ProPara-CRTS: Canonical Referent Tracking for Reliable Evaluation of Entity State Tracking in Process Narratives

Bingyang Ye and **Timothy Obiso** and **Jingxuan Tu** and **James Pustejovsky**
Brandeis University
415 South St, Waltham, MA, 02453
{byye, timothyobiso, jxtu, jamesp}@brandeis.edu

Abstract

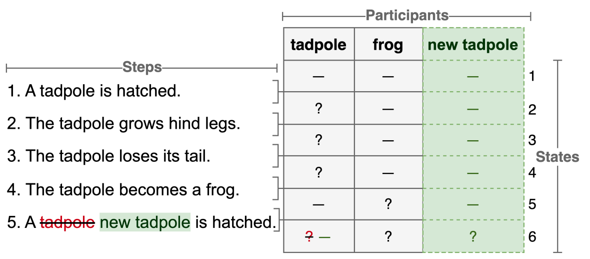
Despite the abundance of datasets for procedural texts such as cooking recipes, resources that capture full process narratives, paragraph-long descriptions that follow how multiple entities evolve across a sequence of steps, remain scarce. Although synthetic resources offer useful toy settings, they fail to capture the linguistic variability of naturally occurring prose. ProPara remains the only sizeable, naturally occurring corpus of process narratives, yet ambiguities and inconsistencies in its schema and annotations hinder reliable evaluation of its core task Entity State Tracking (EST). In this paper, we introduce a Canonical Referent Tracking Schema (CRTS) that assigns every surface mention to a unique, immutable discourse referent and records that referent's existence and location at each step. Applying CRTS to ProPara, we release the re-annotated result as ProPara-CRTS. The new corpus resolves ambiguous participant mentions in ProPara and consistently boosts performance across a variety of models. This suggests that principled schema design and targeted re-annotation can unlock measurable improvements in EST, providing a sharper diagnostic of model capacity in process narratives understanding without any changes to model architecture.

1 Introduction

Comprehending changes in a dynamic world is difficult. It requires the model not only to reason about state transitions across multiple steps but also to infer from knowledge of the world.

There has been considerable recent progress in understanding naturally occurring procedural texts, such as cooking recipes, WikiHow, etc. (Bosselut et al., 2017; Tandon et al., 2020), and establishing benchmarks over these datasets has helped drive research in this area of NLP.

Unlike procedural texts, *process narratives* describe a process in sequential steps in descriptive narratives rather than instructional, imperative steps. Consequently, the boundaries between "actions" are fuzzy, temporal ordering is not always explicit, and many state changes must be inferred from world knowledge rather than extracted from verb–object pairs. These characteristics make process narratives a stricter test of a model's capacity for dynamic, discourse-level reasoning than the formulaic language of recipes or instructional bullet points.


Participants
Steps
States
1. A tadpole is hatched.
2. The tadpole grows hind legs.
3. The tadpole loses its tail.
4. The tadpole becomes a frog.
5. A ~~tadpole~~ new tadpole is hatched.

	tadpole	frog	new tadpole
1	—	—	—
2	?	—	—
3	?	—	—
4	?	—	—
5	—	?	—
6	~~?~~ —	?	?

Figure 1: An annotated paragraph in ProPara. Each row shows the existence and location of participants before and after each step ("?" denotes "unknown", "-" denotes "not exist"). This example demonstrates the problem of referential confusion where the participant "tadpole" can refer to more than one entity. Red denotes the paragraph and annotation we find problematic, green denotes the annotation based on our new schema.

Although there exist a few datasets related to process narratives, many of them are not in natural language but are synthetic texts (Weston et al., 2015; Long et al., 2016). Recently there have been more datasets using natural language, but due to the rarity of the real-world data and expert knowledge needed to create these sets, most of these datasets are in a specific domain (Berant et al., 2014; Bosselut et al., 2017; Tandon et al., 2020; Fang et al., 2022; Rim et al., 2023; Zhang et al., 2024).

ProPara (Mishra et al., 2018; Tandon et al., 2018) is the only existing dataset of process narratives in natural language, covering diverse domains. It is a dataset of human-authored paragraphs of real-world processes, along with annotations about the changing states (existence and location) of entities in these processes. Figure 1 shows an annotated paragraph in ProPara. Annotators first construct the process steps given the prompt *"Describe the life cycle of a frog"*. Then they select entities of interests of the process as participants. Later, annotators track the existence (i.e., Create, Destroy, and None) and location changes of the participants. The resulting benchmark underpins the Entity State Tracking (EST) task, which requires models to predict those step-wise state transitions.

Despite its value as a comparatively large resource in a data-scarce genre, ProPara's original annotation schema is not fully aligned with the formal requirements of EST evaluation. In particular, the schema tolerates referential ambiguity and allows multiple surface mentions to be conflated under a single participant label, leading to inconsistent state chains. Figure 1 illustrates the problem: the label tadpole is used for two distinct entities introduced in Step 1 and Step 5, but both are erroneously merged. Consequently, a system that correctly predicts the destruction of the first tadpole is penalized because the gold annotation wrongly asserts that tadpole is (re)created later in the narrative. Such annotation artifacts obscure true model performance and impede principled analyses of reasoning errors.

To address these problems, we propose a **C**anonical **R**eferent **T**racking **S**chema (CRTS) and introduce a re-annotated dataset ProPara-CRTS. CRTS is a tightly specified annotation framework that assigns every surface mention in a process narrative to a unique, immutable discourse referent and obliges annotators to record that referent's existence and location at every step in the text.

The re-annotation proceeds in three interconnected stages. First, at the paragraph level, we apply Dense Paraphrasing (Tu et al., 2023) to rewrite or split sentences so that every entity mention is self-contained, eliminating referential ambiguity in the running text. Next, at the participant level, we merge coreferential mentions and assign each cluster a single canonical name—its CRTS referent—thereby establishing a strict one-to-one correspondence between discourse entity and participant label. Finally, at the state-label level, we traverse the revised step sequence and re-annotate existence and location, adding previously implicit transitions, sharpening coarse location spans, and guaranteeing that each referent experiences at most one state change per step.

To investigate the effectiveness of our re-annotation on the evaluation of the EST task, we train the state-of-the-art models on the re-annotated data and are able to achieve higher performances and the predictions are more reasonable based on human inspection.

