Predicting Implicit Arguments in Procedural Video Instructions

Anil Batra[1] **Laura Sevilla-Lara**[1] **Marcus Rohrbach**[2] **Frank Keller**[1]

[1]University of Edinburgh, United Kingdom [2]TU Darmstadt & hessian.AI, Germany
a.k.batra@sms.ed.ac.uk l.sevilla@ed.ac.uk
marcus.rohrbach@tu-darmstadt.de keller@inf.ed.ac.uk

Project Page: https://anilbatra2185.github.io/p/ividsrl/

Abstract

Procedural texts help AI enhance reasoning about context and action sequences. Transforming these into Semantic Role Labeling (SRL) improves understanding of individual steps by identifying predicate-argument structure like {*verb*,*what*,*where/with*}. Procedural instructions are highly elliptic, for instance, (i) *add cucumber to the bowl* and (ii) *add sliced tomatoes*, the second step's *where* argument is inferred from the context, referring to where the cucumber was placed. Prior SRL benchmarks often miss implicit arguments, leading to incomplete understanding. To address this, we introduce *Implicit-VidSRL*, a dataset that necessitates inferring implicit and explicit arguments from contextual information in multimodal cooking procedures. Our proposed dataset benchmarks multimodal models' contextual reasoning, requiring entity tracking through visual changes in recipes. We study recent multimodal LLMs and reveal that they struggle to predict implicit arguments of *what* and *where/with* from multi-modal procedural data given the *verb*. Lastly, we propose *iSRL-Qwen2-VL*, which achieves a 17% relative improvement in F1-score for *what-implicit* and a 14.7% for *where/with-implicit* semantic roles over GPT-4o. Dataset and Code are publicly available[1].

1 Introduction

When humans understand procedural videos, they typically rely on verbal instructions. For example, in cooking videos, the narrator will explain the individual recipe steps as they are carried out in the video. Crucially, these verbal instructions are often highly elliptic, as a lot of information can be inferred from the visual and linguistic context.

From Figure 1, take the instruction sequence (1) *brush olive oil on pita bread* (2) *cook and cut it into cubes*. Here, we infer that in step 2, pita bread with oil is cooked and then the oiled pita bread after cooking is cut into cubes. This inference relies on linguistic context, including pronominal reference. In step 5 of Figure 1, we infer that *dressing* refers to the mixture of *olive oil, lemon, garlic, and salt* from step 3, while *bread* refers to the cubes of oiled pita bread from step 2, which are later added to a bowl containing *cucumbers, lettuce, pepper, and onions*. Linguistic context alone is insufficient to support this inference, but the visual context resolves the ambiguity – we are dealing with three distinct mixtures: *dressing, salad, bread.*

Step 1: brush olive oil on pita bread.
Step 2: cook and cut it into cubes.
Step 3: combine olive oil, lemon, garlic and salt in bowl.
Step 4: add cucumbers, lettuce, pepper, and onions to bowl.
Step 5: add dressing and bread ??.

Step 1: brush - olive oil - pita bread
Step 2 (a): cook - olive oil, pita bread - ø
Step 2 (b): cut - olive oil, pita bread - ø
Step 3: combine - olive oil, lemon, garlic, salt - ø
Step 4: add - cucumbers, lettuce, pepper, onions - ø
Step 5: add - olive oil, lemon, garlic, salt, olive oil, pita bread - cucumbers, lettuce, pepper, onions

VERB WHAT WHERE / WITH IMPLICIT

Figure 1: ***Implicit-VidSRL***: A new semantic role labeling (SRL) based dataset, to represent procedural videos using semantic frames ({*verb*,*what*,*where/with*}) with implict arguments. For instance *step 2* is transformed into *step 2(a)* & *step 2(b)*. While in step 5 the arguments are implicit and require both visual and textual context to infer from step 3 & 2. The implicit information is emphasized using a background color.

We need to capture such inferences in order to build models that understand complex, multi-step instructional videos. More specifically, our models need to predict **implicit arguments**, such as *olive oil, pita bread* in step 2 and *cucumbers, lettuce, . . .* in step 5. In this work, we propose to use **semantic role labeling** (SRL) to model the semantics of narrated instructional video as simple predicate-

[1]https://github.com/anilbatra2185/implicit-vid-srl.git

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 30399–30419
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

argument structures. We show how this approach can be used to capture both explicit and implicit arguments in instructional steps, enabling more comprehensive video understanding and more informative system evaluation. We use a domain-specific variant of traditional SRL annotation, using {*verb*, *what*, *where/with*} tuples to represent semantic frames in multi-step videos (see Figure 1). Our focus is on implicit arguments that pertain to recipe ingredients in the *what* and *where/with* roles. We annotate a subset of videos from the standard procedural datasets i.e. YouCook2 (Zhou et al., 2018) and Tasty (Sener et al., 2022), resulting in multimodal procedural data with SRLs, the *Implicit-VidSRL* dataset. We use a cloze task and a next-step prediction task to evaluate a model's comprehension of procedural data. The cloze task requires the model to predict the *what* and *where/with* roles (given the *verb*); this is a way of assessing the model's contextual reasoning skills. In next-step prediction, the model predicts the full semantic frame—including the verb and arguments (both implicit and explicit) — and generates the instruction text for the next step, using the previous steps as context. We evaluate SRL predictions with F1-scores for argument identification, verb recall for next-step prediction, and conventional natural language generation (NLG) metrics for generated text.

We assess various large multimodal models across both tasks and apply chain-of-thought (CoT) prompting (Wei et al., 2022) to enhance the models' ability to predict implicit arguments. However, they still face difficulties in inferring implicit information from temporal contexts. To address this, we automatically create a training dataset with silver-standard SRL annotations using GPT-4o through in-context learning. We show that fine-tuning both the text and video versions of the Qwen2 model on silver-standard semantic labels enhances implicit argument prediction. Our *iSRL-Qwen2-VL* achieves 17% relative F1-score improvement for *what* and 14.7% for *where/with* over GPT-4o with multimodal input.

We summarize our **contributions** as: **(i)** We present *Implicit-VidSRL*, an SRL-based benchmark for the understanding of procedural steps in instructional videos, **(ii)** We show that implicit argument prediction is challenging for multimodal models such as GPT-4o and Qwen2-VL, **(iii)** Using our SRL scheme as an intermediate representation in the next step prediction task, we show that it boosts the Qwen2-VL model's performance in predicting future steps, leading to a $\sim$2% improvement in the METEOR score, **(iv)** We show that large multimodal models achieve good performance in next-step and implicit argument prediction when fine-tuned on silver-standard SRL annotations.

2 Related work

Semantic Role Labels Semantic role labeling (SRL) is a key NLP task that aids applications like document summarization (Fan et al., 2023), building knowledge graphs (Mahon et al., 2020), machine translation (Shi et al., 2016) and question answering (Berant et al., 2013). The goal of SRL is to identify and label the arguments of the semantic predicates in a sentence, providing a shallow semantic representation. The process involves two stages: first, identifying the predicate (typically a verb), and second, determining its arguments along with their specific roles, such as *what* or *where/with*. Previous linguistic benchmarks, such as PropBank (Palmer et al., 2005), FrameNet (Baker et al., 1998), NomBank (Meyers et al., 2004), and the CoNLL SRL task (Carreras and Màrquez, 2005), concentrate on single sentences. These datasets mainly focus on explicit arguments; however, Gerber and Chai (2010) expand NomBank by incorporating implicit information gathered from all preceding sentences in the document through manual annotation. The work focuses on a predefined set of 10 nominal predicates and shows that implicit arguments improve the role coverage of predicates, bridging the gap between human and machine understanding of text. In contrast, we focus on verb predicates in procedural text and annotate implicit arguments using all the entities mentioned in the previous context.

SRL has also been used for video understanding (Yang et al., 2016; Sadhu et al., 2021). Yang et al. (2016) extend the TACoS corpus (Regneri et al., 2013) with bounding box annotations for verb predicate, along with explicit and implicit arguments. This prior research on implicit labels has limitations: (i) it focuses on single short clips, (ii) it only covers "cutting cucumber" and "cutting bread" tasks, and (iii) the implicit information is local in text or video. Later, Sadhu et al. (2021) introduced the VidSitu dataset for video-SRL, and Khan et al. (2022) extended it with argument bounding boxes. However, this prior work either focuses on local implicit information or explicit information in short temporal contexts. In contrast, we incorporate implicit information that requires the model to access an extended temporal context to identify ingredients related to the cooking actions. Moreover, in the cooking domain, the visual appearance and the composition of ingredients can change over time, which makes the task challenging.

Temporal Reasoning As video-LLMs advance, several benchmarks have emerged to evaluate their reasoning abilities. SEED-Bench (Li et al., 2024b) targets spatial or temporal comprehension, with a specific focus on the sequence of actions in procedural understanding. TempCompass (Liu et al., 2024) analyzes short clips featuring a single object, concentrating on aspects like speed, direction, and the order of actions. SOKBench (Wang et al., 2024a) aligns with our interests as it uses videos-captions pairs from YouCook2 (Zhou et al., 2018) to create graphs linking objects and entities. However, it often faces issues with underspecified instructions, leading to incomplete knowledge graphs and QA pairs. Our dataset, in contrast, offers comprehensive information based on which complete graphs and effective QA pairs could be generated. Current work focuses on open vocabulary generative predictions and we plan to introduce a QA-based reasoning benchmark in the future.

Procedural Step Anticipation Predicting human actions has been extensively researched, enabling digital assistants to anticipate behaviors for improved support and protection. Leveraging the planning capabilities of LLMs, recent studies have utilized multimodal LLMs for next-step anticipation for long-term forecasting (Zhao et al., 2024; Islam et al., 2024). Such efforts focus on coarse-grained action anticipation using *verb-noun* pairs (e.g., *take knife*), though some work (Abdelsalam et al., 2023a; Sener et al., 2022) explores fine-grained textual predictions. However, the limitations of natural language generation (NLG) metrics make evaluation difficult. In this work, we use additional metrics based on semantic frames to evaluate the fine-grained next step predictions.

Procedural Understanding Procedural learning is challenging with text-only or multimodal datasets, as it involves parsing recipe steps (Yamakata et al., 2020) and understanding visual dynamics (Zhou et al., 2018). As part of recent work on the LLM-based understanding of procedural text, Diallo et al. (2024) introduce *PizzaCommonSense*. In this dataset, input-output pairs are annotated with intermediate step outputs. For instance, given two instructions, (i) 'combine yeast, sugar, water', (ii) 'stir gently to dissolve', the corresponding annotations are (i) 'combine – yeast, sugar, water – yeast mixture', (ii) 'stir – yeast mixture – yeast mixture'. Such annotations focus on explicit entities and lack an understanding of the compositions, e.g., 'yeast mixture' is made with yeast, sugar and water. In contrast, our annotation contains implicit information and focuses on {*verb*, *what*, *where/with*}, i.e., (i) 'combine – yeast, sugar, water – ∅', (ii) 'stir – yeast, sugar, water – ∅'. Moreover, *PizzaCommonSense* is restricted to pizza recipes and only includes text, whereas our dataset, *Implicit-VidSRL*, is multimodal and features a diverse range of cooking recipes.

