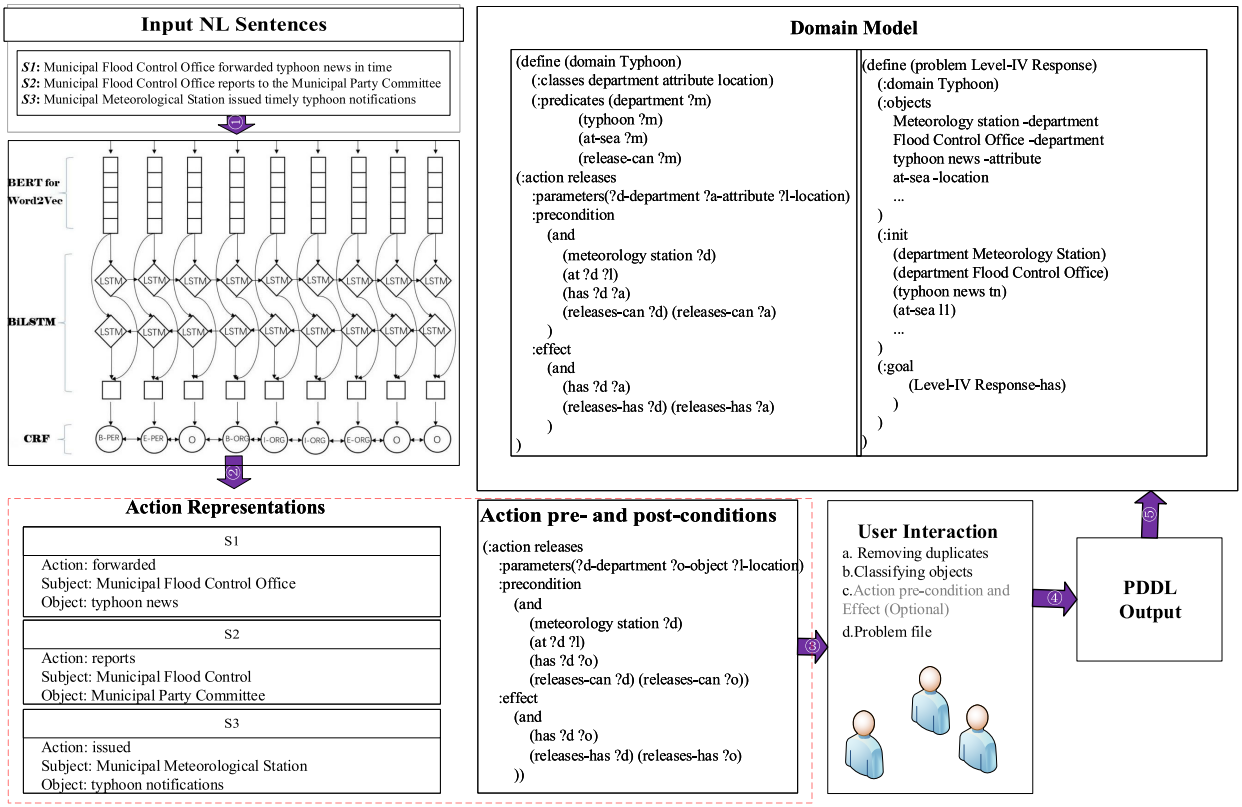

FIGURE 3. A framework of planning typhoon contingency plan (FPTCP) contains five main parts from learning action sequence from NL text to refining the PDDL model with interaction with users.

B. BERT-BiLSTM-CRF MODEL

We adapt the BERT, BiLSTM and CRF models in FPTCP. The BERT model can either subdivide words or form vector representations of sentences by mapping them into semantic feature spaces. Meanwhile, the BiLSTM model can bi-directionally mine the semantic relationships between words in a sentence while CRF can focus on contextual labelling information compared to other models. By exploiting the benefits of the three models, we propose a BERT-BiLSTM-CRF approach to build an action representation in the next section. Its architecture, which consists of three parts, is between ① and ② in Figure 3.

We use the BERT pre-trained language model [17] to better capture words, which adopts a bidirectional transformer as an encoder and can also in parallel decode the input encoder with a satisfactory efficiency. The loss function for a word-level classification task (i.e., Mask-LM task) of BERT is defined below.

$$L(\theta, \theta_w) = -\sum_{i=1}^{M} \log p(w = w_i|\theta, \theta_w), m_i \in [1, 2, \ldots, |V|]$$

where θ is the parameter in the encoder part of BERT and θ_w is the parameter in the output layer of the Mask-LM task that is connected on the Encoder. Thus, if the set of words that are masked is M, it becomes a multiple classification problem on a vocabulary size $|V|$.

For each input sentence, a sentence containing n words is denoted as $X = (x_1, \ldots, x_i, \ldots, x_n)$ and its corresponding label set is denoted as $y = (y_1, \ldots, y_i, \ldots, y_n)$, where x_i represents the i-th word of the current sentence in the vocabulary and y_i is i-th label for word x_i. The first part is word embedding the BERT layer that can embed each word x_i in a d-dimensional dense factor vector $x_i \in R^d$ based on the pre-trained encoder.

The second part is word labeling BiLSTM layer that is able to capture feature factors for its input sentence. The input word sequence is denoted as $X = (x_1, \ldots, x_i, \ldots, x_n)$ and subsequently the output sequence of the forward long-short term memory (LSTM) model $(\overrightarrow{h_1}, \ldots, \overrightarrow{h_i}, \ldots, \overrightarrow{h_n})$ and corresponding output sequence of the backward LSTM $(\overleftarrow{h_1}, \ldots, \overleftarrow{h_i}, \ldots, \overleftarrow{h_n})$ are aggregated at the site $h_s = \left[\overrightarrow{h_s}, \overleftarrow{h_s}\right] \in R^l$ which forms the total output sequence $(h_1, \ldots, h_i, \ldots, h_n) \in R^{n \times l}$ of latent factors. The output sequence will go through a linear layer to make the dimension l of the latent factor be the same as the given dimension k that denotes the number of labels. Under this setting, we can obtain a score matrix $P = (p_1, \ldots, p_i, \ldots, p_n) \in R^{n \times k}$ where $P_{i,j}$ denotes the score of the j-th label upon the i-th word.

