

What Happened? Leveraging VerbNet to Predict the Effects of Actions in Procedural Text

Peter Clark, Bhavana Dalvi, Niket Tandon

Allen Institute for AI, Seattle, WA
{peterc,bhavanad,nikett}@allenai.org

Abstract

Our goal is to answer questions about paragraphs describing processes (e.g., photosynthesis). Texts of this genre are challenging because the effects of actions are often implicit (unstated), requiring background knowledge and inference to reason about the changing world states. To supply this knowledge, we leverage VerbNet to build a rulebase (called the Semantic Lexicon) of the preconditions and effects of actions, and use it along with commonsense knowledge of persistence to answer questions about change. Our evaluation shows that our system PROCOMP significantly outperforms two strong reading comprehension (RC) baselines. Our contributions are two-fold: the Semantic Lexicon rulebase itself, and a demonstration of how a simulation-based approach to machine reading can outperform RC methods that rely on surface cues alone.

Since this work was performed, we have developed neural systems that outperform PROCOMP, described elsewhere (Dalvi et al., 2018). However, the Semantic Lexicon remains a novel and potentially useful resource, and its integration with neural systems remains a currently unexplored opportunity for further improvements in machine reading about processes.

1 Introduction

Our goal is to answer questions about paragraphs describing processes. This genre of texts is particularly challenging because they describe a changing world state, often requiring inference to answer questions about those states. Consider the

Chloroplasts in the leaf of the plant trap light from the sun. The roots absorb water and minerals from the soil. This combination of water and minerals flows from the stem into the leaf. Carbon dioxide enters the **leaf**. Light, water and minerals, and the carbon dioxide all combine into a mixture. This mixture forms **sugar** (glucose) which is what the plant eats.

Q: Where is sugar produced?

A: in the leaf

Figure 1: A paragraph from *ProPara* about photosynthesis (bold added, to highlight question and answer elements). Processes are challenging because questions (e.g., the one shown here) often require inference.

paragraph in Figure 1. While reading comprehension (RC) systems (Seo et al., 2016; Zhang et al., 2017) reliably answer lookup questions such as:

- (1) What do the roots absorb?(A:water,minerals) they struggle when answers are not explicit, e.g.,
- (2) Where is sugar produced? (A:in the leaf)

(e.g., BiDAF (Seo et al., 2016) answers “glucose”). This last question requires knowledge and inference: If carbon dioxide *enters* the leaf (stated), then it will be *at* the leaf (unstated), and as it is then used to produce sugar, the sugar production will be at the leaf too. This is the kind of inference our system, PROCOMP, is able to model.

To perform this kind of reasoning, two types of knowledge are needed:

- (a) what **events** occur in the process (e.g., “CO2 *enters* the leaf”), and
- (b) what **states** those events produced (e.g., “CO2 *is at* the leaf”).

Prior work on event extraction (Berant et al., 2014; Hogenboom et al., 2011; McClosky et al., 2011; Reschke et al., 2014) addresses the former, allow-

Verb	Classes	Rule
assemble	build-26.1	IF (Agent “assemble” Product) THEN before: not exists(Product) & after: exists(Product)
assemble	build-26.1	IF (Agent “assemble” Material “into” Product) THEN before: not exists(Product) & after: exists(Product)
...		
enter	escape-51.1-2	IF (Theme “enter” Destination) THEN before: not is-at(Theme, Destination) & after: is-at(Theme, Destination)
enter	escape-51.1-2	IF (Theme “enter” - (PREP-src Initial.Location)) THEN before: not is-at(Theme, Destination) & after: is-at(Theme, Destination)
...		

Table 1: The Semantic Lexicon, a derivative and expansion of (part of) VerbNet, contains rules describing how linguistically-expressed events change the world. For example, for the sentence “CO2 enters the leaf”, the first rule for “enter” above will fire, predicting that is-at(“CO2”, “leaf”) will be true after that event.

