Step Differences in Instructional Video

Supplementary Material

This section contains supplementary material to support the main paper. The contents include:

- (S1) Training data generation details, including full prompts, description of data filtering implementation and additional examples to supplement Sec. 3.2.
- (S2) Annotation collection details and dataset analysis to supplement Sec. 4 (dataset) and Fig. 4.
- (S3) Full implementation and training details for baselines and our approach to supplement Sec. 4.
- (S4) Additional task formulation details including post-processing implementation for DiffCap (Sec. 4.1) and DiffMCQ negative sampling (Sec. 4.2).
- (S5) Additional experiments and ablations to supplement Sec. 4.5.
- (S6) Qualitative results to add to those presented already in Figures 5 and 6.

S1. Training data generation details

As mentioned in Sec. 3.2, we construct a paired QA dataset using pairs of video clips that share the same step label from HTStep [9]. In this section, we provide detailed descriptions of each phase in the data generation pipeline.

Action and object captioning We use a VCLM model to describe actions and objects in the video clip [31] (see details in Sec. S3). For actions, we sample 8 frames from the clip and use a HowTo100M [29] trained captioning model. For object captions, we sample the center frame of the video clip and use an image captioning model [31]. The full prompt structure for each model is shown below

```
[SYSTEM PROMPT]
You are a multimodal assistant. Designed to provide
 direct answers to users' video related questions.
Here is the video: {video}.

[ACTION PROMPT]
In one short sentence, describe what the person is
doing?

[OBJECT PROMPT]
Give a very short list of all objects that are
visible and their attributes, one per line. Only
list objects being used, NOT in the background.
```

Despite the prompt asking to only list objects being used, the LLM-based captioning models tend to hallucinate object details that are not present in the scene. We therefore post-process the object captions using an off-the-shelf text grounding model [30]. We retain only the object descriptions that have a grounding score greater than zero.

Consolidated step description Next, we consolidate all the information above into a concise step description as shown in Fig. 2 (left panel). For this, we use a text-only LLM model (Llama-2-70b-chat) with the following prompt.

```
[SYSTEM PROMPT]
You are an AI assistant that synthesizes the output
 of narration, action and object captioning models
into a single description of the content.

[PROMPT]
Video narration: {narration}.
Possible activity: {action_caption}.
Possible objects: {object_caption}.
Summarize the captions into a single, descriptive
sentence about what the person is doing, and using
what objects.
```

Paired video QA generation Finally, we select pairs of video clips, along with their generated step descriptions, and query the Llama-2 model to generate questions and answers. We generate questions of three types as shown below.

```
[SYSTEM PROMPT]
You are an AI assistant that asks questions
comparing two videos based on their descriptions,
and then answers them. Each question must be on a
new line starting with "Q:" for question and "A:"
for the answer. Use diverse language.
Video 1: {step_description_1}
Video 2: {step_description_2}

[PROMPT_TYPE1]
Summarize the differences and generate 3 question-
answer pairs comparing the two videos. Answers
should be short and concise.

[PROMPT_TYPE2]
Generate 3 question-answer pairs of the form "Which
 video ... ?". The answer must only refer to one of
 the two videos.

[PROMPT_TYPE3]
Do the two videos share a similar main action?
Answer with a single word: YES or NO.
```

The final training dataset is the composition of question-answer pairs from all three sources. See Fig. S1 for examples of this data. Note that this data is used as weakly supervised training data only. For evaluation, a separate, disjoint set of video clips is manually annotated. See Sec. 4 (dataset) and Sec. S2 for details.

S2. Annotation collection details

In this section, we provide details about the data annotation process outlined in Sec. 4 (dataset).

