Dense Paraphrasing for Textual Enrichment

Jingxuan Tu* and **Kyeongmin Rim*** and **Bingyang Ye** and
Eben Holderness and **James Pustejovsky**
Department of Computer Science
Brandeis University
Waltham, Massachusetts
{jxtu,krim,byye,egh,jamesp}@brandeis.edu

Abstract

Understanding inferences from text requires more than merely recovering surface arguments, adjuncts, or strings associated with the query terms. As humans, we interpret sentences as contextualized components of a narrative or discourse, by both filling in missing information, and reasoning about event consequences. In this paper, we define the process of rewriting a textual expression (lexeme or phrase) such that it reduces ambiguity while also making explicit the underlying semantics that is not (necessarily) expressed in the economy of sentence structure as *Dense Paraphrasing (DP)*. We apply the DP techniques on the English procedural texts from the cooking recipe domain, and provide the scope and design of the application that involves creating a graph representation of events and generating hidden arguments through paraphrasing. We provide insights on how this DP process can enrich a source text by showing that the dense-paraphrased event graph is a good resource to large LLMs such as GPT-3 to generate reliable paraphrases; and by experimenting baselines for automatic DP generation. Finally, we demonstrate the utility of the dataset and event graph structure by providing a case study on the out-of-domain modeling and different DP prompts and GPT models for paraphrasing.

1 Introduction

Two of the most important components of understanding natural languages involve recognizing that many different textual expressions can correspond to the same meaning, and detecting those aspects of meaning that are not present in the surface form of an utterance or narrative. Together, these involve broadly three kinds of interpretive processes: (i) recognizing the diverse variability in linguistic forms that can be associated with the same underlying semantic representation (paraphrases); (ii) identifying semantic factors or variables that accompany or are presupposed by the lexical semantics of the words present in the text, through "hidden" arguments (e.g., *"stir vigorously."*; the argument of *stir* is not in the surface form); and (iii) interpreting or computing the dynamic consequences of actions and events in the text (e.g., *slicing an onion* brings about *onion slices*).

The first of these, the problem of paraphrasing, has been addressed computationally since the early days of natural language processing (NLP). The other two mentioned above, however, are more difficult to model with current machine learning approaches, which rely heavily on explicit textual strings to model semantic associations between the elements in the input. Many Question Answering (QA) systems, for example, rely on such syntagmatic forms in the training data for modeling potential associations that contribute to completion or generation task performance. Hence, if predicates or arguments are missing, implied, or interpreted from context, there is rarely anything to encode, and consequently little to decode as output, as well. Consider the following example from the traditional paraphrasing task. The text difference between the input and output only comes from a lexical substitution, rather than the rephrasing or addition of hidden arguments.

(1) Paraphrasing:
Chop onions, saute until browned. →
Cut *onions, saute until* ***done***.

To solve this problem, some recent attempts have been made to enrich surface forms that are missing information through "decontextualization" procedures that textually supply information which would make the sentence interpretable out of its

*These authors contributed equally to this work.

local context (Choi et al., 2021; Elazar et al., 2021; Wu et al., 2021).

The Focus of the decontextualization is on enriching text through anaphora resolution and knowledge base augmentation, which works well on arguments or concepts that can be linked back to existing knowledge sources, such as Wikipedia. Consider the following example of the this task. It is able to decontextualize *Barilla sauce* in (2a), but does not reintroduce any semantically hidden arguments from the context in (2b), making inferences over such sentences difficult or impossible.

(2) Decontextualization:
a. Add Barilla sauce, salt and red pepper flakes. →
Add Barilla sauce, ***the tomato sauce,*** *salt and red pepper flakes.*
b. Simmer 2 minutes over medium heat. →
Simmer 2 minutes over medium heat.

In this paper, we argue that the problems of paraphrasing and decontextualization are closely related for the purpose of clarifying meaning through verbal, nominal, or structural restatements that preserve (and enhance) meaning (Smaby, 1971; Kahane, 1984; Mel'cuk, 1995; Mel'Čuk, 2012). We propose *Dense Paraphrasing*, the process for the enrichment of the expression through both its lexical semantics and its dynamic contribution to the text in the whole narrative, which are less focused on by other work.

Consider the DPs of the sentences from examples (1) and (2), illustrated below in (3). Compared to the aforementioned tasks, DP aims to recover the semantically hidden arguments that fit the local context of the event (e.g., *pan* for the *saute* event) or carry a broader view of the context of the text (e.g., *sauted chopped onions* shows its transformation through multiple events).

(3) DP:
Chop onions, saute until browned. →
Chop onions ***on a cutting board*** ***with a knife*** *to get* ***chopped onions****, saute* ***chopped onions on a pan with a spatula****, resulting in* ***sauted onions*** *until browned.*

Add Barilla sauce, salt and pepper to the saucepan. Simmer 2 minutes over medium heat. →
Add Barilla sauce, salt and pepper to the saucepan ***by hand to get sauce mixture****. Simmer the* ***sauce mixture*** *2 minutes* ***in the saucepan*** *over medium heat to get* ***simmered sauce mixture****.*

We argue that our work can potentially help and complement these generation tasks by enriching the source text with information that is not expressed in the surface structure. Table 1 shows a complete dense paraphrased document that shows how DP is applied on a multi-sentence level. To show the usage of our method, we experiment with baselines of neural models for text generation tasks that involve dense paraphrased text, based on datasets that are heavily annotated with event-participant structures.

