2019 EC³
2019 European Conference on Computing in Construction
Chania, Crete, Greece
July 10-12, 2019

AUTOMATED UNDERSTANDING OF CONSTRUCTION SCHEDULES: PART-OF-ACTIVITY TAGGING

Fouad Amer and Mani Golparvar-Fard
University of Illinois at Urbana-Champaign, Urbana-Champaign, IL, USA

Abstract

Nowadays, construction planning practices, whether conducted by human planners or artificial intelligence (AI) systems, rely heavily on manually elaborated descriptions of construction means and methods. As part of envisioning a new planning system that automatically learns construction knowledge from previous projects' schedules, this paper introduces Part-of-Activity (POA) Tagging: a construction-specific word-category disambiguation method for decoding the constructional functionalities encoded in activity names. These functionalities represent the roles each token, i.e. word, in an activity name plays in understanding that activity from a construction point of view. The model is built using Bidirectional Long Short Term Memory Recurrent Neural Networks (BI-LSTM RNN). After training on a manually annotated dataset of more than 7000 activities, the model achieved a token accuracy of ~92%. The significance of this method lies in its ability to allow an AI System to decipher construction schedules. This schedule understanding opens the door for further applications such as the automated elaboration of weekly work plans and alignment of master schedules to weekly work plans.

Introduction

The automated generation of construction schedules is a well-established research area in construction literature (Levitt & Kunz (1985), Hendrickson et al. (1987), Darwich et al. (1989), Fischer et al. (1994), Aalami & Fischer (1998) Dong et al. (2012), and Morkos (2014) among others). While this literature is very rich when it comes to construction knowledge formalization (Han et al. 2015, Koo et al. 2006), previous artificial intelligence (AI) systems depend on knowledge databases that are elaborated manually. However, given the variety of construction means and methods and the differences in practice between different companies, the manual population of these databases is tedious and impractical.

On the other hand, many construction companies have already established procedures to propagate their construction scheduling knowledge between different projects. This accumulated knowledge, typically rooted with accomplished construction planners, is maintained in Critical Path Method (CPM)-based project-level construction schedules, lookahead schedules, and weekly work plans. Nowadays, given the progress in Natural Language Processing, Data Mining, and Machine Learning, mining this semi-structured data offers a new opportunity for learning scheduling knowledge automatically and building structured company-specific scheduling knowledge bases. To take advantage of this opportunity, Amer and Golparvar-Fard (2019) proposed a new system that automatically mines company-specific construction sequencing knowledge from previous schedules and stores it in Dynamic Means and Methods Templates. This paper extends their work by detailing a method for decoding activity constituents using only activity names as written in construction schedules.

The model presented in this paper parses an activity name, tokenizes it (i.e. splits it into individual tokens (words)), and identifies the constructional functionality of each token. The constructional functionality is defined here as the role each token plays in understanding the activity from a construction point view. These functionalities reveal the constituents encoded in that activity name. For example, the activity "*place CIP columns level 1*" is split into the following tokens: "*place", "CIP", "columns", "level",* and *"1*". The constructional functionalities of these tokens are respectively "*Action*", "*Object", "Object", "Location", and "Location*". The descriptions of different activities can include all or a subset of the constituents required to understand the task in hand. The level of details of these descriptions vary between different schedules and between different activities of the same schedule. For example, the activity "*prime and paint*" only encodes the action to be done, without explicitly describing the element to be painted, the location where the painting needs to take place, nor the resources required for painting. On the other hand, "*fire sprinkler main laterals area a*" describes the physical elements to be modified and the location of that modification without reciting the exact action to take nor the party responsible for conducting that action. The amount of information that can be learned about a given activity is therefore dependent on its available constituents.

Additionally, the constituents of different activities can in some cases reveal the dependencies among them. For example, "*place CIP columns level 1*" and "*place CIP columns level 2*" differ only in their location constituent. This suggests a resource-based precedence dependency as opposed to a logical means and methods dependency. While the authors envision an AI system that can learn construction knowledge from previous company-specific

DOI:10.35490/EC3.2019.196

schedules, such system needs to understand these schedules and the types of dependencies among their activities. In other words, this system needs to have the capacity of decoding construction activities and pinpointing their constituents. This paper therefore presents a supervised machine learning based method for mining construction activity constituents from raw schedule data.

The following sections will (1) highlight the research methodology and provide a formal definition of the problem, (2) review related literature, (3) detail the proposed method, and (4) validate the model and discuss the results.