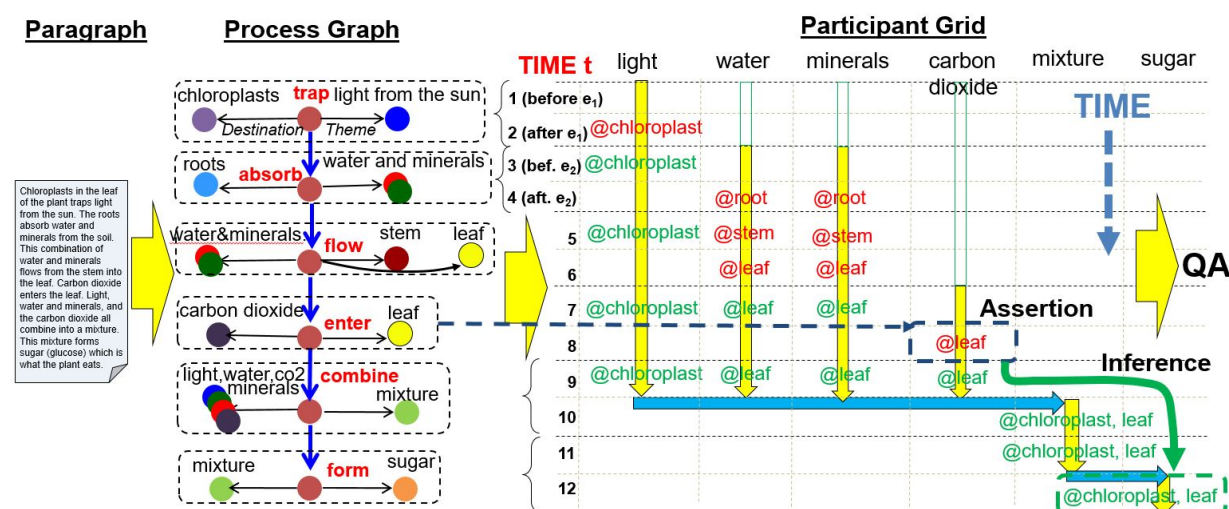


Figure 2: PROCOMP converts the paragraph to the Process Graph (same-colored nodes are coreferential) then to the Participant Grid, displaying the states of the process before/after each event (time vertically downwards). For brevity, @ denotes is-at(), arrows denote exists(), green denotes inferred literals. At line 8, the “CO2 enters leaf” step asserts that CO2 is therefore @leaf after the event. By inference, the sugar must therefore be produced at the leaf too (green box), a fact not explicitly stated in the text.

ing questions about event ordering to be answered. Our work addresses the latter, by also representing the states that occur. We do this with two key contributions:

- (1) a VerbNet-derived rulebase, called the Semantic Lexicon, describing the preconditions and effects of different actions, expressed as linguistic patterns (Table 1).
- (2) an illustration of how state modeling and machine reading can be integrated together, outperform RC methods that rely on surface cues alone.

We evaluate our work on an early version of the ProPara dataset containing several hundred paragraphs and questions about processes (Dalvi et al., 2018), and find that our system PROCOMP significantly outperforms two strong reading compre-

hension (RC) baselines.

Since this work was performed, we have developed neural systems that outperform PROCOMP, described elsewhere (Dalvi et al., 2018). However, PROCOMP provides a strong baseline for other work, the method illustrates how modeling and reading can be integrated for improved question-answering, and the Semantic Lexicon remains a novel and potentially useful resource. Its integration with neural systems remains a currently unexplored opportunity for future work.

2 Related Work

General-purpose reading comprehension systems, e.g., (Zhang et al., 2017; Seo et al., 2016), have become remarkably effective at factoid QA, driven by the creation of large-scale datasets, e.g.,

SQuAD (Rajpurkar et al., 2016), TriviaQA (Joshi et al., 2017). However, they require extensive training data, and can still struggle with queries requiring complex inference (Hermann et al., 2015). The extent to which these systems truly understand language remains unclear (Jia and Liang, 2017).