Passage: Peel and cut apples into wedges. Press apple wedges partly into batter. Combine sugar and cinnamon. Sprinkle over apple. Bake at 425 degF for 25 to 30 minutes.

Dense Paraphrased (DP'ed) Passage:
Using peeler, peel apples, resulting in peeled apples; and using knife on cutting board, cut peeled apples into peeled wedges.
Using hands, press apple wedges partly into batter in the cake pan.
Combine sugar and cinnamon in a bowl, resulting in cinnamon sugar.
Sprinkle cinnamon sugar over apple wedges in batter in cake pan, resulting in appelkoek.
In oven, bake appelkoek at 425 degF for 25 to 30 minutes, resulting in baked appelkoek.

Table 1: Example DP'ed document from our dataset. Color-coded text spans represent locations of events in the input text where dense paraphrases are generated to enrich local context. Underlined text shows the appearance of the ingredient "apple" with transformation in a chain of events. Hidden arguments are added back to the text following simple syntactic rules (e.g., *using X, do Y in/on/at Z, resulting in R*).

In the remainder of the paper, we first review related work and background (§2), and give more detailed definitions of the DP schema (§3). We then apply the DP techniques on a cooking recipe dataset to show its ability to enrich the raw text with paraphrases (§4). §5 provide details of experiments we conducted to validate the utility of the proposed methodology, along with a discussion of our results. §6 explores the case studies on applying DP on the out-of-domain data and the comparison between GPT models on the paraphrasing task. We then conclude our work in §7. The source code and data will be publicly available.

2 Related Work

There is a long history in linguistics, dating back to the early 1960s, of modeling linguistic syntagmatic surface form variations in terms of transformations or sets of constructional variants (Harris, 1954, 1957; Hiż, 1964). (Smaby, 1971) formally defines this process of preserving the meaning from lexical, phrasal, or sentential expressions E_i to E_j as paraphrasing.

For NLP uses, paraphrasing has been a major part of machine translation and summarization system performance (Culicover, 1968; Goldman,

1977; Muraki, 1982; Boyer and Lapalme, 1985; McKeown, 1983; Barzilay and Elhadad, 1999; Bhagat and Hovy, 2013). In fact, statistical and neural paraphrasing is a robust and richly evaluated component of many benchmarked tasks, notably MT and summarization (Weston et al., 2021), as well as Question Answering (Fader et al., 2013) and semantic parsing (Berant and Liang, 2014). To this end, significant efforts have gone towards the collection and compilation of paraphrase datasets for training and evaluation.

In addition to the meaning-preserving paraphrase strategies mentioned above, there are several directions currently explored that use strategies of "decontextualization" or "enrichment" of a textual sequence, whereby missing, elliptical, or underspecified material is re-inserted into the expression. The original and target sentences are compared and judged by an evaluation as a text generation or completion task (Choi et al., 2021; Elazar et al., 2021; Gao et al., 2022; Chai et al., 2022; Eisenstein et al., 2022; Tu et al., 2022b; Ye et al., 2022; Katz et al., 2022). Our work applies both strategies of paraphrasing to the procedural text domain, which is new to the field. Unlike typical paraphrase generation tasks (Zhou and Bhat, 2021) which paraphrase full sentences and favor different wording and structure, our task performs at the entity-level.

Recent studies in procedural texts focus on tracking the state of events and entities in artificial corpora from arbitrary domains (Dalvi et al., 2019; Kazeminejad et al., 2021; Tandon et al., 2020). Some works also treat recipes as a rich resource for procedural texts. (Bosselut et al., 2017; Yamakata et al., 2020) leverage structured representations of domain-specific action knowledge for modeling a process of actions and their causal effects on entities. Other works try to resolve the anaphoric relations between recipe ingredients (Fang et al., 2022; Jiang et al., 2020). While these works all create corpora suitable for their own problems, our work, in contrast, embeds enriched information of both entities and events in the recipe using dense paraphrasing.

Enrichment of VerbNet predicates can be seen as an early attempt to provide a kind of Dense Paraphrasing for the verb's meaning. In Im and Pustejovsky (2009, 2010), the basic logic of *Generative Lexicon*'s subevent structure was applied to VerbNet classes, to enrich the event representation for inference. The VerbNet classes were associated with event frames within an Event Structure Lexicon (ESL), encoding the subevent structure of the predicate. If the textual form of the verb is replaced by the subeventual description itself, classes such as `change_of_location` and `change_of_possession` can help encode and describe event dynamics in the text, as shown in (Brown et al., 2018; Dhole and Manning, 2021; Brown et al., 2022). For example, the VerbNet entry *drive* is enriched with the ESL subevent structure below:

(4) **drive** in *John drove to Boston*
se1: pre-state: not_located_in (john,boston)
se2: process: driving (john)
se3: post-state: located_in (john,boston)

Such techniques will be utilized as part of our Dense Paraphrasing strategy to enrich the surface text available for language modeling algorithms.

