Automatic Entity State Annotation using the VerbNet Semantic Parser

Ghazaleh Kazeminejad[1] **Martha Palmer**[1] **Tao Li**[2] **Vivek Srikumar**[2]

[1] University of Colorado Boulder
[2] University of Utah

{ghazaleh.kazeminejad,martha.palmer}@colorado.edu
{tli,svivek}@cs.utah.edu

Abstract

Tracking entity states is a natural language processing task assumed to require human annotation. In order to reduce the time and expenses associated with annotation, we introduce a new method to automatically extract entity states, including location and existence state of entities, following Dalvi et al. (2018) and Tandon et al. (2020). For this purpose, we rely primarily on the semantic representations generated by the state of the art VerbNet parser (Gung, 2020), and extract the entities (event participants) and their states, based on the semantic predicates of the generated VerbNet semantic representation, which is in propositional logic format. For evaluation, we used ProPara (Dalvi et al., 2018), a reading comprehension dataset which is annotated with entity states in each sentence, and tracks those states in paragraphs of natural human-authored procedural texts. Given the presented limitations of the method, the peculiarities of the ProPara dataset annotations, and that our system, Lexis, makes no use of task-specific training data and relies solely on VerbNet, the results are promising, showcasing the value of lexical resources.

1 Introduction

Question answering in reading comprehension tasks focusing on procedural texts (i.e. texts describing processes) is particularly challenging in natural language processing (NLP), because this type of text describes a changing world state (Clark et al. 2018, Dalvi et al. 2018, Tandon et al. 2018, Du et al. 2019, Gupta and Durrett 2019). Tracking the state of entities in such texts is an important task to enable proper question answering in reading comprehension tasks. The challenging part in such question answering tasks is not where answers are explicitly mentioned in a sentence (Clark et al., 2018). For example, in Figure 1, the sentence with bold elements tells us explicitly that urea and carbon dioxide are located in kidneys by the end of this sentence. However, the fact that the blood itself will also be located at the kidneys is implicit in the semantics of the verb "carry", in which the agent of the action (mover) moves along with the theme (thing moved). Therefore, answering the question in Figure 1 requires knowledge of the particular semantics of this verb. This is the kind of inference our system, Lexis, is able to model using VerbNet.

Paragraph:
Blood delivers oxygen in the body.
Proteins and acids are broken down in the liver.
The liver releases waste in the form of urea.
The ***blood carries*** *the urea and carbon dioxide* ***to the kidneys****.*
The kidneys strain the urea and salts needed from the blood.
The carbon dioxide by product is transported back to the lungs.
Q: *Does blood enter the kidney?*
A: *Yes.*

Figure 1: A paragraph from *ProPara* about blood, showing one kind of inference needed to answer a question. Knowing that blood (as well as the urea and carbon dioxide) will end up in the kidney requires lexical semantics knowledge.

This type of reasoning requires event extraction as a first step, and event participant state extraction as a second step. Our method covers both steps, but is at this point limited to sentence-level inference.

In section 2, we provide an overview of the work related to this research. Section 3 introduces the dataset used to evaluate Lexis, as well as the details of the methods we have used. Section 4 presents our experimental settings, followed by section 5 which illustrates the results of each setting. Section 6 discusses the advantages of Lexis, and provides an error analysis to illuminate the sources of weakness. In section 7, we conclude and briefly discuss our future work.

Proceedings of the Joint 15th Linguistic Annotation Workshop (LAW) and 3rd Designing Meaning Representations (DMR) Workshop, pages 123–132

Sentence:
The water forms a stream.

VNSP VerbNet semantic predicates output:
[{'polarity': False, 'predicateType': 'Has State', 'args': [{'type': 'Material', 'value': ''}, {'type': 'Verbspecific', 'value': 'V_Final_State'}]}, {'polarity': True, 'predicateType': 'Do', 'args': [{'type': 'Agent', 'value': 'The water'}]}, {'polarity': True, 'predicateType': 'Be', 'args': [{'type': 'Result', 'value': 'a stream'}]}]

A more readable format:
¬HAS_STATE(?$_{\text{Material}}$, V_Final_State$_{\text{VerbSpecific}}$)
¬BE(a stream$_{\text{Result}}$)
DO(The water$_{\text{Agent}}$)
BE(a stream$_{\text{Result}}$)
HAS_STATE(?$_{\text{Material}}$, V_Final_State$_{\text{VerbSpecific}}$)

Extracting a CREATE entity state:

In this case, based on the predicateType 'Be' along with a 'Result' argument type, we can conclude that the value 'a stream' is the *created* entity. This is then fed into spaCy to extract the head noun 'stream' as the entity.