The third part is a CRF layer for adding restrictions when a label is transferred. It guarantees that the output labels are valid by learning the adjacencies between them. We can

also obtain a matrix A of transfer score where $A_{i,j}$ denotes a transfer score from the label i to the label j. y_0 and y_n are the beginning and end labels of a sentence and are added into the label set. Hence, A satisfies $A \in R^{k+2,k+2}$.

For a sentence, we can define the score of labels y upon words X as

$$s(X, y) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=0}^{n} P_{i,y_i}$$

where $s(X, y)$ represents the score. The score for the whole sentence equals to the sum of the score for each place that consists of two parts: one is determined by the output P of the BiLSTM layer and the other is based on the transfer matrix A of the CRF layer. Then, given the sentence, the probability of getting the label y is computed as follows.

$$p(y|X) = \frac{\exp s(X, y)}{\sum_{\widetilde{y} \in Y_X} \exp s(X, \widetilde{y})} \quad (1)$$

where $p(y|X)$ denotes the probability and Y_X represents all the possible label sequences corresponding to the sentence X. Each possible sequence of labels corresponding to sentence X has a different score or a probability. Our aim is to maximise the probability that the sentence corresponds to a true label sequence, i.e., maximising p(y|X). However, in general, the loss function is minimised and the original function in Eq. (1) is therefore transformed to a likelihood pattern below.

$$\log p(y|X) = \log \exp s(X, y) - \log \sum_{\widetilde{y} \in Y_X} \exp s(X, \widetilde{y}) \quad (2)$$

$$= s(X, y) - \log \sum_{\widetilde{y} \in Y_X} \exp s(X, \widetilde{y}) \quad (3)$$

where maximising $\log p(y|X)$ is to minimise

$$\log \sum_{\widetilde{y} \in Y_X} \exp s(X, \widetilde{y})$$

because $s(X, y)$ in Eq. (3) is a fixed value. Thus, the final loss function is calculated in Eq. (4),

$$Loss = \log \sum_{\widetilde{y} \in Y_X} \exp s(X, \widetilde{y}) \quad (4)$$

We can solve Eq. (4) via the *Viterbi* algorithm [43].

C. BUILDING ACTION REPRESENTATIONS

The first step in FPTCP is to build templates that are simplified representations of the NL input sentences, i.e., (subject, predicate, object) triples in our design. The triples can represent all of the actions or attributes described in the original sentence, as well as the objects and their roles mentioned in the original sentence. To extract triples from NL sentences, we resort to the aforementioned BERT-BiLSTM-CRF model for syntactic analysis to complete named entity recognition and POS tagging. Among them the closest to our actual needs is the function of POS tagging, which can be divided into three parts. Note that for both BERT and BiLSTM-CRF, we use the parameter settings that achieved best performance in their original papers.

The first part is to perform word embeddings based on BERT model. One important reason why we choose this model is that it comes with a word segmentation mechanism and the segmentation accuracy approaches 100%. Through the word embedding, we can obtain latent low-dimensional semantic representations of each word and also the entire sentence. Each word can be represented by a hidden feature space vector of a given dimension. In the second part, we input the sentence represented by the latent vector into the BiLSTM model, and we can obtain the POS prediction labels for all words in the sentence according to its output. To greatly reduce the invalid sequences in the prediction results, we further input the output of the second part into a CRF model, because it can add some restrictions to the final prediction labels to ensure that the output results are valid and more importantly these restrictions can be automatically learned by the CRF layer during the training process, which will not bring more labor costs.

FPTCP creates an action template ternary based on the model output, which uses verb as an action name and contains all related objects. Figure 4 illustrates an example of a POS result for an NL input. In the figure, the label that contains ''NN'' represents a noun and it generally denotes a possible candidate subject or object (and modifier) for a current domain. The label that contains ''V'' (expect for ''VBN'') represents a verb and ''BZ'' here denotes a third person singular form. It generally indicates a possible candidate action. The ''JJ'' label represents an adjective. It usually acts as a modifier. We do not need to focus on the ''DT'' label since it seems like an article. Notice that the subject of this sentence is ''city meteorology station'', the predicate verb is ''releases'' and the object is ''typhoon message'' of which the modifier is ''orange''. Hence, FPTCP can extract an action ternary from the sentence according to the result. Note that the modifiers will also be added into the action ternary to remain the detail information of the subject or object. The output for this sentence will be (''city meteorology station'',''releases'',''orange typhoon message'') and this ternary contains the key elements of the sentence.

For each action ternary, FPTCP will mark each associated object as a candidate parameter for the output action. During a user interaction phase, the following generic categories will be used to type the categories: department, typhoon-attributes and location. On the other hand, few methods can be used to identify pre- and post-conditions. Inspired by the work [44], we first directly add default predicates to the pre- and post-conditions of the obtained actions, namely (action-can? m) and (action-has? m) to describe a basic level of effect. Secondly, different from the work [44], we face the challenge that when a department completes all its actions and immediately needed to start the next department's actions, the output results of POS tags can not provide any useful information to help the department state transition. We achieve the state transition by implicitly reading the department name at the beginning of the sentence. Under this setting, the relationship is sufficient to ensure that a baseline plan is generated that

FIGURE 4. POS tagging of an example sentence.

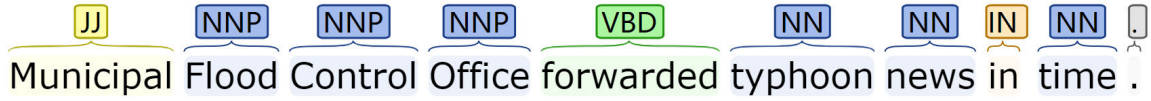

FIGURE 5. POS tagging of S1 sentence.