Research Methodology and Problem Definition

Knowledge Gap

Currently, construction literature lacks a method that can automatically reason about the different constituents of a construction activity given only its name as written in a construction schedule. Moreover, a corresponding formal definition of the activity constituents that can be learned from an activity name and constructional functionalities that different tokens in an activity name can take is also missing.

Problem Definition

Given an activity, we aim at extracting all the activity constituents encoded in the activity name by tagging each token in that name with its constructional functionality. Formally, let's define $\mathbb{F}$ to be the set of all possible constructional functionalities, C to be the set of all possible constituents such as $C \subset \mathbb{F}$, V to be the vocabulary or the set of all possible tokens, and $\int: V \longrightarrow \mathbb{F}$ to be the function that maps individual tokens in an activity name to their respective constructional functionalities. The aim of this paper is therefore to define C and $\mathbb{F}$ and to learn the function $\int$ given V.

The reason C and $\mathbb{F}$ are not identical is that there exists tags in $\mathbb{F}$ such as "Preposition" that are not part of C. For example, some tokens such as "in" which is almost deterministically a "Preposition" are also an "Action" or "Object" as for "in" inside the phrase "rough in".

Literature Review Approach

In order to define C and $\mathbb{F}$, a literature review of the definition of construction activity constituents is conducted and discussed in the following section. On the other hand, given the similarity between this paper's problem and the Part-of-Speech (POS) tagging problem which is well known in the Natural Language Processing (NLP) and Linguistics communities, this work reviews the POS tagging problem definition, the most recent and successful models for solving it, and its coverage in the construction literature specifically.

Data

Similar to the supervised POS tagging problem, learning the mapping function $\int$ requires training data. In this paper, 7300 activities were collected from 3 construction schedules of actual construction projects. The dataset was then built by annotating the data based on the pre-defined $\mathbb{F}$ and using a custom-built annotation tool. An example of the annotated data can be found in Figure 1.

Validation

Validating the model followed a typical machine learning model validation procedure. The labeled data was split into training and testing sets. The model was trained on the training set and then tested on the testing set. The metrics used for validation are Token Accuracy, Sentence Accuracy, Precision, and Recall. These metrics are calculated as follows:

$$Token\ Accuracy\ = \frac{\sum_{i\ =\ 0}^{i\ =\ T} I(y_i\ =\ \overline{y_i})}{\sum_{i\ =\ 0}^{i\ =\ T} I(y_i\ =\ \overline{y_i})\ +\ I(y_i\ \neq\ \overline{y_i})}$$

where T is the total number of tokens, I is an indicator function that takes a value of 1 if its argument is correct and 0 otherwise, y_i is the true label for token i, and $\overline{y_i}$ is the model prediction for token i.

$$Sentence\ Accuracy\ = \frac{\sum_{i\ =\ 0}^{i\ =\ S} I(\boldsymbol{s_i} = \overline{\boldsymbol{s_i}}\ )}{\sum_{i\ =\ 0}^{i\ =\ S} I(\boldsymbol{s_i} = \overline{\boldsymbol{s_i}})\ +\ I(\boldsymbol{s_i} \neq \overline{\boldsymbol{s_i}})}$$

where S is the total number of activities (sentences), $\boldsymbol{s_i}$ is the vector of true labels of the tokens activity i, $\overline{\boldsymbol{s_i}}$ is the vector of the model predictions of the tokens of activity i.

$$Precision_j\ = \frac{\sum_{i\ =\ 0}^{i\ =\ T} I(\overline{y_i}\ =\ y_i\ =\ j)}{\sum_{i\ =\ 0}^{i\ =\ T} I(\overline{y_i}\ =\ j)}$$

Precision of label j is equal to the ratio of the number of tokens correctly predicted as j to the total number of tokens predicted as j.

$$Recall_j\ = \frac{\sum_{i\ =\ 0}^{i\ =\ T} I(\overline{y_i}\ =\ y_i\ =\ j)}{\sum_{i\ =\ 0}^{i\ =\ T} I(y_i\ =\ j)}$$

Recall of label j is equal to the ratio of the number of tokens correctly predicted as j to the total number of tokens that have j as their true label.

Literature Review

Construction activities have been widely discussed in construction literature and especially in the construction scheduling automation and optimization communities. In terms of defining the constituents of those activities, Darwich et al. (1989) defined Object, Action, and Resources as the main arguments required to define and represent an activity. Similarly, Fischer et al. (1994) and Aalami & Fischer (1998) defined these constituents to be Components, Actions, Resources, and Sequencing Constraints. These definitions were also adopted by more recent works such as Dong et al. (2012), Morkos (2014), and Garcia-Lopez (2017). This paper builds on top of these definitions and tailors them towards information that can be mined solely from activity names as opposed to having more contextual data or domain expertise.

