Competence-based Question Generation

Jingxuan Tu, Kyeongmin Rim, James Pustejovsky
Department of Computer Science
Brandeis University
{jxtu,krim,jamesp}@brandeis.edu

Abstract

Models of natural language understanding often rely on question answering and logical inference benchmark challenges to evaluate the performance of a system. While informative, such task-oriented evaluations do not assess the broader semantic abilities that humans have as part of their linguistic *competence* when speaking and interpreting language. We define competence-based (CB) question generation, and focus on queries over lexical semantic knowledge involving implicit argument and subevent structure of verbs. We present a method to generate such questions and a dataset of English cooking recipes we use for implementing the generation method. Our primary experiment shows that even large pretrained language models perform poorly on CB questions until they are provided with additional contextualized semantic information. The data and the source code is available at: https://github.com/brandeis-llc/CompQG.

1 Introduction

Developing an Artificial Intelligence (AI) system with inferencing and reasoning capabilities that enables itself to communicate with intellectual human users has been a holy grail of the research community. For example, there has been considerable effort put on large challenges in question answering formats to measure such inference and communication capabilities of neural end-to-end systems (Ribeiro et al., 2020; Prabhumoye et al., 2020; Rogers et al., 2021; Minaee et al., 2021). However, we argue that a broader effort needs to be put on measurements more focused on linguistic *competencies*, and not just on extractive comprehension skills or "challenge checklisting". Out argument is in line with some moves in this direction (Johnson et al., 2017), but there is still no generally accepted distinction in current natural language processing (NLP) between challenge-based tasks and competence-based (CB) performance (Bentivogli et al., 2017). Analogous to human cognitive competencies, there is both a methodological and modeling advantage to focusing a system's performance on competence-based learning rather than a narrowly defined task or challenge.

The term *competence-based (CB)* has been applied to a number of different concepts, from linguistics (Chomsky, 1965) to both the science of learning and educational communities (Bechtel et al., 1999; Voorhees, 2001; Chyung et al., 2006; Hsiao et al., 2020; Platanios et al., 2019). The common core to both is a concept capturing a coherent set of abilities that an individual has in a specific domain (Doignon and Falmagne, 1985; Heller et al., 2013). More recently, we proposed a new multimodal question answering task R2VQ that focuses on querying competence-based knowledge from cooking recipe texts and videos (Tu et al., 2022). For the text part, we created a dataset with rich annotations of hidden arguments and coreference that focuses on lexical competence as deployed in both single and multiple sentence composition (Pustejovsky, 1995; Marconi, 1997; Geeraerts, 2009).

As a natural complement and extend to the R2VQ task (Tu et al., 2022), here we define the task *competence-based question generation* that aims to generate CB questions that require lexical competence. The lexical competence involves understanding the hidden arguments (due to syntactic ellipsis or semantic defaulting or shadowing) given the event context; and understanding the dynamics of events and objects change in the text. This competence requires non-extractive Question Answering (QA) capabilities of some sort. Thus we define a *competence-based* question as one that queries any aspects of lexical competence mentioned above.

In the rest of the paper, we first review recent related work (Sec. 2), and give more detailed definitions of the CB question generation task (Sec. 3). We then present a dataset of CB knowledge and how we collected and annotated it (Sec. 4). Section 5 provides details of generation of CB questions on the dataset we created. The generated questions are evaluated and the results are discussed in Section 6 and 7. Then we conclude our work in the final Section (8).

2 Related Work

Question Generation (QG) is an essential NLP task that can be used for supplementing educational materials (Heilman and Smith, 2010; Zhao et al., 2022), data augmentation for QA systems (Lyu et al., 2021; Le Berre et al., 2022) and understanding semantic relations within the text (Pyatkin et al., 2020; Klein et al., 2020). With the advent of large pretrained language models and large QA datasets, recent QG approaches are primarily based on transformer-based neural architectures. Dong et al. (2019) finetuned a unified language model for both QG and language understanding tasks. Yuan et al. (2021) enhanced the language model for QG with additional embeddings of linguistic features. To make generated questions more diverse, Murakhovs'ka et al. (2021) took advantage of the fully text-to-text generation model to to generate questions with mixed types of answers. Fei et al. (2022) applied graph networks to generate complex questions requiring broader contexts to answer.

Comparing with the neural methods, traditional rule-based QG systems (Levy and Andrew, 2006; Heilman and Smith, 2009) did not receive much attention due to the lack of diversity of the generated questions. To solve this, He et al. (2015) proposed QA-SRL that use predicate-argument structure to generate more semantic-informed QA pairs. More recent work also has shown the success in incorporating existing explicit semantic annotations and linguistic resources into the rule- or template-based QG systems (Dhole and Manning, 2021; Pyatkin et al., 2021). In this work, we adopt a template-based approach to generate competence-based questions that solicit implicit information and the event dynamics across a broad context.

More broadly related to our general purpose to probe the above-defined competence-based knowledge with QG, Tu et al. (2022) proposed a new QA task that queries competence-based knowledge structures; Rashkin et al. (2018) proposed a new commonsense inference task that involves the reasoning of intents and reactions to the events; Xu et al. (2021) used context information for commonsense QA. There also exist semantic tasks that explore the implicit or underspecified components of a linguistic expression (Roth et al., 2021; Karidi et al., 2021; Ye et al., 2022).

3 Task Definition

We define our task as generating a *competence-based* question set. Unlike traditional QG tasks whose answers are mostly extractive and contexts are largely dependent on surface lexical locality conditions, the goal of our task is to generate questions that represent commonsense semantic inferences from a large context in an abstractive way.

Table 1 shows example CB questions generated from the full text of a cooking recipe. Consider the IMPLICIT question. Our task is to generate such a question-answer pair that reflects the implied state of the object (*peeled* apples) and the missing argument from the context (*knife* to cut). Also consider the LOC. CHANGE and OBJ. LIFESPAN questions: the goal of our task is to solicit information that reflects the dynamics of events and actions through these questions.