Figure 2: VerbNet semantic predicate output of VNSP on the input sentence above, and how it is used to predict a CREATE type change of state for the entity 'stream'

2 Related Work

Our method is similar to Clark et al. (2018) in the sense that we both rely on VerbNet semantic representations to make inferences. However, our point of departure is that previously, VerbNet did not have a neural semantic parser, and Clark et al. (2018) did not disambiguate verb senses, and used the most frequently used verb sense instead. Also, they had to develop a huge rulebase to encode commonsense knowledge about the states that events produce, using a STRIPS-style list of before (preconditions) and after (effects) expressed as possibly negated literals. In order to extract arguments (i.e. event participants), they relied on the syntactic patterns provided in the VerbNet, and used them to instantiate the arguments in the before/after literals. They also performed a manual annotation effort to check and correct the rulebase entries, and added entries for other verbs that affect existence and location. In addition, they added new entries for verbs not existing in VerbNet. In contrast, our method is fully automatic (see section 3.2).

Most work on tracking entity states uses neural methods (Henaff et al. 2016, Ji et al. 2017, Tandon et al. 2018, Tang et al. 2020). However, these models rely on large amounts of annotated data, which is expensive and labor-intensive to provide (Sun et al., 2020). One of the commonly used solutions to the data scarcity problem in NLP tasks is data augmentation. According to Feng et al. (2021), the goal of data augmentation is increasing training data diversity without directly collecting more data. Most strategies for data augmentation consist of creating synthetic data based on the main data. On the same note, automatic training data generation attempts show promise in various NLP tasks, such as event extraction (Chen et al. 2017, Zeng et al. 2018), and named entity recognition (Tchoua et al., 2019).

As explained below, Lexis uses a state-of-the-art semantic parser to automatically generate entity states for each sentence. These entity states are the same as the entity state labels provided by human annotators in the dataset on which we evaluate Lexis. The generated inferences can be used, in future work, to augment the existing training data.

3 Data and Methodology

The main contribution of this work is bringing to the forefront the advantages of using lexical resource based methods. In particular, in this work, these methods are used for automatic annotation of text with labels regarding the location and existence states of entities in a given sentence.

3.1 Data

The ProPara (Process Paragraphs) dataset (Dalvi et al., 2018), developed by the Allen Institute for AI (AI2), contains 183 prompts (with 152 topics) and 488 human authored paragraphs of procedural text in response to these prompts (each paragraph having 10 or less sentences), along with 81k annotations about the changing states (existence and location) of entities in those paragraphs. The training set contains 391 paragraphs (80% of the data). The end-task of the dataset is predicting and tracking location and existence changes for the entities. ProPara is the first dataset of annotated, natural text for real-world processes, which also contains a simple representation of entity states during those

processes.

3.2 Methodology

We used the recently developed BERT-based VerbNet semantic parser (Gung 2020, Gung and Palmer 2021), which is located at the GitHub SemParse site [1], to parse every single sentence in each paragraph. The VerbNet semantic parser (VNSP) returns a json file containing the verb sense disambiguated VerbNet class, the complete logical predicates for that class instantiated with arguments extracted from the sentence, as well as the text spans (phrases) labeled with both VerbNet thematic roles (Schuler, 2005) (if applicable) and PropBank argument role labels (Kingsbury and Palmer 2002, Palmer et al. 2005).

The main idea of using VNSP, and in general what gives VNSP an edge over other semantic parsers, is the logical predicates it generates (for a list of some of the VerbNet predicates used in this work, see Table 1). These predicates are utilized here to infer/predict an entity's change of location and change of existence state (i.e. whether it has been created or destroyed during the course of the sentence). Some of these predicates uncover implicit information about entity states, so our method covers explicit and implicit information, as long as the information is implicit in the semantics of the verb, rather than requiring world knowledge. Figure 2 illustrates the utility of the semantic predicate part of the VNSP output and how it is used to predict that the entity 'stream' is CREATED in the input sentence 'The water forms a stream'. The VNSP output is close to our desired form, but does not match exactly. Certain inferences first need to be drawn, and we have implemented blocks of Python code for this purpose. Figure x gives the block of Python code for 'Results' that extracts "a stream" as the created entity. VNSP also gives us syntactic phrases, and we still have to extract head nouns and objects of prepositions, etc, to get specific entities, as well as before-after states. We use spaCy (Honnibal et al., 2020) to do this.