We evaluate LLMs with a range of reasoning scaffolds and observe modest but consistent gains on ProPara-CRTS relative to the original corpus. This suggests that the cleaner reference mapping removes label noise that previously suppressed zero-shot scores. Even with best performing prompting, however, the model still trails supervised systems by a large margin. When we fine-tune a parameter-efficient Llama-3-8B on the CRTS training split, the model surpasses zero-shot LLMs and approaches the performance of fully supervised baselines, confirming that LLMs can internalize canonical referent tracking once given sufficient task-specific examples. [1]

2 Related Work

Procedural texts comprehension WIQA (Tandon et al., 2019) evaluates models' performance on "What if" questions regarding procedural texts. TRIP (Storks et al., 2021) is a dataset created to evaluate the reasoning ability of language models related to procedural physics texts. MARS (Wang and Song, 2024) evaluates the understanding of event and state changes in processes and how metaphysical changes to certain aspects of the process impact the process.

Entity state tracking Ma et al. (2022) propose a new model, CGLI, that builds local and global representations to track entities in procedural texts showing improvement on ProPara and TRIP. Other datasets focusing on the EST task include OpenPI (Tandon et al., 2020) and its derivations and iterations, OpenPI-C (Wu et al., 2023), and OpenPI2.0 (Zhang et al., 2024). At each step of a process, these datasets ask models to produce the entities involved in that step, the states about them that change, and the before and after states. Elazar et al. (2022) propose a new task, TNE (Text-based NP Enrichment) which aims to collect all relevant information about an NP from a paragraph. Throughout

[1]The source code and dataset is available at `https://github.com/brandeis-llc/ProPara-CRTS`.

a given text, this task challenges models to track the attributes of and participation of entities in events. EST is both a complement to and a component of this task.

Linguitically enriched reannotation High-quality annotation is needed for models to improve performance on tasks related to procedural text comprehension and EST. This is because they require a high level of semantic knowledge about the events and how the entities are involved. These tasks test models on their knowledge of the real-world intricacies of these events and how entities are created, moved, or destroyed through these processes.

Ménard and Mougeot (2019) and Tu et al. (2024) propose heuristics to recognize common annotation errors including typos, expert knowledge errors, protocol ambiguity, etc. These authors propose automatic processes for recognizing common annotation errors to create datasets of higher quality. Rezayi et al. (2021) uses external text to enrich graph representations which suffer from sparsity issues. The enhancement of this additional information improves performance on multiple datasets. Li et al. (2022) enhances social media posts with post metadata appended onto the post text. The authors show improvement over methods using non-enhanced text by fine-tuning a pre-trained language model using the enhanced data.

3 Canonical Referent Tracking Schema

Canonical Referent Tracking Schema (CRTS) is an annotation framework that maps every surface mention to a unique and immutable discourse referent—the single authoritative or *canonical* representation of that entity, corresponding to the participant in ProPara—and obliges annotators to record that referent's existence and location at every step in the discourse. Three constraints follow: (i) **referent uniqueness**—a one-to-one mapping between participants and mentions, so that referent of distinct entities are never conflated; (ii) **temporal atomicity**—a referent can undergo at most one state transition per step; and (iii) **complete state accounting**-consistent state and location values are obligatory for every mention at every step, including transitions that are only implicit in the text. When any of these constraints is violated, gold annotations no longer reflect the ground-truth reasoning problem, and evaluation scores conflate model error with annotation noise.

In this section, we show how ProPara systematically violates each principle and why those violations obscure a model's true competence in EST.

3.1 Violations of Referent Uniqueness

In the EST task, to successfully track the state change of a participant p, given an entity set E of the paragraph, a first and foremost premise is that we know which entity p refers to. Otherwise, our systems can be tracking the states of completely different entities than ProPara intends to.

The original ProPara annotations frequently breach the CRTS requirement of referent uniqueness, producing an ambiguous, non-canonical mapping from the participant list P to the underlying entity set E. These violations manifest in several recurring patterns:

Name confusion Many processes in ProPara are continuous. In a cycling process, it is common to see multiple entities with the same name undergo different actions (i.e., state and location changes). For the purpose of EST, if a participant shares the same name with all these entities, then it is impossible to ascertain which entity is of interest here.

In Figure 1, the paragraphs describe the process of the life cycle of a frog. There are two mentions of tadpole in the paragraph in Step 1 and Step 5 respectively, and the two mentions are parent tadpole and baby tadpole – two different entities. The annotator chooses the first *"tadpole"* as the entity of interest and put the name *"tadpole"* in the participant set then starts labeling its state changes. What seems fine from an annotator's point of view becomes confusing when it comes to someone who wants to use the date to do state tracking. When you asked about what the state of the participant *"tadpole"* is, it is impossible to know which tadpole we want to track without looking at the gold annotation, which is inaccessible during inference.

This also makes annotation error-prone as it is easy to confuse mentions of the same name as one entity. The annotation for participant *"tadpole"* in state 5 is such an example. According to the annotation, the tadpole becomes a frog and then transforms into a tadpole again, which is counterfactual.

Part-whole splits This happens when an entity e undergoes some state changes and splits to a few new entities where each resulting entity holds a part-whole relation with e. Each new split still shares the same name of e. This will also cause reference confusion when doing EST.

(1) *Step t: Water washes the sediment back.*

Step t+1: Some sediment is left as sand.

In example 1, sediment is partially moved in step t. The sediment in step t and step $t + 1$ are not the same entity, neither of which is the same as the sediment before. But since they share the same name, one cannot tell which one the participant *"sediment"* refers to. Furthermore, a situation like this makes annotating the split entities difficult.

Conditional branches Similar to part-whole splits, conditional sentences would create possible worlds where in each world there is a copy of the entity e. When asked about the state of entity e, one cannot tell which possible world the entity belongs to.

(2) *Step t: If the magma building is thick and sticky it will result in an explosive eruption.*

Step t+1: If the magma is thin and runs, the magma results in a low-pressure flow instead of a violent eruption.

In example 2, the changes of the entity magma are conditional in step t and step $t + 1$. When there is only one participant under the name *"magma"*, one cannot tell which magma should be linked to participant *"magma"*.

In each of the cases, referential drift distorts accurate evaluations by scoring correct inferences as errors.

3.2 Violations of Temporal Atomicity

Violations of temporal atomicity further erode evaluation fidelity. It is usually demonstrated in the following two ways.

Multiple transition Since EST is a step-wise prediction task, one entity can only undergo one action (transformation or location change) at each step. To properly evaluate EST on the sentence-level, there should be only one action per step for each entity.