4 Data Description

Due to the lack of existing datasets suitable to our task, we curate a collection of English cooking recipes as the data for our task. Recipes along with other procedural text like instructions tend to be task-oriented and stepwise. The content is composed of step sentences that describe small goals to accomplish the final task. Comparing to text of news or narratives, we think recipe texts are a good fit for our task as it involves the understanding of how to reach the goal locally for each step, as well as how each step contributes to the final task globally. Further, the stepwise progression inherent in the goal-oriented narrative contributes both an interpretative dynamics as well as contextualized elision of arguments. Our proposed dataset has also been used as part of the data resource for the SemEval-2022 Task 9: R2VQ (Tu et al., 2022). Here we describe the annotation process and the annotation alignment step for our QG task.

Data Preprocessing We build our QG dataset with a collection of public domain recipes.[1] We first run the Stanza pipeline (Qi et al., 2020) on the raw text of each recipe to get tokenization and other linguistic features including lemmatization, part-of-speech tagging and dependencies. We also run the state-of-the-art Semantic Role Labeling (SRL) parser from (Conia and Navigli, 2020) to label each recipe sentence with its semantic roles. Subsequently, we ask 2 students annotators to validate and correct both frames and argument labels.

[1] `https://recipes.fandom.com/`, `http://foodista.com/`

Recipe Title: **Appelkoek** *Passage:* Peel and cut apples into eighths (wedges). Sift together flour, baking powder and salt with 4 tablespoons of the sugar. Cut in butter. Combine egg and milk and add to flour mixture. Turn batter into greased 8 inch square cake pan. Press apple wedges partly into batter. Combine remaining 2 tbsp sugar and cinnamon. Sprinkle over apple. Bake at 425 degF for 25 to 30 minutes.

IMPLICIT	How do you cut peeled apples into wedges? - by using a knife
ELISION	What should be sprinkled over apple wedges? - cinnamon sugar
TEMPORAL	For how long should you bake appelkoek? - 20 to 35 minutes
LOC. CHANGE	Where was the batter when you press apple wedges? - *still* in the pan
OBJ. LIFESPAN	What's in the appelkoek? - apples, batter and cinnamon sugar
HAB. STATE	What's already in the bowl when you add egg and milk to it? - butter and dry ingredients

Table 1: Example competence-based questions. Color-coded text spans represent how information has been collected and generated in the questions.

Table 2 shows the basic statistics of the the preprocessed recipes. We have filtered out non-English and non-food recipes, as well as short recipes with only 4 or less sentences. This results in 900 preprocessed recipes in total with an average of 8 sentences per recipe and 13 tokens per recipe sentence. We randomly select 800 recipes for experimenting our QG method, and the other 100 is held out for the evaluation of the generated QA pairs.

	Train	Val
# of recipes	800	100
Avg. # of sentences per recipe	8	7.9
Max./Min. # of sentences	26/4	16/4
Avg. sentence length per recipe	12.5	13.4
Max./Min. sentence length	32/6	25/6

Table 2: Statistics of the train and validation subsets of the QG dataset.

Cooking Role Annotation We prepare the preprocessed recipes for QG by adding Cooking Role Labeling/Linking (CRL), a span-level annotation layer for identifying cooking events from recipe text. We define the event ontology as a set of cooking-related entities and relations. The entity types include the EVENT-HEAD, INGREDIENT, TOOL and HABITAT. The relations include PARTICIPANT-OF and RESULT-OF. Each event has only one predicative verb (EVENT-HEAD), and all the relations within the event are linked from corresponding entities to the predicate. A sample event is provided in the Appendix, Figure 2. More importantly, CRL annotates events that involve implicitly expressed arguments by identifying their hidden entities. For example, consider the sentence *Sprinkle over apple.* from Table 1. In this event, the hidden TOOL *hand*, the hidden HABITAT *cake pan* and hidden INGREDIENTS *cinnamon* and *sugar* are the most plausible participants of the event head *sprinkle*, but are not explicitly stated. The identified hidden entities should be either inferred elsewhere explicitly on the document level, or inferred based on the commonsense.

We start the CRL annotation by labeling explicit entities semi-automatically using a separately trained NER model. Relations and hidden entities are annotated manually. Then we annotate the coreference of tools, habitats and ingredients through the full recipe at the document level. Specifically, we hired 12 trained student annotators for the CRL annotation and validation work.[2] Table 3 shows the average number of annotated entities per recipe. The average number of event verbs (14) are much great than the average recipe length (8) from Table 2, indicating that many recipe sentences tend to involve more than one event. Ingredient participants are the most prevalent entity type under both explicit and hidden settings. Recipes also have more hidden ingredient results, tools and habitats instead of explicit ones, showing the importance of hidden arguments for understanding cooking recipes or instructional text in general.

	Train		Val	
	Exp.	Hidden	Exp.	Hidden
EVENT-HEAD	14.0	N/A	13.6	N/A
INGREDIENT (participant)	13.0	6.9	14.0	10.8
INGREDIENT (result)	0.2	1.5	0.2	1.4
TOOL	0.6	2.1	0.7	2.2
HABITAT	2.8	4.8	2.5	6.2

Table 3: Average number of annotated explicit/hidden entities per recipe from the QG dataset. EVENT-HEAD can only be explicit.

Aligning CRL and SRL We further process the CRL annotation by aligning it with SRL. Specifically, for any given sentence, we align its CRL and SRL annotation that share the same event predicate (marked as `PREDICATE` in SRL and `event-head` in CRL). Semantic roles are merged with corresponding cooking entities if their text spans are overlapped. For example, in the sentence *Transfer peas to the saucepan*, the role *to the saucepan*$_{\texttt{[DESTINATION]}}$ will be merged with the entity *saucepan*$_{\texttt{[HABITAT]}}$, and the text span *peas* is both the entity `INGREDIENT` and the role

[2]Full annotation process can be found in Appendix A.2.

THEME. The un-overlapped roles such as TIME and ATTRIBUTE will be categorized as modifiers to this cooking event. The CRL-SRL aligned events will later be used to generate questions that are able to hold richer context.

5 Experiment Method

Now we describe a template-based method to generate competence-based questions. We explore the data with CRL annotation in two phases: first we adopt the concept of question family in §5.1 by proposing question templates and heuristics to generate QA pairs that involve hidden arguments for each family. For example, given a sentence *Sauté the onions until browned.*, we generate the question *Where should you sauté the onions?*. The correct answer *in the pan* is not in the current sentence, thus needing to be inferred from its context. In §5.2 we propose the concept *dense paraphrasing* that can record previous states of the entities as being processed though events, e.g. *sauté onions* will be rewritten as *sauté **chopped** onions* in generated questions based on the given context. Then we apply it to enhance question families by generating new questions and answers that cover a bigger context and dynamics of events.