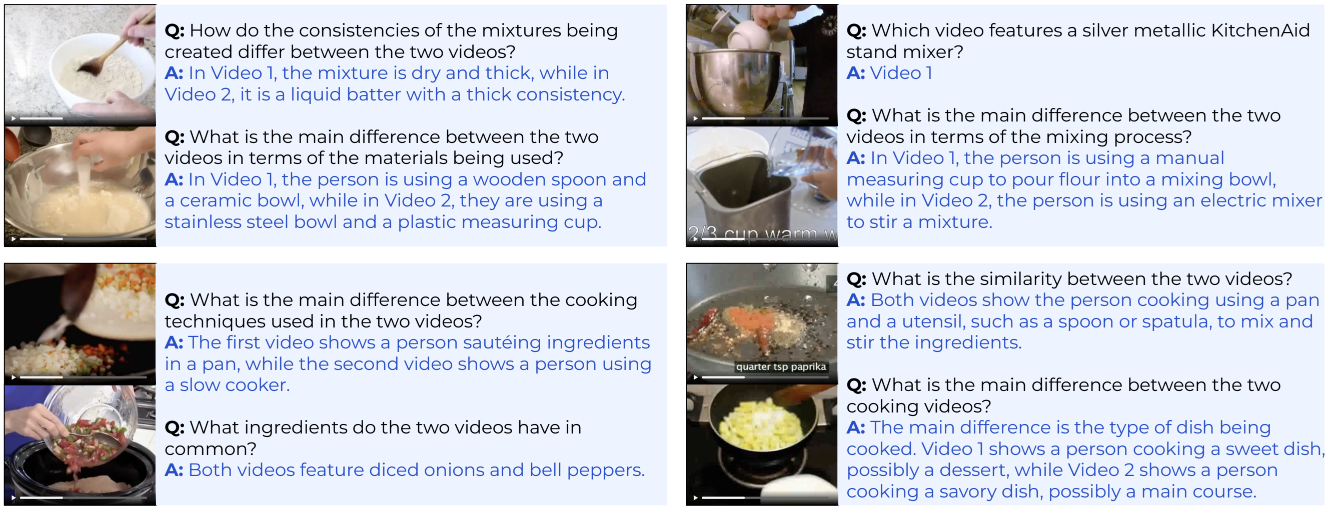

Figure S1. **Generated paired QA data.** Details in Sec. 3.2.

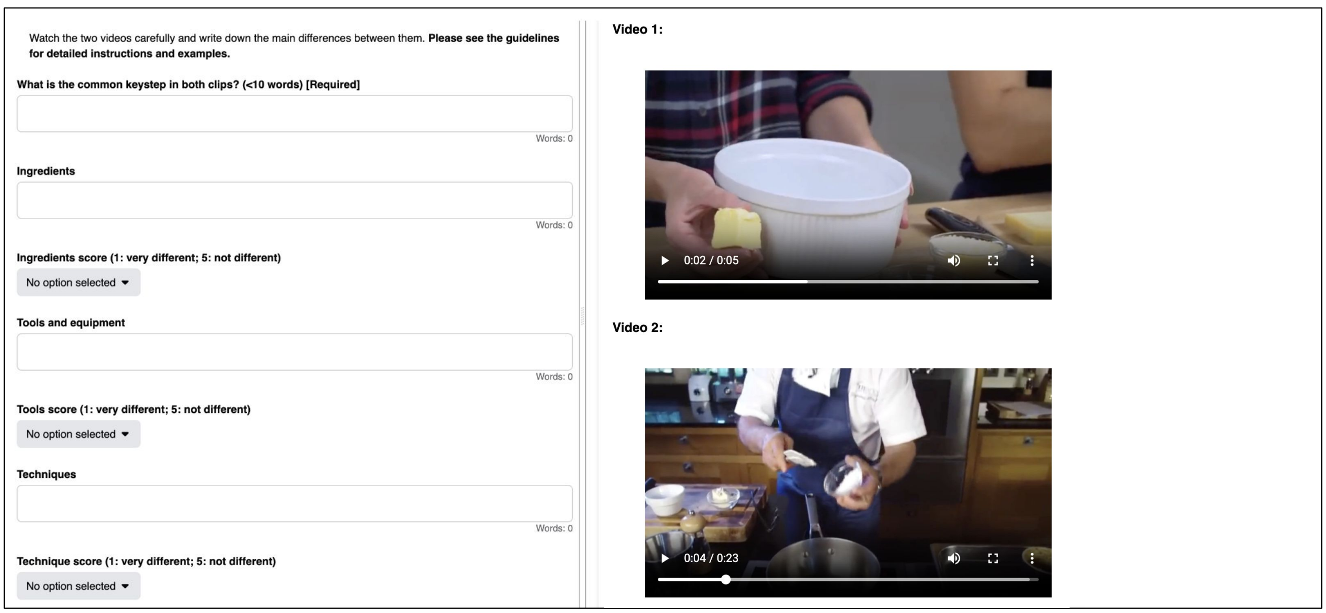

Figure S2. **Data annotation interface** Annotators first watch two short video clips of a keystep performed by two different people (right panel). After that, they write out what they think the common keystep is between the two video clips, and then describe and score the differences between the clips them along various categories (left panel). Annotators can reject clips if they are not comparable (different keysteps, unclear or short videos).