(3) *Fallen rain or snow collects into surface water which will evaporate into water vapor again.*

In example 3, the entity *surface water* is assigned with two actions in one step, i.e., created from rain or snow and transformed into vapor by the annotators. A single timestep now contains two incompatible gold labels. Any model forced to choose one incurs a false negative on the other, capping maximum attainable performance.

Duplicate transition Sometimes, the same change is described across more than one step.

(4) *Step t: The light energy is used to convert carbon dioxide.*
...
Step t+2: The plant uses carbon dioxide in the air to produce glucose.

In example 4, step t and step $t + 2$ together describe the creation of glucose. Annotating CREATE in both steps does not correctly represent the state changes of the involved entities. Models that avoid double-counting a one-off event are scored as under-predicting, while models that parrot the duplication are rewarded.

3.3 Violations of Complete State Accounting

Finally, ProPara often omits or mis-codes required state information.

Overgeneralized locations When annotating the location of a participant, the annotations sometimes are too general when there is a more specific mention of the location.

(5) *Blood returns to left side of your heart.*

In example 5, the location of *"blood"* is annotated as heart, which is technically correct but is not informative enough. So a model that predicts the finer span is marked wrong

Inconsistent state coding The same action does not always get the same annotation. For example, the event *"die"* is sometimes annotated as DESTROY and sometimes annotated as NONE since the annotators think the remains of that entity still exist.

Missing implicit transition When an action can only be inferred from the context or when the input/output is not explicitly stated, the corresponding state transition can be overlooked by the annotators. This happens most frequently when there is a transformation that involves multiple inputs where some inputs being explicitly mentioned while the others not. The state changes of the entities that do not get mentioned explicitly are usually missing in the annotations.

(6) *Step t: A larvae matures inside of the egg.*

Step t+1: The caterpillar hatches from the egg.

In example 6, a transformation of larvae to caterpillar happens in step $t + 1$. However, since the action is not explicitly mentioned in the paragraph, the annotators miss to annotate the state of *"larvae"* as DESTROY. Thus a model that infers the disappearance of the larva is scored as incorrect, disincentivizing genuine causal reasoning

4 Re-annotating ProPara

To align ProPara with the CRTS constraints in §3, we perform a systematic re-annotation based on the Dense Paraphrasing (DP) technique (Ye et al., 2022; Tu et al., 2023). We treat DP as a truth-preserving textual enrichment strategy ϕ that maps a textual unit u (clause/sentence) to an enriched form $u^{+} = \phi(u)$ in which otherwise implicit event roles, entity distinctions, and state transitions licensed by local lexical and discourse context are made explicit, with semantic fidelity $u^{+} \Rightarrow u$. In this work, DP targets explicitness over economy by realizing missing arguments/roles, canonicalizing discourse referents, and expressing step-level state changes that are only implicit in u.

4.1 Re-annotation Actions

Resolving referent confusions via DP In the schema of ProPara-CRTS, we leverage DP to manually enrich entities with the same mention name so that their names become distinguishable while also reflecting the contexts. These enriched names establish a one-to-one mapping from the participant set P to the discourse-entity set E, ensuring that every participant has a single canonical referent and that no reference confusion can arise when models predict its state.

Whenever two lexically identical mentions denoted distinct discourse entities, the annotators should enrich these entities in DP style so that the names of these entities become distinct and can be easily linked to their corresponding entities. The names in the participant sets are subject to change per the names of entities they refer to. If necessary, the annotators will also add more participants to the participant set in one-to-many split situations like part-whole splits and conditional sentences.

In Figure 1, the *"tadpole"* in Step 5 should be re-annotated according to our new schema. It is re-annotated as *"new tadpole"* distinguishing it from the *"tadpole"* in Step 1. Also, *"new"* suggests that it is chronologically created in later steps.

Enforcing temporal atomicity Sentences that assign multiple transitions to one referent are divided into separate steps, or additional DP-distinguished referents are introduced so that each step contained exactly one action per entity. Duplicate transitions express across steps are merged into a single canonical event.

Completing state accounts For every referent at every step annotators record both an existence value and the most specific location span available. Missing implicit transitions (e.g. DESTROY of larva in the larva $\rightarrow$ caterpillar transformation) are inserted. Inconsistent action labels are also corrected.

Paragraph Issues Some paragraphs in ProPara are not a description of a process. One example describes how liver works with each sentence explaining a function of the liver. This kind of paragraphs are usually explanations where steps are not in order and few state or location changes are mentioned rather than process narratives. We believe these paragraphs do not qualify as process narratives and should not be included in the dataset. There are 36 paragraphs that fit in this category, and we exclude them from ProPara-CRTS.

4.2 Annotation Protocol and Qualiaty Control

Three trained graduate students with a background in Computational Linguistics annotate each paragraph in two passes. In Pass 1 the text is screened for non-process narratives; rejected paragraphs are removed. In Pass 2 manual error identification and CRTS corrections are applied, with DP edits logged and adjudicated when necessary. On a stratified sample of 100 paragraphs, we achieve a pairwise Cohen's κ of 0.72 for state labels and a pairwise F1 of 0.56 for location agreement.

4.3 Corpus Statistics

The re-annotated dataset, ProPara-CRTS, consists of 452 paragraphs and 13,417 state annotations. A total of 1,661 re-annotations were performed, encompassing updates to paragraphs, states, and par-

Group	Error type	Count	%
Referent Uniqueness	Name confusion	198	25.1
	Part–whole split	50	6.3
	Conditional branch	63	8.0
	Subtotal	**311**	**39.4**
Temporal Atomicity	Multiple transitions	40	5.1
	Duplicate transitions	45	5.7
	Subtotal	**85**	**10.8**
Complete State Accounting	Missing implicit transition	209	26.5
	Overgeneralized location	81	10.3
	Inconsistent state coding	67	8.5
	Subtotal	**357**	**45.2**
Paragraph Issues	Non-process paragraph	36	4.6
	Subtotal	**36**	**4.6**
Total		**789**	**100**

Table 1: Distribution of error types corrected during the Canonical Referent Tracking re-annotation of ProPara. Groups correspond to the three CRTS principles plus non-process paragraph issues.

ticipants, representing approximately 11% of the annotations in the original ProPara dataset. Most corrections in ProPara-CRTS address missing implicit changes, which accounts for 26.5% of all corrected errors. Name confusion and overgeneralizd location are the next two most frequent issues. The dataset maintains the same data splits as ProPara, with each partition corresponding to a subset of the original dataset partitions. Table 1 shows the statistics of corrected errors in ProPara-CRTS.