5.1 Generating Competence-based Questions

Motivated by the concept of Question Family (QF) that is first outlined and adopted by Johnson et al. (2017) to create the visual question answering dataset, we first introduce our *question families*, which contain a set of text templates and semantic reasoning heuristics that can generate the full spectrum of competence-based questions. We define the question families in Table 4. Each question family is designed to test a specific competence with generated questions using the hand-crafted text templates.

We generate QA pairs by populating templates with slots in a cloze test style. Candidate slots (colored spans in Table 4) are acquired from CRL-SRL aligned events we created earlier. We also set the constraints to only keep the events with at least one hidden cooking entity for template population, so that the generated questions are competence-based, rather than purely "memorizing" original text spans. Each aligned event is composed of cooking entities with semantic roles, relation links between entities and event verb, as well as SRL modifiers to the event. All the required slots and the candidate answer for a template are self-contained within an aligned event.[3]

After a text template is populated, it is further processed to improve the readability of the generated question. We change word inflections and insert articles and agreements. For the templates with [habitat_phrase] and [tool_phrase] slots, we fill those with corresponding LOCATION or INSTRUMENT spans from SRL. If a slot is filled with a hidden entity, we run a BERT-based model (Devlin et al., 2019) to get the most likely preposition given the sentence as context through the masked language modeling task. Modifiers are populated in the same order as they were in the original sentence.

To increase the variety of questions, we allow adjunct slots in the text templates. As shown in Table 4, adjunct slots include tool/habitat phrases and modifiers. For example, one ELISION question can be as short as *What should be sautéed?* or *. . . sautéed in the saucepan with the spatula until browned?* with all the adjunct slots. We argue it is helpful to generate questions more challenging to the systems. Adding more adjunct slots completes the context for the question, but also introduces unseen context if the slots contain hidden entities.

5.2 Generating Enhanced CB Questions

We further improve our QFs by 1) explicating ambiguity of entity coreference, 2) generating questions from larger context. Consider the sentence *Peel and cut apples into wedges.* from Table 1. The entity apple will be transformed to peeled apples after the peel event, and that is what is participating in the following cut event, rather than the same referred entity. We enhance the competence-based QG to capture this layer of underlying semantics. In addition, we also include document-level QFs into our system. For the question *What's in the appelkoek?*, an answer *apple wedges* can be extracted only from the last sentence where the event that contains *applkoek* is annotated. While the answer is not wrong, it's not complete enough to include all the necessary ingredients that can be summarized from the whole recipe (Table 1).

Dense Paraphrasing To reflect the aforementioned difference between *apples vs. peeled apples* in QG, we enrich the source narrative with a dynamic *Dense Paraphrase* of the surface text, which both decontextualizes the expression (Choi et al., 2021; Elazar et al., 2021), and enriches the textual description of both events and participants to reflect

[3]One exception is for *Cardinality*, which requires the count of the co-referred entities on the document level. The entites from *Cardinality* also do not need to be hidden.

QFs	Text Template	Question Sentence
Cardinality	How many toolObj|habitatObj are used? How many times is the toolObj|habitatObj used? How many action does it take to process IngreObj?	How many bowls are used? How many times is the microwave used? How many action does it take to process the pasta?
Elision	What should be verb [habitat_phrase] [tool_phrase] [modifiers]?	What should be baked in the oven at 425 degree?
Implicit	What do you use to verb obj [habitat_phrase] [modifiers]? Where do you verb obj [tool_phrase] [modifiers]?	What do you use to sauté the onions [in the pan]? Where do you arrange the slices into rounds?
Obj. Lifespan	What is in obj? How did you get ingreObj?	What is in the appelkoek? How did you get the appelkoek?
Semantic Role	*For how long* do you verb obj [tool_phrase] [habitat_phrase] [modifiers]? *To what extent* do you verb obj [tool_phrase] [habitat_phrase] [modifiers]?	For how long do you marinate the shrimp? How do you add the water to the bowl?

Table 4: Text templates and generated questions from each question family. ***Cardinality***: integer comparison and counting; ***Elision***: hidden arguments that can be understood from context; ***Implicit***: implicit tools/habitats that require world knowledge; ***Obj. Lifespan***: different object states; ***Semantic Role***: semantic roles from a cooking event. In the templates, bar symbol (...|...) indicates either slot is acceptable; squared brackets ([...]) indicates adjunct slots.

Sense	Verbs	Category
CONVERT	melt, cream, evaporate	Transformation
SPILL_POUR	pour, ladle, drip	Loc. Change
AMELIORATE	enhance, improve, round out	N/A

Table 5: Example event verbs and their senses along with the assigned end state type.

the changes in the object due to the events. This includes descriptions involving subevent decomposition (Pustejovsky, 1995; Im and Pustejovsky, 2010), which tags the event as having three parts: **begin (Be)**, **inside (Ie)**, and **end (Ee)**.

In practice, we only track the begin and end state of an event. We define two types of event end state: TRANSFORMATION and LOCATION-CHANGE. For example, the end state of a *chop onions* event is *chopped onions*, and the end state type is TRANSFORMATION, denoting a change of the object in shape, size or color, etc. To get the end state type of each event, we collected 208 unique verb senses (Di Fabio et al., 2019) that are assigned by the SRL parser to our data, and hand-split those into three categories: transformation, location change or neither. Then we assign the end state type based on the category of the event verb sense. Table 5 shows an example from each category.

We incorporate dense paraphrasing into the QG by replacing the INGREDIENT entities from each aligned event with its dense-paraphrased version before the template population. Specifically, for the event verb from transformation, the paraphrased entities will have the format of `pastParticipleVerb + object`, e.g. *minced garlic*, *heated water*. To retain the naturalness and readability of the questions, we keep the last one or two event end states of an ingredient. For example, *sautéed chopped peeled onions* will be truncated to *sautéed onions* or *sautéed chopped onions*.