More recently, several neural systems have been developed for reading procedural text. Building on the general Memory Network architecture (Weston et al., 2014) and gated recurrent models such as GRU (Cho et al., 2014), Recurrent Entity Networks (EntNet) (Henaff et al., 2016) uses a dynamic memory of hidden states (memory blocks) to maintain a representation of the world state, with a gated update at each step. Similarly, Query Reduction Networks (QRN) (Seo et al., 2017) tracks state in a paragraph, represented as a hidden vector h . Neural Process Networks (NPN) (Bosse-lut et al., 2018), also models each entity’s state as a vector, and explicitly learns a neural model of an action’s effect from training data. NPN then computes the state change at each step from the step’s predicted action and affected entity(s), then updates the entity(s) vectors accordingly, but does not model different effects on different entities by the same action. Finally our own subsequent systems, PROLOCAL and PROGLOBAL, learn neural models of the effects of actions from annotated training data (Dalvi et al., 2018). In (Dalvi et al., 2018), we show that our rule-based system PROCOMP here also outperforms EntNet and QRN, but not PROLOCAL and PROGLOBAL. The integration of PROCOMP’s Semantic Lexicon with neural methods remains an opportunity for future work.

For the background knowledge of how events change the world, there are few broad-coverage resources available, although there has been some smaller-scale work, e.g., (Gao et al., 2016) extracted observable effects from videos for verbs related to cooking. One exception, though, is VerbNet (Schuler, 2005; Kipper et al., 2008). The Frame Semantics part of VerbNet includes commonsense axioms about how events affect the world, but it has had only limited use in NLP to date e.g. (Schuler et al., 2000). Here we show how this resource can contribute the background knowledge required to reason about change.

Finally, there are numerous representational schemes developed for modeling actions and

change, e.g., STRIPS (Fikes and Nilsson, 1971; Lifschitz, 1989), Situation Calculus (Levesque et al., 1998), and PDDL (McDermott et al., 1998). Although these have typically been used for planning, here we use one of these (STRIPS) for language understanding and simulation, due to its simplicity.

3 Approach

We now describe how our system, ProComp (“process comprehension”), infers the states implied by text, and uses those to answer questions. Figure 2 summarizes the approach for the running example in Figure 1.

The input to ProComp is a process paragraph and a question about the process, the output is the answer(s) to that question. ProComp operates in three steps: (1) the sequence of events in the process is extracted from the paragraph (2) ProComp creates a symbolic model of the *state* between each event, using the Semantic Lexicon, a VerbNet-derived database showing how events change the world, and then performs inference over it (3) questions within ProComp’s scope are mapped to computations over this model, and an answer is generated. The process states are displayed as a *Participant Grid*, shown in Figure 2.

3.1 The Semantic Lexicon

Before describing these steps, we describe the Semantic Lexicon itself. The purpose of the Lexicon is to encode commonsense knowledge about the states that events produce. A fragment is shown in Table 1. An **event** is an occurrence that changes the world in some way, and a **state** is a set of literals that describe the world (i.e., are true) at a particular time point. The Semantic Lexicon represents the relationship between linguistic event descriptions and states using using a STRIPS-style list of **before** (preconditions) and **after** (effects) expressed as possibly negated literals (Fikes and Nilsson, 1971)¹. While STRIPS used this knowledge for planning, we use it for simulation: *Given* that an event occurs in the process, its **before** conditions must have been true in the state beforehand, and its **after** conditions after.

Each entry in the Lexicon consists of a (WordNet sense-tagged) **verb** V , a **syntactic pattern** of

¹ The **after** list combines STRIPS’ original “add” and “delete” lists by using negation to denote predicates that become false (“deleted”).

the form (S V O (Prep NP)*) showing how V might be used to describe an event, and the **before** and **after** conditions. Most importantly, the syntactic pattern shows where the arguments of the before/after conditions may appear linguistically, allowing syntactic elements of an event-describing sentence to be mapped to the arguments in the before/after literals.

The lexicon currently models change in existence, location, size, temperature, and phase (solid/liquid/gas). It has entries for the preferred (most frequently used) senses of 2034 verbs². Note that some verbs may have multiple effects (e.g., “melt” affects both phase and temperature), while others may have none within the scope of our predicates (e.g., “sleep”).