By integrating DP into the CRTS workflow, the new corpus removes referential drift without altering the underlying prose, thereby providing a sharply defined benchmark for measuring genuine model capacity in EST.

5 Experiments

In order to investigate the effectiveness of the new annotation, we evaluate different models against ProPara-CRTS and compare the results with those of the original ProPara dataset.

5.1 LLM Prompting

Considering the moderate modifications made to the original ProPara dataset, we believe that utilizing LLMs to perform inference on both test sets provides a valid basis for comparison.

We follow the experiment setups in MeeT (Singh et al., 2023) and frame EST into two subtasks: 1. A multi-choice problem where we ask the LLMs to select the state change of an entity from a fixed label set in step t. Similar to previous works (Zhang et al., 2021; Ma et al., 2022), we define six state types CREATE, NOT_CREATED, EXIST, MOVE, DESTROY, and WAS_DESTROYED. During evaluation, label NOT_CREATED, EXIST and WAS_DESTROYED will be mapped back to NONE. This enrichment of state label space helps the model differentiate the NONE types. 2. An extractive QA task that asks LLMs to extract the location of an entity in step t from the paragraph. Specifically, for each participant p, at each step t in the paragraph, we ask the LLMs two questions: 1) What is the state of p in step t? 2) Where is p located in step t? To preclude information leakage, the paragraph passed to the model is truncated at step t.

We compare four prompting strategies that differ only in the reasoning scaffold they present to the LLM while leaving the task formulation unchanged. The direct prompt elicits a terse answer and is decoded greedily with temperature 0. A Chain-of-Thought (CoT) variant adds the instruction "think step by step" and uses temperature 0.2 to encourage mild lexical diversity. The Self-Consistency setting draws eight independent CoT completions at temperature 0.7 and returns the state chosen by majority vote; when location spans disagree, the longest common subsequence is selected. Finally, a Few-Shot prompt supplies two worked examples drawn from the training split, each consisting of a short paragraph, the target entity, and the gold state and location pair.

We run the experiments using GPT-4o-mini and GPT-4o on the test sets of both ProPara and ProPara-CRTS. The detailed prompting queries to

Prompt	Dataset	Sent-level	Doc-level
Direct	ProPara	37.1	**59.6**
	ProPara-CRTS	**37.5**	58.8
CoT	ProPara	38.4	60.8
	ProPara-CRTS	**40.9**	**61.4**
Self-Cons.	ProPara	40.2	62.1
	ProPara-CRTS	**41.9**	**63.5**
Few-Shot	ProPara	39.0	61.3
	ProPara-CRTS	**40.0**	**62.2**

Table 2: Sentence-level and document-level F1 obtained by GPT-4o under four prompting scaffolds tested on ProPara and ProPara-CRTS test sets respectively.

the LLMs are shown in Appendix A.2.

5.2 LLM Fine-tuning

We also try fine-tuning LLMs for the EST task on the ProPara datasets. Due to the limitation of computing resources, we decide to fine-tune a smaller model `Llama 3.1 8B`[2] twice: once on ProPara and once on ProPara-CRTS. We fine-tune using the training sets of each dataset and evaluate their performance on the test sets of each dataset.

To fine-tune Llama 3.1, we make several adjustments to improve efficiency. We use 4-bit quantized models and LoRA (Hu et al., 2021) layers on all seven target modules available on Llama models. In addition, we use the `unsloth` method for gradient checkpointing. We use the CoT prompt during fine-tuning and Self-Consistency prompt for inference. Full fine-tuning hyperparameter set is reported in Appendix A.1.

5.3 Supervised Learning Models

We evaluate ProPara-CRTS with two supervised learning models MeeT (Singh et al., 2023) and CGLI (Ma et al., 2022), which are the top 2 models on the ProPara leaderboard[3]. MeeT formulates the EST into two subtasks: state prediction and location prediction. Then it fine-tunes the T5 model on both tasks. CGLI uses RoBERTa and leverages a decoding strategy that considers the context of each step on both local and global levels. We reuse the hyperparameters and settings reported by the authors to train the two models on ProPara-CRTS.

6 Results

Table 2 reports GPT-4o's performance under four prompting scaffolds, while Table 3 places those results alongside parameter-efficient fine-tuning and fully-supervised baselines.

Model	Dataset	Sent-level	Doc-level
GPT-4o-mini	ProPara	**24.8**	55.0
	ProPara-CRTS	23.2	**55.6**
GPT-4o	ProPara	40.2	62.1
	ProPara-CRTS	**41.9**	**63.5**
FT Llama 3.1	ProPara	59.5	64.9
	ProPara-CRTS	**63.6**	**65.9**
MeeT	ProPara	54.9	69.4
	ProPara-CRTS	**61.9**	**70.8**
CGLI	ProPara	65.4	72.4
	ProPara-CRTS	**69.5**	**75.1**

Table 3: Sentence-level and document-level evaluation results (F1) of models tested on ProPara and ProPara-CRTS test sets respectively.

Evaluation scheme Following the original ProPara setup, models are judged by their ability to answer three QA categories derived from an entity–step grid: (Cat1) whether an entity is created/destroyed/moved, (Cat2) at which step the change occurs, and (Cat3) where it occurs (Dalvi et al., 2018). In our reporting, *sentence-level* scores evaluate these predictions at the granularity of entity–step pairs, aggregated as macro/micro F_1 over Cat1–3. Complementarily, *document-level* scores treat all answers for a paragraph as a set of predicted tuples (entity, change-type, step, locations) and compute precision/recall/F_1 against the gold set, emphasizing global consistency across the whole process (Tandon et al., 2018).

Effect of canonical reference Across all settings ProPara-CRTS yields higher F1 scores than the original annotation except for GPT-4o-mini at sentence-level. For GPT-4o the absolute gain ranges from +0.4 points with the direct prompt to +1.7 points with self-consistency. The trend is even clearer for trained models: Llama-3-8B fine-tuned on CRTS improves by +4.1 points at the sentence level and by +1.0 points at the document level; CGLI and MeeT register gains of +4.1 and +7.0 points respectively. These consistent improvements confirm that enforcing a canonical, one-to-one referent mapping removes annotation noise that previously capped model scores.