Document-level QG Unlike previous QFs where each template is populated by only one event, we first construct all the CRL-SRL aligned events from each recipe into data views that can provide a global understanding of the whole recipe. Each data view is used to generate questions or answers in document level. Subfigure 1(a) shows a directed graph that connects events by ingredients. `result-of` connects the ingredient participant and result within an event; `coreference-of` connects ingredients from two events where the result of the first one is *directly* used as the participant of the second one. This data view is used to enhance the OBJ. LIFESPAN QF by expanding the answers to include all major ingredients from the whole text, rather than just the ingredients mentioned in current sentence. We describe the steps taken in Algorithm 1 to generate such answer. Given an intermediate ingredient that is asked about, we generate a subgraph of events thorough depth-first-search traversal over the graph from the vertex of the given ingredient. Then we collect the ingredient results from the subgraph as the answer.

We also propose new document-level QF templates that are populated through traversing the other two data views. Table 1 shows example questions from the newly proposed question families, namely LOC. CHANGE and HABITAT STATE. We succinctly describe the steps to generate QA pairs from the new QFs, and provide the corresponding algorithms in the appendix.

Subfigure (b) shows a grid that tracks the change of location of ingredients through events. Each row

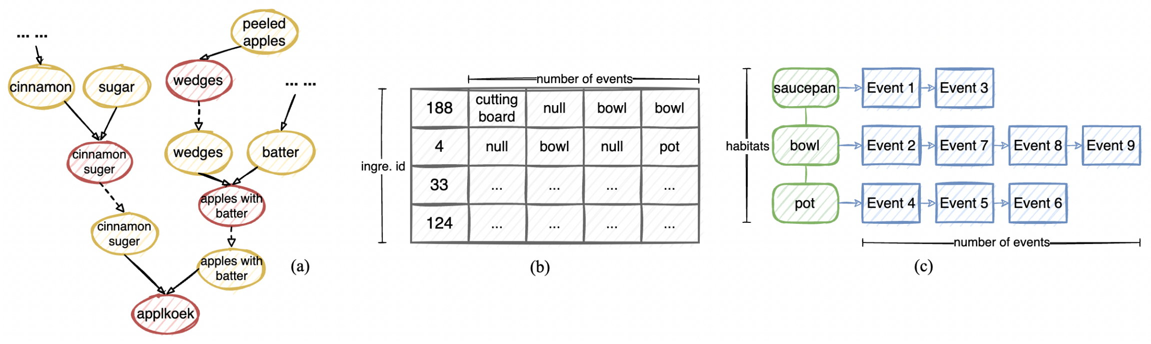

Figure 1: Data views of cooking events from a recipe. (a) A directed graph that connects events by ingredients. (● = ingredient participants, ● = ingredient results, → = result-of, $- - \succ$ = coreference-of); (b) A grid shows the habitat of each ingredient in every event; (c) An index links events by the location where the they took place.

Algorithm 1 Object Lifespan QF Heuristics

```
Require: Graph $G(V, E)$ and ingredient $ingre_i$
  access the ingredient result vertex $v_i$ that stores $ingre_i$
  generate subgraph $g \leftarrow$ predecessors in depth-first-search from $v_i$
  ingredient candidates $C \leftarrow \{\}$
  for $v_k$ in $g$ do
    if $v_k$ stores ingredient result $ingre_k$ and the at least one of parent vertices of $v_k$ stores ingredient participant that is NOT the outcome from other events then
      $add(ingre_k, C)$;
    end if
  end for
  $answer \leftarrow \{C\}$
  $question \leftarrow template(ingre_i)$
```

	EXPLICIT		HIDDEN		ENHANCED	
	Train	Val	Train	Val	Train	Val
SEMANTIC ROLE	5,513	700	8,169	1,064	901	135
ELISION	4,829	622	3,852	598	2,251	340
IMPLICIT	3,618	527	4,287	650	1,385	261
OBJ. LIFESPAN	781	69	1,098	124	2,392	312
CARDINALITY	-	-	4131	529	-	-
LOC. CHANGE	-	-	-	-	6,101	1,020
HAB. STATE	-	-	-	-	4,094	534
ALL	14,741	1,918	21,537	2,965	17,124	2,604

Table 6: Number of generated questions from each subset per QF.

shows the habitat of an ingredient (represented by ingredient id) in every event. *null* means the ingredient is not involved in the given event, thus no habitat needs to be populated. To generate a LOC. CHANGE question with this data view, the habitat from the event grid is used. Then we compare the habitat from the preceding event with current habitat to decide which location change cue to use in the answers. The answer format can be either *STILL habitat_phrase*, *in ANOTHER habitat_phrase* or simply the habitat itself based on the comparison.

Subfigure (c) links events by the location where they took place. Each list contains all the events that have the same habitat given by temporal ordering. To generate HAB. STATE questions, given a habitat and event at a certain time point from the event index view, we look up the grid view to track what ingredients are added or removed from the given habitat dynamically through all the events that happen before the current one.

6 Evaluation and Discussion

In this section we describe the steps to evaluate our system. First we adopt crowd-sourced evaluation to measure the quality of generated questions intrinsically. Then we train a text generation model to perform QA tasks over the subsets of questions to measure the competence of questions and the challenges they post to existing systems. We then compare other existing QG systems with our own.

To prepare the data for the evaluation, we apply our method on both the train and validation set of the QG dataset. The generated questions are categorized into three subset as shown in Table 6. EXPLICIT set contains generated questions of the QFs from Table 4, but the templates are populated with the events where the entities are all *explicitly* stated in the text. This set can be viewed as a non-competence extractive QA dataset. The HIDDEN set also contains the QFs we defined in Table 4 but with the constraint that a valid event should contain at least one hiddden entity. The ENHANCED set includes the dense paraphrased QFs as well as new document-level QFs from §5.2.

Intrinsic Evaluation Similar to other text generation task (Luong et al., 2015; Rush et al., 2015), human evaluation plays a critical role in question generation, as it can provide a more reliable evaluation and capture the nuance between generated text. We measure the generated QA pairs on a 5-point likert scale from *grammatical correctness*, *relevance* and *answer completeness*. The first two metrics are proposed in QG-STEC Task B (Rus et al., 2010) and widely used in previous work (Dhole and Manning, 2021; Pyatkin et al., 2021). They measure the grammaticality and the relevence of the questions to the context, respectively. In addition, we propose answer completeness for our QG task. It measures whether the answer is complete, e.g., some ingredients are missing from the answer or redundant. This metric is helpful to our task as we generate the answers that involve hidden objects which also traverse the whole passage.