In prior work, multi-modal instructional video dataset are explored for learning procedural knowledge (Lin et al., 2022) and procedure planning (Chang et al., 2020). Researchers have proposed various datasets for understanding instructional videos (Damen et al., 2021; Zhukov et al., 2019; Tang et al., 2019; Afouras et al., 2023), typically featuring annotations with fixed labels or brief step captions. In contrast, we focus on extended captions composed in natural English (Zhou et al., 2018; Sener et al., 2022). It is also notable that cooking recipes make up a significant portion of existing datasets such as Epic-Kitchens (Damen et al., 2021) (100%), CrossTask (Zhukov et al., 2019) (82%), HT-Step (Afouras et al., 2023) (100%), Exo-Ego4D (Grauman et al., 2024) (40%), Guide (Liang et al., 2024) (51%). There are only a few datasets where cooking has an equal representation among other domains, such as COIN (Tang et al., 2019), or that concentrate on other domains, like Assembly-101 (Sener et al.). Cooking recipes are conceptually challenging, with many elliptical instructions and state transformations (shape or visual changes). These can be learned from large-scale data on platforms like YouTube and applied to other domains. Thus, we use multimodal cooking recipes, inspired by prior work and existing datasets.

3 The *Implicit-VidSRL* Dataset

We utilize the cooking videos from the validation and test sets of YouCook2 (Zhou et al., 2018) and Tasty (Sener et al., 2022). We have an untrimmed procedural video with multi-step instructions needed to complete a goal (e.g., a person following a cooking recipe). Each instruction is labeled with start and end times, delimiting the instruction temporally and a fine-grained description. Let $\mathcal{V}$ be a corpus of procedural videos with corresponding instructions and goals. Formally, each video is represented as $\mathbf{v} = (\mathcal{G}, \{(t_i^s, t_i^e, s_i)\}_{i=1}^{i=N})$, where $\mathcal{G}$ is title or goal of the video with N instructions, (t_i^s, t_i^e) are start and end time for the video clip, and s_i represents a single multi-step instruction. We choose videos for the test set according to the criteria video duration ($\geq$ 30 sec and $\leq$ 10 min), number of instructions (≥ 4), valid YouTube videos, and presence of implicit arguments.

Following traditional SRL annotation schemes (Gildea and Jurafsky, 2002; Do et al., 2017), we propose to decompose a multi-action instruction s_i paired with video clips into M simple predicate-argument structures $\{a_j\}_{m=1}^{M}$. In our annotation scheme, each a_j is the tuple of the form {*verb*, *what*, *where/with*}, where *verb* is the primary action, *what* denotes the objects impacted by the action, and *where/with* refers to the location, context, or accompanying elements of the action. For example, step 2 in Figure 1 is decomposed into two semantic frames, step 2(a) $\{verb{:}cook, what{:}[olive\ oil, pita\ bread], where/with{:}\emptyset\}$ and step 2(b) $\{verb{:}cut, what{:}[olive\ oil, pita\ bread], where/with{:}\emptyset\}$, where $\emptyset$ signifies an empty argument. Note that the semantic frame can include implicit arguments, i.e., arguments of the verb that are inferred linguistically or visually. For example, in step 5, *what* in the semantic frame references the mixture from steps 3 and 2, which is combined with the vegetable mixture from step 4, needing both multimodal information. In this work, we focus only on implicit arguments for recipe ingredients, limited to the *what* and *where/with* roles.

Data Annotation Our annotation process occurs in three stages. *Stage 1:* The goal of the first stage is to identify the implicit entities from the context in the form of either video or text. To achieve this goal, we hired two PhD students who had linguistics knowledge. They were trained to extract both implicit and explicit information and to identify where information might be unstated. The annotators were given the full videos, as well as the recipe steps and their corresponding timestamps so that they did not have to watch the full video. For instance, for (i) "add onions and tomatoes to the blender and blend them" and (ii) "add spices and garlic to the blender", the annotators turn (ii) into "add spices and garlic to *onions, tomatoes*".

Name	Value
Number of videos	231
Average/max video duration	125.33/588.2
Average/max steps per video	7.47/14
Average/max SRLs per video	11.02/24
Unique *verb*	158
Unique entities	805
Unique *what* arguments	726
Average entities per *what*	4.45
Average implicit entities per *what*	6.29
what emptyset/total count	0/2545
Unique *where/with* arguments	626
Average entities per *where/with*	3.82
Average implicit entities per *where/with*	5.21
where/with emptyset/total count	1393/2545

Table 1: Statistics of ***Implicit-VidSRL*** Dataset. The unit for video duration is *seconds*.

Stage 2: Multi-step instructions featuring these implicit entities are automatically converted into semantic role labels using GPT-4o-Mini. Specifically, we manually annotate five examples that show how to convert multi-step instructions to our scheme of semantic frames, i.e., {*verb*, *what*, *where/with*}. We include them as in-context examples in a chain-of-thought prompt for automatic annotation.

Stage 3: In the final stage, the automatically generated labels undergo manual correction by an annotator (again a PhD student), ensuring the implicit information is accurately specified in the arguments and that all cooking ingredients are included, but any tools are ignored, as per our annotation guidelines. The guidelines, along with the prompt, are provided in the supplementary material.

Dataset Statistics Table 1 presents various statistics of the *Implicit-VidSRL* dataset. We observe that the average number of implicit entities is 6.29 for *what* and 5.21 for *where/with* across the 2.5K semantic frames in 231 videos. There are no empty instances for *what* semantic role, but 54% of *where/with* semantic roles are empty.

4 Task

We define two tasks on our *Implicit-VidSRL* Dataset, namely Semantic Argument Prediction and Next Step Prediction. Their goal is to use context to predict the semantic frames of instruction steps, focusing in particular on implicit arguments. For both tasks, we consider two different scenarios: the input is either (a) a sequence of textual instruction steps or (b) a sequence of trimmed video clips (one clip per step). We refer to this input as context $\mathcal{C}$.

Implicit Argument Prediction A cloze task involves filling in missing words or phrases in a text

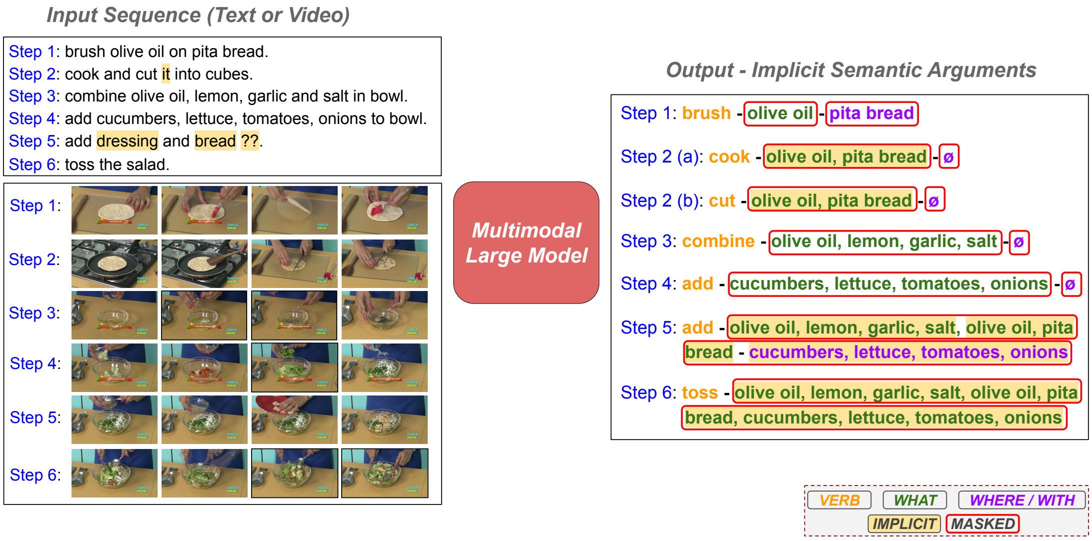


Figure 2: The *Implicit Argument Prediction* task involves providing the input sequence, which may be in the form of text or video or both, to a multimodal large model, alongside masked semantic frames: The arguments that are highlighted with red boxes in the output structure are not provided as part of the input and have to be predicted.

to measure understanding and inference abilities. Rubin (1976) demonstrated that human accuracy in cloze tasks increases with the amount of context available and is also affected by where that context is positioned. Therefore, this task is suitable for assessing the contextual reasoning skills of a model; in our case, the ability to predict the entities in cooking recipes that fill semantic roles.

We are given an input sequence $\mathcal{C}$ and corresponding semantic frames $\{(c_i, a_i)\}_{i=1:K}$. Here c_i refers the single instruction in the text or trimmed video clip, $a_i = \{(v_l, wh_l, ww_l)\}_{l=1:L}$ is the annotated semantic frames. Here, v_l refers to the *verb*-predicate, while wh_l and ww_l represent the arguments for *what* and *where/with* sematic roles consisting of both explicit and implicit entities, i.e., $wh_l = \{e_{explicit} \cup e_{implicit}\}$. To evaluate these sets of arguments, we use the F1-score, for both the combined set and the set of implicit entities.

The cloze task is a sequence-to-sequence problem, where the input is the sequence C and the corresponding semantic frames. We mask the arguments of the *what* and *where/with* semantic roles, while the predicate is given in the prompt, i.e., we have $\{verb{:}v_l,\ what{:}[?],\ where/with{:}[?]\}$. The goal is to predict the masked arguments, which can include both explicit and implicit entities.

Next Step Prediction In this task, we follow the protocol of Abdelsalam et al. (2023b): Given the partial input sequence $\mathcal{C}_{:t}$ for the first t instruction steps either in a video or textual format, and the semantic role labels $(c_{1:t}, a_{1:t})$ for these steps, the task is to output k plausible options for the next instruction step. The next step predictions are expressed as a natural language sentence in conjunction with the corresponding semantic role labels, i.e., $\{(s_{t+1}^{(1)}, a_{t+1}^{(1)}), \ldots, (s_{t+1}^{(k)}, a_{t+1}^{(k)})\}$. We generated 802 samples for next step prediction from 231 annotated instructional videos with $t \geq 3$.

5 The *iSRL-Qwen2-VL* Model

In this section we describe the *iSRL-Qwen2-VL* model, which is designed to infer implicit argument information from the context in the form of multimodal inputs (video frames or text or both).

Silver-standard Dataset To effectively infer implicit arguments from video, we create a silver-standard dataset. To achieve this, we augment the training dataset from Tasty (Sener et al., 2022) that already comes with temporally annotated instructions. By leveraging the chain-of-thought prompt along with the in-context examples we used for the annotation of the test set, we generate the silver-standard dataset for training a model that can identify implicit arguments. Specifically, we prompt the GPT-4o to perform two tasks similar to our manual annotation: (i) split the multi-step instructions into single predicate-argument structures, (ii) automatically infer the implicit entities and update the predicate arguments conditioned by the sequence of textual instructions. Finally, we generated ~2.5K training instructional video samples and formatted the samples as required for next step prediction, leading to ~18K training samples.

Training Objective We perform supervised instruction fine-tuning with LoRA of Qwen2-7B-Instruct and Qwen2-VL-7B-Instruct model, leveraging our auto-generated silver-standard dataset. In this work, we train a model with the next step prediction task, i.e. given a partial sequence of a cooking recipe (in text or video), the model is required to predict the next textual step. Additionally, it must generate semantic frame tuples that include both explicit and implicit entities.

6 Experimental Setup

6.1 Implementation Details

Training and Inference We perform fine-tuning with the default LoRA (Hu et al., 2021) configurations provided in LLama-factory (Zheng et al., 2024) on four A100-80GB GPUs with maximum duration of 48 GPU hours. Further implementation details are available in supplementary materials.