Impact of reasoning scaffolds Moving from the direct question to Chain-of-Thought (CoT) increases GPT-4o's sentence-level score from 37.1 to 38.4 on ProPara and from 37.5 to 40.9 on CRTS. Adding self-consistency sampling delivers a further

[2]`unsloth/Meta-Llama-3.1-8B-Instruct-bnb-4bit`

[3]https://leaderboard.allenai.org/propara/submissions/public

gain, reaching 41.9 / 63.5 on CRTS—the best zero-shot result in our study. Few-shot prompting also helps, though its improvement is slightly smaller than that of self-consistency. The pattern suggests that canonical referent tracking particularly benefits prompts that require multi-step inference: once referential ambiguity is removed, the model's explicit reasoning is more likely to be correct and internally consistent.

Prompting versus training Even with the strongest scaffold, GPT-4o remains 22 points below the best supervised model (CGLI) at the sentence level and 12 points below at the document level. Fine-tuning a relatively small 8-billion-parameter Llama eliminates more than half of that gap, surpassing the older supervised systems MeeT and CGLI on the original corpus and approaching them on CRTS. These results indicate that canonical referent tracking narrows but does not erase the difference between prompt-only and parameter-updated approaches; models still gain substantially from task-specific training.

Canonical referent tracking lifts every method we tested, but the magnitude of the lift is modulated by the model's ability to exploit richer reasoning scaffolds or supervised updates. Prompt-engineering alone can reach the mid-40s F1 at the sentence level, yet fine-tuning remains essential for closing the gap to state-of-the-art supervised systems. We show the full results in Appendix A.3.

	Sent-level		Doc-level	
Model	ProPara	ProPara-CRTS	ProPara	ProPara-CRTS
FT Llama 3.1	61.9	**63.6**	65.0	**65.9**
MeeT	59.9	**61.9**	70.2	**70.8**
CGLI	67.2	**69.5**	71.1	**75.1**

Table 4: Cross-dataset evaluation results (F1) of models trained on the training sets of ProPara and ProPara-CRTS and tested against ProPara-CRTS test set.

	Sent-level		Doc-level	
Model	ProPara	ProPara-CRTS	ProPara	ProPara-CRTS
FT Llama 3.1	62.3	**64.6**	65.4	**67.5**
MeeT	60.8	**62.8**	70.8	**71.3**
CGLI	68.3	**71.2**	73.3	**76.5**

Table 5: Cross-dataset evaluation results (F1) of models trained on the training sets of ProPara and ProPara-CRTS and tested against the shared slice of ProPara and ProPara-CRTS test sets.

7 Analysis

To further investigate the effectiveness of the re-annotation, we look into the prediction differences in ProPara and ProPara-CRTS and see if the models trained on ProPara-CRTS understand process narratives better.

7.1 Cross-Dataset Evaluation

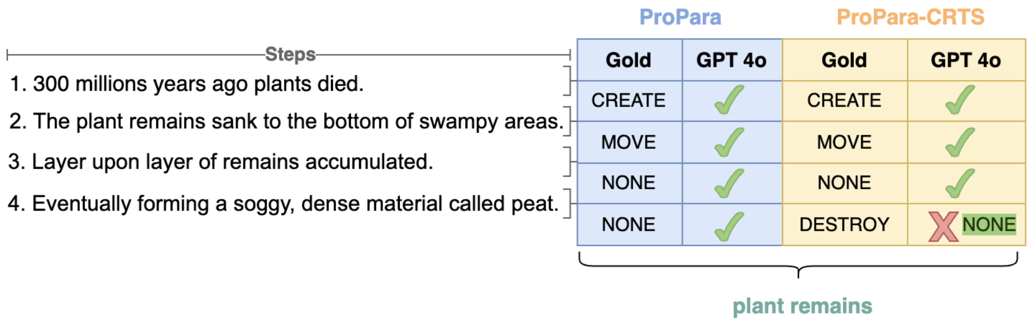
We perform a cross-dataset evaluation by training identical models on the training sets of ProPara and ProPara-CRTS, and then assessing their performance on ProPara-CRTS test set. Notably, models trained on ProPara all exhibit a decline in accuracy compared to those trained and tested exclusively on ProPara-CRTS, underscoring the critical value of high-quality training data and the advantages offered by ProPara-CRTS. The main findings are presented in Table 4.

We further evaluate models on the shared slice of both ProPara and ProPara-CRTS test sets, where no additional CRTS-only annotations are available. The intersection test set contains 42 paragraphs out of 52 of the original ProPara test set. The results are shown in Table 5. Training on ProPara-CRTS improves performance even when evaluation is restricted to the intersection of both test sets. Gains are consistent across all architectures. Because the test slice excludes CRTS-specific enrichment, these improvements indicate better generalization from cleaner training signals rather than artifacts of a richer label space.

Comprehensive results of both experiments are reported in Appendix A.3.

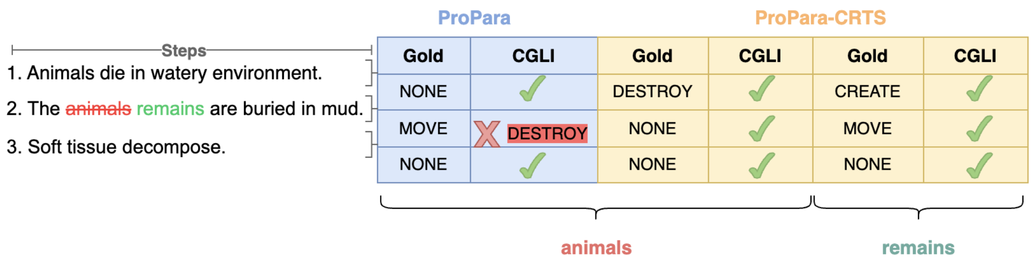
7.2 Qualitative Analysis

Qualitatively, we compare model predictions obtained from training on both datasets and observe that models trained on ProPara-CRTS effectively mitigate the shortcomings inherent in the original annotations. As illustrated in Figure 2a, we show the same paragraph in ProPara and ProPara-CRTS with different state annotations. In Step 4, the entity *plant remains* undergoes a transformation and forms into peat. The annotation in ProPara misses this transition because the input entity, *plant remains*, is not explicitly stated. The annotation in ProPara-CRTS corrects it. The prediction from GPT-4o on Step 4 is NONE as it fails to identify this implicit action as well. This mistake is also regarded as correct when evaluating against ProPara.