We represent the inferred entity states as a triple for change of location cases (entity, AtLoc, location), and a tuple for change of existence cases (entity, Created) or (entity, Destroyed). These generated states can, in future work, be used for data augmentation, to improve the performance of machine learning models on this task.

[1] https://github.com/jgung/verbnet-parser

Location	Existence
Has Location	Appear
Take In	Becomes
Admit	Create Image
Apply Material	Degradation Material Integrity
Contain	Destroyed
Free	Develop
Contact	Emit

Table 1: Some of the VerbNet predicates used in our method to infer state changes.

Sentence:
The roots absorb water and minerals from the soil.

VNSP VerbNet semantic predicates output:
[{'polarity': True, 'predicateType': 'Take In', 'args': [{'type': 'Goal', 'value': ''}, {'type': 'Theme', 'value': 'water and minerals'}]}]

VNSP PropBank parse output:
[{'text': 'The roots', 'pb': 'A0'},
{'text': 'absorb', 'pb': 'V'},
{'text': 'water and minerals',
'pb': 'A1'}, {'text': 'from the soil', 'pb': 'A2'}]}]

Extracting a MOVE entity state:
In this case, based on the predicateType 'Take In' along with a 'Theme' and a 'Goal' argument type, we can conclude that the value 'water and minerals' is an entity that moves to the value for 'Goal'. However, the value for 'Goal' has remained uninstantiated, so we resort to the PropBank parse. Knowing that the 'Goal' argument for the 'Take In' predicate is the same as A0 numbered argument, we extract 'The roots' as the destination for the extracted entity. This is then fed into spaCy to extract 'water' and 'minerals' as two entities, both moving to the destination 'root'. A separate block of Python code draws Take In inferences

Figure 3: VerbNet semantic predicate and PropBank output of VNSP on the input sentence above, and how it is used to predict a MOVE type change of state for the entities 'water' and 'minerals'

This completes the process used for Setting 1 in Section 4.1. In the second setting for our experiments, 4.2, we added information from the PropBank argument roles generated by VNSP to cover cases where there are uninstantiated arguments in the VerbNet semantic parse that PropBank can supply. This should, in theory, increase recall, but may decrease precision, because VerbNet thematic roles

are more fine grained compared to PropBank, and are therefore more accurate in predicting semantic roles. Figure 3 illustrates the utility of the PropBank parse part of the VNSP output and how it is used to predict that the entity 'water' and the entity 'minerals' is MOVED to 'root' in the input sentence 'The roots absorb water and minerals from the soil'. This example illustrates a case in which the VerbNet semantic predicate generated by the VNSP fails to generate a value for the 'Goal' argument, while looking at the PropBank A0 argument supplies us with it. It also illustrates another example of lexically encoded implicit semantic information, since for the *Take In* predicate that is provided in the VNSP output, the 'Goal' argument is the same as 'Agent' argument. For that reason, we can confidently extract the value for A0 from the PropBank parse as the value for the Goal, and therefore the destination of the MOVE event.

In addition, we have also introduced a relaxed setting in which the predicted false positive labels are evaluated by a human judge to be true false positives or correctly predicted labels that are missed in the gold labels in ProPara.

4 Experiments

The text spans that represent arguments are either noun phrases (for entities, e.g. *sediment from the ocean* in the sentence *The waves contain sediment from the ocean.*) or prepositional phrases (for location state tracking, e.g. *in sediment* in the sentence *They are buried in sediment.*). A human annotator labels *sediment* as the destination of *they*, but the parser labels the whole phrase *in sediment* as the destination. On the other hand, there are cases of conjunction, where several entities are conjoined and one predicate is stated about them. In that case, we should be able to separate the conjoined entities and generate a triple or tuple for each. For example, in the sentence *Algae and plankton die*, two change of existence tuples should be generated: (algae, Destroyed), and (plankton, Destroyed).