6.2 Evaluation Metrics

To assess the predictions of semantic arguments for *what* and *where/with* across both tasks, we employ the F1-score as the performance metric. Predicted and actual arguments are treated as sets of entities; the predicted entities are $P = \{w_1^p, w_2^p, \ldots, w_i^p\}$ and the gold-standard entities are $G = \{w_1^g, w_2^g, \ldots, w_j^g\}$, where each w_i can be a phrase. We use the $\hat{\cap}(w_i^p, w_j^g)$ function (Equation 1) to compute precision ($\frac{|P \hat{\cap} G|}{|P|}$) and recall ($\frac{|P \hat{\cap} G|}{|G|}$). We first identify all exact matches between the predicted and gold-standard entity sets and for non-exact matches, we calculate word overlap (intersection over union, IoU). We also compute the F1-score for implicit arguments separately.

$$\hat{\cap}(w_i^p, w_j^g) = \begin{cases} 1, & \text{if } w_i^p = w_j^g \\ \text{IoU}(w_i^p, w_j^g), & \text{otherwise} \end{cases} \quad (1)$$

For the next step prediction task, we follow Abdelsalam et al. (2023a) and predict k plausible next step. Unlike the original method, we enhance the evaluation by employing a sliding window of the next three steps from the gold-standard instructions and use this to identify the best match based on the cosine similarity of embeddings. For the best match, we compute the verb recall $R_{verb}@5$ and the F1-score for semantic arguments as for the cloze task. Our task requires that semantic frames and instruction text are generated at the same time, so in addition to our SRL metrics, we also compute the standard natural language generation metrics BLEU4 (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005) on text.

6.3 Models

Proprietary Model We utilize the most recent OpenAI GPT-4o (Hurst et al., 2024) model (gpt-4o-2024-08-06); all the experiments were conducted before February 2025. We report the results on both text and vision versions of the model.

Open Source Models We use decoder-only models for the evaluation of both tasks. For **text-only inputs**, we use Qwen2 instruct (Yang et al., 2024) as the primary language model, due to its performance and use in recent video-LLMs. We also use the LLama 3.1 (Dubey et al., 2024) instruct model. For **video-only inputs**, we focus on models with long context, i.e., models accepting a large number of frames. Specifically, we utilize LongVA (Zhang et al., 2024) and Qwen2-VL (Wang et al., 2024b). Given Qwen2-VL's capability to handle a context of 32K tokens, we designate it as the primary model for both tasks. In addition to this, we also evaluate LLava-OneVision (Li et al., 2024a) using recursive inference. This involves predicting semantic arguments for a single instruction based on a given input modality and then prepending the response to the prompt for the subsequent prediction. For video inputs, we limit to 320 frames per video at the highest resolution based on available GPU memory.

7 Results and Discussion

7.1 Implicit Argument Prediction

Our experimental results for semantic argument prediction, including implicit entities for *what* and *where/with* semantic roles, are summarized in Table 2. The results are organized based on the model types: Proprietary Model (Rows 1–3), Small Open Source Models (Rows 4–13), and Large Open Source Models (Rows 14–17).

In the scenario with text-only input, we observe that LLama-3.1-70B (Row 15) achieves the highest performance for implicit argument prediction, closely followed by GPT-4o (Row 1). Notably, our Small *iSRL-Qwen2* model (Row 7) also shows competitive results in implicit argument prediction.

When evaluating the scenario with video-only input, large models such as GPT-4o (Row 2)

Row	Model	#Params	Type	FT	SRL		SRL-Implicit	
					$F1_{what}$	$F1_{where}$	$F1_{what}$	$F1_{where}$
Proprietary Model								
1.	GPT-4o	-	T	✗	60.62	54.79	45.50	47.31
2.	GPT-4o	-	V	✗	49.63	42.02	43.38	40.08
3.	GPT-4o	-	V + T	✗	64.83	55.32	50.53	49.01
Small Open Source Models (<10B)								
4.	Qwen2	7B	T	✗	44.51	26.94	19.88	25.81
5.	Qwen2-Math	7B	T	✗	29.33	7.35	15.42	6.75
6.	LLama-3.1	8B	T	✗	52.05	38.27	31.22	39.11
7.	*iSRL-Qwen2*	7B	T	✓	57.82	49.33	51.70	47.74
8.	LongVA	7B	V	✗	9.49	3.16	6.81	3.80
9.	LLava-OV	7B	V	✗	20.78	7.86	11.43	8.44
10.	Qwen2-VL	7B	V	✗	30.20	15.15	22.51	17.07
11.	*iSRL-Qwen2-VL*	7B	V	✓	46.21	33.43	45.76	36.06
12.	Qwen2-VL	7B	V + T	✗	42.07	22.54	22.68	21.96
13.	*iSRL-Qwen2-VL*	7B	V + T	✓	**64.86**	54.54	**59.15**	**56.21**
Large Open Source Models (>10B)								
14.	Qwen2	72B	T	✗	56.10	50.86	41.82	49.61
15.	LLama-3.1	70B	T	✗	63.04	**55.50**	50.46	53.42
16.	LLava-OV	72B	V	✗	39.52	29.05	34.34	36.48
17.	Qwen2-VL	72B	V	✗	44.55	38.48	35.63	39.43

Table 2: **Implicit Argument Prediction:** We report results for different input **Type**s: text-only (**T**), video-only (**V**), and video-text (**V+T**) for predicting *what* and *where/with* given the *verb*. **SRL** includes both explicit and implicit entities, while **SRL-Implicit** focuses only on implicit ones. **FT:** Fine-tuning with our silver-standard dataset. **#Params:** Size of model.

and Qwen2-VL (Row 17) achieved the best F1-scores, although their performance is lacking behind the scenario with text-only input. This highlights two primary challenges in semantic argument prediction from video: recognizing entities in the current step and inferring or tracking entities within the temporal context. For smaller open-source models, performance is poor. However, our model *iSRL-Qwen2-VL* bridges the gap to text-only models by effectively doubling the results when using video-only input, i.e., $F1_{what\text{-}implicit}$ and $F1_{where/with\text{-}implicit}$ improve by 23.25% and 18.99% (comparing Rows 10 and 11).

We examine different models with multimodal input (V+T) and found that it enhances performance over single modality inputs to infer implicit information. Presumably, multimodal input makes it possible to recognize local entities and to resolve visual-linguistic ambiguities, as shown in Figure 1. However, with both modalities, GPT-4o exhibits a bias towards textual entities, incorrectly specifying *dressing* as an argument for *what*, instead of the mixture's ingredients (see Figure 7a). In contrast, our model (Row 13), using multimodal input, achieves the best performance on the implicit metrics, effectively tracking and inferring implicit entities from the temporal context. Overall, *iSRL-Qwen2-VL* achieves a 17% relative improvement in F1-score for *what-implicit* and a 14.7% for *where/with-implicit* semantic roles over GPT-4o.

Qualitative Results The example in Figure 3, highlights common errors in predictions. It includes instructional text sequences and video frames, comparing video-only predictions from GPT-4o and our *iSRL-Qwen2-VL* model with gold-standard semantic frames. A frequent error is the models' inability to temporally track ingredients. For instance, in step 3(a), the *what* arguments are spices rubbed on pork, but the model overlooks that this pork is mixed with spices, differing from step 2. Such inferences are crucial for understanding long temporal contexts due to visual dynamics. In contrast, *iSRL-Qwen2-VL* effectively tracks ingredients but misses that pork with spices is removed from the cider before shredding it in step 5.

7.2 Next Step Prediction

The results for the next step prediction task are shown in Table 3. Sentence prediction metrics emphasize text fluency without considering implicit information, whereas SRL predictions assess the model's ability to infer implicit arguments. The most important observation is that adding our *iSRL* approach to the corresponding base model consistently and significantly improves performance across all metrics (Row 4/5, 8/9, and 11/12). Over-

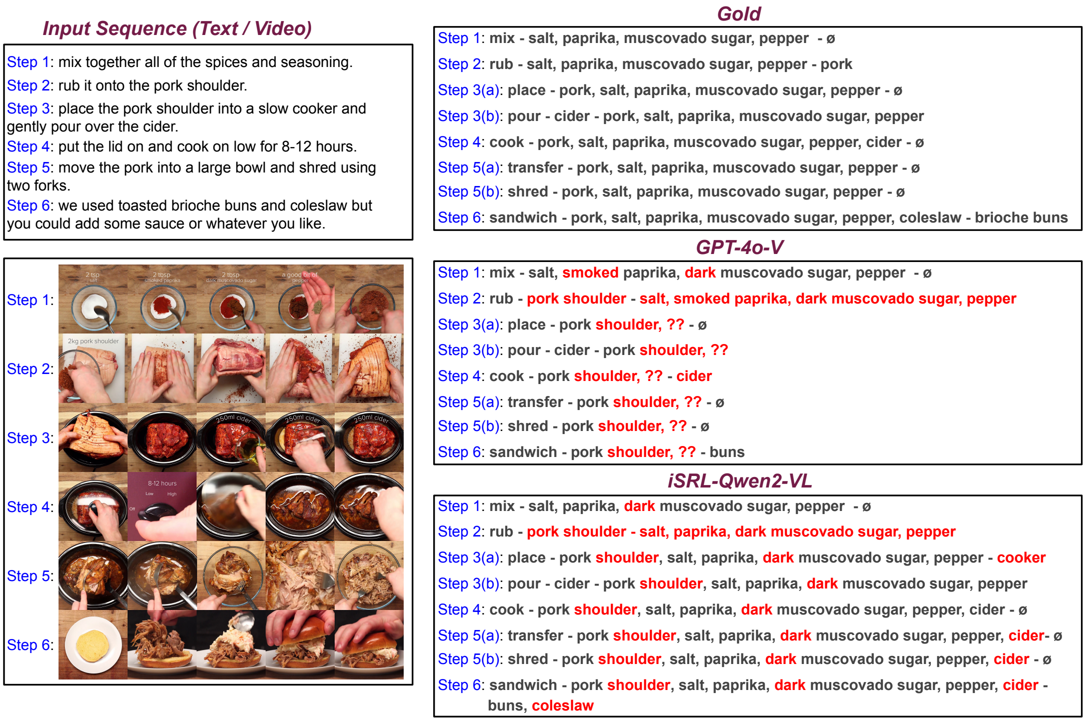


Figure 3: **Qualitative example using video-only predictions.** The example is from TASTY (Sener et al., 2022) with ID-cider-pulled-pork. The examples highlight common errors in the predictions, i.e., a failure to track the mixture ingredients, as in step 3(a) pork is mixed with spices. Incorrect predictions are highlighted in red and missing ingredients are indicated using '??'.

all, we achieve state-of-the-art performance across modalities on NLG metrics and remain competitive with, or better than, large models in semantic argument predictions. The text-only and multimodal models (Qwen2/GPT-4o) perform better than the specialized model (Row 1) for next step prediction trained on Recipe1M+ (Marın et al., 2021) dataset.

7.3 Ablation and Analysis

Chain-of-thought Prompt We experimented with various prompts to enhance the performance of the Qwen2 model using both text-only and video input, as detailed in Table 4. The chain-of-thought prompt defines semantic roles alongside implicit and explicit entities, summarizing our annotation scheme as a thought sequence. Table 4 shows that chain-of-thought (CoT) prompting significantly improves *where/with* argument identification. Without our CoT prompt, the model tends to focus on cooking tools for the *where/with* semantic role. This highlights the effectiveness of our structured prompt in guiding the model to a more accurate understanding of the task.