On the other hand, In Linguistics and NLP, Part-of-Speech (POS) Tagging is the process of identifying the syntactic

part of speech, i.e. Noun, Verb, Adjective, etc., of any token in any given context. Similar to POS Tagging, this paper introduces Part-of-Activity (POA) Tagging where the aim is at identifying the constructional functionality of each of the terms that appear in an activity name as opposed to their syntactic roles. Research in POS Tagging started in the 70s where rule-based models and finite-state models were mostly implemented (Greene & Rubin 1971). These models where deterministic in their nature, and they took advantage of preset rules and manually designed features. On the other hand, the next generation of models was stochastic. Hidden Markov Models (Cutting et al., 1992, Kupiec, 1992) and Maximum Entropy Models (Ratnaparkhi, 1996) became the state of the art and achieved token accuracies of ~96%. These models still used predesigned features combined with probabilistic distributions over tags and tokens. More recently, Neural models such as Huang et al. (2015) and Choi (2016) achieved state of the art performance with accuracies above 97.5%. This paper implements one of the neural models tested in Huang et al. (2015) which is the Bidirectional Long Short Term Memory Recurrent Neural Network (BI-LSTM RNN). The reason this model was chosen is its high performance combined with its intuitiveness which makes it ideal for a baseline prototype.

In construction literature, POS Tagging has been used as part of NLP pipelines for information extraction for compliance checking (Zhang & El-Gohary, 2014, 2015a, and 2017, Zhou & El-Gohary, 2017, Li et al., 2016), contract requirements extraction (Zhou & El-Gohary, 2016), IFC schema extension (Zhang & El-Gohary, 2016), and information retrieval from Building Information Models (BIM) (Zhang & El-Gohary, 2015b, Lin et al., 2016, Oraee et al., 2017). However, to the best of the authors' knowledge, using approaches inspired by POS tagging to decipher construction activities was never attempted before.

The following section (1) defines activities' constructional functionalities and constituents, (2) describes the dataset used for training the model, and lastly (3) details the model architecture.

Proposed Method: POA Tagging

Constructional Functionalities (Ƒ) and Activity Constituents (C)

As seen in the literature review section, construction activities are often defined by an Action, Object/Component, Resources, and Sequencing Constraints. Since the aim of this research is to ultimately learn construction scheduling knowledge by mining existing schedules, the sequencing constraints constituent of an activity is already available through the precedence links provided by those schedules. Therefore, formalizing a schema for learning sequencing constraints from individual activity names is not necessary. The focus of this paper is therefore on the Action, Object, and Resources constituents of an activity. More specifically, Resources are further split into Location and Responsible Party and the set of possible constituents is defined as:

$$C = \{Action, Object, Location, Responsible\ Party\}$$

While the Action and Object constituents provide a strong indication on the type of the ongoing activity and allow for generalization across different projects, Location and Responsible Party are more project specific.

Moreover, as mentioned earlier in the problem definition section, activity names include ambiguous terms such as "in" which can be part of the Object or Action, or it can simply be a Preposition. Therefore, the set of constructional functionalities Ƒ expands on C, and it is defined as follows:

$$Ƒ = \{Action, Object, Location, Responsible\ Party, Conjunction, Preposition, Extra\ Information\}$$

where Conjunction, Preposition, and Extra Information are only complementary tags that help in the identification of activity constituents.

Dataset

Using the predefined set of constructional functionalities and constituents, the authors built a labeled dataset of 7300 activities from 3 different schedules. Each activity name is tokenized and manually tagged as shown in Figure 1. This labeled dataset is split into training (85%) and testing (15%) sets.

Figure 1: Labeled Dataset Example

As shown in Figure 2, the distribution of different tags in the original training dataset is skewed. While more than 47% of the tokens are labeled as Object, only ~1% are labeled as Responsible Party. Additionally, while 84% of the activities include an Object, less than 4% of all activities include an explicit indication of the Responsible Party.

To target this data imbalance problem, the original training dataset was augmented by randomly pulling from the subset of activities that include Responsible Party but does not include Object. The new tags distribution can be seen in Figure 3. While data augmentation also increased the portion of tokens labeled as Action, the distribution of different tags is more uniform when compared to the original dataset: the standard deviation of the number of different tags in the augmented dataset is 5.6 as opposed to being 16.5 in the original dataset.