Such syntactic problems were solved using the spaCy (Honnibal et al., 2020) dependency parser. Nouns were lemmatized, the nominal heads of the noun phrases in prepositional phrases were extracted, the conjoined noun phrases were disjoined.

4.1 Setting 1

In setting 1, we only relied on the predicates from the VerbNet semantic representations part of the output of the VNSP. Given the definition of each predicate and the types of arguments it can take across the VerbNet, we categorized the predicates into those indicating a change of location and those indicating a change of existence (in particular, creation and destruction). We also used the VerbNet thematic role hierarchy to collect all thematic roles that could point to an event participant (e.g. Agent, Co-Agent, Pivot, Patient, etc.), a location (e.g. Destination, Location, Goal, Source, etc.), an Undergoer (e.g. Patient, Theme, Pivot, etc.), as well as those particularly indicating a Destination-type location and a Source-type location. For example, in the sentence *Water from oceans, lakes, swamps, rivers, and plants turns into water vapor.*, the semantic predicate output of the VNSP is summarized (by our algorithm) into the following:

[(True, 'Has State', [('Patient', 'Water from oceans , lakes , swamps , rivers , and plants'), ('Initial State', '')]),
(True, 'Has State', [('Patient', 'Water from oceans , lakes , swamps , rivers , and plants'), ('Result', 'into water vapor')])]

From this logical statement, since the final sub-predicate is a 'Has State' predicate and it has a 'Result' argument, our algorithm extracts the value for the 'Result', i.e. into water vapor, as the entity created.

Another example of the semantic predicate output that explains the inference in Figure 1 follows:

[(True, 'Has Location', [('Theme', 'the urea and carbon dioxide'), ('Initial Location', '')]), (True, 'Has Location', [('Agent', 'The blood'), ('Initial Location', '')]), (True, 'Do', [('Agent', 'The blood')]), (True, 'Motion', [('Theme', 'the urea and carbon dioxide'), ('Verbspecific', 'Trajectory')]), (True, 'Motion', [('Agent', 'The blood'), ('Verbspecific', 'Trajectory')]), (True, 'Has Location', [('Theme', 'the urea and carbon dioxide'), ('Destination', 'to the kidneys')]), (True, 'Has Location', [('Agent', 'The blood'), ('Destination', 'to the kidneys')])]

The last two lines show that both the Theme (the urea and carbon dioxide) and the Agent (The blood) will end up in the Destination (to the kidneys), uncovering the implicit information in the semantics of the verb carry.

Some predicates required special handling. For example, the *Take In* predicate (representing verbs such as inject, drink, eat, smoke) is fairly opaque, and does not explicitly indicate a change of location in VerbNet. However, we know that as a result of

Block of Python Code:

```
elif 'Take In' in preds:
    for p in predlist:
        if p['predicateType'] == 'Location' and p['polarity']:
            for a in p['args']:
                if a['type'] == 'Theme' and a['value']:
                    sp = SpacyObject(a['value'])
                    motion_dict['entity'] = sp.treatNP()
                    motion_dict['before'] = []
                elif a['type'] in all_srl('goal') or \
                a['type'] in all_srl('agent') and a['value']:
                    sp = SpacyObject(a['value'])
                    motion_dict['after'] = sp.PP2NP()
```

Figure 4: Python code for extracting a MOVED entity, along with its location states, from the VerbNet semantic predicate `Take In`.

Pseudo Code:

If `Take In` is in the list of generated VNSP semantic predicates for this sentence:
    loop over the generated predicates
        If there is a `Take In` predicate with positive polarity:
            loop over the arguments of that predicate:
                If the type of an argument is `Theme`:
                    extract the head noun of the value as an entity which has MOVED.
                If the type of an argument is `Goal` or `Agent`:
                    extract the noun head of the prepositional object as the *after* state.

this type of event, an undergoer (such as a Theme) would move to the Goal, Agent, or Recipient of the event, whichever is specified in the predicates. The block of Python code for Take In (Figure 4) includes an explicit expansion of Take In to indicate the Goal is now At The Agent. Such semantic expansions needed to be assumed in the semantics of such predicates in the entity extraction algorithm, as VerbNet fails to do its normal expansion of the predicates. These expansions will be added to the next VerbNet release.