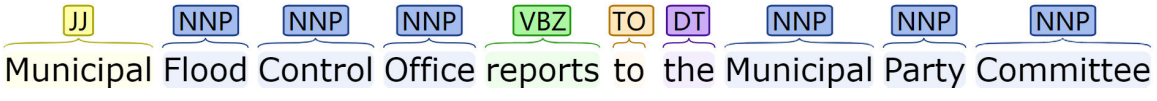

FIGURE 6. POS tagging of S2 sentence.

FIGURE 7. POS tagging of S3 sentence.

corresponds to the NL sentence of the original input. For example, it may be effective to adopt some of those predicates as stated in a goal specification. Other predicates are added by FPTCP when a PDDL domain file is output after a user interaction.

D. USER INTERACTION

User interaction is important since it could be used to validate a learned domain and ensure the correct input to a PDDL model. We describe main needs from a user in order to refine the FPTCP output.

1) REMOVE DUPLICATES

Different expressions in NL sentences can reflect the same meaning, which is still a hard problem in NLP research. We need users to pick out terms that are different, but have the same meaning, for a disambiguation purpose. Consequently, we can ensure that there are no two different terms that have the same meaning in a single domain.

2) CLASSIFY OBJECTS

We need users to divide objects into different categories. In our settings, objects would be divided into three types, i.e., department, typhoon-attributes and location. Department category is necessary because if a department completed all the actions and immediately needed to start the next department's actions, we need the department information to connect actions between the two departments. In other words, if an action, involving with a state transition, is completed, its post-conditions must contain the pre-condition of one of the next department's actions.

3) ACTION PRE-CONDITION AND EFFECT (Optional)

We need users to think and decide pre-conditions and effects of actions in a domain. In our settings, there are two initial predicates with each action and this consideration may not be comprehensive. Hence a user may help to identify predicates to make the pre-conditions more suitable. Besides, a sentence may not always express location attributes of objects whereas we made a rule that an object must always come with a location attribute. Hence the user should help to clarify all the attributes if FPTCP does not obtain the information from NL inputs. At last, as long as an action has multiple parameters of the same class, FPTCP will make these parameters not be equal and the user could add a pre-condition such as (not (= $?m_1$ $?m_2$)).

4) PROBLEM FILE

We need to specify a goal in order to generate a plan in a problem domain. Hence users are asked to set up both an initial state and a goal state for a planning problem. Each predicate detected by FPTCP is true and indicates a possible initial state or a planning goal. FPTCP will show a user many initial states and some of them will not be appropriate. We need the user to screen out possible initial states or specify other any possible ones that are not extracted by FPTCP. For example, a user is often asked for the selection of predicates that give location information of departments in the initial state. For a goal state, FPTCP expects the same from the users as the former.

E. PDDL OUTPUT AND PLANNING

In the last step, we create a domain file using actions and predicates obtained by FPTCP. The user interaction stage provides necessary information on actions and classes of a domain object. By considering the actual situation of a typhoon contingency plan, FPTCP can also add some other necessary parameters to the action at this stage since all the actions should be performed in the right location. Therefore,

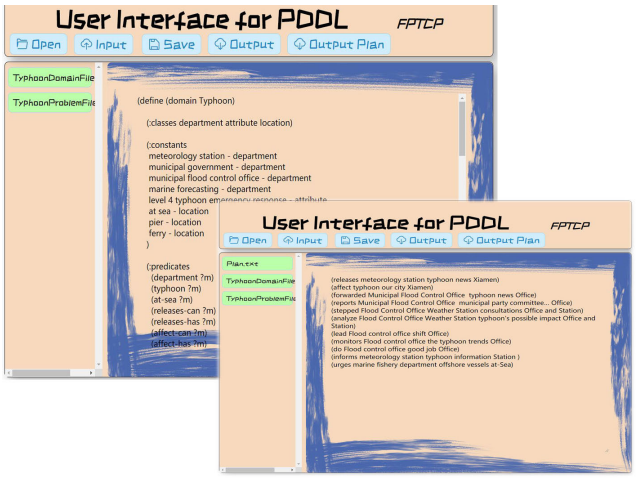

FIGURE 8. The user interface for modifying PDDL domain and probem files and generating an optimal plan in a problem domain.

Actions	Predicates	Objects
releases affect forwarded reports consult analyze work monitor do inform urge	releases-can affect-has forwarded-has strengthen-consultation strengthen duty do well ...	city meteorology station meteorology station municipal government municipal flood-control office marine forecasting level 4 typhoon-emergency response at sea pier ferry city-pier

Removing duplicates

(city meteorology station, meteorology station)
(city-pier, pier)

Classifying objects

city meteorology station . . . → department
level 4 typhoon . . . → typhoon-attributes
at sea → location

FIGURE 9. Results of the step of building action representations through FPTCP. It illustrates the sets of action names, action predicates and their objects before the step user interaction. The current step is a fully automatic. In the middle of the above figure, it shows the results of removing duplicates by a user interaction where the results are coloured in blue and the duplicates are coloured in red. At the bottom it shows the results of classifying objects into three types.

location parameters are automatically added to those actions that do not have the location information in the NL text.

For a domain file, the objects and parameters of an action can be obtained from a ternary template, and the user may also add the categories of action parameters if necessary. The pre-conditions and effects are by default composed of predicates extracted from the NL text, which are subsequently considered and selected by the user to ensure reasonableness. After building the domain and problem files, we use the tool developed by C. Muise[3] to solve the PDDL planner.