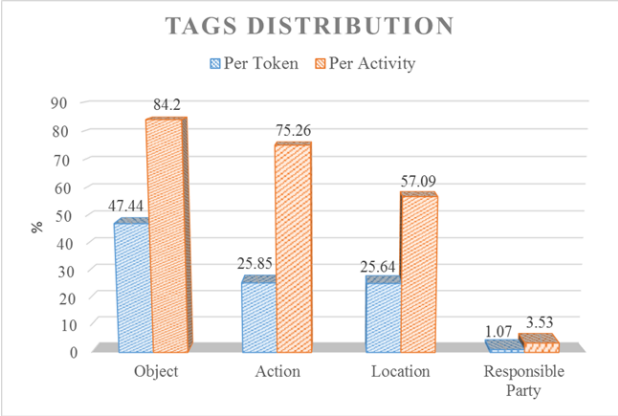

Figure 2: Tags Distribution in the Original Dataset

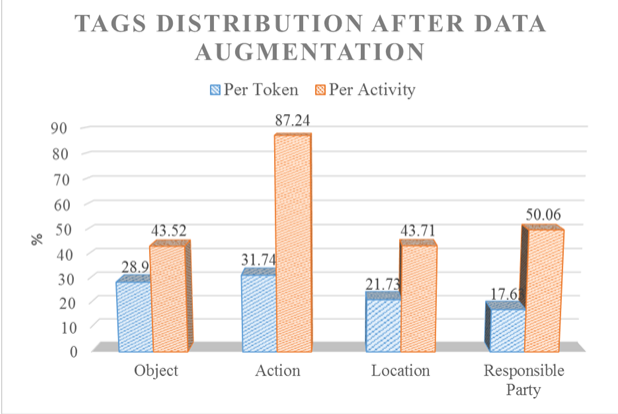

Figure 3: Tags Distribution in the Augmented Dataset

Model Architecture

The model architecture is shown in Figure 4. After preprocessing and tokenization, the input list of tokens is fed to a pre-trained embedding layer that transforms each token into a vector.

The output of the embedding layer is then fed into a BI-LSTM which is composed of two LSTMs that parse the input sequence both from left to right and from right to left. Additionally, each LSTM is stacked vertically to form a 4-layer deep LSTM (illustrated as layers 1 to k in Figure 4). The number of layers is based on empirical testing and fine tuning. The bi-directional parsing allows the contextual information to be better learned, and it is particularly useful for the purpose of this paper given the non-standardized and inconsistent writing conventions of activity names in the construction industry. The power of LSTMs lies in their ability to model sequential data by passing information from one time step to the following one while implementing the same set of weights to the input token at each time step. This means, irrespective of the length of the input sequence, which in this case is the number of tokens in an activity name, the model can handle that sequence without the need to change the architecture. This flexibility cannot be achieved with feedforward networks for example where the length of the input features must be fixed a priori.

The output of the BI-LSTM is used as input for a fully connected (FC) layer that reduce the dimensionality of the output vector into the size of set Ʃ. A Softmax layer is then applied on the output of the fully connected layer to create a probability distribution over Ʃ. For each token, the model returns the constructional functionality with the highest probability given the activity context.