3 Method

In this section, we detail the procedure involved in creating DPs. The DP method can be seen as the method for creating sets of semantically "enriched, but consistent" expressions, that can be exploited by either human consumption (e.g., natural language paraphrases) or machine consumption (e.g., configurable graphs). Specifically, we currently adopt a template-based method along with heuristics to generate DPs that account for hidden entities and entity subevent structure.

Sub-Event Structure DP starts by identifying events from the text. As mentioned above, ESL represents an event as having three parts: **begin** (B_e), **inside** (I_e), and **end** (E_e). In our method, we use this subevent structure not only to track the begin and end state of an event, but to create *textual redescriptions* of the changed event arguments. To illustrate, in Table 1 the *peel* and *cut* events form a two-event sequence through the DP subevent descriptions of the beginning and ending entities (*apples* $\rightarrow$ *peeled apples* $\rightarrow$ *apple wedges*).

Hidden Arguments DP also recovers hidden arguments that are not present in the surface form of the text to ensure the richness of the subevents. The changed entities associated with the begin or end events can be either hidden or explicit. For example, the *bake* event from Table 1 has both the hidden beginning and ending entity. In addition,

DP also recovers relevant arguments in the same context of the event (e.g., the *bake* event occurs in the *oven*).

4 Experiment 1: Dense Paraphrasing from annotation

We use the text data from the subdomain of cooking recipes to demonstrate the application of the DP. Compared to texts of news or narratives, procedural text such as recipes tend to be task-oriented and highly contextualized, allowing the DP to focus on the hidden information and changes that are taking place in the course of a sequence of events in the narrative. Specifically, we apply the DP on the existing Coreference under Transformation Labeling (CUTL) dataset (Rim et al., 2023). CUTL consists of a subset of 100 cooking recipes from a larger Recipe-to-Video Questions (R2VQ) dataset (Tu et al., 2022a). It contains rich annotation of the cooking-related events and entities (both explicit and hidden), as well as the coreference relations between the entities.

4.1 Event structure for Dense Paraphrasing

To prepare the CUTL dataset for the DP, we transform the annotation into a set of "events", as events are primary anchors for applying DP. Adapted from (Rim et al., 2023), we define an event as an event predicate, a set of cooking-related entities and relations. The ingredient entities are associated with the begin and end subevents (of the event predicate) and re-described to show the subevent change. An example is shown in Figure 1. The entity can be hidden or explicit, and the entity types include the EVENT-HEAD, INGREDIENT, TOOL and HABITAT. The relations include BEGINNING and ENDING for ingredients, as well as PARTICIPANT-OF for tools and habitats. Each event has only one predicative verb (EVENT-HEAD), and all the relations within the event are linked from corresponding entities to the predicate. In addition, the event must have at least one beginning ingredient entity and one ending ingredient entity. Table 2 shows the statistics of the events in the dataset. The high ratio of the hidden entities makes it effective to demonstrate the utility of the DP.

4.2 Paraphrasing Hidden Entities

In this stage, we propose a semi-automatic approach to paraphrase the hidden entities that are annotated and represented in text placeholders

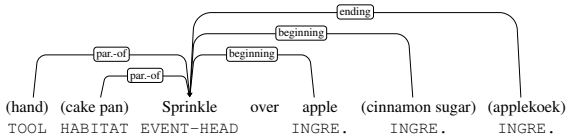


Figure 1: Annotated event example (combined R2VQ and CUTL annotations). Hidden entities are enclosed in parenthesis.

Avg. # of entities per recipe	Explicit	Hidden
EVENT-HEAD	10.6	N/A
TOOL	0.8	2.7
HABITAT	2.1	4.0
INGREDIENT (beginning)	12.0	9.4
INGREDIENT (ending)	1.0	10.4

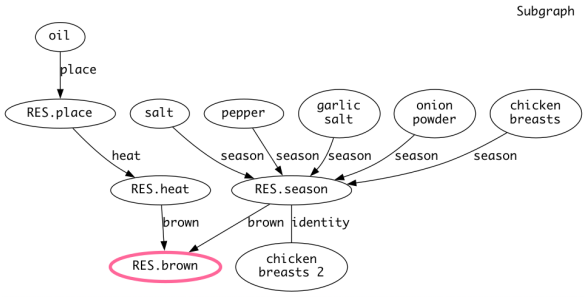
Table 2: Statistics of the events in the CUTL dataset.

(`verb.RES`) from the CUTL annotation. Formally, it involves two steps: generate text realizations of the hidden entities, and paraphrase the text realization to be useful for DP or other downstream tasks. We propose two methods to create the text realization of hidden entities: prefix paraphrasing (PP) and subgraph linearization. For the latter, we apply GPT-3 (Brown et al., 2020) on the text realizations to generate paraphrases, and then compare the generated PP paraphrase, the subgraph paraphrase, and the PP text directly used as the paraphrase.