Steps
1. 300 millions years ago plants died.
2. The plant remains sank to the bottom of swampy areas.
3. Layer upon layer of remains accumulated.
4. Eventually forming a soggy, dense material called peat.
plant remains

ProPara		ProPara-CRTS	
Gold	GPT 4o	Gold	GPT 4o
CREATE	✓	CREATE	✓
MOVE	✓	MOVE	✓
NONE	✓	NONE	✓
NONE	✓	DESTROY	✗ NONE

(a) State predictions of entity *plant remains* by GPT-4o.


Steps
1. Animals die in watery environment.
2. The ~~animals~~ remains are buried in mud.
3. Soft tissue decompose.
animals
remains

ProPara		ProPara-CRTS			
Gold	CGLI	Gold	CGLI	Gold	CGLI
NONE	✓	DESTROY	✓	CREATE	✓
MOVE	✗ DESTROY	NONE	✓	MOVE	✓
NONE	✓	NONE	✓	NONE	✓

(b) State predictions of entities *animals* and *remains* by CGLI.

Figure 2: State predictions on the same paragraphs or their re-annotated counterparts in ProPara and ProPara-CRTS. The check denotes that the predictions match the gold. The cross denotes a mismatch with the gold. Green background of the prediction means it is factually correct, red otherwise.

This shows that if the gold annotations are problematic, the evaluation results can be misleading.

Figure 2b demonstrates an example where the re-annotation help CGLI better predict the states. In the example, CGLI is asked to track the state of participant *"animals"*. However, CGLI fails to identify that *die* is an action of DESTROY, and predicts that there is no state change for *"animals"* in Step 1. We suspect that this is because there is another mention of *"animals"* in Step 2 so CGLI assumes that the animals are still alive in Step 1. This is a mistake caused by reference confusion where the *"animals"* in Step 1 refers to living animals while the mention in Step 2 refers to animal remains, which should be differentiated in EST. Hence, CGLI is actually tracking the states of the wrong entity. By decontextualizing the *"animals"* in Step 2, we distinguish the two entities which share the same name. And the example shows that CGLI is able to predict the states of both participants correctly. This indicates that the canonical referent tracking schema help model to better comprehend process narratives.

8 Conclusion

We have presented ProPara-CRTS, a rigorously re-annotated version of the ProPara corpus that replaces the original, ambiguity-prone schema with a Canonical Referent Tracking Schema. CRTS enforces one-to-one mention–referent mapping, stepwise atomicity, and exhaustive state accounting. DP supplies the minimal lexical edits needed to make colliding mentions distinguishable while preserving the original prose. During re-annotation we also corrected recurrent paragraph and state-label errors, yielding 452 paragraphs with 13,417 noise-free state triples. Experiments spanning from LLM prompting, LLM fine-tuning and supervised models show consistent gains on CRTS. The results confirm three claims: (i) referential canonicalization removes label noise that previously suppressed scores; (ii) prompts that elicit multi-step reasoning profit most from the cleaner supervision; and (iii) despite these gains, EST remains challenging—supervised models still outperform purely prompted LLMs, underscoring the importance of dedicated training data. We release ProPara-CRTS, annotation guidelines, and validation scripts to facilitate future work on robust, semantics-aware evaluation of EST in natural-language process narratives.

9 Limitation

Due to resource constraints, only a subset of 100 paragraphs from ProPara underwent dual re-annotation, while the remaining paragraphs were subjected to single re-annotation. Consequently, the inter-annotator agreement was calculated solely based on this limited sample. Furthermore, for each paragraph, both the re-annotation of the paragraph text and the state labels were conducted by the same annotator, which could introduce potential bias into the annotations. Despite the involvement of three specially-trained annotators, the possibility of unintentional errors or subjective judgments remains.

References

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 *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1510, Doha, Qatar. Association for Computational Linguistics.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2017. Simulating action dynamics with neural process networks. *arXiv preprint arXiv:1711.05313*.

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, New Orleans, Louisiana. Association for Computational Linguistics.

Yanai Elazar, Victoria Basmov, Yoav Goldberg, and Reut Tsarfaty. 2022. Text-based NP enrichment. *Transactions of the Association for Computational Linguistics*, 10:764–784.

Biaoyan Fang, Timothy Baldwin, and Karin Verspoor. 2022. What does it take to bake a cake? the RecipeRef corpus and anaphora resolution in procedural text. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3481–3495, Dublin, Ireland. Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Jinning Li, Shubhanshu Mishra, Ahmed El-Kishky, Sneha Mehta, and Vivek Kulkarni. 2022. NTULM: Enriching social media text representations with non-textual units. In *Proceedings of the Eighth Workshop on Noisy User-generated Text (W-NUT 2022)*, pages 69–82, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Reginald Long, Panupong Pasupat, and Percy Liang. 2016. Simpler context-dependent logical forms via model projections. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1456–1465, Berlin, Germany. Association for Computational Linguistics.

Kaixin Ma, Filip Ilievski, Jonathan Francis, Eric Nyberg, and Alessandro Oltramari. 2022. Coalescing global and local information for procedural text understanding. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1534–1545, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Pierre André Ménard and Antoine Mougeot. 2019. Turning silver into gold: error-focused corpus reannotation with active learning. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 758–767, Varna, Bulgaria. INCOMA Ltd.

Bhavana Dalvi Mishra, 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. *arXiv preprint arXiv:1805.06975*.

Saed Rezayi, Handong Zhao, Sungchul Kim, Ryan Rossi, Nedim Lipka, and Sheng Li. 2021. Edge: Enriching knowledge graph embeddings with external text. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2767–2776, Online. Association for Computational Linguistics.

Kyeongmin Rim, Jingxuan Tu, Bingyang Ye, Marc Verhagen, Eben Holderness, and James Pustejovsky. 2023. The coreference under transformation labeling dataset: Entity tracking in procedural texts using event models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12448–12460, Toronto, Canada. Association for Computational Linguistics.

Janvijay Singh, Fan Bai, and Zhen Wang. 2023. Entity tracking via effective use of multi-task learning model and mention-guided decoding. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1255–1263, Dubrovnik, Croatia. Association for Computational Linguistics.

Shane Storks, Qiaozi Gao, Yichi Zhang, and Joyce Chai. 2021. Tiered reasoning for intuitive physics: Toward verifiable commonsense language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4902–4918, Punta Cana, Dominican Republic. Association for 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, Brussels, Belgium. Association for Computational Linguistics.