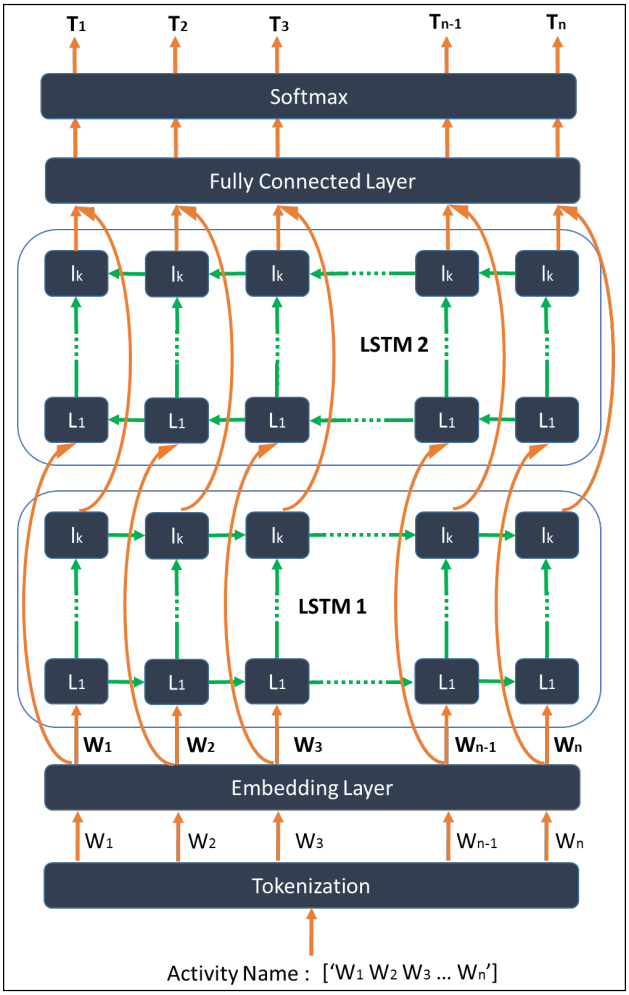

Figure 4: POA Tagger Model Architecture

Preprocessing and Tokenization

Before passing an activity name into the model, it goes through a preprocessing and tokenization step. During preprocessing, punctuations and unwanted characters such as {()?!*$#} are removed, the characters are set to lower case, and numbers are replaced by a special token among other preprocessing steps. This preprocessing takes advantage of a predefined dictionary of construction-specific terms to be skipped in the cleaning process such as 'f/r/p' (which stands for form/reinforce/place). Additionally, the tokenization step splits the input string 'W_1 W_2 W_3 … W_4' into a list of individual tokens ['W_1', 'W_1', 'W_1', …, 'W_4'].

Embedding

After tokenization, each token of the activity name is first represented as a vector using an embedding layer trained specifically on construction scheduling data. These vectors are formally known as Word Embeddings in the NLP community (Mikolov et al. 2013). The idea is to learn n-dimensional vectors that capture the semantic similarities between different words in a given corpus (set

of documents or schedule activities in this case).

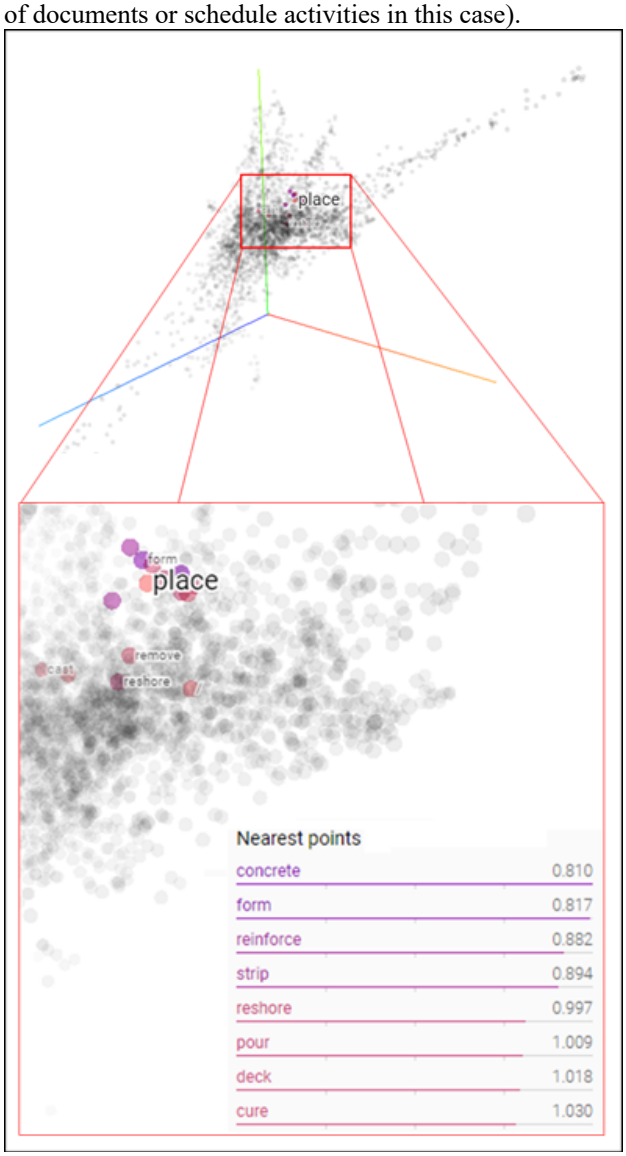

Figure 5: Learned Embedding Example - "place" and its nearest neighbors

While there are many existing trained word embeddings for different languages including English, these models do not capture the specifics of the construction scheduling terms. An example is "place" and "pour" which have very different meanings in English, but they are both used in construction to describe concrete related activities. Moreover, some of the technical terms used in construction such as "f/r/p" are also missing from a general English model. Building a specific construction scheduling model is therefore essential to the proposed approach. Details related to building and training the embedding model used in this paper can be found in Amer and Golparvar-Fard (2019). An example of the embedding results is shown in Figure 5 displaying a three-dimensional plot of the word vectors after reducing their dimensionality using t-SNE (t-Distributed Stochastic Neighbor Embedding) (Van Der Maaten & Hinton, 2008). It can be seen in the figure that the nearest points to "place" were found to be {"concrete", "form", "reinforce", "strip", "reshore", "pour", "deck", "cure"}.