Text Realization PP is a heuristic method introduced by (Tu et al., 2022b) for question generation, which enriches the textual description of entities to reflect changes due to actions. We adopt this idea by first separating all the event predicates appearing in the data into three categories: TRANSFORMATION, LOCATION-CHANGE, and neither. For transformation events, the paraphrased entity has the format `eventPrefix + entity` (e.g. *boiled water, drained soaked peas*). For location change events or neither, the paraphrased entity has the same text form as the event input.

Given the graphical nature of the coreference graph from the DP events, we also use linearized graphs as the text realization, which has shown to be useful in various tasks such as syntactic parsing and AMR parsing (Vinyals et al., 2015; Bevilacqua et al., 2021). Specifically in our task, we extract the subgraph that is rooted in the hidden entity mention node, and then linearize it into a string literal. Examples from text realization methods are presented in Figure 2. PP converts transformation verbs into

prefixes (e.g., *heated, seasoned*) and drops location change verbs (e.g., *place*). It also uses the identity link from the graph to find single entity texts that can substitute parts of the prefix-paraphrased text (e.g., `chicken breast 2` at the bottom of fig. 2 replaces the PP text for `RES.season` in the target realization.). Subgraph realization, on the other hand, records all the subevent state changes relevant to the target entity, and the events are also typed with the relations based on the verb sense and the number of beginning and ending ingredients that are connected to the verb.



Implicit mention provenance:
Brown the chicken breasts on each side. → `RES.Brown`

PP realization:
```
'browned seasoned chicken breasts and heated oil'
```

Subgraph realization:
```
['brown-AGG',
 [['season-AGG',
   ['chicken breasts', 'salt', 'pepper',
    'garlic salt', 'onion powder']],
  ['heat-TRANS', [['place-COL', ['oil']]]]]]
```

Figure 2: Text realization from PP and subgraph. Subgraph realization is wrapped and indented for readability. Event verbs are typed with: `AGG` (aggregation), `TRANS` (transformation), `COL` (change of location), etc.

Paraphrase Generation We prepare the paraphrasing data for evaluation by extracting all the ingredient mention nodes from the graph that satisfy: (1) the node is linked to a begin subevent, and to another end subevent; (2) the node has explicit text form. Such a node is connected to its placeholder text with the IDENTITY relation, as shown in Figure 2. Then we use the text of such nodes as the gold paraphrase to the hidden entity placeholder. In the end, we collected 273 gold paraphrase pairs from our dataset. Considering the scarcity of gold paraphrase in the dataset (2.7 pairs per recipe), we formalize the task as few-shot prompting and apply the GPT-3-davinci model to generate the paraphrases. Figure 3 shows the example prompts used in the GPT-3 paraphrasing methods. In each prompt, we use a single set of eight exemplars from the gold pairs and a human-created instruction on the task and how to interpret the input from different text realizations.

Paraphrase	BERTScore	Intrinsic
PREFIXP	81.15	3.08
PREFIXP-GPT	84.45 (±0.46)	3.97 (±0.08)
SUBGRAPH-GPT	**86.08** (±0.15)	**4.15** (±0.02)

Table 3: Paraphrase generation results on the gold paraphrase pairs. PREFIXP uses PP realization directly as the paraphrase; PREFIXP/SUBGRAPH-GPT uses DP/-subgraph realizations as exemplars in GPT-3 prompting.

Evaluation We use BERTScore (Zhang et al., 2019) for automatic evaluation and a 5-point Likert scale as intrinsic evaluation for the correctness, relevance, and appropriateness. For each type of realization, we perform two rounds of GPT-3 prompting with different sets of gold exemplars, and present the overall results in Table 3. While ROUGE (Lin, 2004) has been widely used in text-generation tasks, it is shown that these token-matching metrics do not align well with human annotation (Shen et al., 2022), and this finding aligns with what we observed in our experiments.

The BERTScore from all paraphrases is over 80, indicating the higher semantic similarity between the gold and model output. PREFIXP has the lowest BERTScore due to the text addition from verb prefixes and the lack of summarization ability over a list entities in the input. For intrinsic evaluation, SUBGRAPH-GPT performs better than PREFIXP-GPT, suggesting that the subgraph realization is a better resource for GPT-3 to recover and summarize the essential information in paraphrasing. PREFIXP performs the worst in the intrinsic evaluation. From the summary of annotators' feedback on the evaluation, we observe that the PP paraphrase of the entity from later steps tends to be lengthy and redundant without signaling the salient entity (average token numbers of PP paraphrase is 7.4, whereas it is 2.4 in GPT-generated paraphrases). In addition, PP paraphrase alone is less natural and less understandable to humans.[1] At the end, we validate the paraphrasing results from SUBGRAPH-GPT, and incorporate them into the following experiments.