We conduct the intrinsic evaluation on the HIDDEN and the ENHANCED question subsets of the QG validation set. We assign the generated questions to 4 trained annotators who had the experience with CRL annotation, and ask them to score each QA pair. Table 7 shows the result from intrinsic evaluation. For grammatical correctness, SEMANTIC ROLE and HAB. STATE have lower scores than the others. We suspect this is because CRL has more templates for different roles and HAB. STATE has a rather complicated question structure, while others have relative fixed templates. The overall change in relevance score tends to be consistent with grammatical correctness. However, OBJ. LIFESPAN from ENHANCED has a low relevance score compared to its grammaticality score. Through analysis, we found the system tends to generate less useful and less relevant QA pairs such as *What is in the chopped tomatoes? - tomatoes*. OBJ. LIFESPAN from ENHANCED also has the lowest score on AC. This is expected since the desired answers for this QF may include a whole list of entities that are scattered over the text. As a comparison, IMPLICIT has the highest score on AC because of the simplicity of answer format (either the hidden habitat or tool). Compared to HIDDEN, the score also drops on QFs in ENHANCED. By checking the examples, we found some dense-paraphrased objects can be "unnatural" to read such as *mixed apple* and *drained water*, which may lead to this drop. Overall, by comparing the scores to other QG systems (Dhole and Manning, 2021; Pyatkin et al., 2021) that also adopt the human evaluation, the result proves the validity of our method.[4]

Question Answering Performance We also evaluate the competence of generated questions by performing the QA task on the question subsets. We fine-tune the T5-BASE text generation model

	HIDDEN			ENHANCED		
	Gram.	Rel.	AC	Gram.	Rel.	AC
SEMANTIC ROLE	4.71	4.59	4.65	4.52	4.50	4.56
ELISION	4.93	4.77	4.57	4.92	4.79	4.53
IMPLICIT	4.92	4.87	4.83	4.84	4.76	4.78
OBJ. LIFESPAN	4.83	4.70	4.52	4.75	4.45	4.36
LOC. CHANGE	-	-	-	4.80	4.51	4.48
HAB. STATE	-	-	-	4.51	4.44	4.57
ALL	4.81	4.69	4.66	4.74	4.56	4.53

Table 7: Results from intrinsic evaluation on questions families with / without enhancement. CARDINALITY is not included as there is little variance between questions from this set, and the answer only contains numbers.

(Raffel et al., 2020) on each question set from the training set and evaluate on the validation set using exact match (EM) and token-level F1 score (F1) following Rajpurkar et al. (2018). The model training detail is provided in the Appendix A.3.

Table 8 shows the QA results from fine-tuned T5 on question subsets per question family. Overall, the model performs best on EXPLICIT set and performs worse on HIDDEN and ENHANCED. This observation confirms our assumption that CB questions indeed pose new challenges to existing systems. The model performs best on SEMANTIC ROLE and the score difference between question sets is much smaller comparing to the overall. This is because the answers for this QF are explicit semantic roles that can be retrieved in an extractive manner. For OBJ. LIFESPAN, the score difference between questions sets is much greater. Compared with EXPLICIT which only account for explicit ingredients, HIDDEN set requires a correct answer to include all hidden ingredients from the given event. ENHANCED set further accounts for dense-paraphrased answers from the document-level, thus resulting in the most challenging set for OBJ. LIFESPAN. Comparing the two multi-sentence question families in ENHANCED, the model performs much better on LOC. CHANGE. By examining the examples, we found that some LOC. CHANGE questions involve events that are next to each other, which requires local context for the model to pick up the necessary information to answer it. In contrast, HAB. STATE is the most difficult question family as it asks the model to track the relative position between habitat and ingredients globally to answer it correctly.

To understand whether CRL annotation can help answer CB questions, we fine-tune the T5 model on the ENHANCED set again, but we modify the input string to include the CRL by inserting hidden entities as a human-readable format to the in-

[4] Dhole and Manning (2021) and Pyatkin et al. (2021) also evaluated the result on grammatical correctness and relevance, resulting in 3.93/4.34 and 4.57/4.29 on the data presented in their work. Despite the difference in data and templates makes it difficult to compare the exact scores, we argue the relative score (e.g. all above 3.5) is still useful to look at due the evaluation is based on human judgement.

	EXPLICIT			HIDDEN			ENHANCED			- w/ CRL	
	EM	F1	Count	EM	F1	Count	EM	F1	Count	EM	F1
SEMANTIC ROLE	92.59	96.92	700	90.51	96.05	1064	84.44	89.68	135	60.00	72.46
ELISION	66.72	75.63	622	27.59	50.93	598	25.16	46.20	340	40.88	59.07
IMPLICIT	62.43	71.95	527	59.69	69.15	650	48.28	58.13	261	70.88	75.78
OBJ. LIFESPAN	40.58	58.38	69	23.39	54.43	124	10.58	48.82	312	14.10	60.58
CARDINALITY	-	-	-	71.08	71.08	529	-	-	-	-	-
LOC. CHANGE	-	-	-	-	-	-	56.47	74.05	1020	60.00	74.45
HAB. STATE	-	-	-	-	-	-	8.96	42.69	536	18.66	52.36
ALL	73.72	81.69	1918	64.79	74.86	2965	37.73	60.15	2604	44.59	66.26

Table 8: Results from fine-tuning T5 model for the question-answering task on three different subsets of generated questions. For ENHANCED subset, we collect two sets of results from the models trained with or without CRL annotation. CARDINALITY does not have counterparts from EXPLICIT and ENHANCED sets. LOC. CHANGE and HAB. STATE are only introduced in ENHANCED for multi-sentence QG.

put.[5] Table 8 also shows the results from the model trained with CRL, which outperforms the result from baseline T5. It shows the utility of CRL to carry competence-based knowledge. Interestingly, the scores for SEMANTIC ROLE questions drop significantly (24.44 on EM) with the new model. This is because we reconstruct the context text to include CRL, but that breaks the extractive nature of SEMANTIC ROLE questions. However, it can be avoided by training the system with different input.

Context: ... **Add**$_{[2]}$ to it potatoes and red pepper. **Stir**$_{[1]}$ well for 2 minutes.