For existence and location, the Lexicon was first initialized using data from VerbNet, transforming its Frame Semantics entries for each verb. VerbNet’s assertions of the form $pred(start(E), a_1, \dots, a_n)$ are converted to a **before** assertion $pred(a_1, \dots, a_n)$, when $pred$ is either `exists()` or `location()` (VerbNet does not explicitly model size, temperature, or phase changes). Similarly, assertions with `end(E)` or `result(E)` arguments become **after** assertions. For example, for the pattern (Agent “carve” Product), VerbNet’s Frame Semantics state that:

NP V NP

EXAMPLE	"Martha carves toys."
SYNTAX	<u>AGENT</u> V <u>PRODUCT</u>
SEMANTICS	NOT(EXIST(START(E), PRODUCT)) EXIST(RESULT(E), PRODUCT)

We transform this to a rule:

IF (Agent “carve” Product)
THEN before: not exists(Product)
& after: exists(Product)

We then performed a manual annotation effort over several days to check and correct those entries, and add entries for other verbs that affect existence and location. At least two annotators checked each entry, and added new entries for verbs not annotated in VerbNet. This was necessary as VerbNet is both incomplete and, in places, incorrect (its assumption that all verbs in the same Levin Class (Levin, 1993) have the same semantics only partially holds). Specifically, for the

² These are all verbs that occurred in a collection of high-school-level textbooks that we assembled, with the most frequently used VerbNet sense assigned to each.

2034 verbs in our Semantic Lexicon, VerbNet only covered 891 of them (using 193 Levin classes). The manual annotation involved checking and correcting these entries at the verb level, and added change axioms for the remaining verbs as appropriate (note that many verbs have a “no change” entry, if the verb’s effects are outside those modelable using our predicates).

For changes in temperature, size, and phase (not modeled in VerbNet’s Frame Semantics), we first identified candidate verbs that might affect each through corpus analysis (collecting verbs in the same sentence as temperature/size/phase adjectives), and then manually created and verified lexicon entries for them (~20 hours of annotation effort).

While most verbs describe a specific change (e.g., “melt”), there are a few general-purpose verbs (e.g., “become”, “change”, “turn”, “increase”, “decrease”) whose effects are argument-specific. These are described using multiple patterns with type constraints on their arguments.

The final lexicon contains 4162 entries (including empty “no change” entries; note that a single verb may have multiple entries for different sub-categorizations). It represents a substantial operationalization and expansion of (part of) VerbNet.

3.2 Step 1. Process Extraction (the Process Graph)

ProComp’s first step is to extract the events in the paragraph, and assemble them into a Process Graph representing the process. In ProComp a **process** is a sequence of events, represented as a **process graph** $G = (E, A, R_{ee}, R_{ea})$ where E and A are nodes denoting events (here verbs) and verb arguments respectively, and R_{ee} and R_{ea} are edges denoting event-event (e.g., next-event, depends-on) and event-argument (semantic role) relations respectively.

ProComp first breaks the paragraph into clauses using ClausIE (Corro and Gemulla, 2013). Then, each clause is processed in two ways:

- OpenIE (Banko et al., 2007) plus normalization rules convert each clause into a syntactic tuple of the same (S V O PP*) form as in the Lexicon. The tuple is then matched against patterns in the Lexicon to find the before/after assertions about the described event.
- Semantic role labeling (SRL) is performed to identify the participants and their roles in

the event that the clause describes. To increase the quality, we use an ensemble of SRL systems: neural network based DeepSRL (He et al., 2017), linguistic feature based EasySRL (Lewis et al., 2015), and OpenIE (Etzioni et al., 2011), with manually tuned heuristics to aggregate the signals together. Standard NLP techniques are used to normalize phrases, and a stop-list of abstract verbs is used to remove non-events. For example, in “CO2 enters the leaf”, “CO2” is labeled as the Agent and leaf as the Destination:



These roles are used to reason about the process (Step 2).

Finally, event (verb) nodes are connected by next-event links in the order they appeared in the text, making the assumption that events will be presented in chronological order (non-chronological event ordering is out of scope of this work). Argument nodes with the same headword are merged (i.e., coreferential). The full process graph G for the earlier paragraph is shown in Figure 2.