Annotation instructions and rubrics As mentioned in the main paper, annotators are presented with pairs of video clips from the same keystep category and asked to identify the main differences across 5 categories (ingredients, tools/equipment, techniques, visual differences) and then score how severe the differences per category are on a scale of 1-5. The annotation interface presented to the user is shown in Fig. S2. Scoring how severe the differences are is a fairly subjective task. To avoid ambiguity in this scoring, we present annotators with a scoring matrix (Fig. S3) that provides a rubric for scoring differences in each category. We conducted pilot experiments to calculate inter-annotator agreement. We found that two out of three annotators agree 82% of the time (Cohen's kappa = 0.64 on a [-1, 1] scale). Moreover, disagreements when present are small (on average within 1.2 points from each other).

Dataset statistics and analysis Overall, we collect 35,988 difference captions across 6,292 video clip pairs involving 8,396 unique video clips. Fig. S5 (left) shows the distribution

	5	4	3	2	1
Ingredients	No differences	• Substitution within the same category (e.g citrus fruits, Rock salt vs iodized) • Quantity/volume, shape, size, consistency	• 1-2 instances of Addition/Skipping of minor ingredients (e.g condiments)	• 3 or more instances of Addition / Skipping of minor ingredients (e.g condiments) • Substitution of completely 2 different items. (e.g pork to fish) • Addition/Skipping of major ingredients but all other elements are the same	-
Tools / Equipment	Color, size	• Substitution within the same category but serves the same function (e.g kitchenware: ordinary pan vs non-stick pan) • Substitution of tool used for the same purpose (e.g, Poured stock using a ladle instead of a measuring cup) • Difference in material (e.g metal vs wood)	• Replacement but the outcome will be the same (e.g., baking in microwave vs conventional oven) • 2-3 instances of tool substitution/material difference that serves the same purpose	• 4 or more instances of tool substitution / material difference that serves the same purpose	-
Techniques	No differences	• Procedural manner (e.g speed, frequency) • Slicing vs dicing similar ingredient	• 2- 3 instances of technique difference	• Difference in temperature / level of heat • Difference in method (e.g boiling vs poaching)	-
Action /steps	No difference	• Sequence is different but all the elements are the same	• Addition/Skipping of any steps related to the activity	• Difference of steps e.g., preparation of ingredients vs cooking or cooking vs serving • 2-3 additonal / skipped steps related to the activity • The sequence is different as well as the ingredients	4 or more additional / skipped steps
Visual differences	No difference	• Difference in color is just for small portion of the items • Difference in small details (e.g crisp sides of egg)	• Insignificant difference in shade (e.g, dark brown vs light brown sauce) • Significant difference in the size of portioned dishes like (cookies, dumplings, meatballs etc.)	Consistency, significant difference in color e.g, brown and white, red and blue	-

Figure S3. **Difference scoring matrix** Annotators score how severe the differences are on a scale of 1-5 (1 = very different; 5 = nearly identical) using the scoring matrix as reference to avoid ambiguity across annotators.

of difference captions collected over the five categories, with *Tools/Equipment* being the most popular category. There are fewer differences in *Actions* which involves variations in step order, however they still account for a significant proportion of annotated differences (12%). Fig. S5 (middle) shows the aggregate difference score for video pairs in the dataset, computed by averaging the difference score across all categories. While all clip pairs are expected to be similar overall by design, since they are paired together if they share the same step label (on average, this aggregate score is 3.9), they often have significant differences in one or more individual category. Fig. S5 (right) shows the distribution of difference scores only for categories where annotators label difference text, highlighting the spread in scores.