SynQG	[1] For how long does someone stir well? [2] What vegetable is added to it?
RoleQ	[1] What stirs well? / Why does someone stir something something? [2] What does something add something to?
CompQG	[1] What do you use to stir the fried potatoes and red pepper? / For how long do you stir the fried potatoes and red pepper in the pot? [2] Where are the fried potatoes before you add them to the pot? / What is already in the pot when you stir the fried potatoes?

Table 9: Examples of questions generated by SynQG, RoleQ and CompQG (this work).

Comparison to Other QG Systems As a qualitative study of our system, we compare it to two QG systems: Syn-QG (Dhole and Manning, 2021) and RoleQG (Pyatkin et al., 2021). Due to the fact that all three systems are essentially different regarding their domain and model I/O, we focus on comparing how the systems differ in the final outcome and how our system fits into the larger scope of QG. Syn-QG leverages syntactic and semantic rules to generate template-based questions. RoleQG proposes a hybrid method of templates and seq2seq models to generate questions for any semantic role that may exist in the context. We adopt Syn-QG and RoleQG on a sample data and summarize our observations from comparing results.[6]

Procedural vs. descriptive text Both SynQG and RoleQ are developed on datasets with narratives, such as OntoNotes 5.0 (Weischedel et al., 2013) and SQuAD (Rajpurkar et al., 2016). However we choose to apply our method on procedural text like cooking recipes. This difference in data format and scope can impact the operations for QG tasks. Procedural text tends to be imperative and instructional. This may lead to certain semantic roles being naturally omitted from sentences. For example in Table 9, the questions for Agent and Cause (RoleQ [1]) may not fit for the given sentence. Procedural text also involves a specific task under a given setting, which could lead to text elision and ambiguity as the same arguments may appear multiple times across the text. This hinders existing systems from generating accurate questions. In Table 9, if the *stir* action is performed more than once in the context, question SynQG [1] would be ambiguous to ask. The corresponding answers (*potatoes and red pepper*) also need to be inferred from other sentences. Thus we propose CRL aligned with SRL that is able to account for the above mentioned properties from procedural text. The questions from our work (CompQG [1]) tend to either include hidden text slots in the questions, or solicit them in the answer.

Competence vs. ubiquitous QG Traditional QG methods are designed to generate as many questions as possible to have a wide coverage of the context. To achieve that, SynQG includes named entities and hypernyms into SRL arguments to increase the variety of SRL templates (SynQG [2]); RoleQ generates questions for all possible semantic roles of the predicate even though the answers can not be identified in the text (RoleQ [1]). As a com-

[5]E.g., input `sauté until browned` is changed to `{using a spatula} {on the cutting board} sauté {chopped onions} until browned {resulting in sautéed onions}`

[6]For SynQG, We directly follow their proposed SRL templates and heuristics in §3.2 of their paper. For RoleQ, we follow their instruction to run the model provided in `github.com/ValentinaPy/RoleQGeneration`.

parison, we focus on competence-based QG with special attention to answer completeness. Answers to SRL questions are just semantic roles that can be retrieved locally with ease (SRL row in Table 8). They lack the capability to uncover underlying semantics and solicit new information globally, which are what we are trying to address with extra steps to connect sentences and generate complete answers from data views. For example, questions from CompQG [2] leverage the multi-sentence information. This move could be especially suitable to procedural text as it is task-oriented and further steps need to build on past actions.

7 Error Analysis and Postprocessing

Based on the intrinsic evaluation results from Table 7 and manual inspection of the generated questions, we summarize the common errors that we identified from the system output.

Error propagation from the annotation Some wrong or incomplete answers are from the data annotation. This may happen more frequently in OBJ. LIFESPAN when certain hidden ingredients are not annotated. Missing of the annotation also reduces the number of QA pairs that can be generated automatically.

Unnatural and Uninformative Questions Pronouns and simple events can result in questions that are less helpful. For example, question *To what extent do you turn it?* and *What should be cooked* cannot be answered due to the lack to specific reference and proper context. Another case is when the ingredient participant and result is the same and is the only ingredient in the QA pair. Example like that is *What is in the chopped tomatoes? - tomatoes*, which reads unnatural.

Question or Answer Ambiguity Question ambiguity may happen when the question structure is simple and the same event action is performed multiple times in the same context. For example, the question *What should be sliced?* is ambiguous if the action *slice* is mentioned in different sentences from the same recipe. Answer ambiguity happens more frequently in questions that involve the world knowledge or commonsense from the annotation. Given the questions *How do you roll the dough into rounds?*, if the candidate tool is not explicitly mentioned elsewhere in the same recipe, the annotator should resort to their world knowledge to fill it in. However, different annotators may have different answers such as *rolling pin* or *bare hands*.

To solve or alleviate aforementioned errors, we postprocess the data based on some heuristics to filter out the low-quality QA pairs. Before the events are populated into the templates, we fill in possible missing annotation for hidden tools or habitats. We assign the most likely tool or habitat to the given event based on the frequency distribution of the entity from the existing annotation. For example, if the event head *cut* has no hidden tool annotated, *knife* as the most frequent tool to this event head will be assigned. To make the questions less ambiguous, the templates that are used to generate questions involve common verbs such as *add, cook, put, place* need to contain as least one adjunct slot. This can generate more specific questions. We also remove QA pairs that contain pronouns and the pairs whose answer is only one ingredient and overlapped with the entities that are asked about.

Although the errors from the annotation and automation process are hard to avoid, the annotation overall can be improved through more careful post validation and adjudication of the data. Answer ambiguity can also be partially solved by setting rules in the specification to favor more specific entities over the general ones (e.g. *rolling pin vs. hands*, *large spoon vs. spoon*), or allowing multiple valid answers. We leave these to the future discussion.

8 Conclusion

We have proposed a method for generating competence-based questions that leverages lexical semantic knowledge involving, implicit arguments, subevent structure of verbs, and document-level event dynamics. To that end, we have constructed a new dataset that encodes manually annotated lexical semantic knowledge in a corpus of cooking recipes. Our proposed method includes a set of rich targeted question families and a suite of dense paraphrasing operations that facilitate systems to generate answers both locally and globally. We have conducted intrinsic evaluation of the system to show the quality and usefulness of CB questions. We have also performed QA tasks by letting systems answer generated questions; this in turn has provided potential insights into further improvement of existing large language models for understanding such questions. By comparing to existing QG systems, our work has focused on generating questions from the subdomain of procedural text, and the questions are designed to query *competence-based* knowledge. In future research, we intend to further improve our method by minimizing the annotation errors and ambiguous answers as indicated in the error analysis, and then apply the method to a broader range of text genres.