Fine-tuning Input We investigated the impact of fine-tuning the Qwen2 model with text-only input for next step prediction task, with and without semantic role prediction. The results in Table 5 reveal that naive finetuning without incorporating SRL predictions in the output diminishes the model's ability to infer and track explicit and implicit arguments. Conversely, when SRL predictions are included, the model's performance significantly improves, nearly doubling in effectiveness. Notably, for the $F1_{\textit{what-implicit}}$ metric, the performance sees an impressive increase of 31.82% points.

Effect of Semantic Frame Position Figure 4a,4b illustrates that the average number of entities overall, as well as the average number of implicit arguments. Both averages are higher in later semantic frame positions, requiring models to use temporal context for accurate tracking and prediction of semantic arguments. This emphasizes that tracking entities in our proposed dataset, *Implicit-VidSRL*, becomes challenging as sequence length increases.

We also study the effect of argument prediction across the semantic frame position in multimodal procedural inputs (*video and text*) for two models, i.e., GPT-4o and ours *iSRL-Qwen2-VL*. Figure 4c, 4d presents *SRL* prediction score and shows that both GPT-4o and *iSRL-Qwen2-VL* achieve

Row	Model	#Params	Type	FT	SRL Prediction			Sentence Predictions	
					$R_{verb}@5$	$F1_{what}$	$F1_{where}$	B4	METEOR
1.	GEPSAN	-	T	-	-	-	-	0.30	9.88
2.	GPT-4o	-	T	✗	**52.05**	18.48	15.04	3.18	17.93
3.	Qwen2	72B	T	✗	47.84	17.59	13.44	4.92	20.22
4.	Qwen2	7B	T	✗	45.39	9.49	9.63	3.34	17.86
5.	*iSRL-Qwen2*	7B	T	✓	50.01	**20.29**	**15.99**	**6.48**	**20.54**
6.	GPT-4o	-	V	✗	**49.89**	**18.02**	16.14	2.14	16.20
7.	Qwen2-VL	72B	V	✗	41.19	14.76	12.42	1.56	15.31
8.	Qwen2-VL	7B	V	✗	40.41	8.33	9.47	2.05	15.77
9.	*iSRL-Qwen2-VL*	7B	V	✓	44.56	16.22	**16.58**	**3.69**	**17.63**
10.	GPT-4o	-	V + T	✗	**53.36**	**20.51**	16.32	4.34	18.99
11.	Qwen2-VL	7B	V + T	✗	38.56	7.62	9.43	2.40	16.18
12.	*iSRL-Qwen2-VL*	7B	V + T	✓	47.76	19.74	**17.44**	**5.22**	**19.38**

Table 3: **Next Step Anticipation.** We report results for input **Types**: text-only (**T**), video-only (**V**), and video-text (**V+T**) to predict {*verb*,*what*,*where/with*} with explicit and implicit arguments, plus natural text for the next step. **SRL prediction** evaluates the predicate-argument structure, while **Sentence Predictions** uses NLG metrics for predicted text. **FT:** Fine-tuning with our silver-standard dataset. **#Params:** Size of model.

Model	Pompt	SRL		SRL-Implicit	
		$F1_{what}$	$F1_{where}$	$F1_{what}$	$F1_{where}$
Qwen2	CoT	44.51	26.94	19.88	25.81
	w/o CoT	42.90	20.60	18.02	19.30
Qwen2-VL	CoT	30.20	15.15	22.51	17.07
	w/o CoT	30.09	8.60	22.93	9.95

Table 4: Impact of Chain-of-Thought Prompting (CoT).

Model	FT	SRL		SRL-Implicit	
		$F1_{what}$	$F1_{where}$	$F1_{what}$	$F1_{where}$
Qwen2	Zero-shot	44.51	26.94	19.88	25.81
	w/o SRL	21.90	13.17	8.38	10.88
	w/ SRL	57.82	49.33	51.70	47.74

Table 5: Effectiveness of fine tuning (**FT**) with **SRL**. **Zero-shot** is the performance without fine tuning.

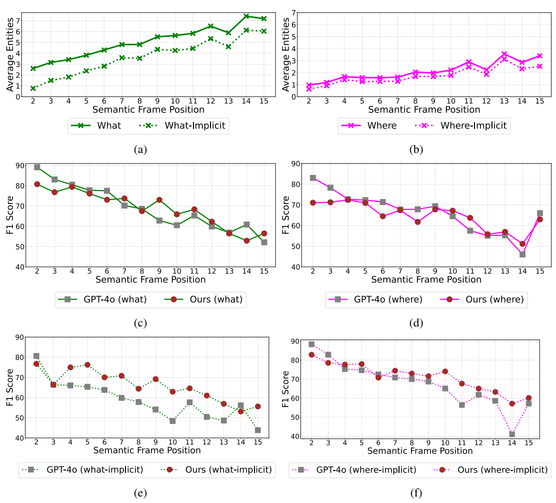


Figure 4: Comparison of GPT-4o and our *iSRL-Qwen2-VL* model for argument prediction across *semantic frame* positions in multi-modal procedural inputs (V+T).

strong results at early semantic positions, but performance decreases gradually as semantic position increases. In contrast, *iSRL-Qwen2-VL* achieves performance for implicit arguments across the board for *what* arguments, and for later semantic frame positions for *where* arguments (see Figure 4e, 4f). This shows that while GPT-4o excels at local contextual reasoning, *iSRL-Qwen2-VL* is robust for longer contextual reasoning and better at tracking implicit entities. Refer to the appendix (see Section D) for video-only inputs.

8 Conclusion

In this work, we introduced *Implicit-VidSRL*, a semantically annotated dataset for procedural video understanding, focusing on inferring implicit information from multimodal contexts. Our dataset is a step towards interpreting elliptical instructions, facilitating applications like personalizing cooking instruction (e.g., to account for allergies) and tracking entities in human-robot interaction. Through a cloze task, we assessed the ability of recent multimodal models, including GPT-4o, to infer implicit information and evaluate contextual reasoning using our dataset. Our experiments reveal that these models have difficulty tracking implicit entities. We used a chain-of-thought prompting approach with in-context examples to automatically generate a silver-standard dataset of semantic frames. We then proposed *iSRL-Qwen2-VL*, a model fine-tuned with our silver-standard data, to predict semantic frames as intermediate representations. We tested the model for next step prediction and showed that it enhances contextual reasoning and entity tracking ability in longer sequences.

Limitations

While our work advances multimodal procedural understanding, we discuss a few limitations. Automated semantic role labeling (SRL) uses GPT-4o and may introduce biases or inaccuracies in the SRL distribution. The proposed method relies heavily on the quality of the initial SRLs — suboptimal SRLs can lead to inaccurate implicit argument predictions and contextual reasoning skills. Our method also assumes that the multi-step instructions of cooking recipes can be meaningfully decomposed into simple predicate-argument structure, which may not hold for other procedural data and require further decompositions.

Acknowledgements

This work was supported in part by the UKRI Centre for Doctoral Training in Natural Language Processing, funded by UKRI grant EP/S022481/1 and the University of Edinburgh, School of Informatics. Marcus Rohrbach was funded in part by an Alexander von Humboldt Professorship in Multimodal Reliable AI sponsored by Germany's Federal Ministry for Education and Research.

References

Mohamed A Abdelsalam, Samrudhdhi B Rangrej, Isma Hadji, Nikita Dvornik, Konstantinos G Derpanis, and Afsaneh Fazly. 2023a. Gepsan: Generative procedure step anticipation in cooking videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2988–2997.

Mohamed A. Abdelsalam, Samrudhdhi B. Rangrej, Isma Hadji, Nikita Dvornik, Konstantinos G. Derpanis, and Afsaneh Fazly. 2023b. Gepsan: Generative procedure step anticipation in cooking videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2988–2997.

Triantafyllos Afouras, Effrosyni Mavroudi, Tushar Nagarajan, Huiyu Wang, and Lorenzo Torresani. 2023. HT-step: Aligning instructional articles with how-to videos. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the conll-2005 shared task: Semantic role labeling. In *Proceedings of the ninth conference on computational natural language learning (CoNLL-2005)*, pages 152–164.

Chien-Yi Chang, De-An Huang, Danfei Xu, Ehsan Adeli, Li Fei-Fei, and Juan Carlos Niebles. 2020. Procedure planning in instructional videos. In *European Conference on Computer Vision*, pages 334–350. Springer.

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. 2021. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 43(11):4125–4141.

Aïssatou Diallo, Antonis Bikakis, Luke Dickens, Anthony Hunter, and Rob Miller. 2024. Pizzacommonsense: A dataset for commonsense reasoning about intermediate steps in cooking recipes. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12482–12496.

Quynh Ngoc Thi Do, Steven Bethard, and Marie-Francine Moens. 2017. Improving implicit semantic role labeling by predicting semantic frame arguments. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 90–99, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Jing Fan, Dennis Aumiller, and Michael Gertz. 2023. Evaluating factual consistency of texts with semantic role labeling. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics (*SEM 2023)*, pages 89–100, Toronto, Canada. Association for Computational Linguistics.

Matthew Gerber and Joyce Chai. 2010. Beyond nombank: A study of implicit arguments for nominal predicates. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1583–1592.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational linguistics*, 28(3):245–288.

Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, Eugene Byrne, Zach Chavis, Joya Chen, Feng Cheng, Fu-Jen Chu, Sean Crane, Avijit Dasgupta, Jing Dong, Maria Escobar, Cristhian Forigua, Abrham Gebreselasie, Sanjay Haresh, Jing Huang, Md Mohaiminul Islam, Suyog Jain, Rawal Khirodkar, Devansh Kukreja, Kevin J Liang, Jia-Wei Liu, Sagnik Majumder, Yongsen Mao, Miguel Martin, Effrosyni Mavroudi, Tushar Nagarajan, Francesco Ragusa, Santhosh Kumar Ramakrishnan, Luigi Seminara, Arjun Somayazulu, Yale Song, Shan Su, Zihui Xue, Edward Zhang, Jinxu Zhang, Angela Castillo, Changan Chen, Xinzhu Fu, Ryosuke Furuta, Cristina Gonzalez, Prince Gupta, Jiabo Hu, Yifei Huang, Yiming Huang, Weslie Khoo, Anush Kumar, Robert Kuo, Sach Lakhavani, Miao Liu, Mi Luo, Zhengyi Luo, Brighid Meredith, Austin Miller, Oluwatumininu Oguntola, Xiaqing Pan, Penny Peng, Shraman Pramanick, Merey Ramazanova, Fiona Ryan, Wei Shan, Kiran Somasundaram, Chenan Song, Audrey Southerland, Masatoshi Tateno, Huiyu Wang, Yuchen Wang, Takuma Yagi, Mingfei Yan, Xitong Yang, Zecheng Yu, Shengxin Cindy Zha, Chen Zhao, Ziwei Zhao, Zhifan Zhu, Jeff Zhuo, Pablo Arbelaez, Gedas Bertasius, Dima Damen, Jakob Engel, Giovanni Maria Farinella, Antonino Furnari, Bernard Ghanem, Judy Hoffman, C.V. Jawahar, Richard Newcombe, Hyun Soo Park, James M. Rehg, Yoichi Sato, Manolis Savva, Jianbo Shi, Mike Zheng Shou, and Michael Wray. 2024. Ego-exo4d: Understanding skilled human activity from first- and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19383–19400.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Md Mohaiminul Islam, Tushar Nagarajan, Huiyu Wang, Fu-Jen Chu, Kris Kitani, Gedas Bertasius, and Xitong Yang. 2024. Propose, assess, search: Harnessing llms for goal-oriented planning in instructional videos. In *European Conference on Computer Vision*, pages 436–452. Springer.