In Fig. S6, we show word clouds of prominent concepts captured in each difference category, sorted by their TF-IDF scores. We exclude words with a document frequency > 0.25 (e.g., person, instead, prefers etc.) to highlight category-specific concepts. We can see these concepts emerge for Tools/Equipment (e.g., materials, textures), Ingredients (e.g., ingredient names and properties), Visuals (e.g., visual attributes), Technique (e.g., motion-heavy words) and Actions (e.g., actions and verbs).

Examples of these annotations can be seen in Fig. 4 and Fig. S4. Note that none of these video clips are used in our automatic training data generation pipeline. These are a held-out subset of videos that are manually annotated for evaluation purposes only.

S3. Full implementation and training details

In this section, we present complete implementation details for our approach and all baselines listed in Sec. 4.

VCLM baselines As mentioned in Sec. 4 (baselines), we train our in-house VCLM and Interleaved baselines on clips from HowTo100M. To re-iterate, following prior work [31], M_V is an Internvideo [55] video encoder that inputs 8 uniformly sampled frames from each video clip and generates 2048 spatio-temporal tokens. M_{Proj} is a 2-layer Perceiver [15] module followed by a linear layer head to output 32 tokens in the LLM's input dimension. During training, all parameters are frozen except for M_{Proj}.

For the VCLM models, we extract (video, ASR) pairs from automatically aligned ASR data from prior work [13]. We use a batch size of 512 for 50k iterations. We use the AdamW optimizer, with a learning rate of 1e-4. For the Interleaved models, we sort (video, ASR) instances by their end timestamp and interleave sequences of 3 clips along with their ASR (clip1, ASR1, clip2, ASR2 ...). The Perceiver model converts each of the clips into 32 tokens. In addition to HowTo100M, we also train on single image captioning instances using filtered images from LAION2B [44] to improve the diversity of the training data beyond instructional video content. We duplicate the single image 8 times to feed to our video backbone. During training, we sample instances from each dataset in a round-robin manner. The batch size and number of iterations follow the VCLM models.

StepDiff training details As mentioned in Sec. 4 (implementation details), we initialize our models from the Interleaved checkpoints above. In addition to LAION and HT100M data, we also train on our generated PairQA data from Sec. 3.2. As before, we sample instances in a round-robin manner. We use a batch size of 256 for and train for 20k iterations based on validation data.

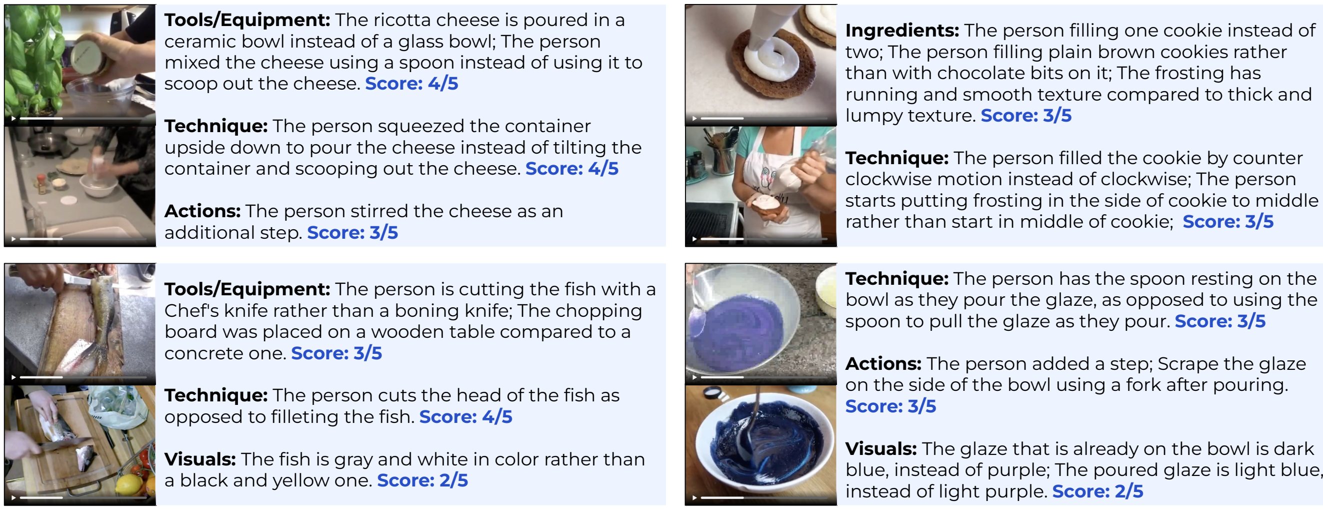

Figure S4. **Manually collected step differences.** Details in Sec. S2.