Niket Tandon, Bhavana Dalvi, Keisuke Sakaguchi, Peter Clark, and Antoine Bosselut. 2019. WIQA: A dataset for "what if..." reasoning over procedural text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6076–6085, Hong Kong, China. Association for Computational Linguistics.

Niket Tandon, Keisuke Sakaguchi, Bhavana Dalvi, Dheeraj Rajagopal, Peter Clark, Michal Guerquin, Kyle Richardson, and Eduard Hovy. 2020. A dataset for tracking entities in open domain procedural text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6408–6417, Online. Association for Computational Linguistics.

Jingxuan Tu, Kyeongmin Rim, Eben Holderness, Bingyang Ye, and James Pustejovsky. 2023. Dense paraphrasing for textual enrichment. In *Proceedings of the 15th International Conference on Computational Semantics*, pages 39–49, Nancy, France. Association for Computational Linguistics.

Jingxuan Tu, Keer Xu, Liulu Yue, Bingyang Ye, Kyeongmin Rim, and James Pustejovsky. 2024. Linguistically conditioned semantic textual similarity. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1161–1172, Bangkok, Thailand. Association for Computational Linguistics.

Weiqi Wang and Yangqiu Song. 2024. Mars: Benchmarking the metaphysical reasoning abilities of language models with a multi-task evaluation dataset.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart Van Merriënboer, Armand Joulin, and Tomas Mikolov. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*.

Xueqing Wu, Sha Li, and Heng Ji. 2023. OpenPI-C: A better benchmark and stronger baseline for open-vocabulary state tracking. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7213–7222, Toronto, Canada. Association for Computational Linguistics.

Bingyang Ye, Jingxuan Tu, Elisabetta Jezek, and James Pustejovsky. 2022. Interpreting logical metonymy through dense paraphrasing. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 44.

Li Zhang, Hainiu Xu, Abhinav Kommula, Chris Callison-Burch, and Niket Tandon. 2024. OpenPI2.0: An improved dataset for entity tracking in texts. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 166–178, St. Julian's, Malta. Association for Computational Linguistics.

Zhihan Zhang, Xiubo Geng, Tao Qin, Yunfang Wu, and Daxin Jiang. 2021. Knowledge-aware procedural text understanding with multi-stage training. In *Proceedings of the Web Conference 2021*, pages 3512–3523.

rank	16
lora_alpha	16
lora_dropout	0
target_modules	q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj
max_seq_length	2048
use_gradient_checkpointing	unsloth
per_device_train_batch_size	2
gradient_accumulation_steps	4
warmup_steps	5
num_train_epochs	1
learning_rate	2e-4
optim	adamw_8bit
weight_decay	0.01
lr_scheduler_type	linear

Table A1: Hyperparameters for Unsloth fine-tuning.

A Appendix

A.1 LLMs Fine-tuning

We report the hyperparameters for fine-tuning in Table A1. During fine-tuning Llama 3.1 on both datasets, the loss quickly decreases and then stabilizes during the first epoch of training. Therefore, we stop fine-tuning after one epoch. As shown in Figure A1, the loss functions for both datasets are nearly identical. This is expected as the task itself is not changing. The average loss for ProPara is 0.06618. For ProPara-CRTS, it is 0.06341, only a 4% difference.

While it is not a difficult task for humans, LLMs struggle to be competitive on the EST task. We believe this is due to the ambiguity associated with the task. Even with a limited set of responses, annotators will interpret these labels differently. The sharp initial decrease in loss is where the model learns the expected format of the answers. Very soon after starting training, the model produces correctly formatted responses, but they are less accurate than those collected after fine-tuning has concluded.

A.2 Prompts

Figure A2 illustrates the direct prompts we feed to LLMs for inference. Figure A3 demonstrates the CoT prompts we use for LLMs inference and fine-tuning. We also use the same prompt for self-consistency setting. Figure A4 shows the few-shot prompts for LLMs inference.

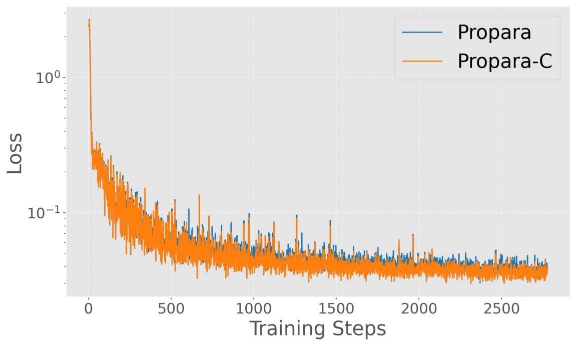


Figure A1: Fine-tuning loss on ProPara and ProPara-CRTS using Llama 3.1.

A.3 Results

We report the full sentence-level and document-level evaluation results of models on ProPara and ProPara-CRTS in Table A2. We report the cross-dataset evaluation results in Table A3 where EST models are trained on ProPara and ProPara-CRTS training sets respectively and tested on ProPara-CRTS test set. We report the cross-dataset evaluation results in Table A4 where EST models are trained on ProPara and ProPara-CRTS training sets respectively and tested on the shared slice of ProPara and ProPara-CRTS test sets.

Model / Train set		Sentence-level					Document-level		
Model	Train	Cat1	Cat2	Cat3	Macro	Micro	P	R	F1
GPT-4o-mini	ProPara	59.3	9.5	0.6	23.4	**24.8**	70.2	44.2	55.0
	ProPara-CRTS	53.7	10.8	0.6	21.7	23.2	68.2	41.9	**55.6**
GPT-4o	ProPara	70.8	36.4	11.5	39.6	40.2	63.1	61.2	62.1
	ProPara-CRTS	67.7	36.5	13.8	39.3	**41.9**	62.3	53.4	**63.5**
FT Llama 3.1	ProPara	78.1	55.2	45.6	59.6	59.5	66.4	63.5	64.9
	ProPara-CRTS	79.9	61.6	50.2	63.9	**63.6**	64.9	66.9	**65.9**
MeeT	ProPara	77.0	50.8	37.8	55.1	54.9	79.0	61.9	69.4
	ProPara-CRTS	81.1	62.6	43.9	62.5	**61.9**	78.5	64.5	**70.8**
CGLI	ProPara	81.1	61.7	53.8	65.5	65.4	74.9	70.0	72.4
	ProPara-CRTS	83.9	70.5	55.6	70.0	**69.5**	80.3	70.6	**75.1**

Table A2: Sentence-level and document-level evaluation results of models on ProPara and ProPara-CRTS.