Zeeshan Khan, CV Jawahar, and Makarand Tapaswi. 2022. Grounded video situation recognition. *Advances in Neural Information Processing Systems*, 35:8199–8210.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. 2024a. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*.

Bohao Li, Yuying Ge, Yixiao Ge, Guangzhi Wang, Rui Wang, Ruimao Zhang, and Ying Shan. 2024b. Seed-bench: Benchmarking multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13299–13308.

Jiafeng Liang, Shixin Jiang, Zekun Wang, Haojie Pan, Zerui Chen, Zheng Chu, Ming Liu, Ruiji Fu, Zhongyuan Wang, and Bing Qin. 2024. Guide: A guideline-guided dataset for instructional video comprehension. *IJCAI '24: Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*.

Xudong Lin, Fabio Petroni, Gedas Bertasius, Marcus Rohrbach, Shih-Fu Chang, and Lorenzo Torresani. 2022. Learning to recognize procedural activities with distant supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13853–13863.

Yuanxin Liu, Shicheng Li, Yi Liu, Yuxiang Wang, Shuhuai Ren, Lei Li, Sishuo Chen, Xu Sun, and Lu Hou. 2024. TempCompass: Do video LLMs really understand videos? In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8731–8772, Bangkok, Thailand. Association for Computational Linguistics.

Louis Mahon, Eleonora Giunchiglia, Bowen Li, and Thomas Lukasiewicz. 2020. Knowledge graph extraction from videos. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 25–32.

Javier Marın, Aritro Biswas, Ferda Ofli, Nicholas Hynes, Amaia Salvador, Yusuf Aytar, Ingmar Weber, and Antonio Torralba. 2021. Recipe1m+: A dataset for learning cross-modal embeddings for cooking recipes and food images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):187–203.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The nombank project: An interim report. In *Proceedings of the workshop frontiers in corpus annotation at hlt-naacl 2004*, pages 24–31.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. 2013. Grounding action descriptions in videos. *Transactions of the Association for Computational Linguistics (TACL)*, 1:25–36.

David C Rubin. 1976. The effectiveness of context before, after, and around a missing word. *Perception & Psychophysics*, 19:214–216.

Arka Sadhu, Tanmay Gupta, Mark Yatskar, Ram Nevatia, and Aniruddha Kembhavi. 2021. Visual semantic role labeling for video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5589–5600.

F. Sener, D. Chatterjee, D. Shelepov, K. He, D. Singhania, R. Wang, and A. Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. *CVPR 2022*.

Fadime Sener, Rishabh Saraf, and Angela Yao. 2022. Transferring knowledge from text to video: Zero-shot anticipation for procedural actions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7836–7852.

Chen Shi, Shujie Liu, Shuo Ren, Shi Feng, Mu Li, Ming Zhou, Xu Sun, and Houfeng Wang. 2016. Knowledge-based semantic embedding for machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2245–2254.

Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. 2019. Coin: A large-scale dataset for comprehensive instructional video analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1207–1216.

Andong Wang, Bo Wu, Sunli Chen, Zhenfang Chen, Haotian Guan, Wei-Ning Lee, Li Erran Li, and Chuang Gan. 2024a. Sok-bench: A situated video reasoning benchmark with aligned open-world knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13384–13394.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024b. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

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

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Shaohua Yang, Qiaozi Gao, Changsong Liu, Caiming Xiong, Song-Chun Zhu, and Joyce Chai. 2016. Grounded semantic role labeling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 149–159.

Peiyuan Zhang, Kaichen Zhang, Bo Li, Guangtao Zeng, Jingkang Yang, Yuanhan Zhang, Ziyue Wang, Haoran Tan, Chunyuan Li, and Ziwei Liu. 2024. Long context transfer from language to vision. *arXiv preprint arXiv:2406.16852*.

Qi Zhao, Shijie Wang, Ce Zhang, Changcheng Fu, Minh Quan Do, Nakul Agarwal, Kwonjoon Lee, and Chen Sun. 2024. Antgpt: Can large language models help long-term action anticipation from videos? *ICLR*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

Luowei Zhou, Chenliang Xu, and Jason J Corso. 2018. Towards automatic learning of procedures from web instructional videos. In *AAAI Conference on Artificial Intelligence*, pages 7590–7598.

Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. 2019. Cross-task weakly supervised learning from instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3537–3545.

Appendix

The supplementary material contains the following:

1. Annotation Guidelines (Section A);
2. Additional Dataset sources and Implementation details (Section B).
3. Prompt Details for Silver-standard dataset and Task inference (Section C);
4. Additional Analysis and Qualitative Results (Section D);

A The *Implicit-VidSRL* Dataset - Annotations

A.1 Annotation Guidelines

The annotation tool is shown in Figure 5

Stage 1 - Implicit Entities: We define the Implicit Argument as an ingredient implied by earlier steps or video clips, not directly visible in the current clip or text, but it must appear in either the text or video of a previous step. The Explicit Argument is an argument directly mentioned or visible in the current video clip. The additional annotation guidelines are as follows.

1. Focus only on cooking ingredients and ignore the tools or cooking utensils.
2. Only use the previously mentioned entities for the implicit arguments.
3. Only add implicit entities for nouns and pronouns like “them” and “it”, and disregard all other cases.
4. If the verb is di-transitive i.e. need two objects such as {*verb*,*what*,*where/with*}. For example: in the sentence “add spices and onions to the bowl”, the verb “add” has two arguments “spices and onions” (i.e. items need to be added) and “the bowl” (i.e. where the items will be added. However in the sentences “add spices and onions” the second argument is missing, which is implicit and we need to add the implicit argument.
5. If the verb is transitive i.e. need one object. For example, (i) “add flour and spices in a bowl and mix”. Here “mix” verb requires an implicit object i.e. mix flour and spices.

Stage 2 - GPT-4o-Mini Labels: The prompt to pre-annotate and split the multi-step instructions is shown in Figure 8.

Stage 3 - Manual Refinement: Along with the Stage 1 guidelines, we share the following additional instructions for semantic role label refinement.

1. Use previously mentioned entities for implicit arguments. For instance, (i) “add onions and tomatoes to the blender and blend them.”, (ii) “add spices and garlic to the blender” the second instruction becomes $\{verb{:}add,\ what{:}[spices, garlic],\ where/with{:}[onions, tomatoes]\}$, as the blender already contains onions and tomatoes.
2. Ignore instructions without *what* and *where/with* arguments, such as “repeat”.
3. Avoid instructions without visible ingredients like "turn heat to low" unless they are visible in the video. However, instructions like "bake tomatoes" are considered even if they are implicit.
4. Determine *what* based on the verb. For example, “add sausages on top of potatoes” becomes $\{verb{:}add,\ what{:}sausages,\ where/with{:}potatoes\}$, and “top potatoes with sausages” turns into $\{verb{:}top,\ what{:}potatoes,\ where/with{:}sausages\}$.

A.2 Annotator Recruitment and Payments

We hired three final-year PhD students to annotate the implicit entities in Stage 1. Initially, we conducted a session to explain the task, guidelines, and provide a walk-through of the Label Studio [2] tool. Despite this, one annotator struggled due to a lack of linguistic expertise. To address this, we refined our guidelines and held interactive sessions to resolve any queries. We equally divide the 700 samples among two annotators, and there is no overlap, i.e., each annotator worked on 350 samples each. This allowed us to continue effectively with the two annotators who had strong linguistic skills. They completed the work within 55 hours, and the entire process cost approximately $1200. In the final stage, the automatically generated semantic frames using GPT-4o-Mini are verified and corrected by an annotator who is different from

[2] https://labelstud.io/

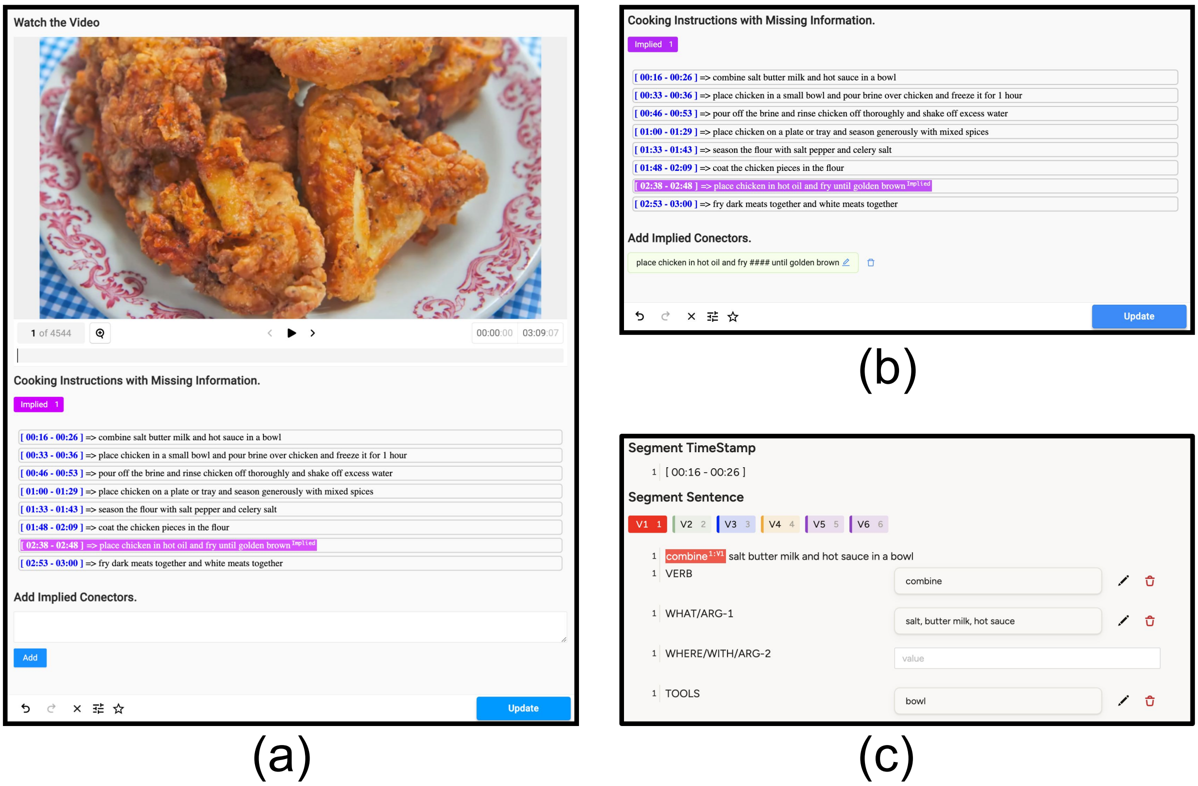


Figure 5: **Annotation Tool**. The images in (a) & (b) shows the tool interface to annotate the implicit entities during Stage 1. While the image in (c) shows the tool interface for semantic role labeling in Stage 3 (the video is omitted for clarity).

the annotators involved in the Stage 1. The PhD student spent 40 hours reviewing and refining the semantic role labels generated after Stage 2, costing approximately $800. This multi-stage annotation scheme ensures that the semantic frames are accurate. All annotators were trained in-house annotators (PhD students).