[3]Editor:http://planning.domains/

```
1  (:action releases
2      :parameters(?d-department ?a-attribute ?l-location)
3      :precondition
4          (and
5              (meteorology station ?d)
6              (at ?d ?l)
7              (has ?d ?a)
8              (releases-can ?d) (releases-can ?a)
9          )
10     :effect
11         (and
12             (has ?d ?a)
13             (releases-has ?d) (releases-has ?a)
14         )
15 )
```

FIGURE 10. Example of FPTCP generating action `releases`. After user interaction, the parameters for object names stated in NL text are replaced with variables of the corresponding classes. The pre-conditional predicates are selected by a user (line 5) and the location and default predicates recommended by FPTCP are accepted by the user (lines 6-8). In terms of post-conditions, the user also accepted the location and default predicates (lines 12-13).

V. EXPERIMENTS

We implement FPTCP with the integration of the PDDL tool[3], and develop a user interface (in Figure 8) to facilitate user interaction in running the planner. The system receives the input of NL text, allows users to refine the learned domain and problem files, and runs the PDDL planer to provide a final plan. We will evaluate the FPTCP performance in generating a level 4 typhoon contingency plan given public documents for typhoon contingency plan that was executed in the past. The documents are published by a forecast office in two cities, namely Xiamen and Shenzhen in China.

We organise three sets of experiments to demonstrate the FPTCP performance. We first test FPTCP on learning domain models from the existing documents of the typhoon contingency plan in Xiamen and then investigate the transferring of planning knowledge from Xiamen to another city Shenzhen given the learned domain model. Finally, we compare FPTCP with two state-of-the-art approaches on learning domain models for narrative generation purpose.

A. EXPERIMENT 1: DOMAIN LEARNING FOR TYPHOON CONTINGENCY PLAN IN XIAMEN

In this experiment, we focus on the tests for a typhoon contingency plan in Xiamen. There are four domains in the published documents and each domain represents a level of typhoon that occurred in Xiamen. We choose level 4 typhoon contingency plan in our tests.

1) THE AUTOMATIC PROCESS BY FPTCP

FPTCP parses the NL text of a typhoon contingency plan document and provides the sequence labelling results of each sentence in the document. Based on the results, we can extract the ternary of subject, predicate verb and object as action and its objects. It is worth mentioning that FPTCP supports processing multiple sentences simultaneously and returns the sentences with annotations respectively. Taking the following input sentences as an example:

S1 "Municipal Flood Control Office forwarded typhoon news in time"

Output plans	Original NL sentences
(releases, meteorology station, typhoon news, Xiamen)	The city meteorology station releases a typhoon news.
(affect, typhoon, our city, Xiamen)	affect may affect our city in the future.
(forwarded, Municipal Flood Control Office , typhoon news, Office)	Municipal Flood Control Office forwarded the typhoon news to all districts in a timely manner
(reports, Municipal Flood Control Office , municipal party committee…, Office)	And reports to the municipal party committee, city government and command leadership
(stepped, Flood Control Office Weather Station, consultations, Office and Station)	City Flood Control Office, City Weather Station stepped up consultations
(analyze, Flood Control Office Weather Station, typhoon's possible impact, Office and Station)	Analyze the extent of the typhoon's possible impact on the city
(lead, Flood control office, shift, Office)	Flood control office leaders lead the shift
(monitors, Flood control office, the typhoon trends, Office)	Closely monitors the typhoon trends
(do, Flood control office, good job, Office)	Do a good job of uploading and sending
(informs, meteorology station, typhoon information, Station)	Meteorology station informs the typhoon information to the city flood control office
(urges, marine fishery department, offshore vessels, at-Sea)	The marine fishery department urges offshore vessels to do windproof and safety work

FIGURE 11. FPTCP generates the plan results for the Xiamen city. The corresponding NL sentences are also shown in the right-hand side.

S2 "Municipal Flood Control Office reports to the Municipal Party Committee"

S3 "Municipal Meteorological Station issued timely typhoon notifications"

Figures 5 to 7 illustrate the POS tagging results of sentences S1, S2 and S3, respectively. We subsequently extract actions and objects from the above sentences according to the steps in Section IV-C. We list the obtained action names and their objects and parameters in Table 1. For example, for the sentence S1 in Figure 5, we first pick out the words continuously labeled as "NN"- and "V"-, i.e., "Flood Control Office forwarded typhoon news" here. Then, we take the "NN"- labeled words that appear before the word labeled as "V"- (i.e., words labeled "NNP") as the first term of the action ternary. Naturally, the word labeled "VBD" becomes the second term and the last remaining word labeled "NN"- which appears after "V"- (i.e., words labeled "NN") is the third term. At this point, the action ternary becomes ("Flood Control Office","forwarded","typhoon news") that almost expresses the meaning of the original sentence S1. Last but not least, we add modifiers to the subject and object to avoid semantic loss if they exist. The modifier in S1 is the word labeled "JJ". Thus, the output ternary for S1 is ("Municipal Flood Control Office","forwarded","typhoon news").

For the sentence ternary, FPTCP identifies the main action factors and parameters, and simultaneously adds the *m*-can and *m*-has predicates to pre- and post-conditions of an action to guarantee its effectiveness. This default setting of predicates of FPTCP is necessary and reasonable because for the application case the predicates generally only contain information about whether an action is completed or not.

After an automatic process on the documents, the sets of action names and their objects, corresponding predicates and domain objects are shown in Figure 9.