[1]One low-scored example of the DP paraphrase: *stirred egg and water and black pepper and garlic granules*.

```
The task is to generate short and accurate paraphrase of the given noun phrases.
The input noun phrase describes the event state change of the food ingredients
through processing in a recipe, and the output paraphrase should summarize the
combination or state change of the ingredients.

input: stirred butter mixture and flour and cocoa and baking soda and salt
output: dough

[7 more exemplars]


input: squeezed horseradish
output:
```

```
The task is to generate short and accurate paraphrase of the given logical expression.
The input logical expression describes the cooking events and state change of the food ingredients
through processing in a recipe, and the output paraphrase should summarize the combination or state
change of the ingredients.

event types in the logical expression:
TRANSFORMATION: event that transforms the state, shape and etc. of an ingredient
AGGREGATION: event that combines multiple ingredients together
SEPARATION: event that separate an ingredient, or remove part of the ingredient
LOC: move the ingredient to another location

input: ['reserve-TRANSFORMATION', [['combine-AGGREGATION', ['onion', 'chilies', 'cilantro', 'salt']]]]
output: reserved onion mixture

[7 more exemplars]

input: ['squeeze-TRANSFORMATION', ['horseradish']]
output:
```

Figure 3: GPT-3 Prompt templates for the PREFIXP-GPT (top) and the SUBGRAPH-GPT (bottom).

5 Experiment 2: End-to-end DP

In this section, we present experiments of the task for automatic generation of the DP text. we explore baselines from language models and provide further insights on our data. We formalize DP generation as the task of identifying textual event mentions from cooking recipe text as well as their associated hidden entities or text mentions.

Experiment Setup We use the recent sequence-to-sequence generation model T5 (Raffel et al., 2020) as the baseline. We set the output sequence to be 'label-enclosed' text with special symbols to mark up the patterns that can be effectively processed by the models (Zhai et al., 2022). An example sequence is shown in Figure 4. We randomly sample 80 recipes for training and hold out 20 for testing. Model performance was evaluated using F1-score. We fine-tune the T5-base model on the training set, and leverage the effect from either using single sentence or aggregated sentences as the input sequence, and using additional recipe data for the augmentation.

Model Details We fine-tune the T5 text generation model (Raffel et al., 2020) to perform the task on the training set with a maximum of 512 input and out tokens. For each experiment run, we fine-tune T5-BASE model for 8 epoches on 4 NVIDIA Titan Xp GPUs. It took roughly an hour to finish the training [2]. For the augumentation setting, we map the ingredient entities that are linked with the PARTICIPANT-OF and RESULT-OF relations from the R2VQ dataset (Tu et al., 2022a) to the BEGINNING and ENDING subevents. R2VQ didn't assume the event participant/result is necessary so the mapping can only recover partial annotations under our subevent definition. In practce, we first use the entities and mapped relations from the 900 recipes as the "silver" data to pretrain the T5 model, and then fine-tune/train the pretrained T5 with the 80 recipes from the CUTL dataset.

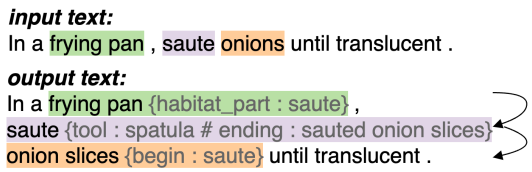


Figure 4: Example of T5 model input and output for DP generation task. Each cooking role is wrapped by a pair of curly brackets ({...}). Cooking roles at the same position are separated by hashtags (#).

[2] training script adopted from https://huggingface.co/valhalla/t5-base-qa-qg-hl

	SINGLE-T5		AGG.-T5		AGG.+AUG.-T5		
Label	E.	H.	E.	H.	E.	H.	Count
TOOL	71.42	60.28	72.63	61.59	**75.09**	**64.50**	73
HABITAT	73.62	64.28	73.87	64.69	**80.93**	**68.69**	129
INGREDIENT (beginning)	81.33	31.92	82.18	**32.63**	**88.22**	32.13	405
INGREDIENT (ending)	**60.03**	44.68	59.13	45.59	59.53	**46.19**	221
ALL	73.57	42.56	73.89	43.64	**78.27**	**44.43**	828

Table 4: DP generation results from T5 under different settings. F1 score is reported for both explicit (E.) and hidden (H.) entities. SINGLE-T5 uses one sentence as single model input; AGG.-T5 aggregates every three continuous sentences as single input and only evaluates on the third sentence from each input; AGG.+AUG.-T5 uses the rest of 900 R2VQ recipes as augmented data for training.

Results Table 4 shows the model results on the DP generation task. Compared to SINGLE-T5, AGG.-T5 gains a better performance (73.9/43.6 F1), suggesting the importance of contextual information from previous sentences in procedural text. AGG.+AUG.-T5 performs the best overall (78.3/44.3 F1 F1) due to the additional data from the R2VQ annotation. For individual labels, identifying hidden entities are still challenging to the baseline model, especially for the INGREDIENT. AGG.+AUG.-T5 performs worse on hidden beginning ingredients than explicit ones by a large margin (53.1 F1). Compared to the hidden TOOL and HABITAT, hidden INGREDIENT has more variants from the context of DP events (e.g., *onions, onion slices, sauted onions, etc*). In addition, each DP event can have multiple beginning or ending ingredients (e.g., *mix water and flour*), which also increases the difficulty of the task.

Overall, the above experiment shows that the inference and reasoning over all the hidden text remains a very challenging task to current large language models. For our data specifically, the higher ratio of the hidden entities and the entity variance from the dense paraphrasing makes it a challenging task to the model. Attempts to improve the results may include multi-task learning to generate entity types and values separately, and iterative training to utilize the data more efficiently. We further explore the DP method and data by showing the case study on out-of-domain DP text generation and GPT-3 paraphrasing.