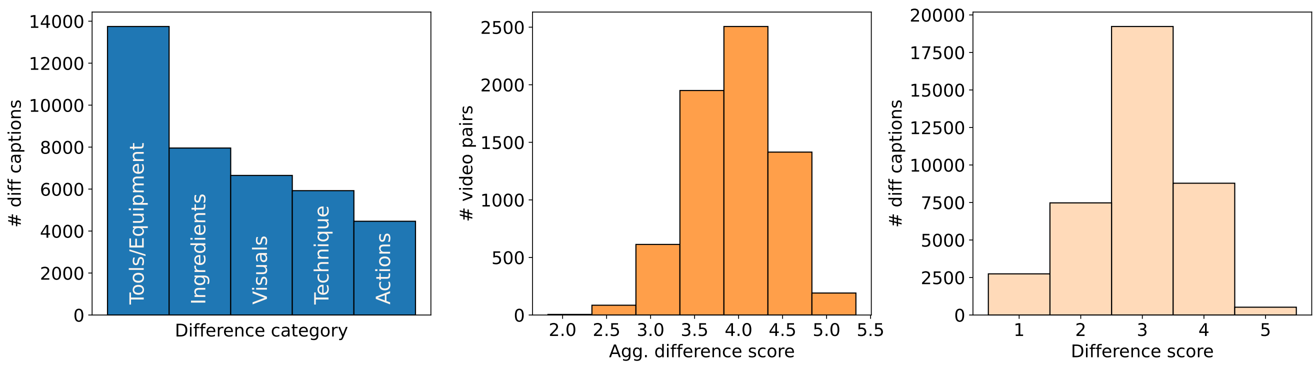

Figure S5. **Annotated data statistics. Left:** Distribution of difference captions by category. **Middle:** Aggregate difference score distribution for video pairs (averaged over categories). **Right:** Distribution of difference scores for categories that have annotated differences (1 = very different; 5 = nearly identical).

S4. Additional task formulation details

In Sec. 3.4, we described the prompts used for downstream tasks. To ensure that the outputs generated are in a consistent style with the collected annotations, we seed the generation step with partial text, and require the model to complete it. For DiffCap, we seed with "The main difference in category is that in Video 2,", and for DiffMCQ, we seed with "In Video 2," followed by the difference caption text that is being evaluated.

Additionally, as mentioned in Sec. 4.1, we post-process the outputs of each captioning baseline to match the annotated difference structure. This is important given the sensitivity of captioning metrics to even small structural changes. Even with careful prompting, the baselines tend to produce captions of the form "In Video 1/2, the person ..., while in Video 2/1, ...", while the annotations are collected in a specific format "action in candidate video compared to action in reference video" (see Fig. 4). The parsing involves simple text matching and replacing (e.g., replacing "whereas in Video 1, the person" with "instead of"). Note that all models benefit from the same partial completion and output post-processing strategies listed above to ensure fair comparison. In Table S1 we show results without any additional parsing. All methods perform considerably worse compared

	BLEU	CIDER	ROUGE-L
Socratic (BLIP-2) [22]	0.122	0.016	0.139
Socratic (LLaVA) [27]	0.117	0.015	0.135
Socratic (Step desc.)	0.113	0.009	0.139
VCLM (LLaVA) [27]	0.143	0.037	0.144
VCLM (AnyMAL) [31]	0.183	**0.079**	0.181
Interleaved (IDEFICS) [19]	0.156	0.041	0.160
Interleaved (AnyMAL)	0.184	0.068	0.185
StepDiff	**0.193**	0.061	**0.191**

Table S1. **DiffCap results without output parsing.** All methods perform worse on the generation metrics that are sensitive to sentence structure, though our method still has the best performance.