2) THE MANUAL INPUTS FROM USERS

As we mentioned above, we still demand the inputs from users to refine the FPTCP outputs. First, a user should pick out duplicate objects that express the same meaning in different ways and remove redundant duplicates. For example, as shown in Figure 9, there are some duplicates in the third column of actions objects, such as pier and city-pier. The users are asked to pick one of them and remove extra duplicates. After duplicate problems are resolved, users can categorise the types of objects. In our experiments, we limit the selection of departments, typhoon-attributes and locations. For the current domain, the main task of user interaction is to modify objects, i.e., classifying different objects into the given classes. The results are also illustrated in Figure 9

Output plans	Original NL sentences
(strengthen, Municipal Third Defense Command 24-hour duty, MTDC)	Municipal Third Defense Command (MTDC): strengthen the 24-hour duty;
(track, Municipal Third Defense Command, typhoon information , MTDC)	track typhoon information throughout the day;
(meet, Municipal Third Defense Command, Municipal Meteorological Bureau, MTDC)	meet with the Municipal Meteorological Bureau;
(issue, Municipal Third Defense Command, defense notice, MTDC)	issue a defense notice;
(deploy, Municipal Third Defense Command, city, MTDC)	and deploy the city for coming typhoon.
(report, Municipal Meteorological Bureau, typhoon information , MMB)	Municipal Meteorological Bureau (MMB): timely report typhoon information such as center location, intensity, direction of movement, and speed to the MTDC
(command, City Marine Bureau, fishing boats , at-sea) (take, fishing boats , shelter, at-sea)	City Marine Bureau (CMB): command fishing boats to take shelter from the wind;
(check, City Marine Bureau, offshore fishing personnel , CMB)	and check offshore fishing personnel ashore.
(issue, Municipal Housing Construction Bureau, early warning information , MHCB)	Municipal Housing Construction Bureau (MHCB): issue early warning information for the city's building construction units,
(urge, Municipal Housing Construction Bureau, them, MHCB)	urge them on defense work in strict accordance with the relevant safety norms;
(urge, Municipal Housing Construction Bureau, property management agencies , MHCB)	urge property management agencies through the broadcast,;
(post, Municipal Housing Construction Bureau, wind and flood prevention reminders, MHCB)	post wind and flood prevention reminders;
(remind, Municipal Housing Construction Bureau, residents, MHCB)	remind residents of defense;
(urge, Municipal Housing Construction Bureau, gas companies , MHCB)	and urge gas companies on inspection and other defense preparation.
(urge, Municipal Education Bureau, primary and secondary schools, kindergartens and nurseries , School)	Municipal Education Bureau (MEB): urge primary and secondary schools, kindergartens and nurseries for defense;
(eliminate, Municipal Education Bureau, safety hazards, School)	eliminate safety hazards;
(suspend, Municipal Education Bureau, outdoor teaching activities, School)	suspend outdoor teaching activities;
(ensure, Municipal Education Bureau, safety, School)	and ensure the safety of students in schools

FIGURE 12. FPTCP generates a level 4 typhoon contingency plan for another city - Shenzhen.

TABLE 1. The actions obtained from sentences S1- S3 by FPTCP.

Sentence	Action Name	Action Objects	Pre-condition	Post-condition
S1:	forwarded	Municipal Flood Control Office, typhoon news	forwarded-can	forwarded-has
S2:	reports	Municipal Flood Control Office, Municipal Party Committee	reports-can	reports-has
S3:	issued	Municipal Meteorological Station, typhoon notifications	issued-can	issued-has

in which the sections that have user interaction have been coloured. Notice that the parts that require the user inputs is relatively trivial since FPTCP has provided reasonably correct information in this application.

Second, a user could refine the pre-conditions and post-conditions of the output action by supplementing and modifying the appropriate predicates with background knowledge of the problem domain. The complement of predicates could consider the following aspects.

- Predicates are identified by the typhoon-attributes from the NL text and
- Predicates about locations must be defined by the user if they are not represented in the NL sentences.

As shown in Figure 10, the user is able to autonomously select or modify predicates of the action `releases` and finally constructs an output action. It is worth mentioning that FPTCP is able to output an effective complete action representation except for line 5 in Figure 10, which also demonstrates the FPTCP advantages.

The last step of user interaction uses the predicates to design both the initial and goal states to complete a problem file. In our setting, we aim to rebuild the plan through the above predicates to compare with the original typhoon contingency plan to demonstrate the effectiveness of the proposed FPTCP.

3) PLANNING OUTPUT

After an automatic FPTCP process and the user refinement, we can obtain the domain files according to the action set and predicate set, which includes a domain file and a problem file. The final stage of FPTCP is to adopt an appropriate planner to produce a sequence of actions from an initial state to a goal state in the problem file. We apply an online planner [3] to the domain and problem files to generate a new typhoon contingency plan which includes 11 actions illustrated in Figure 11, along with their NL sentences, respectively. In the left-hand side, a set of 11 actions are generated by the obtained domain model while in the right-hand side, we show the original inputs of the NL sentences (based on which we learn the domain model). From the results, we can see that the output from FPTCP is almost the same as the original typhoon contingency plan in the document.

B. EXPERIMENT 2: TRANSFERRING PLANS TO SHENZHEN BY FPTCP

We conduct the second experiment to test whether the learned domain model could be re-used to generate a typhoon contingency plan for other cities. This is to transfer the knowledge of a typhoon contingency plan among different cities. We extend the pre-trained model (for the typhoon contingency plan in the Xiamen city) to learn new domain and problem files in order to provide a typhoon contingency plan in the Shenzhen city (close to Xiamen and is a typhoon hotspot in the South of China). We run FPTCP upon the Shenzhen typhoon contingency plan document [4] and generate a level 4 typhoon contingency plan in Figure 12. The left-hand side is a part of the output plan and the right-hand side is the corresponding NL sentences in the dataset. We observe that the FPTCP output plan is almost the same as the original contingency plan document although missing some multiple compound objects such as ''defence work'' for action `urge`. This is mainly because multiple compound objects generally do not appear consecutively in the sentence and there are prepositions between them, which dis-enables their role in the sentence in terms of a grammatical composition. The experimental results demonstrate potential capability of FPTCP in transferring knowledge between different applications.

"Timmy died.
Carl the shopkeeper healed Timmy using his medicine.
Hank stole antivenom from the shop, which angered Sheriff William.
Hank healed his son Timmy using the stolen antivenom.
Sheriff William shot Hank for his crime.
Hank intended to heal his son Timmy using the stolen antivenom.
Sheriff William intended to shoot Hank for his crime.
Hank got bitten by a snake.
Hank intended to heal himself using the stolen antivenom."