4.2 Setting 2

In setting 2, we used the PropBank argument roles included in the VNSP output in addition to the VerbNet predicates. This covered cases of verbs that exist in VerbNet, but VNSP fails to instantiate the entity or location. This is most likely due to the small size of the VerbNet labeled training data compared to PropBank labeled training data. For example, in the sentence *The stream becomes a river.* from the training data, here is the output of the VNSP:

[(False, 'Has State', [('Patient', 'The stream'), ('Result', '')]),
(True, 'Has State', [('Patient', 'The stream'), ('Result', '')])]

Here, the value for the 'Result' semantic role has remained uninstantiated, which means that setting 1 yields no output (i.e. does not extract 'river' as a created entity). Therefore, we look at the A2 argument in the PropBank parse output of the VNSP, summarized below:

['text': 'The stream', 'pb': 'A1', 'vn': 'Patient',
'text': 'becomes', 'pb': 'V', 'vn': 'Verb',
'text': 'a river', 'pb': 'A2', 'vn': '']

Linguistically, the A2 argument for this verb is described as 'new state', which is exactly what we need. Since the numbered arguments in PropBank are too coarse-grained, we use SemLink (Palmer 2009, Bonial et al. 2013, Stowe et al. 2021) to find their mapping into VerbNet thematic roles and find those that suit our purposes in this task.

In general, we assumed that A1 (proto-patient, which is typically the undergoer) is the entity moved or created. This assumption should be accurate in theory at least for verbs indicating creation or motion, because an entity in motion is theoretically a "theme", and an entity created could be a "result", a "product", or "theme", which are all labeled A1 in PropBank. However, there are also change of state verbs such as become, for which the 'Result' (new state) is not the entity undergoing change or created, so it is marked as A2 rather than

A1. It should also be noted that since the PropBank numbered arguments A2 and above are very coarse-grained and overloaded. That is the reason we consult SemLink to find the correct mapping.

4.3 Relaxed Setting

Since Lexis generates more states than gold standard annotations in ProPara. For example, for the sentence 'Magma rises from deep in the earth.' in the training data, no motion has been predicted for *magma*, while as a human judge, we know that magma moves in this sentence. Lexis predicts this movement of magma in this sentence, and this is counted as a false positive in evaluation, reducing the precision and F1 score. Therefore, we examined a third setting in which we do not count the extra annotations as false positives, if they are correct. Knowledge of whether these extra predicted entities are correct or wrong requires human judgment.

For that matter, we set up a judgment task on the system output (i.e. the predicted created/destroyed/moved entities) on the test data, and asked two unbiased human annotators to make judgments. The results were then examined by the authors to identify the sources of false negatives. These will be discussed in section 6.2, and illustrated in Table 6.

5 Results

Tables 2, 3, and 4 illustrate the evaluation results. In addition to the cumulative/general state tracking results, we have evaluated location tracking and existence tracking separately in order to monitor the sources of error. We have also provided the results for the relaxed setting in Table 4 (see 4.3).

As expected, the inclusion of PropBank argument roles (in setting 2) increases recall from 0.29 (Table 2) to 0.39 (Table 3). Interestingly, this is achieved through existence tracking, not location tracking. Within location tracking, not much change is observed between the two settings, as discussed in section 6.2.

While performing the human judgment task for the relaxed setting, we also came across cases of gold label errors (see 'GLE' in Table 6), as well as useless gold labels in location tracking, labeling the location of an entity as *unk* (i.e. unknown). Both the GLE and superfluous labels were considered unlabeled in the gold data for evaluation purposes. Therefore, the results in the top section of Table 4 are different from the results in setting 2 (VerbNet + PropBank, see Table 3), because here we removed the gold label errors. These will be discussed further in section 6. Within Table 4, it is notable that the precision significantly increases (by 20%) in the relaxed setting (compared to the strict setting in the same table).

Within the Allen AI leaderboard for the ProPara task (see Table 5), our general recall in setting 2 (VerbNet + PropBank) beats Facebook AI Research EntNet, University of Washington QRN, and AI2 ProLocal. The recall increases from 0.39 to 0.48 in the relaxed setting (Table 4). There are 6 teams that have a higher recall, and Lexis Relaxed would stand on the 7th place with respect to recall. For the general F1 score, too, Lexis Relaxed would take the 7th place on the leaderboard.