B Source Datasets and Implementation Details

Source Datasets: We utilize two instructional video datasets, YouCook2 (Zhou et al., 2018) and Tasty (Sener et al., 2022). These datasets consist of cooking videos with detailed annotations specifying instruction steps in English sentences along with temporal boundaries. YouCook2 videos are characterized by an unconstrained environment and a third-person viewpoint, while Tasty videos are captured using an overhead camera. The videos in YouCook2 are sourced from YouTube and there are on average ≈7.7 instructions per video. The videos in Tasty are relatively short, with an average duration of 54 seconds, and each recipe incorporates an average of nine instructions. An illustrative example can be found on the Tasty website[3].

Initially, we selected videos from YouCook2's validation set and Tasty's test set based on the following criteria: video duration (between 30 seconds and 10 minutes), at least 4 instructions, and valid YouTube video. Later, we selected 165 videos from Tasty's test set and 66 from YouCook2, focusing on those containing implicit information obtained after stage 1 of annotation.

The Tasty dataset is a collection of 2511 unique recipes distributed across 185 tasks, such as making cakes, pies and soups, which is used to fine tune our *iSRL-Qwen2-VL* model after generating silver-standard semantic labels.

Implementation Details: We fine-tune our model with the default configurations provided in LLama-factory (Zheng et al., 2024) specifically for LoRA (Hu et al., 2021). We perform distributed training on four A100-80GB GPUs with maximum duration of 48 GPU hours. This involves setting the rank to 8, a learning rate of 10^{-4}, and a batch size of 64. For Qwen2-VL, we use video inputs to predict the next step, with a batch size of 16 and

[3]https://tasty.co/

a learning rate of 10^{-5}. Eight frames per clip are sampled at 224 pixels. The text model is fine-tuned for three epochs and the video model for two, taking 16 and 48 GPU hours, respectively. We use NLTK for simple text processing such as lemmatization.

C Prompt Designs

Prompt for Silver-standard Dataset: The prompt to generate silver-standard dataset is shown in Figure 11.

Prompt for Implicit Argument Prediction: The prompt to perform implicit argument prediction using a cloze task is shown in Figure 9.

Prompt for Next Step Prediction: The prompt to perform next step prediction is shown in Figure 10.

D Additional Analysis and Qualitative Results

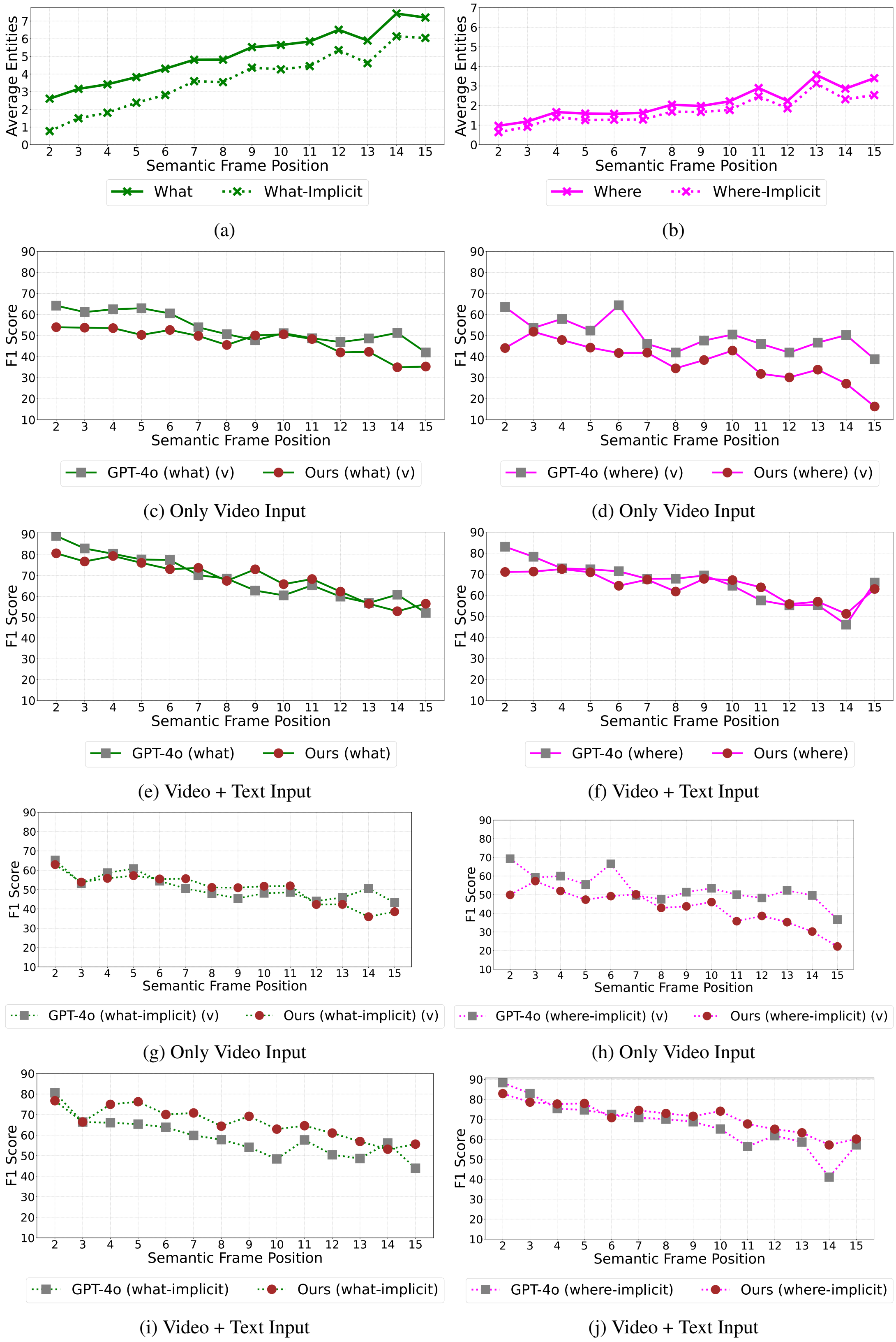
Fine-tuning Strategy We ablate different components of Qwen2-VL model for fine-tuning with proposed silver-standard dataset generated by GPT-4o and show the results in Table 6.

FT	SRL		SRL-Implicit	
	$F1_{what}$	$F1_{where}$	$F1_{what}$	$F1_{where}$
ZS	30.20	15.15	22.51	17.07
L	46.07	33.72	45.59	36.38
V+L	45.41	35.08	43.32	35.73
2-Stage	46.21	33.43	45.76	36.06

Table 6: Ablation to study Fine-tuning *iSRL-Qwen2-VL* model with silver-standard dataset generated by GPT4o. **ZS**: zero-shot results for Qwen2-VL model. **L**: Only fine-tune the LLM. **V+L**: fine-tune visual encoder and LLM together. **2-Stage**: In first stage fine-tune only LLM and in next stage fine-tune only visual encoder of Qwen2-VL model.

Tracking Entities We further extend our analysis for argument prediction across the semantic frame position in video only inputs in Figure 6 along with multimodal inputs.

Qualitative Results The additional qualitative results are shown in Figure 7.



(a) (b)

(c) Only Video Input (d) Only Video Input

(e) Video + Text Input (f) Video + Text Input

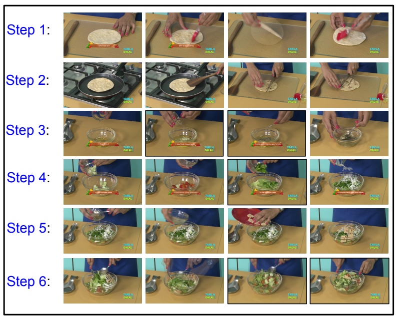
(g) Only Video Input (h) Only Video Input

(i) Video + Text Input (j) Video + Text Input

Figure 6: Comparison of GPT-4o and ours *iSRL-Qwen2-VL* models for argument prediction across *semantic frame* positions.

Input Sequence (Text / Video)

Step 1: brush olive oil on pita bread and place on pan..
Step 2: cut it into cubes.
Step 3: combine olive oil, lemon juice, garlic and salt in bowl.
Step 4: add cucumbers, lettuce, tomatoes, onions to bowl.
Step 5: add dressing and bread pieces to salad.
Step 6: toss the salad.



Gold

Step 1 (a): brush - olive oil - pita bread
Step 1 (b): place - olive oil, pita bread - ø
Step 2: cut - olive oil, pita bread - ø
Step 3 (a): add - olive oil, lemon juice, garlic, salt - ø
Step 3 (b): mix - olive oil, lemon juice, garlic, salt - ø
Step 4: add - cucumbers, lettuce, tomatoes, onions - ø
Step 5: add – olive oil, lemon juice, garlic, salt, olive oil, pita bread – cucumbers, lettuce, tomatoes, onions
Step 6: toss – olive oil, lemon juice, garlic, salt, olive oil, pita bread, cucumbers, lettuce, tomatoes, onions – ø

GPT-4o (V+T)

Step 1 (a): brush - olive oil - pita bread
Step 1 (b): place - olive oil, pita bread - ø
Step 2: cut - olive oil, pita bread - ø
Step 3 (a): add - olive oil, lemon juice, garlic, salt - ø
Step 3 (b): mix - olive oil, lemon juice, garlic, salt - ø
Step 4: add - cucumbers, lettuce, tomatoes, onions - ø
Step 5: add – **dressing, bread pieces** – cucumbers, lettuce, tomatoes, onions
Step 6: toss – **dressing, bread pieces**, cucumbers, lettuce, tomatoes, onions – ø

iSRL-Qwen2-VL (V+T)

Step 1 (a): brush - olive oil - pita bread
Step 1 (b): place - olive oil, pita bread - **pan**
Step 2: cut - olive oil, pita bread - ø
Step 3 (a): add - olive oil, lemon juice, garlic, salt - **bowl**
Step 3 (b): mix - olive oil, lemon juice, garlic, salt - ø
Step 4: add - cucumbers, lettuce, tomatoes, onions - ø
Step 5: add – olive oil, lemon juice, garlic, salt, olive oil, pita bread – cucumbers, lettuce, tomatoes, onions
Step 6: toss – olive oil, lemon juice, garlic, salt, olive oil, pita bread, cucumbers, lettuce, tomatoes, onions – ø

(a) **Qualitative example using multimodal (Video + Text) inputs.** The example is from Tasty (Sener et al., 2022) with ID-cider-pulled-pork. The examples highlights that GPT-4o fail to identify *dressing* as mixture output from step 3. Incorrect predictions are highlighted in red.

Input Sequence

Step 1: in a large bowl combine waffle mix, rosemary,egg, oil and milk.
Step 2: stir until well combined.
Step 3: pour about ½ cup of the batter into waffle iron.
Step 4: cook until the waffle is golden brown.
Step 5: add slices of cheese on half of the waffle.
Step 6: carefully fold the other half on top and close the lid.
Step 7: cook until the cheese is melted.