FIGURE 13. The sentences are commonly used in an old American west story.

C. EXPERIMENT 3: COMPARISON WITH STATE-OF-THE-ARTS

We evaluate FPTCP against two state-of-the-art PDDL domain learning methods over the commonly used benchmark - an old American west story [45]. We extract the inputs of NL descriptions in Figure 13. One of the competing methods is called *StoryFramer* [13] that takes the NL input for building a story-telling domain through NLP techniques while the other *Ware* is a planning model for narrative generation and inference [45]. For all the methods, the key is to extract actions and corresponding objects from the NL text since they are the main components in building a domain file. We show the results by the three methods (FPTCP, StoryFramer and Ware) in Figure 14. By comparing the results to the original sentences in Figure 13, we observe that the proposed FPTCP can extract all the actions from the given text while neither StoryFramer nor Ware could

[4] http://www.sz.gov.cn/shuiwj/qt/yjgl

Plan	Ware Actions	StoryFramer Actions	FPTCP Actions (this paper)
1	(die, Timmy)	(died, timmy)	(died, Timmy)
2	(heal, Carl Timmy)	(healed, carl-shopkeeper timmy medicine)	(healed, Carl shopkeeper, Timmy) (using, Carl shopkeeper, medicine)
3	(shoot, Hank Timmy)	(shot, timmy hank)	(shot, Hank, son Timmy)
4	(steal, Hank Antivenom Carl William)	(stole, hank sheriff-william antivenom)	(stole, Hank, antivenom) (angered, Hank, Sheriff William)
5	(heal, Hank Antinvenom Timmy)	(healed, hank timmy antivenom)	(healed, Hank, son Timmy) (using, Hank, stolen antivenom)
6	(shoot, William Hank)	(shot, sheriff-william hank)	(shot, Sheriff William, Hank)
7	(heal, Hank Antivenom Timmy)	(healed, hank timmy antivenom) (intended, hank)	(intended to heal, Hank, son Timmy) (using, Hank, stolen antivenom)
8	(shoot, William Hank)	(intended, sheriff-william)	(intended to shoot, Sheriff William, Hank)
9	(snakebite, Hank)	(bitten, hank)	(got , Hank, bitten snake)
10	(heal, Hank Antivenom Hank)	(healed, hank timmy antivenom) (intended to heal, hank)	(intended to heal, Hank, himself) (using, Hank, stolen antivenom)

FIGURE 14. Performance comparison on action extractions between our FPTCP and its competing methods.

extract the full action sequence. For example, StoryFramer does not obtain the action `shoot` (line 8) and its objects *Sheriff William and Hank*, and Ware does not extract the action `intended` (lines 7,8 and 10). Moreover, neither StoryFramer nor Ware could extract the actions of `using` (lines 2 and 7) and `angered` (lines 4), which is mainly due to the fact that the CoreNLP tool used in StoryFramer does not have a sufficient recognition accuracy for identifying actions and corresponding objects from the NL sentences directly. In contrast, FPTCP extracts actions in an indirect POS tagging manner for which the BERT-BiLSTM-CRF model has an adequate recognition accuracy, not to mention the Ware model.

The results, together with the experimental study of the typhoon contingency plan in both Xiamen and Shenzhen cities, demonstrate that FPTCP perform satisfactorily in generating plans in applications of a natural disaster prevention and it outperforms the state-of-the-art methods in learning the domain models for a narrative generation.

VI. DISCUSSION AND CONCLUSION

We proposed a novel framework, namely FPTCP, to automate the domain learning in a PDDL model, and particularly focused on generating typhoon plans from the inputs of NL process manuals. This is the first attempt at extracting action sequences for learning a domain model by using sequence labelling techniques in NLP. FPTCP adopts the BERT-BiLSTM-CRF model to solve the sequence labelling problem. We develop a comprehensive set of rules to extract action ternaries from the output of sequence labelling and automate the action extraction process for AI planning. In the empirical study, we have demonstrated the effectiveness of FPTCP by comparing the its output plan with the ground-truth. The new method can also see the benefit of transferring knowledge between different application scenarios. Although the framework targets at the application of generating typhoon contingency plans, we also showed its utility in domain learning for narrative generation - another popular application where a PDDL planner is often used.

As we are deploying this framework in a practical application of generating typhoon contingency plans, there is a need to further reduce user interaction and develop a more automatic process. However, we shall notice that a semi-automatic procedure is still expected since the user interaction is necessary in many AI planning systems, e.g., user can help to input a goal specification and remove semantic duplicates in our new framework. We will continue to look into more intelligent ways of identifying actions accurately in complex compound sentences. This will immediate alleviate manual effort from users in a complex, time-critical application. One possible way to solve this problem is to do grammar parsing, e.g., using *Stanford Parser*[5] to do dependency syntactic analysis and output the dependency relationship of a sentence.

We will also be keen to explore knowledge transfer such as from the Xiamen city to the Shenzhen city through the AI planning approach and share disaster management experience in various regions. To this end, we may consider adopting disaster emergency plans from multiple regions as input to gradually establish a large domain that contains much more

[5]https://nlp.stanford.edu/software/lex-parser.shtml

disaster relief information - subsequently using the large domain model to build more diverse plans.

For a general direction of learning an AI planning model, we consider to generalise FPTCP to automate other planning models e.g. partially observable Markov decision process and influence diagrams. We have seen similar thoughts in the latest work [46] on learning influence diagrams from data generated in a problem domain. We could perceive further improvement of their techniques by using FPTCP to identify more relevant variables in the planing model. This will contribute to the learning of AI planning models in a wide community.

ACKNOWLEDGMENT

The authors would like to thank Dr. Q. Zhou and M. Tang for valuable inputs to the work.