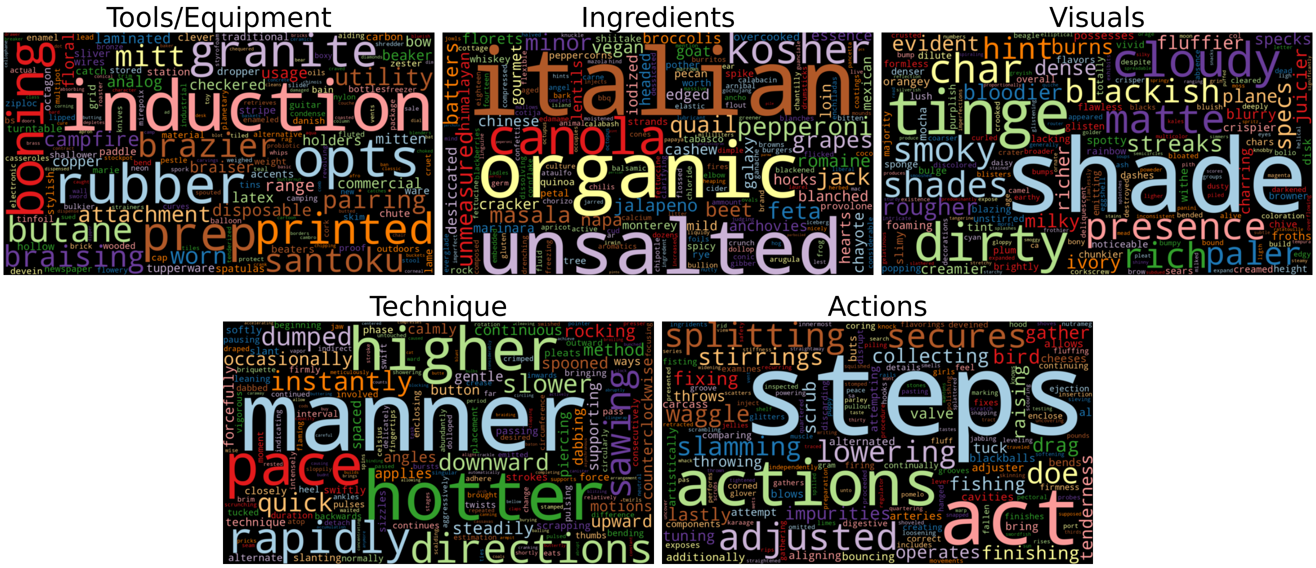

Figure S6. **Prominent concepts captured in difference captions per category.** Tools/Equipment features tool materials and attributes (e.g., rubber, granite, butane), while techniques feature motion-related words (e.g., rapidly, quick, slower).

	CLIP [40]	InternVideo [55]
V_r only	0.359	0.424
V_c only	0.353	0.413
$avg(V_r, V_c)$	**0.396**	**0.451**

Table S2. **VLEmbed variants.** Matching the difference caption to both the reference and the candidate video features results in the best performance.

to their counterparts with output parsing in Table 1 (left), however our approach still achieves the highest performance among them.

S5. Additional experiments

We present additional experiments to supplement the main paper results in Sec. 4.

Alternate variants of VLEmbed In our experiments, we assumed that the embeddings of a *pair of videos* can be represented as the average of their video embeddings. We evaluate other alternatives where a difference caption is matched to a single video (either the reference or the candidate) for DiffMCQ. Note that these variants are not applicable to DiffRank, where the difference caption is not an input. Our results in Table S2 show that including information from both video clips results in the best performance, though there is a small bias in the queries towards the reference video features.

Alternate variants of the DiffMCQ task As mentioned in Sec. 4.2, we construct the task from the DiffCap annotations by sampling three *negative* video pairs for every difference caption that are visually similar to the true video pair, but that do not exhibit the true difference. We identify the negatives as follows. First, we compute the average visual embedding (CLIP features) for each reference and candidate pair in the dataset, and sort the video pairs based on this distance to the positive pair embedding. Then, we go down this list and select pairs that obey two criteria: (1) they do not involve the true reference or candidate videos and (2) they do not share equivalent difference descriptions. For (2), we measure the sentence similarity between the ground truth difference and all of the differences for the selected pair in the category of interest, using MPNet [47] embeddings. If any difference text is too similar (above a threshold of 0.8 cosine similarity), then we ignore the pair. We continue this process until we collect three negatives.