BI-LSTM RNN

After the input activity name is vectorized, it is fed to the two LSTMs. The first parses the input vectors from left to right, and the second parses it from right to left independently. Each cell in both LSTMs is characterized by a set of "gates" that control the importance of different pieces of information. There are exactly 8 sets of weights for each cell: input (W_{ig}), output (W_{io}), update (W_{ii}), and forget (W_{if}) weights applied to the current input, and (W_{hg}), output (W_{ho}), update (W_{hi}), and forget (W_{hf}) weights applied to the previous "hidden state". This hidden state is responsible for maintaining important information from previous time steps to capture distributional dependencies. The function graph along with the different computations and definitions of individual LSTM cells can be found in Figure 6. At each time step, the cell hidden state combined with the current input token contribute to the prediction of the constructional functionality of that token.

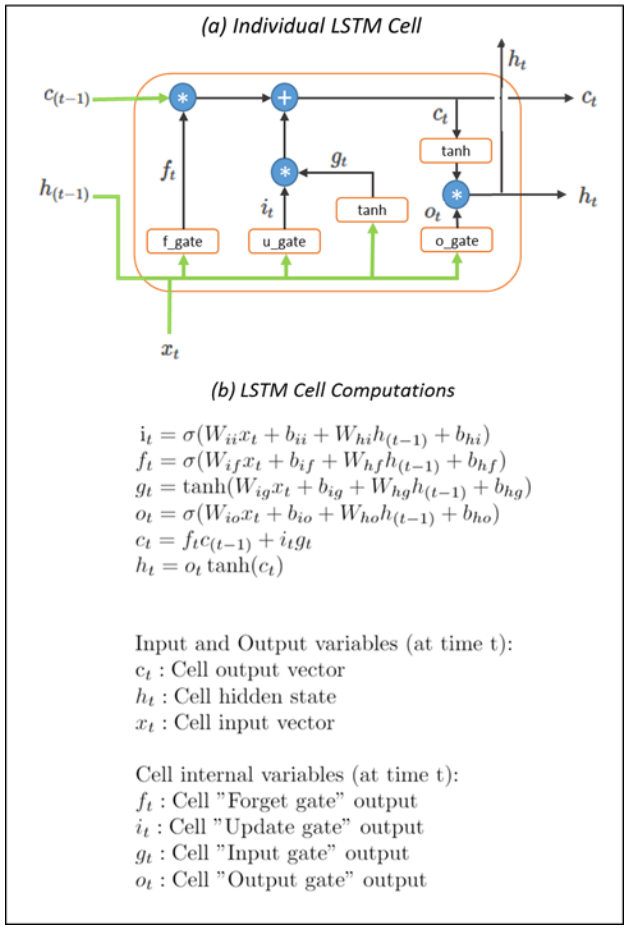

Figure 6: LSTM Cell Configuration and Computations

Fully Connected and Softmax Layers

The FC and Softmax layers are the last step of the tagging pipeline. The FC layer is used to reduce the dimensionality of the BI-LSTM output to the size of set Ƒ which is equal to 7. Softmax is then used to normalize the output of the FC layer and create a probability distribution over all 7 possible constructional functionalities.

Mapping function (ʃ)

The mapping function ʃ: V ⟶ Ƒ is therefore the entire learned model that takes an activity name "a" and identifies the constructional functionalities of its individual tokens by passing it through the neural network layers: Tokenization, Embedding, BI-LSTM, FC, and Softmax.

Results

Qualitatively, sample results of the proposed model are shown in Figure 7. Examples 1 and 2 illustrate the model's ability to accurately predict both "Install" and "Installation" as Actions irrespective of their location in the sentence and their different inflections. Similarly, examples 1 and 3 showcase how "level 1" and "lv 1" were both predicted as Location irrespective of the abbreviation used by the planner while writing "lv". Moreover, "drywall" in example 1, "Doors hardware" in example 2, and {"walls", "columns"} in example 3 were all correctly labeled as Object irrespective of their location in the sentence and their relative location when occurring together in the same sentence.

(1) Install drywall at level 1

[Action, Object, Preposition, Location, Location]

(2) Doors hardware installation

[Object, Object, Action]

(3) f/r/p walls and columns at lv 1

[Action, Object, Conjunction, Object, Preposition, Location, Location]

Figure 7: Examples of Qualitative Results

Quantitatively, as shown in Table 1, the model achieved a token accuracy of 92% and sentence accuracy of 72%. A sentence in our context is an activity name. It is considered correct only if all of its individual tokens are correctly predicted as previously described in the Research Methodology section.