Gold

Step 1: combine - waffle mix, rosemary, egg, oil, milk - ø
Step 2: stir - waffle mix, rosemary, egg, oil, milk - ø
Step 3: pour - waffle mix, rosemary, egg, oil, milk - ø
Step 4: cook - waffle mix, rosemary, egg, oil, milk - ø
Step 5: add - cheese slices - waffle mix, rosemary, egg, oil, milk
Step 6: fold - waffle mix, rosemary, egg, oil, milk, cheese slices - ø
Step 7: cook - waffle mix, rosemary, egg, oil, milk, cheese slices - ø

GPT-4o (V)

Step 1: combine - waffle mix, rosemary, egg, **vegetable** oil, milk - ø
Step 2: stir - waffle mix, rosemary, egg, **vegetable** oil, milk - ø
Step 3: pour - waffle mix, rosemary, egg, **vegetable** oil, milk - ø
Step 4: cook - waffle mix, rosemary, egg, **vegetable** oil, milk - ø
Step 5: add - cheese - waffle **??**
Step 6: fold - waffle **??** - **cheese**
Step 7: cook - waffle, cheese **??** - ø

GPT-4o (V + T)

Step 1: combine - waffle mix, rosemary, egg, oil, milk - ø
Step 2: stir - waffle mix, rosemary, egg, oil, milk - ø
Step 3: pour - **batter** - ø
Step 4: cook - **batter** - ø
Step 5: add - **slices of** cheese - **half of** waffle
Step 6: fold - **other half of** waffle - **slices of** cheese
Step 7: cook - waffle, cheese **??** - ø

iSRL-Qwen2-VL (V)

Step 1: combine - waffle mix, rosemary, egg, **vegetable** oil, milk - ø
Step 2: stir - waffle mix, rosemary, egg, **vegetable** oil, milk - ø
Step 3: pour - waffle mix, rosemary, egg, **vegetable** oil, milk - **waffle iron**
Step 4: cook - waffle mix, rosemary, egg, **vegetable** oil, milk - ø
Step 5: add - **cheddar** cheese - waffle mix, rosemary, egg, **vegetable** oil, milk
Step 6: fold - waffle mix, rosemary, egg, **vegetable** oil, milk, **cheddar** cheese - ø
Step 7: cook - waffle mix, rosemary, egg, **vegetable** oil, milk, **cheddar** cheese - ø

iSRL-Qwen2-VL (V + T)

Step 1: combine - waffle mix, rosemary, egg, oil, milk - ø
Step 2: stir - waffle mix, rosemary, egg, oil, milk - ø
Step 3: pour - waffle mix, rosemary, egg, oil, milk - **waffle iron**
Step 4: cook - waffle mix, rosemary, egg, oil, milk - ø
Step 5: add - cheese **??** - waffle mix, rosemary, egg, oil, milk
Step 6: fold - waffle mix, rosemary, egg, oil, milk, cheese **??** - ø
Step 7: cook - waffle mix, rosemary, egg, oil, milk, cheese **??** - ø

(b) **Qualitative example using video (Video) and multimodal (Video + Text) inputs.** Example is from Tasty (Sener et al., 2022) with ID-waffle-grilled-cheese. Incorrect predictions are highlighted in red. The examples show that GPT-4o, with multimodal inputs (V+T), bias towards text, leading to incorrect predictions compared to video-only inputs.

Figure 7: Qualitative Examples from *Implicit-VidSRL* dataset.

You are help assistant in a cooking recipe task. Before the task, understand the definitions given below:

Definitions:
Semantic Role Labels: A list of dictionaries, each containing a verb and its corresponding arguments (WHAT, WHERE_OR_WITH). Each semantic role label is created by splitting each instruction into individual actions based on verbs, creating single-action sentences.
Semantic Roles:
- **SRL_ID**: A unique identifier for each action sentence. It is a combination of the step number and action number (e.g., "1-1" for the first action in step 1).
- **VERB**: The main action word (e.g., "chop," "place").
- **WHAT**: The primary ingredient(s) or object(s) affected by the action.
- **WHERE_OR_WITH**: Specify as WHERE (e.g., "in pan") or WITH (e.g., "with salt"). It is optional. Do not mention both WHERE and WITH in the same argument, give priority to WHERE. Use transivity of the verb to determine the argument type.

Example Format:
1. [{"srl_id":"STEP_ID-ACTION_ID", "verb": "verb_action", "what": "main_object", "where_or_with": "WITH or WHERE arguments"}]

Implicit Argument: An ingredient implied from previous steps that isn't directly stated in the current instruction. Implicit arguments can be in either the WHAT or WHERE_OR_WITH field but should only include ingredients, not tools (e.g., pan, bowl).
Explicit Argument: An argument directly stated in the sentence.

Task:
In this task, you will be presented with the list of numbered recipe steps from a cooking recipe and your goal is to predict the semantic role labels for each recipe step, based on the following instructions and examples.

INSTRUCTIONS: Steps to Follow:
1. Split, Track and Predict Implicit Ingredients:
   1.1 Split: Split each numbered recipe step into individual actions based on verbs. Each action should be a single-action sentence. For example - "1. add the flour and water in the bowl and mix." should be split into [{"srl_id":"1-1", "verb": "add", "what": "flour, water", "where_or_with":"bowl"}, {"srl_id":"1-2", "verb": "mix", "what": "flour, water", "where_or_with":""}]
   1.2 Track: Carefully read the list of numbered recipe steps. Note ingredients introduced in semantic role labels and video clip, as they may persist in future steps. For example - Given two recipe steps - "1. add onions and butter to the pan. 2. add tomatoes.", the semantic role label for step 2 should be tracked as -> 2. [{"srl_id":"2-1", "verb": "add", "what": "tomatoes", "where_or_with":"onions, butter"}
   1.3 Apply: For each action, determine if any previous ingredients are implied in the current step. Include these ingredients as arguments:
      - In WHAT if they are directly affected by the action (e.g., “mix” implies mixing all previous ingredients).
      - In WHERE_OR_WITH if they define the action's context (e.g., adding something “to the mixture”).
   1.4. Do not include implicit tools (e.g., pan, bowl) for the arguments, only mention ingredients and explicit tools.
   1.5. Carefully observe Example 1, Example 2 to understand the steps and format.
   1.6. Do not add prepositions (e.g., "in," "on"), articles, determiners, or quantities to the arguments.
2. Do not include any additional information in the response. Do not generate any additional examples.
3. Make sure **SRL_ID** is unique for each action sentence.
4. The response should be in the same format as the examples provided and semantic role labels should be numbered with same step number as the recipe steps.

QUERY - START:

Task: <<task>>
Recipe Steps:
<<recipe_steps>>
Semantic Role Labels:

Figure 8: Dataset Prompt.

You are help assistant in a cooking recipe task. Before the task, understand the definitions given below:

Definitions:
<<OMITTED DUE TO SPACE>>

Task:

In this task, you will be presented with concatenated video clips and its corresponding steps from a cooking recipe. Each video clip is seperated by a blank frame and visualizes the recipe step to perform the given task.
The corresponding semantic role labels for the each clip are presented only with verb, but the arguments are missing and marked with "??".
Based on the concatenated video clips and its corresponding steps, your goal is to fill the arguments of each verbs mentioned by "??". Do not repeat previously mentioned semantic role labels.

Steps to Follow:
1. You majorly focus on the ingredients used in the recipe. You do not focus on the tools used to prepare the ingredients.
2. Carefully watch each video clip and understand the recipe steps. Make sure to observe the blank frame that separates the video clips depicting the recipe steps.
3. Carefully understand and track the ingredients along with composite entities representing the composition of ingredients.
4. Based on the verb, decide wheather you need to fill the WHAT or WHERE_OR_WITH or both arguments.
4.1. *BOTH ARGUMENTS*: The verb such as add, mix, chop, etc. is used, then you need to fill the both arguments. However, if argument contains a tool, then you need to ignore the tool and fill it in "TOOLS".
We only use the previously mentioned entities for the implied arguments and avoid new name given to composite entities. Consider the following scenarios:
(a). Scenario 1 (Implied argument is previous entity):
(i) add onions and tomatoes to the blender and blend them to have a consistent mixture. ==> {"verb": "add", "what": "onions, tomatoes", "where_or_with":"", “tools”: “blender”}
(ii) add spices and garlic ##mixture## ==> {"verb": "add", "what": "spices, garlic", "where_or_with":"onions, tomatoes"}
(b). Scenario 2 (implied argument is list of ingredients):
(i) add onions and tomatoes to the blender and blend them. ==> {"verb": "add", "what": "onions, tomatoes", "where_or_with":"", “tools”: “blender”}
(ii) add spices and garlic ##onions, tomatoes## ==> {"verb": "add", "what": "spices, garlic", "where_or_with":"onions, tomatoes"}
(c). Scenario 3 (No implied argument):
(i) add onions and tomatoes to the blender and blend them. ==> {"verb": "add", "what": "onions, tomatoes", "where_or_with":"", “tools”: “blender”}
(ii) add spices and garlic to the blender. ==> {"verb": "add", "what": "spices, garlic", "where_or_with":"onions, tomatoes"}
4.2. *ONLY WHAT ARGUMENT*: The verb such as slice, pound, lay, etc. is used, then you need to fill the WHAT argument only and ignore the WHERE_OR_WITH argument. Consider the following example:
(a). Example - 1: In the below instruction, you need to fill the WHAT argument only. Here “mix” verb requires an implied object i.e. mix flour and spices.
(i) add flour and spices in a bowl and mix. ==> [{"verb": "add", "what": "flour, spices", "where_or_with":"", “tools”: “bowl”}, {"verb": "mix", "what": "flour, spices", "where_or_with":""}]
(b). Example - 2: The arguments are filled based on the verb.
(i). add sausages on top of potatoes ⇒ {"verb": "add", "what": "sausages", "where_or_with":"potatoes"}.
(ii). top potatoes with sausages ⇒ {"verb": "top", "what": "potatoes", "where_or_with":"sausages"}.
4.3. *COMPOSITE ENTITIES*: Memorize the ingredients and corresponding composite entities from the previous steps. Use the commonsense knowdledge to fill the arguments. Consider the following example:
(a). Example - 1: Carefully read the below recipe steps and semantic role labels. The output of step (ii) is referred as "sauce" in step (v), we call this as composite entity.
In step (v), when we dip the chicken in the sauce, we need to reason that chicken is already coated with the flour in step (iv), which is a part of enitity for composition of "flour, salt, pepper" based on the step (iii).
Recipe Steps and Semantic Role Labels:
(i) combine salt, mustard, and pepper in water. ==> [{"verb": "combine", "what": "salt, mustard, pepper", "where_or_with":"water"}]
(ii) add soya sauce, oyster sauce, and sugar. ==> [{"verb": "add", "what": "soya sauce, oyster sauce, sugar", "where_or_with":"salt, mustard, pepper, water"}]
…….
4.4. Do not include implicit tools (e.g., pan, bowl) for the arguments, only mention ingredients.
5. Only fill in the missing arguments marked with "??" in the "Response Semantic Role Labels".
………..

THE FOLLOWING EXAMPLES ASSUME THE VIDEO CLIPS ARE SUMMARIZED INTO TEXTUAL RECIPE STEPS AND GOAL IS TO FILL THE ARGUMENTS OF VERBS IN THE SEMANTIC ROLE LABELS.:

Example 1 - START:

Task: Make Falafel
Recipe Steps:
1. put chickpeas parsley chopped onion chili powder ground cumin in food processor.
………..
Masked Semantic Role Labels:
{
"srl":[
{"srl_id":"1-1", "verb": "put", "what": "??", "where_or_with":"??"},
………..
]
}
Response Semantic Role Labels:
{
"srl":[
{"srl_id":"1-1", "verb": "put", "what": "chickpeas, parsley, chopped onion, chili powder, ground cumin", "where_or_with":"", "tools":"food processor"},
………..
]
}
Example 1 - END:

FOR QUERY, YOU WILL BE PROVIDED WITH CONCATENATED VIDEO CLIPS AND TEXTUAL STEPS FROM A RECIPE WITH A TASK AND MASKED SEMANTIC ROLE LABELS. EACH VIDEO CLIP IS SEPARATED BY A BLANK FRAME. YOUR GOAL IS TO FILL THE ARGUMENTS FOR EACH VERB IN THE SEMANTIC ROLE LABELS, MENTIONED BY "??". DO NOT INCLUDE TOOLS IN ARGUMENTS AND FOCUS ONLY ON INGREDIENTS.