Note that this is not the only method to construct the DiffMCQ task. For example, we can sample video pairs regardless of whether they share a reference or candidate video (as long as they are not the exact same pair). This results in a more difficult variant of DiffMCQ, but runs the risk of selecting negatives that may share differences. A third alternative is to fix either the reference or candidate clip and randomly sample the other, regardless of visual similarity or difference text similarity. We present all three alternatives in Table S4. Across the first two variants, our approach outperforms baselines. In the third alternative, the second clip is selected randomly, and so the VLEmbed baselines are sufficient for identifying outliers, and all baselines perform similarly. Moreover, the lack of constraints may permit negatives that still match the difference caption, making this version unsuitable for benchmarking our models.

	DiffCap			DiffMCQ	DiffRank
	BLEU	CIDER	ROGUE-L	Acc %	τ
Socratic (BLIP-2) [22]	0.164	0.035	0.174	0.341	0.000
Socratic (LLaVA) [27]	0.155	0.027	0.169	0.332	0.000
Socratic (Step desc.)	0.138	0.019	0.169	0.400	0.006
VCLM (LLaVA) [27]	**0.235**	0.072	0.199	0.385	0.009
VCLM (AnyMAL) [31]	0.193	0.106	0.196	0.496	0.041
Interleaved (IDEFICS) [19]	0.187	0.058	0.189	0.340	0.022
Interleaved (AnyMAL)	0.221	0.105	**0.216**	0.475	0.048
StepDiff	0.216	**0.124**	0.205	**0.527**	**0.175**

Table S3. **Results with lower capacity models.** Socratic (Llama 13B), AnyMAL (13B), LLaVA (7B) and IDEFICS (9B). Smaller models perform reasonably on the captioning task, but under-perform on the discriminative and ranking tasks.

	V1	V2	V3
VLEmbed (CLIP) [40]	0.396	0.311	0.657
VLEmbed (InternVideo) [55]	0.451	0.336	**0.683**
Socratic (BLIP-2) [22]	0.335	0.219	0.644
Socratic (LLaVA) [27]	0.332	0.217	0.646
Socratic (Step desc.)	0.392	0.258	0.648
VCLM (LLaVA) [27]	0.381	0.319	0.561
VCLM (AnyMAL) [31]	0.471	0.344	0.648
Interleaved (IDEFICS) [19]	0.376	0.304	0.638
Interleaved (AnyMAL)	0.497	0.351	0.653
StepDiff	**0.541**	**0.382**	0.654

Table S4. **DiffMCQ variants for selecting negatives.** V1 excludes negatives that share the true reference or candidate video clip. This is the version reported in Table 1. V2 permits overlaps in reference / candidate clips as long as the pair is not identical. V3 fixes either the reference or candidate clip and randomly selects the other.

Ablation experiments with lower capacity baselines In Sec. 4.5 of the main paper, we presented our method with a 13B parameter LLM backbone. In Table S3, we show results of all baseline models with smaller variants, including Socratic (LLama-13B), AnyMAL-13B, LLaVA-7B, and IDEFICS-9B. Our results show that while smaller capacity models perform reasonably well in the captioning task (even outperforming their 70B model alternatives on the BLEU metric), they perform worse overall on the discriminative and ranking tasks.

S6. Additional qualitative results

We show additional qualitative samples of our method's outputs in Fig. S7. We show various kinds of supported prompts. These are standard difference captioning used to evaluate our models (panel 1), comparative reasoning (panel 2) and mistake reasoning (panel 3). Panel 4 highlights some failure cases. These typically arise due to two reasons. First, the underlying LLM naturally hallucinates details that are not present. This can happen due to inaccurate recognition (e.g., identifying a bell pepper as a jalapeno), or incomplete context information (e.g., without knowing the full recipe, the model assumes the dish is a dessert and the white powder is sugar). The second failure mode occurs when the model is forced to produce an output when differences in that category do not necessarily occur. This forces the model to hallucinate details as it is not trained to reject a query (e.g., asking "what mistake did I make" in the last row). More diverse automatically generated training data that explicitly handles these situations will likely address these failure modes. Despite these limitations, our approach can answer a wide variety of questions and requires reasoning over multiple videos, as shown in the figure.