6 Discussion

6.1 Advantages

First and foremost, it should be noted that in this work, we have relied solely on a lexical resource, and no task-specific learning-based methods have been utilized. Also, our method generates more entity states than human annotators generated. That increases our system's false positives, which in turn decreases the precision and F1 scores. For this reason, we decided to create a relaxed setting (cf. section 4.3) and judge which one of the false positives are true false positives and which ones are correct labels which are missing from the train set. In the judgment task to find true false positives, out of 235 false positives (in 42% of the train set), 129 were in fact judged as correct labels missed in the gold data. Therefore, the count of false positives in the relaxed settings was reduced from 235 to 106 (i.e. 45% of the false positives were not true false positives).

Another advantage of Lexis is that it predicts states for all possible entities in a sentence, and is not limited to the set of entities labeled in ProPara. However, for evaluation purposes, we only compared our results against the gold labeled entities.

What makes our method valuable is the fact that, unlike learning models which need annotated data, Lexis does not require training. As such, this method can be used for automatic generation of entity states, potentially with higher recall than human annotation. In addition, this is not limited to the ProPara dataset, or even procedural texts. Since VerbNet is a general semantic lexicon for

	Precision	Recall	F1 Score
Lexis			
General	0.38	0.29	0.33
Location	0.41	0.21	0.28
Existence	0.85	0.40	0.55

Table 2: Evaluation Results for Setting 1 - using only VerbNet semantic predicates and semantic representations provided in the VNSP output.

	Precision	Recall	F1 Score
Lexis			
General	0.36	0.39	0.38
Location	0.41	0.22	0.29
Existence	0.64	0.65	0.64

Table 3: Evaluation Results for Setting 2 - using the VerbNet predicates, and utilizing PropBank argument roles (numbered arguments) when an argument is uninstantiated in the semantic representations. Both the VerbNet semantic representations and PropBank argument labels are provided by the VNSP.

English verbs, the VNSP output can be utilized to predict entity states in any domain. However, we should also acknowledge the sources of error and weaknesses of this method, which are discussed in section 6.2 below.

6.2 Error Analysis

In this section, we discuss and analyze the sources of error and weakness of our method and results, illuminating the limitations.

The first noticeable source of weakness is the limited coverage of the VerbNet lexicon itself. There are currently 4588 unique verbs in VerbNet. For the verbs that are absent from the VerbNet, the VNSP output is empty. Out of 2639 sentences in the train set, 158 remained unparsed (6%). This increases false negative counts and therefore decreases our recall and F1 score.

Another source of weakness of VNSP, and therefore our method, is the relatively small size of VerbNet-labeled data for training the VNSP. This is in comparison to the size of the PropBank-labeled data.

One of the most important sources of error was use of world knowledge in the gold standard human annotation in addition to the explicit linguistic knowledge – something that is beyond the scope of VerbNet. For a complete picture of the false negative underlying reasons, see Table 6. An illustrative

	Precision	Recall	F1 Score
Lexis (Gold Label Errors Removed)			
General	0.44	0.48	0.46
Location	0.54	0.44	0.49
Existence	0.37	0.53	0.44
Lexis (Relaxed: True False Positives Filtered)			
General	0.64	0.48	0.55
Location	0.79	0.44	0.57
Existence	0.53	0.53	0.53

Table 4: Evaluation Results for the Relaxed Setting. At the top, we have the predicted labels from Table 3, except that we have removed the gold label errors. At the bottom, in addition to removing gold label errors, we have also filtered the false positives and kept only the true false positives (not counting the correct labels that are only missing from the gold data).

example follows. Given the paragraph

> *Get some seeds. Pick a spot to plant them. Dig a hole in the dirt. Put the seed in the hole. Pour some water on the seed and hold.* ***Cover up the hole****. Press down on it. Spray some plant food on it.*

The annotator has annotated the highlighted step with a 'Move' state change of the 'seed' entity to 'dirt'. This knowledge requires recovering of information from step 4 to know that the seed is in the hole, from step 3 to know that the hole is dug in the dirt, and from world knowledge to know that covering up the seed in the hole which is in the dirt normally happens with the dirt that has been dug. Another example of the same type is in the paragraph below.

> *Water from the surface seeps below the soil. The water comes into contact with the rock below.* ***The water over the years carves through the rock.*** *The space in the rock becomes larger and forms into a cave. As more rather rushes through the cave becomes a cavern.*

One of the annotations is that in the highlighted sentence, the entity 'space' has been 'Created', whereas, space is not even mentioned in this sentence (or before that). The annotator infers, from the following sentence, that a space must have been created at this point. These types of inference require real world knowledge that goes way beyond lexical semantics.