Acknowledgements

We would like to thank our colleagues at Brandeis and Sapienza Università di Roma, who were co-organizers of the 2022 SemEval challenge, from which the QG dataset has been drawn. In particular, we would like to acknowledge the help of Marco Maru, Simone Conia, and Roberto Navigli. We would also like to thank Brandeis LLC lab members: Eben Holderness, Kelley Lynch, and Richard Brutti for their help in orchestrating, compiling, and annotating the QG dataset; and Ayla Taylor for her initial work on creating a decontextualized generation task from the recipe dataset. All views expressed in this paper are those of the authors and no others.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Gregory A Bechtel, Ruth Davidhizar, and Martha J Bradshaw. 1999. Problem-based learning in a competency-based world. *Nurse Education Today*, 19(3):182–187.

Luisa Bentivogli, Ido Dagan, and Bernardo Magnini. 2017. The recognizing textual entailment challenges: Datasets and methodologies. In Nancy Ide and James Pustejovsky, editors, *Handbook of Linguistic Annotation*, pages 1119–1147. Springer.

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.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*, volume 11. MIT Press.

Seung Youn Chyung, Donald Stepich, and David Cox. 2006. Building a competency-based curriculum architecture to educate 21st-century business practitioners. *Journal of Education for Business*, 81(6):307–314.

Simone Conia and Roberto Navigli. 2020. Bridging the gap in multilingual semantic role labeling: a language-agnostic approach. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1396–1410, Barcelona, Spain (Online). International Committee on Computational Linguistics.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

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

Andrea Di Fabio, Simone Conia, and Roberto Navigli. 2019. VerbAtlas: a novel large-scale verbal semantic resource and its application to semantic role labeling. 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 627–637, Hong Kong, China. Association for Computational Linguistics.

Jean-Paul Doignon and Jean-Claude Falmagne. 1985. Spaces for the assessment of knowledge. *International journal of man-machine studies*, 23(2):175–196.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*.

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

Zichu Fei, Qi Zhang, Tao Gui, Di Liang, Sirui Wang, Wei Wu, and Xuanjing Huang. 2022. CQG: A simple and effective controlled generation framework for multi-hop question generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6896–6906, Dublin, Ireland. Association for Computational Linguistics.

Dirk Geeraerts. 2009. *Theories of lexical semantics*. OUP Oxford.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 643–653, Lisbon, Portugal. Association for Computational Linguistics.

Michael Heilman and Noah A. Smith. 2009. Question generation via overgenerating transformations and ranking.

Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

Jürgen Heller, Thomas Augustin, Cord Hockemeyer, Luca Stefanutti, and Dietrich Albert. 2013. Recent developments in competence-based knowledge space theory. In *Knowledge spaces*, pages 243–286. Springer.

Cheng-Ting Hsiao, Fremen ChihChen Chou, Chih-Cheng Hsieh, Li Chun Chang, and Chih-Ming Hsu. 2020. Developing a competency-based learning and assessment system for residency training: analysis study of user requirements and acceptance. *Journal of medical Internet research*, 22(4):e15655.

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

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2901–2910.

Taelin Karidi, Yichu Zhou, Nathan Schneider, Omri Abend, and Vivek Srikumar. 2021. Putting words in BERT's mouth: Navigating contextualized vector spaces with pseudowords. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10300–10313, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ayal Klein, Jonathan Mamou, Valentina Pyatkin, Daniela Stepanov, Hangfeng He, Dan Roth, Luke Zettlemoyer, and Ido Dagan. 2020. QANom: Question-answer driven SRL for nominalizations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3069–3083, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Guillaume Le Berre, Christophe Cerisara, Philippe Langlais, and Guy Lapalme. 2022. Unsupervised multiple-choice question generation for out-of-domain Q&A fine-tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 732–738, Dublin, Ireland. Association for Computational Linguistics.

R. Levy and Galen Andrew. 2006. Tregex and tsurgeon: tools for querying and manipulating tree data structures. In *LREC*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Chenyang Lyu, Lifeng Shang, Yvette Graham, Jennifer Foster, Xin Jiang, and Qun Liu. 2021. Improving unsupervised question answering via summarization-informed question generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4134–4148, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Diego Marconi. 1997. *Lexical competence*. MIT press.

Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep learning based text classification: A comprehensive review.

Lidiya Murakhovs'ka, Chien Sheng Wu, Tong Niu, Wenhao Liu, and Caiming Xiong. 2021. Mixqg: Neural question generation with mixed answer types. *ArXiv*, abs/2110.08175.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1162–1172, Minneapolis, Minnesota. Association for Computational Linguistics.

Shrimai Prabhumoye, Ruslan Salakhutdinov, and Alan W Black. 2020. Topological sort for sentence ordering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2783–2792, Online. Association for Computational Linguistics.

James Pustejovsky. 1995. *The Generative Lexicon*. MIT Press, Cambridge, MA.

Valentina Pyatkin, Ayal Klein, Reut Tsarfaty, and Ido Dagan. 2020. QADiscourse - Discourse Relations as QA Pairs: Representation, Crowdsourcing and Baselines. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2804–2819, Online. Association for Computational Linguistics.

Valentina Pyatkin, Paul Roit, Julian Michael, Reut Tsarfaty, Yoav Goldberg, and Ido Dagan. 2021. Asking it all: Generating contextualized questions for any semantic role.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *ACL*.

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.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual*

Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *EMNLP*.

Hannah Rashkin, Maarten Sap, Emily Allaway, Noah A. Smith, and Yejin Choi. 2018. Event2Mind: Commonsense inference on events, intents, and reactions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 463–473, Melbourne, Australia. Association for Computational Linguistics.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2021. Qa dataset explosion: A taxonomy of nlp resources for question answering and reading comprehension.

Michael Roth, Reut Tsarfaty, and Yoav Goldberg, editors. 2021. *Proceedings of the 1st Workshop on Understanding Implicit and Underspecified Language*. Association for Computational Linguistics, Online.

Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Christian Moldovan. 2010. The first question generation shared task evaluation challenge. In *Proceedings of the 6th International Natural Language Generation Conference*. Association for Computational Linguistics.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*.

Jingxuan Tu, Eben Holderness, Marco Maru, Simone Conia, Kyeongmin Rim, Kelley Lynch, Richard Brutti, Roberto Navigli, and James Pustejovsky. 2022. 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.

Richard A Voorhees. 2001. Competency-based learning models: A necessary future. *New directions for institutional research*, 2001(110):5–13.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. OntoNotes release 5.0 LDC2013T19.

Yichong Xu, Chenguang Zhu, Ruochen Xu, Yang Liu, Michael Zeng, and Xuedong Huang. 2021. Fusing context into knowledge graph for commonsense question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1201–1207, Online. 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.

Wei Yuan, Tieke He, and Xinyu Dai. 2021. Improving neural question generation using deep linguistic representation. In *Proceedings of the Web Conference 2021*, WWW '21, page 3489–3500, New York, NY, USA. Association for Computing Machinery.

Zhenjie Zhao, Yufang Hou, Dakuo Wang, Mo Yu, Chengzhong Liu, and Xiaojuan Ma. 2022. Educational question generation of children storybooks via question type distribution learning and event-centric summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5073–5085, Dublin, Ireland. Association for Computational Linguistics.

A Appendix

A.1 Algorithms

This section shows the algorithms describe in §5.2. Algorithm 2 and 3 describe the steps to populate LOC. CHANGE and HAB. STATE QFs.

Algorithm 2 Location Change QF Heuristics

```
Require: Grid $G$, ingredient $ingre_i$ and event $e_j$
  habitat $h_{ij} \leftarrow G(ingre_i, e_j)$
  for $k = j - 1$ to 0 do
    if $h_{ik} \neq null$ then
      break loop
    end if
  end for
  if $id(h_{ij}) = id(h_{ik})$ then
    $answer \leftarrow \{'still', h_{ij}\}$
  else
    if $text(h_{ij}) = text(h_{ik})$ then
      $answer \leftarrow \{'another', h_{ij}\}$
    else
      $answer \leftarrow \{h_{ij}\}$
    end if
  end if
  $question \leftarrow template(e_j)$
```

A.2 Annotation Process

we posted annotation positions within several University-wide distribution lists, available to all students within the various departments targeted. We hired 8 student annotators for the recipe annotation work. They were paid at the University-

Algorithm 3 Habitat State QF Heuristics

```
Require: Grid $G$, Index $Q$, habitat $h$ and event $e_j$
  events $E \leftarrow \{e_k \in Q(h) \mid k < j\}$
  ingredient candidates $C \leftarrow \{\}$
  for $e_k$ in $E$ do
      get ingredient result set $R$
      for every result $r$ in $R$ do
          for $i = k - 1$ to 0 do
              if $G(r, e_i) = h \wedge r \notin C$ then
                  $add(r, C)$; break;
              else
                  if $G(r, e_i) = null$ then
                      continue;
                  else
                      break;
                  end if
              end if
          end for
      end for
  end for
  $answer \leftarrow \{C\}$
  $question \leftarrow template(h, e_j)$
```

mandated rate of $15/hour for student research assistants. All annotators were students at a US-based university, ranging from undergraduate to master's program.

We start by running the SRL parser (Conia and Navigli, 2020) on the full dataset to label each recipe sentence with its semantic roles. Subsequently, we ask 2 students annotators to validate and correct both frames and argument labels. We then train Flair named entity recognition (NER) model[7] (Akbik et al., 2019) on 100 recipes annotated with cooking entities only. The model takes a tokenized sentence and outputs the entity tag for each token in BIO format. We apply the trained NER model to the rest of recipes to generate all the entities for the further validation. CRL annotation task includes the annotation for relations, hidden entities and coreference of entities. Each recipe is annotated once at full by one annotator. All the annotators are trained to be familiarized with the annotation guideline and annotation examples before they start the task. The full annotation guidelines for the CRL task will be available along with the publication of this paper. For the intrinsic evaluation, we assign the generated questions to 4 annotators who had the experience with CRL annotation. Annotators are familiarized with the scoring for sample questions before doing the evaluation. Annotators are also required to provide a short comment to account for low scored QA pairs.

Figure 2 shows a final CRL annotation. The ingredient `apples` as a direct object to the verb, is linked as a participant of `cut`. The ingredient `wedges` as the outcome, is linked as a result of `cut`. Both habitat and tool can only be participants of an event, not results.

[7] https://github.com/flairNLP/flair

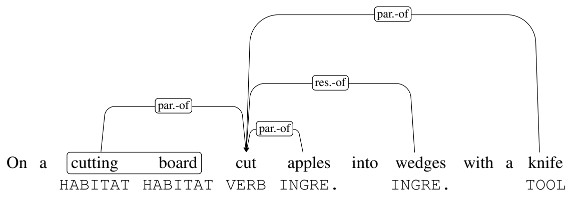
par.-of
res.-of
par.-of
par.-of
On a cutting board cut apples into wedges with a knife
HABITAT HABITAT VERB INGRE. INGRE. TOOL

Figure 2: CRL annotation.

A.3 Model Detail

We fine-tune the T5 text generation model (Raffel et al., 2020) to perform question answering task on different question subsets. To make the model output comparable, we format each input instance to `"question: {question_str} context: {recipe_str}"` that includes the raw text of the whole recipe as context regardless of the question scope or implicity. For fine-tuning T5 with CRL annotation, we modify the context string to include the CRL by inserting hidden entities as a human-readable format to the input. For example, given a piece of the context: `sauté until browned`, we change it to `{using a spatula} {on the cutting board}, sauté {chopped onions} until browned {resulting in sautéed onions}` to recover the hidden objects to a human-readable format that can be consumed by T5. All the hidden entities from each event are wrapped with squared brackets. For each experiment run, we fine-tune T5-BASE model for 15 epoches on 4 NVIDIA Titan Xp GPUs. It took roughly an hour to finish the training. We adapt the training script from https://huggingface.co/valhalla/t5-base-qa-qg-hl.

A.4 Open Source License

- Flair NER - MIT License
- RoleQGeneration - Github Default
- Deep Event & Entity Palette - GPL3
- Docanno - MIT License