3.3 Step 2. Simulation (the Participant Grid)

In Step 2, ProComp uses the process graph G and Lexicon to infer the before and after states of each event, and then reasons over these states, essentially simulating the process. To do this, for each event e_i in G that matched an entry in the Lexicon, ProComp records facts true in the event’s before/after states using $holds-at(L, t)$ assertions in a state database, where L is a before/after literal and t is a time. For event e_i , we define the before time $t = (2i-1)$ and the after time as $t = 2i$. For example, for “carbon dioxide enters the leaf” step e_4 , ProComp finds and asserts

$$holds-at(is-at(carbon\ dioxide, leaf), 8)$$

i.e., after e_4 the carbon dioxide is *at* the leaf.

The contents of the database are displayed as a *Participant Grid*, a 2D matrix with a column for each participant p_j (verb argument), a row for each time step t (time proceeding vertically downwards), and each cell (j,t) containing the literals true of that participant p_j at that time t (i.e., where p_j is the first argument of the literal).

For inference, second-order frame axioms project facts both forward and backwards in time:

$$\begin{aligned} \forall L, t\ holds-at(L, t + 1) \leftarrow \\ holds-at(L, t) \ \& \ not \ \sim holds-at(L, t + 1) \\ \forall L, t\ holds-at(L, t - 1) \leftarrow \\ holds-at(L, t) \ \& \ not \ \sim holds-at(L, t - 1) \end{aligned}$$

i.e., if literal L holds at time t , and it is not inconsistent that L holds at time $t + 1$, then conclude that L holds at $t + 1$; similarly for $t - 1$; where *not* and \sim denote negation as failure and strong negation respectively. Note that this formalism tolerates inconsistency in the process paragraph: if a projected fact clashes with what is known, the $holds-at()$ assertions are not made and omitted from the intermediate Grid cells, denoting lack of knowledge. Some examples of inferred facts are shown in green in Figure 2, for example given CO2 is at the leaf at $t=8$, it is still at the leaf at $t=9$.

We also use seven rules expressing **commonsense knowledge** to complete the Grid further, including using information from events that do not have explicit change effects. These rules codify additional commonsense laws (the first 4 rules) and plausible inferences that follow from the pragmatics of discourse (the remaining 3 rules):

Location: If X is the Patient of event E, and E has initialLocation (resp. finalLocation) L (even if E doesn’t change a location), then X is at L before (resp. after) E.

Existence: If X is an Agent/Patient of event E, then (if no contrary info) X exists before and after E.

Colocation: If X is converted to Y (i.e., X consumed + Y produced), and X is at L, then Y is at L (and vice versa). For example, given that the CO2 is at the leaf at $t=9$, and is converted to a mixture, the mixture will be at the leaf after the mixing at $t=10$ (collocation rule), Figure 2.

Creation: If only one event E involves participant X, and X is the Patient in E, then X is produced (created) at E.

Destruction: If at least two events involves participant X, and X is the Patient in at least one of these, then X is consumed (destroyed) in the last of these events.

Dependency: If Event E_i is the first event involving participant X, and E_j is the next event that has X as its Agent/Patient, then E_j depends on E_i .

Default Dependency: In the absence of other information, event E_i depends on its previous event E_{i-1} .

3.4 Step 3. Question Answering

Given the participation grid, ProComp can answer 7 classes of question about the process:

- What is produced/consumed/moved during this process?
- Where is X produced/consumed?
- Where does X move from/to?
- What increases/decreases in temperature?
- What increases/decreases in size?
- What changes from solid/liquid/gas to a solid/liquid/gas?
- What step(s) does step X depend on?

Questions are posed as instantiated templates (ProComp is designed for state modeling, and does not currently handle linguistic variability in questions). Each template is twinned with a straightforward answer procedure that computes the answer from the participant grid. For example, in Figure 2, “Where is sugar produced?” is answered by the procedure for “Where is X produced?”. This procedure scans the X (“sugar”) column in Figure 2 to find where it comes into existence, and the location reported (chloroplasts, leaf). Each procedure thus codifies the semantics of each question class, i.e., maps the (templated) English to its meaning in terms of the predicates used to model the process.

Although these technology components (VerbNet, event extraction, and process modeling) are familiar, this is the first time all three have been integrated together, allowing latent states in language to be recovered and a new genre of questions to become answerable via modeling.