Table 1: Token and Sentence Accuracies

Token Accuracy	Sentence Accuracy
0.92	0.72

The Precision and Recall of different constituents are illustrated in Figure 8. As previously described in the Research Methodology section, Precision illustrates the model's accuracy in predicting each of the constituents out of the total number of times each constituent was predicted. It is equal to the number of True Positives (TP) divided by the sum of TP and False Positives (FP).

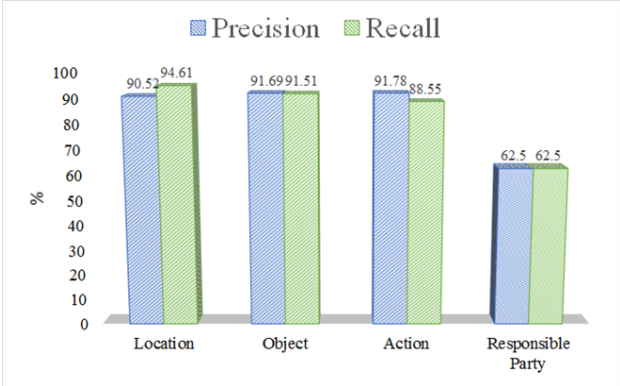

Figure 8: Constituents Precision and Recall

Recall on the other hand represents the model's efficiency in retrieving each correct constituent out of the number of times each constituent should have been predicted. It is therefore equal to the number of TP divided by the sum of TP and False Negatives (FN). In the context of this paper of TP, FP, True Negatives (TN), and FN are defined per label as shown in Figure 8 and as discussed in the Research Methodology section.

(1) Insulation level 5
True Labels:
[Action, Location, Location]
Prediction:
[Object, Location, Location]

(2) Elevator inspections
True Labels:
[Object, Action]
Prediction:
[Location, Action]

(3) re pave alley north
True Labels:
[Action, Action Object, Location]
Prediction:
[Action, Object, Object, Location]

Figure 9: Examples of incorrect predictions

On the other hand, Figure 9 displays some of the incorrect predictions made by the model. For example "Insulation" is labeled as Object as opposed to being correctly labeled as Action in this context. In the dataset, "Insulation" can take both Action and Object as labels depending on the context. In the activities where another action is specified such as "install insulation", "insulation" will be an Object. However, in the contexts where no explicit Action is specified, "insulation" reflects the Action of "installing insulation" more than reflecting the insulation Object itself. Similarly, "Elevator" can be either a Location or an Object depending on the context. Moreover, "pave" in "re pave" should reflect an Action, but the model mistakenly classified it as an Object.

In terms of FN and FP, classifying "Elevator" as Location in the second activity will add 1 to the count of Object FN and 1 to the count of Location FP. In other words, this prediction of Location instead of Object decreased the Precision of Location and the Recall of Object.

Conclusions and Future Work

As part of the authors' vision for a new construction planning system that can automatically learn construction knowledge from previous projects' schedules, this paper presented a new method for deciphering construction activities and automatically understanding their different constituents. This paper defined {Action, Object, Location, and Responsible Party} as the main constituents of a construction activity that can be directly learned from the activity name. Inspired by POS Tagging, the paper also presented a Part-of-Activity (POA) tagging model that built on a BI-LSTM RNN architecture to identify activity constituents with high accuracies. This work constitutes a part of an envisioned overall automated schedule generation and revision system that can leverage the available construction scheduling knowledge encoded in schedules of previous projects. Moreover, when used as a part of project control system, the automated understanding of construction activities as described in this paper allows for the identification of similar activities across multiple schedules, the alignment of short term and long term plans, and for the elaboration of weekly work plans from master schedules. Future work will focus on improving the robustness of the proposed algorithm and merging it with a larger system for learning scheduling knowledge from previous projects.

Acknowledgments

The authors would like to acknowledge the help of Harsh Bansal in annotating part of the dataset. This work was supported by the National Science Foundation through grant CMMI-1446765. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Aalami, F., & Fischer, M. (1998). Joint product and process model elaboration based on construction method models. *Cib Report, 1–12*

Amer, F. & Golparvar-Fard, M. (in press). Formalizing Construction Sequencing Knowledge and Mining Company-Specific Best Practices from Past Project Schedules. *International Conference on Computing in Civil Engineering, I3CE, June 2019, Atlanta, Georgia.*

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching Word Vectors with Subword Information. *arXiv:1607.04606v2*

Choi, J. D. (2016). Dynamic Feature Induction: The Last Gist to the State-of-the-Art, *Proceedings of NAACL 2016.*