Model / Train set		Sentence-level					Document-level		
Model	Train	Cat1	Cat2	Cat3	Macro	Micro	P	R	F1
FT Llama 3.1	ProPara	77.2	58.0	50.8	62.0	61.9	68.1	62.2	65.0
	ProPara-CRTS	79.9	61.6	50.2	63.9	**63.6**	64.9	66.9	**65.9**
MeeT	ProPara	78.6	55.6	46.0	60.0	59.9	81.3	61.7	70.2
	ProPara-CRTS	81.1	62.6	43.9	62.5	**61.9**	78.5	64.5	**70.8**
CGLI	ProPara	81.6	65.5	55.3	67.5	67.2	75.7	67.0	71.1
	ProPara-CRTS	83.9	70.5	55.6	70.0	**69.5**	80.3	70.6	**75.1**

Table A3: Cross-dataset evaluation results of the setting where EST models are trained on ProPara and ProPara-CRTS respectively and tested on ProPara-CRTS.

Model / Train set		Sentence-level					Document-level		
Model	Train	Cat1	Cat2	Cat3	Macro	Micro	P	R	F1
FT Llama 3.1	ProPara	78.8	58.4	50.6	62.6	62.3	67.8	63.1	65.4
	ProPara-CRTS	78.5	61.5	52.3	64.1	**64.6**	69.2	65.9	**67.5**
MeeT	ProPara	79.1	56.2	46.3	60.6	60.8	81.0	62.9	70.8
	ProPara-CRTS	81.3	63.4	44.7	63.1	**62.8**	78.2	65.4	**71.3**
CGLI	ProPara	82.3	68.0	55.3	68.5	68.3	77.8	69.2	73.3
	ProPara-CRTS	84.4	72.3	57.9	71.5	**71.2**	79.2	73.9	**76.5**

Table A4: Cross-dataset evaluation results of the setting where EST models are trained on ProPara and ProPara-CRTS respectively and tested on the shared slice of ProPara and ProPara-CRTS test sets.

System Prompt

You are an intelligent assistance that can track the state of an given entity from a document. The following is the document.

Document:
```
[1] A plant or animal dies in a watery environment.
[2] The remain is buried in mud and silt.
```

State Tracking Instruction

Based on the document, answer the question: What is the state of `<plant; animal>` exactly after step [2]?
- Create — entity absent before step [2] but present after
- Not_Created — entity has not been created yet after step [2]
- Move — entity present before step [2] and after, but in a different location
- Destroy — entity present before step [2] but absent after
- Was_Destroyed — entity had already been destroyed before step [2]
- Exist — entity present before step [2] and after, in the same location

You should only answer with Create, Move, Destroy, Not_created, Was_destroyed and Exist without saying anything else.

Response

Move; gold: Exist

Location Tracking Instruction

Based on the document, answer the question: Where is `<plant; animal>` located exactly after step [2]?
If you cannot tell the location of `<plant; animal>` after [2], answer with 'unknown'. If `<plant; animal>` is either not created yet or was destroyed before [2], answer with 'not_exist'. The extracted text should be exactly the same with that in the document. Only respond with the extracted text, do not say any other words or explain.

Response

in mud and silt; gold: not_exist

Figure A2: Direct prompt for LLMs inference.

System Prompt

You are an intelligent assistance that can track the state of an given entity from a document. The following is the document.

Document:
```
[1] A plant or animal dies in a watery environment.
[2] The remain is buried in mud and silt.\
```

State Tracking Instruction

Based on the document, answer the question: What is the state of `<plant; animal>` exactly after step [2]?
- Create — entity absent before step [2] but present after
- Not_Created — entity has not been created yet after step [2]
- Move — entity present before step [2] and after, but in a different location
- Destroy — entity present before step [2] but absent after
- Was_Destroyed — entity had already been destroyed before step [2]
- Exist — entity present before step [2] and after, in the same location

You should only answer with Create, Move, Destroy, Not_created, Was_destroyed and Exist without saying anything else.

Think step by step about the entity's existence and location from step 1 through [2].

Reasoning:
(Your chain-of-thought here.)

Answer:
State = <LABEL>
Location = <SPAN | unknown | not_exist>

Response

Move; gold: Exist

Location Tracking Instruction

Based on the document, answer the question: Where is `<plant; animal>` located exactly after step [2]?
If you cannot tell the location of `<plant; animal>` after [2], answer with 'unknown'. If `<plant; animal>` is either not created yet or was destroyed before [2], answer with 'not_exist'. The extracted text should be exactly the same with that in the document. Only respond with the extracted text, do not say any other words or explain.

Think step by step about the entity's existence and location from step 1 through [2].

Reasoning:
(Your chain-of-thought here.)

Answer:
State = <LABEL>
Location = <SPAN | unknown | not_exist>

Response

in mud and silt; gold: not_exist

Figure A3: CoT Prompt for LLMs inference and fine-tuning.

System Prompt

You are an intelligent assistance that can track the state of an given entity from a document.
You are first given two examples.

Example 1
<Example1>

Example 2
<Example2>

The following is the target document.
Document:

```
[1] A plant or animal dies in a watery environment.
[2] The remain is buried in mud and silt.
```

State Tracking Instruction

Based on the document, answer the question: What is the state of `<plant; animal>` exactly after step [2]?
- Create — entity absent before step [2] but present after
- Not_Created — entity has not been created yet after step [2]
- Move — entity present before step [2] and after, but in a different location
- Destroy — entity present before step [2] but absent after
- Was_Destroyed — entity had already been destroyed before step [2]
- Exist — entity present before step [2] and after, in the same location

You should only answer with Create, Move, Destroy, Not_created, Was_destroyed and Exist without saying anything else.

Response

Move; gold: Exist

Location Tracking Instruction

Based on the document, answer the question: Where is `<plant; animal>` located exactly after step [2]?
If you cannot tell the location of `<plant; animal>` after [2], answer with 'unknown'. If `<plant; animal>` is either not created yet or was destroyed before [2], answer with 'not_exist'. The extracted text should be exactly the same with that in the document.
Only respond with the extracted text, do not say any other words or explain.

Response

in mud and silt; gold: not_exist

Figure A4: Few-shot prompt for LLMs inference.