4 Experiments

4.1 Dataset

To evaluate this work, we used an earlier version of the ProPara dataset³ (Dalvi et al., 2018). ProPara is a new process dataset consisting of 488 crowdsourced paragraphs plus questions about different processes. The earlier version used here, called OldProPara, is similar but contains only 382 annotated paragraphs, uses a different train/dev/test split of 40/10/50, has additional annotations for size, temperature, phase, and event dependencies, and uses different (and easier) question templates (Section 3.4). OldProPara is available on request from the authors.

³available at <http://data.allenai.org/propara>

	Single Answer Questions (1675 questions)			All Questions (3562 questions)		
	P	R	F1	P	R	F1
ProRead	33.9	33.9	33.9	17.8	16.8	17.3
BiDAF	62.2	62.2	62.2	34.5	31.7	33.0
ProComp	63.8	70.5	67.0	73.7	76.0	74.8

Table 2: Results for ProComp and two recent RC systems (ProRead and BiDAF) on the OldProPara data (test set). Scores are macroaveraged, F1 differences are statistically significant ($p < 0.05$).

4.2 Baselines

We compared ProComp with two recent RC systems, BiDAF and ProRead. For subsequent comparisons of ProComp with the neural systems EntNet and QRN, see (Dalvi et al., 2018).

BiDAF (Seo et al., 2016) is a neural reading comprehension system that is one of the top performers on the SQuAD (paragraph QA) dataset⁴. We retrained BiDAF on our data, and found continued training (train on SQuAD then our data) produced the best results⁵, so we report results with that configuration.

ProRead (Berant et al., 2014) is a system explicitly designed for reasoning about processes, in particular answering questions about event ordering and event arguments⁶. We used ProRead’s pre-trained model, trained on their original data of annotated process paragraphs, appropriate for our task. ProRead is not easily extensible/retrainable as it requires extensive expert authoring of the entire process graph for each paragraph.

Because BiDAF and ProRead assume there is exactly one answer to a question (while OldProPara questions can have 0, 1, or more answers), we report scores on both the entire test set, and also the subset with exactly 1 answer.

4.3 Results

The results are shown in Table 2. ProComp significantly outperforms the baselines on both the full test set, and the single answer questions subset (removing ProRead and BiDAF’s disadvantage

⁴ <https://rajpurkar.github.io/SQuAD-explorer/>

⁵ BiDAF F1 scores on Single Answer Questions were 62.2 (train on SQuAD then continue training on OldProPara, Table 2), compared with 55.2 (retrain using only OldProPara), and 17.5 (original model, trained only on SQuAD). Similar differences hold on All Qns.

⁶ As ProRead only answers binary multiple choice (MC) questions, we converted each question into an N-way MC over all NPs in the passage, then used an all combinations binary tournament to find the winner (Landau, 1953).

of producing just one answer). The low ProRead numbers reflect that it does not model states (it was primarily designed to reason about event arguments and event ordering), with other questions, including most of the OldProPara questions, answered by an Information Retrieval fallback (Berant et al., 2014). ProComp also significantly outperforms BiDAF, but by a smaller amount. We analyze the respective strengths and weaknesses of ProComp and BiDAF in detail in the Analysis Section.

4.4 Ablations

We are also interested in two related questions:

1. How much have our extensions to the original VerbNet-derived lexicon helped?
2. How much has additional commonsense inference, beyond computing direct consequences of actions, helped?

To evaluate this, we performed two (independent) ablations: (1) removing the extensions and corrections made to the original VerbNet axioms in the Lexicon (2) disabling the inference rules about change; thus only states that directly follow from the Semantic Lexicon, rather than inferred via projection etc., are recovered. Table 3 shows the results of these ablations. For the first ablation, the results indicate that the VerbNet extensions have significantly improved performance (+5.9% F1 single answer questions, +2.4% F1 all questions). The second ablation similarly demonstrates additional inference using the commonsense rules improves performance (+6.6% F1 single answer questions, +2.1% F1 all questions). The relatively high performance even without this additional inference suggests that many questions in OldProPara ask about direct effects of actions. We analyze these further below.