QUERY - START:

Task: <<task>>
Recipe Steps:
<<recipe_steps>>
Masked Semantic Role Labels:
<<masked_semantics>>
Response Semantic Role Labels:

Figure 9: Prompt for Implicit Argument Prediction as Cloze task.

You are help assistant in a cooking recipe task. Before the task, understand the definitions given below:

Definitions:
<<OMITTED DUE TO SPACE>>

Thoughts to Understand Semantic Roles:
1. It majorly focus on the ingredients used in the recipe and avoid the focus on the tools used to prepare the ingredients.
2. Based on the verb, the WHAT or WHERE_OR_WITH or both arguments are updated. Consider the following cases:
2.1. *BOTH ARGUMENTS*: The verb such as add, mix, chop, etc. is used, then the both arguments are mandatory. However, if argument contains a tool, then we ignore the tool in WHAT or WHERE_OR_WITH arguments and fill it in "TOOLS".
We only use the previously mentioned entities for the implied arguments and avoid new name given to composite entities. Consider the following scenarios:
(a). Scenario 1 (Implied argument is previous entity):
(i) add onions and tomatoes to the blender and blend them to have a consistent mixture. ==> {"verb": "add", "what": "onions, tomatoes", "where_or_with":"", "tools": "blender"}
(ii) add spices and garlic ##mixture## ==> {"verb": "add", "what": "spices, garlic", "where_or_with":"onions, tomatoes"}
(b). Scenario 2 (implied argument is list of ingredients):
(i) add onions and tomatoes to the blender and blend them. ==> {"verb": "add", "what": "onions, tomatoes", "where_or_with":"", "tools": "blender"}
(ii) add spices and garlic ##onions, tomatoes## ==> {"verb": "add", "what": "spices, garlic", "where_or_with":"onions, tomatoes"}
(c). Scenario 3 (No implied argument):
(i) add onions and tomatoes to the blender and blend them. ==> {"verb": "add", "what": "onions, tomatoes", "where_or_with":"", "tools": "blender"}
(ii) add spices and garlic to the blender. ==> {"verb": "add", "what": "spices, garlic", "where_or_with":"onions, tomatoes"}
2.2. *ONLY WHAT ARGUMENT*: The verb such as slice, pound, lay, etc. is used, then we fill the WHAT argument only and ignore the WHERE_OR_WITH argument. Consider the following example:
(a). Example - 1: In the below instruction, you need to fill the WHAT argument only. Here "mix" verb requires an implied object i.e. mix flour and spices.
(i) add flour and spices in a bowl and mix. ==> [{"verb": "add", "what": "flour, spices", "where_or_with":"", "tools": "bowl"}, {"verb": "mix", "what": "flour, spices", "where_or_with":""}]
(b). Example - 2: The arguments are filled based on the verb.
(i). add sausages on top of potatoes ⇒ {"verb": "add", "what": "sausages", "where_or_with":"potatoes"}.
(ii). top potatoes with sausages ⇒ {"verb": "top", "what": "potatoes", "where_or_with":"sausages"}.
2.3. *COMPOSITE ENTITIES*: We memorize the ingredients and corresponding composite entities from the previous steps. Use the commonsense knowdledge to fill the arguments. Consider the following example:
(a). Example - 1: We carefully read the recipe steps and semantic role labels. The output of step (ii) is referred as "sauce" in step (v), we call this as composite entity.
In step (v), when we dip the chicken in the sauce, we need to reason that chicken is already coated with the flour in step (iv), which is a part of enitity for composition of "flour, salt, pepper" based on the step (iii).
Recipe Steps and Semantic Role Labels:
(i) combine salt, mustard, and pepper in water. ==> [{"verb": "combine", "what": "salt, mustard, pepper", "where_or_with":"water"}]
(ii) add soya sauce, oyster sauce, and sugar. ==> [{"verb": "add", "what": "soya sauce, oyster sauce, sugar", "where_or_with":"salt, mustard, pepper, water"}]
2.4. Do not include implicit tools (e.g., pan, bowl) for the arguments, only mention ingredients.
2.5. Do not add prepositions (e.g., "in," "on"), articles, determiners, or quantities to the arguments.

Task:
In this task, you will be presented with the partial list of numbered recipe steps from a cooking recipe along with its semantic role labels. You are also given corresponding video clips, separated by a blank frame.
Carefully read the definitions and thoughts to understand semantic roles. Based on the list of numbered recipe steps and semantic role labels, your goal is to predict the next one instruction and corresponding semantic role label.
Carefully observe the examples below and follow the same format to predict the next step and corresponding semantic role label(s).

Thoughts to Predict Semantic Roles:
1. Read and Watch the given steps in text and video clips. Also, observe their corresponding semantic role labels. Note that for multi-step actions, the semantic role labels are splited into individual actions.
2. Similar to the examples, first predict the single or multiple semantic role labels for the next step based on the given steps.
3. Make sure to include the verb, what, where_or_with, and tools in the semantic role labels.
4. Later, predict the next instruction based on the semantic role label.
Don'ts
1. Do not include any additional information in the response. Do not generate any additional examples.
2. Do not change or remove the **SRL_ID** in the "Semantic Role Labels".
3. Make sure **SRL_ID** is unique for each action sentence.
4. Do not change the format of the output.

Please note that the examples do not include video clips, as the examples are for illustration purposes only.
Example 1 - START:
<<IN CONTEXT EXAMPLE - OMITTED DUE TO SPACE>>

Example 1 - END:

Now, predict the next instruction and the corresponding semantic role label(s) using the partial recipe steps provided in the text and video clips, along with the given semantic role labels.:
QUERY - START:

Task: <<task>>
Partial Recipe Steps:
<<recipe_steps>>
Partial Semantic Role Labels:
<<semantics>>
Future Step and Semantic Role Label(s):

Figure 10: Prompt for Next Step Prediction with Semantic Frame.

You are help assistant in a cooking recipe task. Before the task, understand the definitions given below:
Definitions:
<<OMITTED DUE TO SPACE>>

Task:
In this task, you will be presented with the list of numbered recipe steps from a cooking recipe and your goal is to predict the semantic role labels for each recipe step, based on the following instructions and examples.

INSTRUCTIONS: Steps to Follow:
1. You majorly focus on the ingredients used in the recipe. Do not focus on the tools used to prepare the ingredients.
2. For each recipe step, validate if the instruction is complex or simple. If it is complex or contains multiple actions, then split it into distinct single-action steps in your mind.
2.1 Break down the instruction into distinct single-action steps in your mind. Consider the following Examples:
- Complex Instruction 1: "Squeeze some lime juice into the food processor and add some olive oil."
- Sub Instructions 1: ["Squeeze some lime juice into the food processor", "add some olive oil"]
- Complex Instruction 2: "thinly slice the apple and place in a medium bowl with the water and lemon juice to prevent browning."
- Sub Instructions 2: ["thinly slice the apple", "place sliced apple in a medium bowl", "add the water and lemon juice to the sliced apple"]
- Simple Instruction 3: "place the falafel in a pan."
- Sub Instructions 3: ["place the falafel in a pan"]

2.2 Carefully observe the examples and split each instruction into distinct single-action steps. This will help you understand the structure of the recipe and prepare the response accordingly.
3. Once you have split the instructions into distinct single-action steps, create semantic role labels for each step. Based on the verb of each step, decide wheather you need to fill the WHAT or WHERE_OR_WITH or both arguments.
3.1. *BOTH ARGUMENTS*: The verb such as add, mix, chop, etc. is used, then you need to fill the both arguments. However, if argument contains a tool, then you need to ignore the tool and fill it in "TOOLS".
We only use the previously mentioned entities for the implied arguments and avoid new name given to composite entities. Consider the following scenarios:
(a). Scenario 1 (Implied argument is previous entity):
(i) add onions and tomatoes to the blender and blend them to have a consistent mixture. ==> {"verb": "add", "what": "onions, tomatoes", "where_or_with":"", "tools": "blender"}
(ii) add spices and garlic ##mixture## ==> {"verb": "add", "what": "spices, garlic", "where_or_with":"onions, tomatoes"}
(b). Scenario 2 (implied argument is list of ingredients):
(i) add onions and tomatoes to the blender and blend them. ==> {"verb": "add", "what": "onions, tomatoes", "where_or_with":"", "tools": "blender"}
(ii) add spices and garlic ##onions, tomatoes## ==> {"verb": "add", "what": "spices, garlic", "where_or_with":"onions, tomatoes"}
(c). Scenario 3 (No implied argument):
(i) add onions and tomatoes to the blender and blend them. ==> {"verb": "add", "what": "onions, tomatoes", "where_or_with":"", "tools": "blender"}
(ii) add spices and garlic to the blender. ==> {"verb": "add", "what": "spices, garlic", "where_or_with":"onions, tomatoes"}
3.2. *ONLY WHAT ARGUMENT*: The verb such as slice, pound, lay, etc. is used, then you need to fill the WHAT argument only and ignore the WHERE_OR_WITH argument. Consider the following example:
(a). Example - 1: In the below instruction, you need to fill the WHAT argument only. Here "mix" verb requires an implied object i.e. mix flour and spices.
(i) add flour and spices in a bowl and mix. ==> [{"verb": "add", "what": "flour, spices", "where_or_with":"", "tools": "bowl"}, {"verb": "mix", "what": "flour, spices", "where_or_with":""}]
(b). Example - 2: The arguments are filled based on the verb.
(i). add sausages on top of potatoes ⇒ {"verb": "add", "what": "sausages", "where_or_with":"potatoes"}.
(ii). top potatoes with sausages ⇒ {"verb": "top", "what": "potatoes", "where_or_with":"sausages"}.
3.3. *COMPOSITE ENTITIES*: Memorize the ingredients and corresponding composite entities from the previous steps. Use the commonsense knowdledge to fill the arguments. Consider the following example:
(a). Example - 1: Carefully read the below recipe steps and semantic role labels. The output of step (ii) is referred as "sauce" in step (v), we call this as composite entity.
In step (v), when we dip the chicken in the sauce, we need to reason that chicken is already coated with the flour in step (iv), which is a part of enitity for composition of "flour, salt, pepper" based on the step (iii).
Recipe Steps and Semantic Role Labels:
(i) combine salt, mustard, and pepper in water. ==> [{"verb": "combine", "what": "salt, mustard, pepper", "where_or_with":"water"}]
(ii) add soya sauce, oyster sauce, and sugar. ==> [{"verb": "add", "what": "soya sauce, oyster sauce, sugar", "where_or_with":"salt, mustard, pepper, water"}]
3.4. Do not include implicit tools (e.g., pan, bowl) for the arguments, only mention ingredients.
3.5. Do not add prepositions (e.g., "in," "on"), articles, determiners, or quantities to the arguments.
4. Make sure **SRL_ID** is unique for each action sentence.
5. Carefully observe the examples given below and format the response accordingly. Finally, predict the semantic role labels for the given QUERY recipe steps.

<<ICL_EXAMPLES>>

NOW IT'S YOUR TURN! PREDICT THE SEMANTIC ROLE LABELS FOR THE FOLLOWING RECIPE STEPS:
QUERY - START:

Task: <<task>>
Recipe Steps:
<<recipe_steps>>
Semantic Role Labels:

Figure 11: Prompt to generate Silver-standard dataset using GPT-4o (Hurst et al., 2024) model.