6 Case Study

6.1 Out-of-Domain DP Modeling

We explore the scenarios that the DP strategy and datasets can be adapted to raw data in the same style (e.g., procedural text) but out of the domain under a transfer learning setting. We show a case study of the results by applying the DP generation model that is fine-tuned on our training set to WikiHow articles. For this experiment, we use the articles from the WikiHow corpus curated by (Zhang et al., 2020) that is originally for the goal-step inference tasks. Specifically, we pick four articles from different domains and apply the fine-tuned DP generation model from §5 on these articles.

The generation results on the four unseen WikiHow articles are shown in Figure 5. The first article is an in-domain recipe (shortened in the Figure), so the model performs very well on identifying the relations and hidden entities. The ingredient entities also show the subevent state change through sentences (e.g., *fried arepas* to *baked arepas*). The results on the second article shows the effectiveness of the DP strategy being applied to out-of-domain data. Our defined DP event structure can be naturally transferred to text with clear steps and intermediate goals (e.g., *Mix a mild cleaner with warm water*). The model could mispredict the actual values of the hidden entities due to the limitations from the domain-specific vocabulary inventory. E.g., the predicted hidden entity is *oil* from the sentence *"Scrub down the brush ..."*. The subevent entity paraphrasing, however, is still effective. For example, the hidden result ingredient of the event *mix* is *cleaner water*. Similarly in the last sentence, we are able to generate *rinsed brush* that carries the subevent state effectively.

Compared to the first two, we find the last two articles to be more challenging to the model. Although the text is short, the third article involves rather complex spatial actions (e.g., *snap off, peel downward, etc.*) that may confuse the model. The part-whole relations of entities (e.g., *banana vs. skin vs. stem*) can also lead to semantically ambiguous subevent paraphrases such as *snapped stem / banana, peeled skin / banana*. The last article is

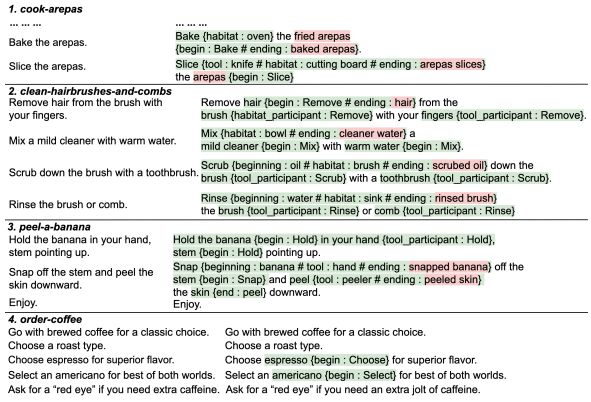


Figure 5: DP generation on example WikiHow articles. The left shows the article title and steps; the right shows the model output. Green spans mark the entities and relations; red spans mark the paraphrased entities.

different from the others in the sense that it has a less clear step-goal structure and the events are not actions interacting with physical objects. These differences make texts of this type less suitable to the proposed method. In general, the case study shows the usefulness of the DP strategy and the dataset we created under a transfer learning scenario to procedural texts with the similar format. Future work includes expanding the DP evaluation on general procedural texts so that a quantitative study can be conducted.

6.2 Subgraph for GPT-3 Paraphrasing

We briefly characterize the common differences in the output paraphrases between PREFIXP-GPT and SUBGRAPH-GPT, and present several examples in Table 5. In comparison, PREFIXP-GPT tends to generate paraphrase as noun-noun components, while PREFIXP-GPT tends to generate an adjectival verb as the modifier to the entity. Score-wise, both output format are acceptable, but minor syntactic errors (mushroom[s] slices) and semantic ambiguity (meat [mixture]) are spotted from the NN components. PREFIXP-GPT also has a strong tendency to rewrite or hallucinate new text. This may be due to the fact that prefix-paraphrase has no special symbol or text structure to regulate the generation. Compared to SUBGRAPH-GPT which preserves the event type and structure in the model input, PREFIXP-GPT uses the ‘flattened’ text that may put extra weight on the local event that is closest to the entity to be paraphrased. Consider the gold *salad* from the table. Based on the event text ***season*** *with salt and pepper*, the PREFIXP-GPT generates the realization such as *seasoned pepper*

	GOLD	PREFIXP-GPT	SUBGRAPH-GPT
NN Comp.	mushrooms	mushrooms slices (4)	sliced mushrooms (5)
	cooked bacon	bacon bites (5)	chopped bacon (5)
	meat	meat mixture (4)	sauteed meat (5)
Hallucination	sewian	fried noodles (2)	fried sewian (5)
	meat	ground beef (4)	minced meat (4)
	soup	stew (3)	vegetable broth (4)
Locality	fish	marinated chunks (4)	marinated fish (5)
	salad	vinaigrette (2)	salad (5)

Table 5: Common difference between the output paraphrase from PREFIXP-GPT and SUBGRAPH-GPT, and their intrinsic scores.

and salt and combined lemon juice and ..., whic features the latest event and entities. A subgraph allows one to trace all the visited events and thus increase the model reasoning capability.