5 Analysis

To understand the respective strengths of ProComp and BiDAF, we analyzed answers on 100 randomly drawn questions. The relative performance on these is below:

	ProComp incorrect	ProComp correct
BiDAF incorrect	21%	15%
BiDAF correct	14%	50%

5.1 ProComp Successes

There were 15 cases where ProComp was correct and BiDAF failed. For the majority of these,

ProComp Configuration	Single Answer Questions			All Questions		
	P	R	F1	P	R	F1
Full system	63.8	70.5	67.0	73.7	76.0	74.8
no VN extensions	58.2	64.3	61.1	71.2	73.6	72.4
basic inference only	59.4	61.4	60.4	72.8	72.6	72.7

Table 3: Results of two (separate) ablation experiments: (a) only use the original VerbNet axioms (ignore extensions/corrections) (b) only use inferences from the Semantic Lexicon (ignore inference rules about projection, collocation, etc.). F1 differences between the full and ablation versions are statistically significant ($p < 0.05$).

BiDAF either did not recognize a verb’s relation to the question, or was distracted by other parts of the paragraph. For example (where **T** is part of the paragraph, **QA** is the question and the systems’ answers, and bold is the correct answer):

T: ...A **roof** is built on top of the walls...
QA: What is created during this process?
roof (ProComp), concrete (BiDAF)

Here ProComp used the knowledge in the Lexicon that “build” is a creation event. In 2 cases, ProComp performed more complex, multi-event reasoning, e.g.,:

T: ...transport the aluminum to a **recycling facility**. The aluminum is melted down... The melted aluminum is formed into large formations called ingots. The ingots are transported to another facility...
QA: Where are the ingots moved from? a **recycling facility** (ProComp), another facility (BiDAF)

Here ProComp infers the aluminum is at the recycling facility (semantics of “transport to”, first sentence), projects this to the formation of the ingots, and thus that the ingots are also at the recycling facility before they are moved.

5.2 ProComp Errors

There were also 35 cases where ProComp made mistakes. We identified four classes of error, described below. The impact percentages below were judged using additional questions as well as the 35 failures.

1. NLP errors (~20%): Basic NLP errors can cause ProComp to fail, for example:

T: ...Rising **air** cools...

QA: What decreases in temperature? [no answer] (ProComp), **air** (BiDAF)

For this question, OpenIE fails to produce a tuple for “Rising air cools”, hence no semantic implications of the event could be inferred. In general the NLP pipeline can cause cascading errors; we later discuss how neural methods may be used to map directly from language to state changes, to reduce these problems.

2. Verb Semantics (~30%): In some cases there were subtle omissions in the Lexicon, for example:

T: Water evaporates up to the **sky**...

QA: Where does the water move to? [no answer] (ProComp), **sky** (BiDAF)

Here “evaporate” had been annotated as a change of phase and temperature, but not as a change of location. In contrast, BiDAF is likely using the surface cue “to” or “up to” in the text to answer correctly.

3. Complex knowledge and reasoning (~40%):

In some cases, knowledge and reasoning beyond ProComp is required to answer the question. For example:

T: ...The tiny ice particle...gathers more ice on the surface. Eventually the **hailstone** falls...

QA: What is created? supercooled water molecule (ProComp), updrafts (BiDAF)

In this example, the conversion of the ice particle to hailstone is implicit, and hence ProComp does not recognize the hailstone’s creation. Similarly:

T: ...Fill the tray with **water**....Place the tray in the freezer. Close the freezer door...

QA: What decreases in temperature? [no answer] (ProComp), **water** (BiDAF)

Reasoning that the water cools requires complex world knowledge (of freezers, containers, doors, and ramifications), beyond ProComp’s abilities. In contrast, BiDAF guesses water - correctly, in this case, illustrating that sometimes surface cues can suffice.

4. Annotation and scoring errors (~10%): In a few cases, the annotator-provided answers were questionable/incorrect (e.g., that tectonic plates are “consumed” during an earthquake), or our headword-based scoring incorrectly penalized the systems for correct answers.

6 Conclusion

Our goal is to answer questions about change from paragraphs describing processes, a challenging genre of text. We have shown how this can be done with two key contributions: (1) a VerbNet-derived rulebase (the Semantic Lexicon), describing how events affect the world and (2) an integration of state-based reasoning with language processing, allowing PROCOMP to infer the states that arise at each step. We have shown how this outperforms two state-of-the-art systems that rely on surface cues alone. The Semantic Lexicon is available from the authors on request.