REFERENCES

[1] A. Perrault, F. Fang, A. Sinha, and M. Tambe, "AI for social impact: Learning and planning in the data-to-deployment pipeline," 2020, *arXiv:2001.00088*. [Online]. Available: https://arxiv.org/abs/2001.00088

[2] S. Sohrabi, "AI planning for enterprise: Putting theory into practice," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 6408–6410.

[3] S. Peng, H. Chen, C. Du, J. Li, and N. Jing, "Onboard observation task planning for an autonomous earth observation satellite using long short-term memory," *IEEE Access*, vol. 6, pp. 65118–65129, 2018.

[4] H. Qie, D. Shi, T. Shen, X. Xu, Y. Li, and L. Wang, "Joint optimization of multi-UAV target assignment and path planning based on multi-agent reinforcement learning," *IEEE Access*, vol. 7, pp. 146264–146272, 2019.

[5] J. Porteous, F. Charles, and M. Cavazza, "Networking: Using character relationships for interactive narrative generation," in *Proc. Int. Conf. Auton. Agents Multi-Agent Syst.*, 2013, pp. 595–602.

[6] C. W. Coley, D. A. Thomas, III, J. A. M. Lummiss, J. N. Jaworski, C. P. Breen, V. Schultz, T. Hart, J. S. Fishman, L. Rogers, H. Gao, R. W. Hicklin, P. P. Plehiers, J. Byington, J. S. Piotti, W. H. Green, A. J. Hart, T. F. Jamison, and K. F. Jensen, "A robotic platform for flow synthesis of organic compounds informed by AI planning," *Science*, vol. 365, no. 6453, p. 557, 2019.

[7] R. E. Fikes and N. J. Nilsson, "Strips: A new approach to the application of theorem proving to problem solving," *Artif. Intell.*, vol. 2, nos. 3–4, pp. 189–208, Dec. 1971.

[8] A. R. Sankar, P. Doshi, and A. S. Goodie, "Evacuate or not? A POMDP model of the decision making of individuals in hurricane evacuation zones," in *Proc. 35th Conf. Uncertainty Artif. Intell.*, 2019, pp. 234–235.

[9] Y. Zeng and P. Doshi, "Exploiting model equivalences for solving interactive dynamic influence diagrams," *J. Artif. Intell. Res.*, vol. 43, pp. 211–255, Feb. 2012.

[10] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins. (1998). *PDDL—The Planning Domain Definition Language*. [Online]. Available: http://www.cs.yale.edu/homes/dvm

[11] M. Fox and D. Long, "PDDL2.1: An extension to PDDL for expressing temporal planning domains," *J. Artif. Intell. Res.*, vol. 20, pp. 61–124, Dec. 2003.

[12] D. N. M. Ghallab and P. Traversoz, *Automated Planning: Theory and Practice*. Amsterdam, The Netherlands: Elsevier, 2004.

[13] T. Hayton, J. Porteous, J. Ferreira, A. Lindsay, and J. Read, "StoryFramer: From input stories to output planning models," in *Proc. Workshop Knowl. Eng. Planning Scheduling, 27th Int. Conf. Automat. Planning Scheduling*, 2017, pp. 1–9.

[14] S. Cresswell, T. L. McCluskey, and M. West, "Acquisition of object-centred domain models from planning examples," in *Proc. 19th Int. Conf. Automat. Planning Scheduling*, 2009, pp. 19–23.

[15] W. Feng, H. H. Zhuo, and S. Kambhampati, "Extracting action sequences from texts based on deep reinforcement learning," 2018, *arXiv:1803.02632*. [Online]. Available: http://arxiv.org/abs/1803.02632

[16] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," 2018, *arXiv:1802.05365*. [Online]. Available: http://arxiv.org/abs/1802.05365

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*. [Online]. Available: http://arxiv.org/abs/1810.04805

[18] M. A. Hawas, "Are we intentionally limiting urban planning and intelligence? A causal evaluative review and methodical redirection for intelligence systems," *IEEE Access*, vol. 5, pp. 13253–13259, 2017.

[19] M. Asai and A. Fukunaga, "Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary," 2017, *arXiv:1705.00154*. [Online]. Available: http://arxiv.org/abs/1705.00154

[20] Z. Liu, T. Yang, N. Sun, and Y. Fang, "An antiswing trajectory planning method with state constraints for 4-DOF tower cranes: Design and experiments," *IEEE Access*, vol. 7, pp. 62142–62151, 2019.

[21] R. Jilani, "Automated domain model learning tools for planning," in *Knowledge Engineering Tools and Techniques for AI Planning*. Cham, Switzerland: Springer, 2020, pp. 21–46.

[22] J. Porteous, J. F. Ferreira, A. Lindsay, and M. Cavazza, "Extending narrative planning domains with linguistic resources," in *Proc. 19th Int. Conf. Auton. Agents Multiagent Syst.*, 2020, pp. 1081–1089.

[23] S. Janghorbani, A. Modi, J. Buhmann, and M. Kapadia, "Domain authoring assistant for intelligent virtual agent," in *Proc. 18th Int. Conf. Auton. Agents MultiAgent Syst.*, 2019, pp. 104–112.

[24] M. MacMahon, B. Stankiewicz, and B. Kuipers, "Walk the talk: Connecting language, knowledge, and action in route instructions," *Def*, vol. 2, no. 6, p. 4, 2006.

[25] D. L. Chen, "Fast online lexicon learning for grounded language acquisition," in *Proc. 50th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012, pp. 430–439.

[26] J. Kim and R. J. Mooney, "Unsupervised PCFG induction for grounded language learning with highly ambiguous supervision," in *Proc. Joint Conf. Empirical Methods Natural Lang. Process. Comput. Natural Lang. Learn.* Stroudsburg, PA, USA: Association for Computational Linguistics, 2012, pp. 433–444.

[27] J. Kim and R. Mooney, "Adapting discriminative reranking to grounded language learning," in *Proc. 51st Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2013, pp. 218–227.

