A Benchmark for Structured Procedural Knowledge Extraction from Cooking Videos

Anonymous authors

Abstract

Procedural knowledge, which we define as concrete information about the sequence of actions that go into performing a particular procedure, plays an important role in understanding real world tasks and actions. Humans often learn this knowledge from instructional text and video, and in this paper we aim to perform automatic extraction of this knowledge in a similar way. As a concrete step in this direction, we propose the new task of inferring procedures in a *structured form* (a data structure containing verbs and arguments) from *multimodal* instructional video contents and their corresponding transcripts. We first create a manually annotated, large evaluation dataset including over 350 instructional cooking videos along with over 15,000 English sentences in transcripts spanning 89 recipes. We conduct analysis of the challenges posed by this task and dataset with experiments with unsupervised segmentation, semantic role labeling, and visual action detection based baselines. The dataset and code will be publicly available at https://anonymous.blind.

1. Introduction

Instructional videos are a convenient way to learn a new skill. Although learning from video seems natural to humans, it requires identifying and understanding procedures and grounding them to the real world. In this paper, we propose a new task and dataset for extracting procedural knowledge into a fine-grained *structured representation* from *multimodal* information contained in a *large-scale* archive of *open-domain* narrative videos with transcripts. While there is a significant amount of related work (summarized in Section 7), to our knowledge there is no dataset similar in scope, with previous attempts focusing only on a single modality (e.g. text only [Kiddon et al., 2015] or video only [Zhukov et al., 2019, Alayrac et al., 2016]), using closed-domain taxonomies [Tang et al., 2019], or lacking structure in the procedural representation [Zhou et al., 2018a].

In our task, given a narrative video, say a cooking video on YouTube about *making clam chowder* as shown in Figure 1, our goal is to extract a series of tuples representing the procedure, e.g. (heat, cast iron skillet), (fry, bacon, with heated skillet), etc. We created a manually annotated, large dataset for evaluation of the task, including over 350 instructional cooking videos along with over 15,000 English sentences in the transcripts spanning over 89 recipe types. This verb-argument structure using arbitrary textual phrases is motivated by open information extraction [Schmitz et al., 2012, Fader et al., 2011], but focuses on procedures rather than entity-entity relations.
This task is challenging with respect to both video and language understanding. For video, it requires understanding of video contents, with a special focus on actions and procedures. For language, it requires understanding of oral narratives, including understanding of predicate-argument structure and coreference. In many cases it is necessary for both modalities to work together, such as when resolving null arguments necessitates the use of objects or actions detected from video contents in addition to transcripts. For example, the cooking video host may say “just a pinch of salt in”, while adding some salt into a boiling pot of soup, in which case inferring the action “add” and its argument “pot” requires visual understanding.

Along with the novel task and dataset, we propose several baseline approaches that extract structure in a pipelined fashion. These methods first identify key clips/sentences using video and transcript information with unsupervised and supervised multimodal methods, then extract procedure tuples from the utterances and/or video of these key clips. On the utterances side, we utilize an existing state-of-the-art semantic role labeling model [Shi and Lin, 2019], with the intuition that semantic role labeling captures the verb-argument structures of a sentence, which would be directly related to procedures and actions. On the video side, similarly, we utilize existing state-of-the-art video action/object recognition model trained in kitchen settings to further augment utterances-only extraction results. The results are far from perfect, demonstrating that the proposed task is challenging and that structuring procedures requires more than just state-of-the-art semantic parsing or video action recognition.

2. Problem Definition

We show a concrete example of our procedural knowledge extraction task in Figure 1. Our ultimate goal is to automatically map unstructured instructional video (clip and utterances) to structured procedures, defining what actions should be performed on which objects, with what arguments and in what order. We define the input to such an extraction system:

- Task $R$, e.g. “Create Chicken Parmesan” and instructional video $V_R$ describing the procedure to achieve task $R$, e.g. a video titled “Chicken Parmesan - Let’s Cook with ModernMom”.\(^1\)
- A sequence of $n$ sentences $T_R = \{t_0, t_1, \ldots, t_n\}$ representing video $V_R$’s corresponding transcript. According to the time stamps of the transcript sentences, the video is also segmented into $n$ clips $V_R = \{v_0, v_1, \ldots, v_n\}$ accordingly to align with the sentences in the transcript $T_R$.