Finally, true entity state tracking requires entity-reference tracking in a given paragraph. For this, we need to go beyond sentence-level semantic analysis (i.e. what we have achieved in this work) and

ProPara Leaderboard			
	Precision	**Recall**	**F1 Score**
KOALA	0.777	0.644	0.704
TSLM	0.684	0.689	0.686
DynaPro	0.752	0.579	0.655
NCET	0.671	0.585	0.625
KG-MRC	0.693	0.493	0.576
LACE	0.753	0.454	0.566
Lexis Relaxed	0.64	0.48	0.55
ProStruct	0.743	0.430	0.545
AQA	0.620	0.451	0.523
ProGlobal	0.488	0.617	0.519
ProLocal	0.817	0.368	0.507
QRN	0.609	0.311	0.411
EntNet	0.547	0.307	0.394
Lexis (VN+PB)	0.36	0.39	0.38
Lexis (VN)	0.38	0.29	0.33

Table 5: ProPara Leaderboard and where different settings of Lexis would be placed.

False Negative Type	**Frequency**
RT	34
WK	30
GLE	22
REF	7
UP	3
ONT	2
Other	100
Total	198

Table 6: Underlying reasons for predicted labels judged as false negative. **RT** (*Reverse Tracing*): the annotator has assumed the existence or motion of an entity in a prior state for it to have an effect later. **WK** (*World Knowledge*): the annotator has used world knowledge to generate this label. **GLE** (*Gold Label Error*): This was judged as erroneous – something that is not true even given the world knowledge or reverse tracing. In the judgment task, this was considered an empty gold label. **REF** (*Reference Resolution*): the gold label is achievable given a reliable reference resolution system. **UP**: (*Unparsed*): state extraction failure due to parsing failure. **ONT** (*Ontology*): the gold label is achievable given a reliable entity ontology. **Other**: failure in predicting the state due to none of the above reasons.

analyze the whole paragraph as one unit of discourse. This is mandatory to uncover anaphora (or even cataphora) cases in the text, which is required to track the state of entities in a paragraph. For example, in the sentence *"Spray some plant food on it."*, our system predicts that the location of 'plant food' at the end of this sentence will be 'it', without finding the reference to 'seed'. State-of-the-art reference resolution is clearly indicated to improve results.

7 Conclusion and Future Work

The goal of this work was demonstrating how lexical resource-based methods using existing NLP resources could be utilized in automatic annotation of text for tracking entity states. The results were quite encouraging and demonstrated the potential of leveraging pre-existing rich lexical resources for challenging inference tasks. The limitations of our approach were examined, including VerbNet's lack of coverage compared to PropBank, another semantic role labeling lexical resource. We also found certain VerbNet predicates that were particularly opaque and benefited from expansion. In addition, and as expected, using the VerbNet semantic parser does not always yield a 100% accurate parse, resulting in downstream errors. In addition, there is a clear need for document-level entity state tracking, which requires automatic reference resolution.

Given the limitations of the VerbNet semantic parser discussed in section 6.2, we can expect to increase the recall by backing-off to a PropBank semantic parser (e.g. Li et al. 2020), that has been trained on much larger training data. This is the next step we plan to undertake. However, it is important to note that the reason we chose VNSP for this task is that it is the only semantic parser that generates the rich semantic representations in the predicate logic format that we employed in our method. Therefore, another possibility that looks more promising (but is a much longer-term solution) is continuing to update the VerbNet lexical resource based on the error analysis introduced in this work, annotating new data to cover examples from these updates, and re-training the VerbNet semantic parser.

Another area for planned future work is to use the generated entity states to augment the ProPara data and feed it into a machine learning model to assess performance improvement. Finally, as proposed in section 6.2, we will be adding a state-of-the-art reference resolution system to achieve document-level entity state tracking. All of these additions will be evaluated and reported on.