Since this work was performed, we have subsequently developed two neural systems that outperform PROCOMP, described elsewhere (Dalvi et al., 2018). However, the Semantic Lexicon remains a novel and potentially useful resource, PROCOMP provides a strong baseline for other work, and the method illustrates how modeling and reading can be integrated for improved question-answering. An integration of PROCOMP’s Semantic Lexicon with a neural system remains a currently unexplored opportunity for further improvements in machine reading about processes.

References

- Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *IJ-CAI*. volume 7, pages 2670–2676.
- Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D Manning. 2014. Modeling biological processes for reading comprehension. In *EMNLP*.
- Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. *6th International Conference on Learning Representations (ICLR)*.
- Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proc. Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST’14)*. pages 103–111.
- Luciano Del Corro and Rainer Gemulla. 2013. Clausie: clause-based open information extraction. In *WWW*.
- 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 *Proc. NAACL'18*.
- Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam. 2011. Open information extraction: The second generation. In *Proc. IJCAI'11*.
- Richard E Fikes and Nils J Nilsson. 1971. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence* 2(3-4):189–208.
- Qiaozi Gao, Malcolm Doering, Shaohua Yang, and Joyce Yue Chai. 2016. Physical causality of action verbs in grounded language understanding. In *Proc. ACL'16*.
- Luheng He, Kenton Lee, Mike Lewis, and Luke S. Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proc. ACL'17*.
- Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. 2016. Tracking the world state with recurrent entity networks. *CoRR* abs/1612.03969.
- Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Su-leyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*. pages 1693–1701.
- Frederik Hogenboom, Flavius Frasincar, Uzay Kaymak, and Franciska De Jong. 2011. An overview of event extraction from text. In *DeRiVE 2011 Workshop*. Citeseer, volume 779, pages 48–57.
- Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proc. EMNLP'17*.
- Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proc. ACL'17*.
- Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. 2008. A large-scale classification of english verbs. *Language Resources and Evaluation* 42(1):21–40.
- HG Landau. 1953. On dominance relations and the structure of animal societies: Iii the condition for a score structure. *Bulletin of Mathematical Biology* 15(2):143–148. Also see [https://en.wikipedia.org/wiki/Tournament_\(graph_theory\)](https://en.wikipedia.org/wiki/Tournament_(graph_theory)).
- Hector Levesque, Fiora Pirri, and Ray Reiter. 1998. Foundations for the situation calculus.
- Beth Levin. 1993. *English verb classes and alternations: A preliminary investigation*. University of Chicago press.
- Mike Lewis, Luheng He, and Luke S. Zettlemoyer. 2015. Joint a* ccg parsing and semantic role labelling. In *EMNLP*.
- Vladimir Lifschitz. 1989. On the semantics of STRIPS. In *Reasoning about Actions and Plans*. Morgan Kaufmann, pages 1–9.
- David McClosky, Mihai Surdeanu, and Christopher D Manning. 2011. Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, pages 1626–1635.
- Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. 1998. Pddl-the planning domain definition language. Technical report, Yale Univ.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proc. EMNLP'16*.
- Kevin Reschke, Martin Jankowiak, Mihai Surdeanu, Christopher D. Manning, and Daniel Jurafsky. 2014. Event extraction using distant supervision. In *LREC*.
- Karin Kipper Schuler. 2005. *VerbNet: A broad-coverage, comprehensive verb lexicon*. Ph.D. thesis, Univ Colorado at Boulder.
- William Schuler, Liwei Zhao, and Martha Palmer. 2000. Parameterized action representation for virtual human agents. *Embodied conversational agents* page 256.
- Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *CoRR* abs/1611.01603.
- Minjoon Seo, Sewon Min, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Query-reduction networks for question answering. In *ICLR*.
- Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory networks. *arXiv preprint arXiv:1410.3916*.
- Junbei Zhang, Xiaodan Zhu, Qian Chen, Lirong Dai, and Hui Jiang. 2017. Exploring question understanding and adaptation in neural-network-based question answering. *arXiv preprint arXiv:1703.04617*.