The output will be:

- A sequence of $m$ procedure tuples $S_R = \{s_0, s_1, \ldots, s_m\}$ describing the key steps to achieve task $R$ according to instructional video $V_R$.
- An identified list of key video clips and corresponding sentences $V'_R \subseteq V_R$, to which procedures in $S_R$ are grounded.

\(^1\)https://www.youtube.com/watch?v=nWGpCmD1NU4
Each procedural tuple \( s_j = (\text{verb}, \text{arg}_1, \ldots, \text{arg}_k) \in S_R \) consists of a verb phrase and its arguments. Only the “verb” field is required, and thus the tuple size ranges from 1 to \( k + 1 \). All fields can be either a word or a phrase.

Not every clip/sentence describes procedures, as most videos include an intro, an outro, non-procedural narration, or off-topic chit-chat. Key clips \( V'_R \) are clips associated with one or more procedures in \( P_R \), with some clips/sentences associated with multiple procedure tuples. Conversely, each procedure tuple will be associated with only a single clip/sentence.

3. Dataset & Analysis

While others have looked at creating related datasets, they fall short on key dimensions which we remedy in our work. Specifically, In Table 3 we compare to AllRecipes [Kiddon et al., 2015] (AR), YouCook2 [Zhou et al., 2018b] (YC2), CrossTask [Zhukov et al., 2019] (CT), COIN [Tang et al., 2019], How2 [Sanabria et al., 2018], HAKE [Li et al., 2019] and TACOS [Regneri et al., 2013]. Additional details about all datasets are included in the Appendix A. In summary, none have both structured and open extraction annotations for the procedural knowledge extraction task, since most focus on either video summarization/captioning or action localization/classification tasks.

3.1 Dataset Creation

To address the limitations of existing datasets, we created our own evaluation dataset by annotating structured procedure knowledge given the video and transcript. Native English-speakers annotated four videos per recipe type (e.g. clam chowder, pizza margherita, etc.) in the YouCook2 dataset into the structured form presented in §2 (totaling 356 videos). Annotators selected key clips as important steps and extracted corresponding fields to fill in verbs and arguments. Filling in the fields with the original tokens was preferred but not required (e.g. in cases of coreference and ellipsis). The result is a series of video clips labeled with procedural structured knowledge as a sequence of steps \( s_j \) and series of short sentences describing the procedure.

Figure 2 shows the user interface of annotation tool. The process is divided into 3 questions per clip: 

\( Q1 \): Determine if the video clip is a key step if: (1) the clip or transcript contains at least one action; (2) the action is required for accomplishing the task (i.e. not a self introduction); and (3) for if a clip duplicates a previous key clip, choose the one with clearer visual and textual sig-

<table>
<thead>
<tr>
<th>General domain?</th>
<th>✓</th>
<th>✓</th>
<th>✓</th>
<th>✓</th>
<th>✓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multimodal input?</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Use transcript?</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Use noisy text?</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Open extraction?</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Structured format?</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 1: Comparison to current datasets.

<table>
<thead>
<tr>
<th>Ours</th>
<th>AR</th>
<th>YC2</th>
<th>CT</th>
<th>COIN</th>
<th>How2</th>
<th>HAKE</th>
<th>TACOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verbs</td>
<td>Arguments</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total #</td>
<td>4004</td>
<td>7007</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average # per key clip</td>
<td>1.12</td>
<td>1.70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average #words</td>
<td>1.07</td>
<td>1.43</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>% directly from transcript</td>
<td>69.8</td>
<td>75.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>% coreference (pronouns)</td>
<td>N/A</td>
<td>14.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>% ellipsis</td>
<td>30.2</td>
<td>10.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Statistics of annotated verbs and arguments in procedures.

2. A common dataset we do not include here is HowTo100M [Miech et al., 2019] as it does not contain any annotations.
Table 3: Annotations of structured procedures and summaries. Coreference and ellipsis are marked with italics and are resolved into referred phrases also linked back in the annotations.