6.3 Does GPT-4 solve everything?

We further explore the performance of different GPT models on the the task of paraphrase generation. In table 6, we select five examples which SUBGRAPH-GPT performs poorly on (with an intrinsic score of 3 or lower), and anecdotally compare the results with the paraphrases generated by the latest GPT-4 (SUBGRAPH-GPT4). In the first example, both GPT-3 and GPT-4 generate the same incorrect paraphrase. This might be due to the model has been trained biased towards a strong connection between the action *squeeze* and the *juice*. GPT-4 also doesn’t generate ideal paraphrase on the second and the third example due to the noise from the context, e.g., *mussels* or *peas* is a more salient feature than the *water*. However, it is able to provide more details in the paraphrases (*cooked peas* v.s. *peas*). On the last two examples, GPT-4 performs better than GPT-3 by generating more accurate entities (e.g., *fillet* v.s. *fish*) and richer states (*seasoned. marinated*).

7 Conclusion

In this paper we define *Dense Paraphrasing (DP)*, the task of enriching a text fragment (lexeme, phrase, or sentence) such that contextual ambiguities are eliminated, contextual anchors or variables are supplied, and any implied arguments are made textually explicit. We outlined our DP procedure that can be applied to enrich the textual dataset, and provided insights on the transformer-based models as baselines for the DP text generation task. We presented the case study for generating DP under the out-of-domain setting, and the analysis on paraphrasing from event graphs, which show the feasibility of modeling DP and the challenges it

CONTEXT	SUBGRAPH-GPT	SUBGRAPH-GPT4	GOLD
Prepared horseradish, squeeze it dry through a kitchen towel to get [SQUEEZE.RES]. Combine the horseradish, sour cream ...	horseradish juice	horseradish juice	horseradish
Wash mussels and de-beard, bring a pot of water to a boil to get [BOIL.RES] and cook the mussels ...	cooked mussels	de-bearded and cooked mussels	boiled water
In a large pot, bring peas and water to boil over high heat and reduce to simmer until tender to get [SIMMER.RES].	peas	cooked peas	soup
Add the salt and pepper ... place the fillets under the broiler, about 2 inches from the heat source and cook for 2 minutes to get [COOK.RES]	fish	seasoned cooked fillets	salmon fillets
Cut chicken thighs in half ...Combine the paste with the chicken and mix well; refrigerate several hours or overnight to get [REFRIGERATE.RES]	chicken patties	marinated chicken	chicken

Table 6: Output paraphrase comparison between different GPT models on five examples. Paraphrases are generated for entities represented as [VERB.RES].

poses to current large language models.

We believe that DP has the potential to help in a broad range of NLP applications. In particular, applications and tasks involving abstractive inferencing can benefit from the dynamic tracking and decontextualized redescriptions of entities appearing in a coreference chain. The notion of following an entity as it changes through a developing narrative or text can be computationally encoded using the technique described here, giving rise to a history or biographical model of an entity. We hope to extend the DP procedure to include creating vector representations of DP that can be fit into a broader range of computational models. We also intend to include reference to the "vertical typing" of an expression (type inheritance) from online resources with definitional texts, such as Wikipedia or WordNet (e.g., onion $\in$ vegetable, poodles $\in$ dogs). This would further enhance the utility of the resulting DP'ed data for logical inference tasks.

References

Regina Barzilay and Michael Elhadad. 1999. Using lexical chains for text summarization. *Advances in automatic text summarization*, pages 111–121.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One spring to rule them both: Symmetric amr semantic parsing and generation without a complex pipeline. In *AAAI Conference on Artificial Intelligence*.

Rahul Bhagat and Eduard Hovy. 2013. What is a paraphrase? *Computational Linguistics*, 39(3):463–472.

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*.

Michel Boyer and Guy Lapalme. 1985. Generating paraphrases from meaning-text semantic networks. *Computational Intelligence*, 1(1):103–117.

Susan Windisch Brown, Julia Bonn, Ghazaleh Kazeminejad, Annie Zaenen, James Pustejovsky, and Martha Palmer. 2022. Semantic representations for nlp using verbnet and the generative lexicon. *Frontiers in artificial intelligence*, 5.

Susan Windisch Brown, James Pustejovsky, Annie Zaenen, and Martha Palmer. 2018. Integrating generative lexicon event structures into verbnet. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Haixia Chai, Nafise Sadat Moosavi, Iryna Gurevych, and Michael Strube. 2022. Evaluating coreference resolvers on community-based question answering: From rule-based to state of the art. In *CRAC*.

Eunsol Choi, Jennimaria Palomaki, Matthew Lamm, Tom Kwiatkowski, Dipanjan Das, and Michael Collins. 2021. Decontextualization: Making sentences stand-alone. *Transactions of the Association for Computational Linguistics*, 9:447–461.

Peter W Culicover. 1968. Paraphrase generation and information retrieval from stored text. *Mech. Transl. Comput. Linguistics*, 11(3-4):78–88.