Cutting, D., Kupiec, J., Pedersen J., & Sibun P., (1991), A practical part-of-speech tagger. *Submitted to The 3rd Conference on Applied Natural Language Processing.*

Darwiche, A., Levitt R., E., & Hayes-Roth B., (1989), OARPLAN:Generating Project Plans by Reasoning about Objects, Actions, and Resources, *CIFE Technical Report Nb. 2.*

Dong, N., Fischer, M., Ge, D., & Levitt, R. E. (2012). Automated look-ahead schedule generation and optimization for the finishing phase of complex construction projects. *CIFE Technical Report Nb. 211*

Fischer, M., Aalami, F., & Evans, M., O., (1994), Model Based Constructibility Analysis: The MOCA System, *CIFE Working Paper Nb. 34*

Garcia-Lopez, N. P. (2017). "An Activity and Flow-Based Construction Model for Managing On-Site Work.", *PhD Dissertation*, *Stanford University*

Green, B. and Rubin, G., (1971), Automated Grammatical Tagging of English. *Department of Linguistics, Brown University, 1971.*

Hendrickson C., Zozaya-Gorostiza C., Rehak, D., Baracco-Miller E., & Lim P., (1987), Expert System for Construction Planning, *Journal of Computing in Civil Engineering,* Vol. 1, Issue 4

Huang, Z. H., Xu, W. and Yu, K. (2015) Bidirectional LSTM-CRF Models for Sequence Tagging. *In arXiv:1508.01991. 2015.*

Levitt, R. E.& Kunz J. C., (1985), Using Knowledge of Construction and Project Management for Automated Schedule Updating, *Project Management Journal, Vol. 16, No. 5, December.*

Li, S., Cai, H., & Kamat, V.R., (2016), Integrating Natural Language Processing and Spatial Reasoning for Utility Compliance Checking, *Journal of Construction Engineering and Management, 2016, 142(12): 04016074*

Lin, J., Hu., Z., Zhang, J., & Yu., F., (2016) A Natural-Language-Based Approach to Intelligent Data Retrieval and Representation for Cloud BIM, *Computer-Aided Civil and Infrastructure Engineering 31 (2016) 18-33*

Mikolov, T. Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. (2013). Distributed representations of words and phrases and their compositionality. *NIPS, pages 3111–3119*

Morkos, R. (2014), Operational Efficiency Frontier: Visualizing, Manipulating, and Navigating the Construction State Space with Precedence, Discrete, and Disjunctive Constraints, *PhD Dissertation, Stanford University.*

Oraee, M., Hosseini, M.R., Papadonikolaki E., Palliyaguru R., & Arashpour, M., (2017), Collaboration in BIM-based Construction Networks: A Bibliometric-qualitative Literature Review, *International Journal of Project Management 35 (2017) 1288-1301*

Ratnaparkhi, A., (1996), A Maximum Entropy Model for Part-of-Speech Tagging. *Proc. EMNLP. New*

Brunswick, New Jersey: Association for Computational Linguistics.

Van Der Maaten, L. & Hinton, G. (2008), Visualizing Data Using t-SNE, *Journal of Machine Learning Research, 2008 vol.9 pp: 2579-2605*

Zhang J. & El-Gohary N., (2017), Integrating Semantic NLP and Logic Reasoning into a Unified System for Fully-Automated Code Checking, *Automation in Construction 73 (2017) 45–57*

Zhang J. & El-Gohary N., (2016), An Automated Relationship Classification to Support Semi-Automated IFC Extension, *Proceedings of Construction Research Congress 2016, p 829-838*

Zhang J. & El-Gohary N., (2015b), Automated Extraction of Information from Building Information Models into a Semantic Logic-Based Representation, *Proceedings of Computing in Civil Engineering 2015, p 173-180*

Zhang J. & El-Gohary N., (2015a), Semantic NLP-Based Information Extraction from Construction Regulatory Documents for Automated Compliance Checking, *Journal of Computing in Civil Engineering 2016, 30(2): 04015014*

Zhang J. & El-Gohary N., (2014), Automated Reasoning for Regulatory Compliance Checking in the Construction Domain, *Proceedings of Construction Research Congress 2014, p 907-916*

Zhou, P., & El-Gohary, N., (2017), Ontology-Based Automated Information Extraction from Building Energy Conservation Codes, *Automation in Construction 74 p 103-117*

Zhou, P., & El-Gohary, N., (2016), Automated Extraction of Environmental Requirements from Contract Specifications, *Proceedings of the 16th International Conference on Computing in Civil and Building Engineering p 1669-1676*