[28] W. Jing, C. F. Goh, M. Rajaraman, F. Gao, S. Park, Y. Liu, and K. Shimada, "A computational framework for automatic online path generation of robotic inspection tasks via coverage planning and reinforcement learning," *IEEE Access*, vol. 6, pp. 54854–54864, 2018.

[29] Z. Jiao, P. Yao, J. Zhang, L. Wan, and X. Wang, "Capability construction of C4ISR based on AI planning," *IEEE Access*, vol. 7, pp. 31997–32008, 2019.

[30] H. Mei, M. Bansal, and M. R. Walter, "Listen, attend, and walk: Neural mapping of navigational instructions to action sequences," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 2772–2778.

[31] H. H. Zhuo and S. Kambhampati, "Model-lite planning: Case-based vs. model-based approaches," *Artif. Intell.*, vol. 246, pp. 1–21, May 2017.

[32] M. Pomarlan, S. Koralewski, and M. Beetz, "From natural language instructions to structured robot plans," in *Proc. Joint German/Austrian Conf. Artif. Intell.* Cham, Switzerland: Springer, 2017, pp. 344–351.

[33] H. H. Zhuo, J. Peng, and S. Kambhampati, "Learning action models from disordered and noisy plan traces," 2019, *arXiv:1908.09800*. [Online]. Available: https://arxiv.org/abs/1908.09800

[34] A. Sil, F. Huang, and A. Yates, "Extracting action and event semantics from Web text," in *Proc. AAAI Fall Symp. Ser.*, 2010, pp. 1–6.

[35] A. Sil and A. Yates, "Extracting strips representations of actions and events," in *Proc. Int. Conf. Recent Adv. Natural Lang. Process.*, 2011, pp. 1–8.

[36] S. R. Branavan, H. Chen, L. S. Zettlemoyer, and R. Barzilay, "Reinforcement learning for mapping instructions to actions," in *Proc. Joint Conf. 47th Annu. Meeting ACL, 4th Int. Joint Conf. Natural Lang. Process. AFNLP*, vol. 1. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, pp. 82–90.

[37] B. Li, S. Lee-Urban, G. Johnston, and M. Riedl, "Story generation with crowdsourced plot graphs," in *Proc. 27th AAAI Conf. Artif. Intell.*, 2013, pp. 598–604.

[38] S. Sina, A. Rosenfeld, and S. Kraus, "Generating content for scenario-based serious-games using crowdsourcing," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 522–529.

[39] R. Nazar and M. Janssen, "Combining resources: Taxonomy extraction from multiple dictionaries," in *Proc. 7th Int. Conf. Lang. Resour. Eval.*, 2010, pp. 1–7.
[40] R. Swanson and A. S. Gordon, "Say anything: Using textual case-based reasoning to enable open-domain interactive storytelling," *ACM Trans. Interact. Intell. Syst.*, vol. 2, no. 3, pp. 1–35, Sep. 2012.
[41] T. Hayton, J. Porteous, J. F. Ferreira, and A. Lindsay, "Narrative planning model acquisition from text summaries and descriptions," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 1–8.
[42] A. Lindsay, J. Read, J. F. Ferreira, T. Hayton, J. Porteous, and P. Gregory, "Framer: Planning models from natural language action descriptions," in *Proc. 27th Int. Conf. Automat. Planning Scheduling*, 2017, pp. 1–9.
[43] A. J. Viterbi, "A personal history of the viterbi algorithm," *IEEE Signal Process. Mag.*, vol. 23, no. 4, pp. 120–142, Jul. 2006.
[44] K. Yordanova, "From textual instructions to sensor-based recognition of user behaviour," in *Proc. Companion Publication 21st Int. Conf. Intell. User Interfaces (IUI Companion)*, 2016, pp. 67–73.
[45] S. G. Ware *et al.*, "A plan-based model of conflict for narrative reasoning and generation," Ph.D. dissertation, Dept. Comput. Sci., North Carolina State Univ., Raleigh, NC, USA, 2014.
[46] A. R. Sankar, P. Doshi, and A. Goodie, "Evacuate or not? A POMDP model of the decision making of individuals in hurricane evacuation zones," in *Proc. 35th Conf. Uncertainty Artif. Intell. (UAI)*, 2019, pp. 234–238.

YONGFENG HUO received the Bachelor of Engineering degree from Xiamen University, China, where he is currently pursuing the degree with the Department of Automation. His current research interests include recommendation systems and natural language processing.

JING TANG received the Ph.D. degree from Nanyang Technological University, Singapore, in 2006. She is currently a Senior Lecturer with the Newcastle Business School, Northumbria University, U.K. Her publications has appeared in CEC, GECCO, the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATIONS, and so on. Her research interests include evolutionary algorithms, memetic algorithms, and artificial intelligence.

YINGHUI PAN received the Ph.D. degree from Xiamen University, in 2012. She was an Associate Professor with the Jiangxi University of Finance and Economics, China. She is currently a Research Professor with Shenzhen University, China. Her articles often appear in AAMAS, AAAI, and other top AI conferences. Her research interests include intelligent agents and machine learning.

YIFENG ZENG received the Ph.D. degree from the National University of Singapore, Singapore, in 2006. He was a Professor with Teesside University, U.K. He was also an Affiliate Professor with Xiamen University, Xiamen, China. He is currently a Professor with the Department of Computer and Information Sciences, Northumbria University, U.K. His publications appear with the most prestigious international academic journals and conferences, including *Journal of Artificial Intelligence Research*, *Journal of Autonomous Agents and Multi-Agent Systems*, the International Conference on Autonomous Agents and Multi-Agent Systems, the International Joint Conference on Artificial Intelligence, and the Association for the Advancement of Artificial Intelligence. His research interests include intelligent agents, decision making, social networks, and computer games.

LANGCAI CAO received the Ph.D. degree in automation from Xiamen University, in 2011. He is currently an Associate Professor with Xiamen University. His research interests include research and development of information systems, process intelligence, and machine learning.