Bhavana Dalvi, Niket Tandon, Antoine Bosselut, Wentau Yih, and Peter Clark. 2019. Everything happens for a reason: Discovering the purpose of actions in

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 4496–4505, Hong Kong, China. Association for Computational Linguistics.

Kaustubh D. Dhole and Christopher D. Manning. 2021. Syn-qg: Syntactic and shallow semantic rules for question generation.

Jacob Eisenstein, Daniel Andor, Bernd Bohnet, Michael Collins, and David Mimno. 2022. Honest students from untrusted teachers: Learning an interpretable question-answering pipeline from a pretrained language model. *ArXiv*, abs/2210.02498.

Yanai Elazar, Victoria Basmov, Yoav Goldberg, and Reut Tsarfaty. 2021. Text-based np enrichment. *arXiv e-prints*, pages arXiv–2109.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1608–1618.

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.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, N. Lao, Hongrae Lee, Da-Cheng Juan, and Kelvin Guu. 2022. Rarr: Researching and revising what language models say, using language models.

Neil M Goldman. 1977. Sentence paraphrasing from a conceptual base. *Sentence Paraphrasing from a Conceptual Base*, pages 481–507.

Zellig S Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Zellig S Harris. 1957. Co-occurrence and transformation in linguistic structure. *Language*, 33(3):283–340.

Henry Hiż. 1964. The role of paraphrase in grammar. In *Monograph Series on Language and Linguistics 17*.

Seohyun Im and James Pustejovsky. 2009. Annotating event implicatures for textual inference tasks. In *The 5th Conference on Generative Approaches to the Lexicon*.

Seohyun Im and James Pustejovsky. 2010. Annotating lexically entailed subevents for textual inference tasks. In *Twenty-third international flairs conference*.

Yiwei Jiang, Klim Zaporojets, Johannes Deleu, Thomas Demeester, and Chris Develder. 2020. Recipe instruction semantics corpus (RISeC): Resolving semantic structure and zero anaphora in recipes. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 821–826, Suzhou, China. Association for Computational Linguistics.

Sylvain Kahane. 1984. The meaning-text theory.

Uri Katz, Mor Geva, and Jonathan Berant. 2022. Inferring implicit relations in complex questions with language models. *ArXiv*, abs/2204.13778.

Ghazaleh Kazeminejad, Martha Palmer, Tao Li, and Vivek Srikumar. 2021. Automatic entity state annotation using the VerbNet semantic parser. In *Proceedings of the Joint 15th Linguistic Annotation Workshop (LAW) and 3rd Designing Meaning Representations (DMR) Workshop*, pages 123–132, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Kathleen McKeown. 1983. Paraphrasing questions using given and new information. *American Journal of Computational Linguistics*, 9(1):1–10.

Igor Mel'cuk. 1995. Phrasemes in language and phraseology in linguistics. *Idioms: Structural and psychological perspectives*, pages 167–232.

Igor Mel'Čuk. 2012. Phraseology in the language, in the dictionary, and in the computer. *Yearbook of phraseology*, 3(1):31–56.

Kazunori Muraki. 1982. On a semantic model for multilingual paraphrasing. In *Coling 1982: Proceedings of the Ninth International Conference on Computational Linguistics*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

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*, Toronto, Canada. Association for Computational Linguistics.

Lingfeng Shen, Lemao Liu, Haiyun Jiang, and Shuming Shi. 2022. On the evaluation metrics for paraphrase generation.

RM Smaby. 1971. Paraphrase grammars, volume 2 of formal linguistics series. dordrecht: D.

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, Eben Holderness, Marco Maru, Simone Conia, Kyeongmin Rim, Kelley Lynch, Richard Brutti, Roberto Navigli, and James Pustejovsky. 2022a. SemEval-2022 Task 9: R2VQ – Competence-based Multimodal Question Answering. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 1244–1255, Seattle, United States. Association for Computational Linguistics.

Jingxuan Tu, Kyeongmin Rim, and James Pustejovsky. 2022b. Competence-based question generation. In *International Conference on Computational Linguistics*.

Oriol Vinyals, Ł ukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Jack Weston, Raphael Lenain, Udeepa Meepegama, and Emil Fristed. 2021. Generative pretraining for paraphrase evaluation. *arXiv preprint arXiv:2107.08251*.

Zeqiu Wu, Yi Luan, Hannah Rashkin, David Reitter, and Gaurav Singh Tomar. 2021. Conqrr: Conversational query rewriting for retrieval with reinforcement learning. *arXiv preprint arXiv:2112.08558*.

Yoko Yamakata, Shinsuke Mori, and John Carroll. 2020. English recipe flow graph corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 5187–5194, Marseille, France. European Language Resources Association.

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.

Weihe Zhai, Mingqiang Feng, Arkaitz Zubiaga, and Bingquan Liu. 2022. Hit&qmul at semeval-2022 task 9: Label-enclosed generative question answering (leg-qa). In *SEMEVAL*.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020. Reasoning about goals, steps, and temporal ordering with WikiHow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4630–4639, Online. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *ArXiv*, abs/1904.09675.

Jianing Zhou and Suma Bhat. 2021. Paraphrase generation: A survey of the state of the art. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5075–5086, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.