Q2: For each key video clip, annotate the key procedural tuples. We have annotators indicate which actions are both seen and mentioned by the instructor in the video. The actions should correspond to a verb and its arguments from the original transcript except in the case of ellipsis or coreference where they have to refer to earlier phrases based on the visual scene. Q3: Construct a short fluent sentence from the annotated tuples for the given video clip.

We have two expert annotators and a professional labeling supervisor for quality control and deciding the final annotations. To improve the data quality, the supervisor reviewed all labeling results, and applied several heuristic rules to find anomalous records for further correction. The heuristic is to check the annotated verb/arguments that are not found in corresponding transcript text. Among these anomalies, the supervisor checks the conflicts between the two annotators. 25% of all annotations were modified as a result. On average annotators completed task Q1 at 240 sentences (clips) per hour and task Q2 and Q3 combined at 40 sentences per hour. For Q1, we observe an inter-annotator agreement with Cohen’s Kappa of 0.83. Examples are shown in Table 3.

3.2 Dataset Analysis

Overall, the dataset contains 356 videos with 15,523 video clips/sentences, among which 3,569 clips are labeled as key steps. Sentences average 16.3 tokens, and the language style is oral English. For structured procedural annotations, there are 347 unique verbs and 1,237 unique objects in all. Statistics are shown in Table 2. Figure 3 lists the most commonly appearing verbs and entities. The action add is most frequently performed, and the entities salt and onions are the most popular ingredients.

In nearly 30% of annotations, some verbs and arguments cannot be directly found in the transcript. An example is “(add) some salt into the pot”, and we refer to this variety of absence as ellipsis. Arguments not mentioned explicitly are mainly due to (1) pronoun references, e.g. “put it (fish) in the pan”; (2) ellipsis, where the arguments are absent from the oral language, e.g. “put the mixture inside” where the argument “oven” is omitted. The details can be found in Table 2. The coreferences and ellipsis phenomena add difficulty to our task, and indicate the utility of using multimodal information from the video signal and contextual procedural knowledge for inference.

4. Extraction Stage 1: Key Clip Selection

3. We use the Jaccard ratio between the annotated tokens of two annotators for Q2’s agreement. Verb annotations have a higher agreement at 0.77 than that of arguments at 0.72.
In this and the following section, we describe our two-step pipeline for procedural knowledge extraction (also in Figure 4). This section describes the first stage of determining which clips are “key clips” that contribute to the description of the procedure. We describe several key clip selection models, which consume the transcript and/or the video within the clip and decide whether it is a key clip or not.

4.1 Parsing-Based Heuristic Baselines

Given our unsupervised setting, we first examine two heuristic parsing-based methods that focus on the transcript only, one based on semantic role labeling (SRL) and the other based on an unsupervised segmentation model Kiddon et al. [2015].

Before introducing heuristic baselines, we note that having a lexicon of domain-specific actions will be useful, e.g. for filtering pretrained model outputs, or providing priors to the unsupervised model described later. In our cooking domain, these actions can be expected to consist mostly of verbs related to cooking actions and procedures. Observing recipe datasets such as AllRecipes [Kiddon et al., 2015] or WikiHow [Miech et al., 2019, Zhukov et al., 2019], we find that they usually use imperative and concise sentences for procedures and the first word is usually the action verb like “add”, e.g. add some salt into the pot. We thus construct a cooking lexicon by aggregating the frequently appearing verbs as the first word from AllRecipes, with frequency over a threshold of 5. We further filter out words that have no verb synsets in WordNet [Miller, 1995]. Finally we manually filter out noisy or too general verbs like “go”. Note that when applying to other domains, the lexicon can be built following a similar process of first finding a domain-specific corpus with simple and formal instructions, and then obtaining the lexicon by aggregation and filtering.

Semantic role labeling baselines. One intuitive trigger in the transcript for deciding whether the sentence is a key step should be the action words, i.e. the verbs. In order to identify these action words we use semantic role labelling [Gildea and Jurafsky, 2002], which analyzes natural language sentences to extract information about “who did what to whom, when, where and how?” The output is in the form of predicates and their respective arguments that acts as semantic roles, where the verb acts as the root (head) of the parse. We run a strong semantic role labeling model [Shi and Lin, 2019] included in the AllenNLP toolkit [Gardner et al., 2018] on each sentence in the transcript. From the output we get a set of verbs for each of the sentences. Because not all verbs in all sentences represent actual key actions for the procedure, we additionally filter the verbs with the heuristically created cooking lexicon above, counting a clip as a key clip only if at least one of the SRL-detected verbs is included in the lexicon.