Acknowledgements

We thank the reviewers for their helpful feedback. We gratefully acknowledge the support of NSF under grants #1801446 (SATC) and #1822877 (Cyberlearning), and of DARPA CwC (subcontracts from UIUC and SIFT), DARPA AIDA Award FA8750-18-2-0016 (RAMFIS), DARPA KAIROS FA8750-19-2-1004-A20-0047-S005 (RESIN, sub to RPI, PI: Heng Ji), as well as DTRA HDTRA1-16-1-0002/Project 1553695 (eTASC - Empirical Evidence for a Theoretical Approach to Semantic Components). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of any government agency.

References

Claire Bonial, Kevin Stowe, and Martha Palmer. 2013. Renewing and revising semlink. In *Proceedings of the 2nd Workshop on Linked Data in Linguistics (LDL-2013): Representing and linking lexicons, terminologies and other language data*, pages 9–17.

Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao. 2017. Automatically labeled data generation for large scale event extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 409–419.

Peter Clark, Bhavana Dalvi, and Niket Tandon. 2018. What happened? leveraging verbnet to predict the effects of actions in procedural text. *arXiv preprint arXiv:1804.05435*.

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

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

Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*.

James Gung. 2020. *Abstraction, Sense Distinctions and Syntax in Neural Semantic Role Labeling*. Ph.D. thesis, University of Colorado at Boulder.

James Gung and Martha Palmer. 2021. Predicate representations and polysemy in verbnet semantic parsing. In *Proceedings of the 14th International Conference on Computational Semantics (IWCS)*, pages 51–62, Groningen, The Netherlands (online). Association for Computational Linguistics.

Aditya Gupta and Greg Durrett. 2019. Tracking discrete and continuous entity state for process understanding. In *Proceedings of the Third Workshop on Structured Prediction for NLP*, pages 7–12.

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. 2016. Tracking the world state with recurrent entity networks. *arXiv preprint arXiv:1612.03969*.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Yangfeng Ji, Chenhao Tan, Sebastian Martschat, Yejin Choi, and Noah A Smith. 2017. Dynamic entity representations in neural language models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1830–1839.

Paul R Kingsbury and Martha Palmer. 2002. From treebank to propbank. In *LREC*, pages 1989–1993. Citeseer.

Tao Li, Parth Anand Jawale, Martha Palmer, and Vivek Srikumar. 2020. Structured tuning for semantic role labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8402–8412, Online. Association for Computational Linguistics.

Martha Palmer. 2009. Semlink: Linking propbank, verbnet and framenet. In *Proceedings of the generative lexicon conference*, pages 9–15. GenLex-09, Pisa, Italy.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Karin Kipper Schuler. 2005. *VerbNet: A broad-coverage, comprehensive verb lexicon*. University of Pennsylvania.

Kevin Stowe, Jenette Preciado, Kathryn Conger, Susan Windisch Brown, Ghazaleh Kazeminejad, and Martha Palmer. 2021. Semlink 2.0: Chasing lexical resources. In *30th Annual Meeting of the Association for Computational Linguistics*, pages 222–227, Gronigen, Netherlands. Association for Computational Linguistics.

Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, Philip Yu, and Lifang He. 2020. Mixup-transformer: Dynamic data augmentation for NLP tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3436–3440, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Niket Tandon, Bhavana Dalvi, Joel Grus, Wen-tau Yih, Antoine Bosselut, and Peter Clark. 2018. Reasoning about actions and state changes by injecting commonsense knowledge. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 57–66.

Niket Tandon, Keisuke Sakaguchi, Bhavana Dalvi Mishra, Dheeraj Rajagopal, Peter Clark, Michal Guerquin, Kyle Richardson, and Eduard Hovy. 2020. A dataset for tracking entities in open domain procedural text. *arXiv preprint arXiv:2011.08092*.

Jizhi Tang, Yansong Feng, and Dongyan Zhao. 2020. Understanding procedural text using interactive entity networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7281–7290, Online. Association for Computational Linguistics.

Roselyne B Tchoua, Aswathy Ajith, Zhi Hong, Logan T Ward, Kyle Chard, Alexander Belikov, Debra J Audus, Shrayesh Patel, Juan J de Pablo, and Ian T Foster. 2019. Creating training data for scientific named entity recognition with minimal human effort. In *International Conference on Computational Science*, pages 398–411. Springer.

Ying Zeng, Yansong Feng, Rong Ma, Zheng Wang, Rui Yan, Chongde Shi, and Dongyan Zhao. 2018. Scale up event extraction learning via automatic training data generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.