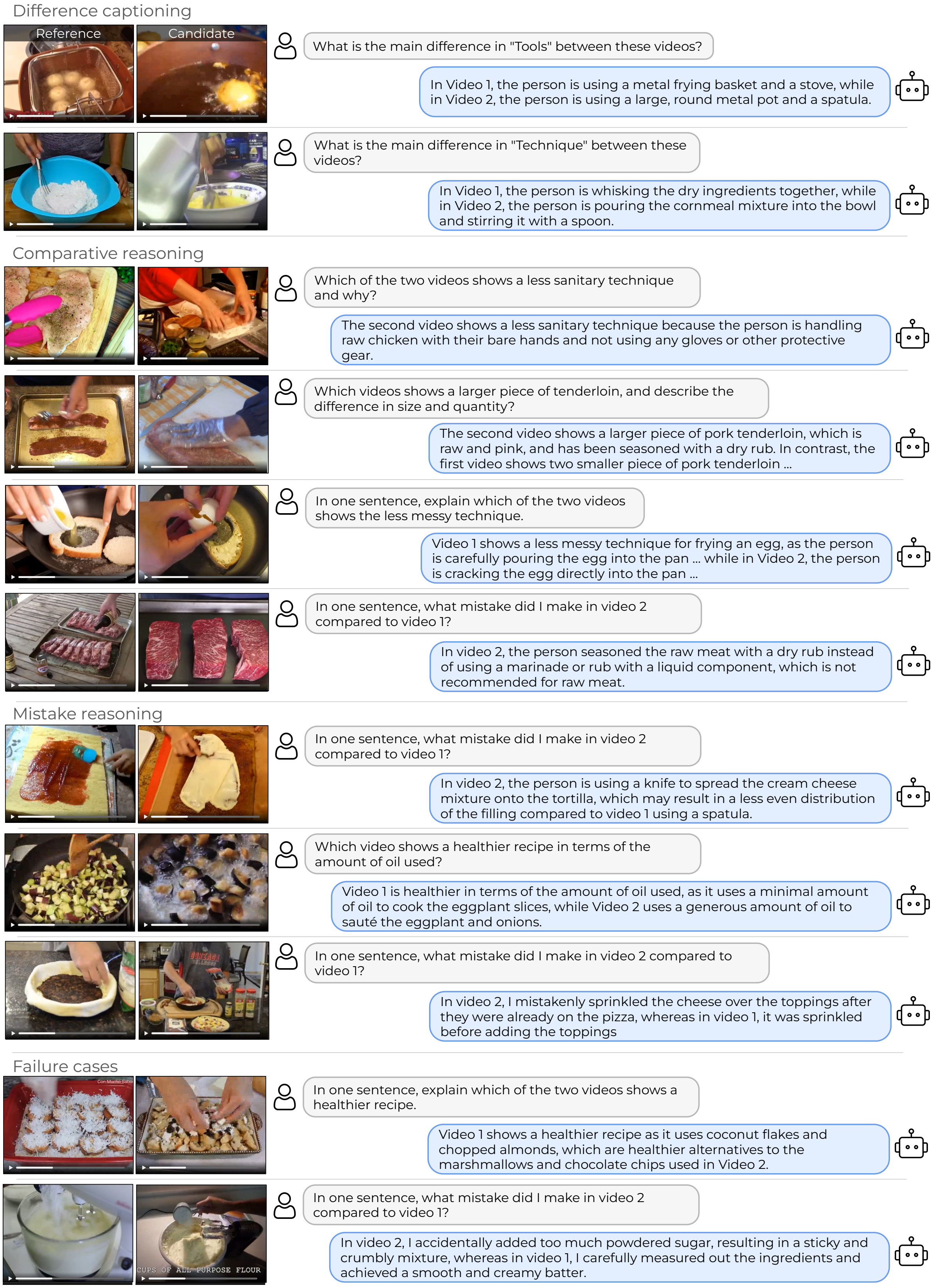

Figure S7. **Additional QA results on video pairs** See Sec. S6 for discussion. Failure cases are shown in the last two rows.