Unsupervised recipe segmentation baseline [Kiddon et al., 2015]. The second baseline is based on the outputs of the unsupervised recipe sentence segmentation model in Kiddon et al. [2015]. Briefly speaking, the model is a generative probabilistic model where verbs and arguments, together with their numbers, are modeled as latent variables. It uses a bigram model for string selection. It is trained on the whole transcript corpus of YouCook2 videos iteratively for 15 epochs using a hard EM approach before the performance starts to converge. The count of verbs in the lexicon created

---

4. https://demo.allennlp.org/semantic-role-labeling

5. The SRL model is used in this stage only as a verb identifier, with other output information used in stage 2.
in §4.1 is provided as a prior through initialization. We then do inference to parse the transcripts in our dataset using the trained model. Following the same heuristics as the SRL outputs, we treat sentences with non-empty parsed predicates after lexical filtering as key sentences, and those without as negatives.

4.2 Neural Selection Baseline

Next, we implement a supervised neural network based model that incorporates visual information, which we have posited before may be useful in the face of incomplete verbal utterances. We first extract the features of the sentence and each video frame using pretrained feature extractors respectively. Then we perform attention [Bahdanau et al., 2014] over each frame feature, using the sentence as a query, in order to acquire the representation of the video clip. Finally, we combine the visual and textual features to predict whether the input is a key clip. The model is trained on another general domain instructional key clip selection dataset with no overlap with ours, and our annotated dataset is used for evaluation only. Additional details about the model and training dataset are included in the Appendix B.

5. Extraction Stage 2: Structured Knowledge Extraction

With the identified key clips and corresponding transcript sentences, we proceed to the second stage that performs clip/sentence-level procedural knowledge extraction from key clips. In this stage, the extraction is done from clips that are identified at first as “key clips”.

5.1 Extraction From Utterances

We first present two baselines to extract structured procedures using transcripts only, similarly to the key-clip identification methods described in §4.1.

Semantic role labeling. For the first baseline, we use the same pretrained SRL model introduced in §4.1 to conduct inference on the sentences in key clips identified from stage 1. Because they consist of verb-argument structures, the outputs of the SRL model are well aligned with the task of extracting procedural tuples that identify actions and their arguments. However, not all outputs from the SRL model are the structured procedural knowledge we aim to extract. For example, in the sentence “you ’re ready to add a variety of bell peppers” from the transcript, the outputs from SRL model contains two parses with two predicates, “are” and “add”, where only the latter is actually part of the procedure. To deal with this issue we first perform filtering similar to that used in stage 1, removing parses with predicates (verbs) outside of the domain-specific action lexicon we created in §4.1. Next, we filter out irrelevant arguments in the parse. For example, the parse from the SRL model for sentence “I add a lot of pepper because I love it.” after filtering out irrelevant verb “love” is “[ARG0: I] [V: add] [ARG1: a lot of pepper] [ARGM–CAU: because I love it]”, some arguments such as ARG0 and ARGM–CAU are clearly not contributing to the procedure. We provide a complete list of the filtered argument types in Appendix C.

Unsupervised recipe segmentation [Kiddon et al., 2015]. The second baseline is to use the same trained segmentation model as in §4.1 to segment selected key transcript sentences into verbs and arguments. We treat segmented predicates in the key sentence as procedural verbs, and segmented predicate arguments plus preposition arguments as procedural arguments.
5.2 Extraction From Video

We also examine a baseline that utilizes two forms of visual information in videos: actions and objects. We predict both verbs and nouns of a given video clip via a state-of-the-art action detection model TSM [Lin et al., 2019], trained on the EpicKitchen [Damen et al., 2018a] dataset. For each video, we extract 5-sec video segments and feed into the action detection model. The outputs of the models are in a predefined set of labels of verbs (actions) and nouns (objects). We directly combine the outputs from the model on each video segment, aggregate and align them with key clips/sentences through timestamps in the video, forming the final output.

5.3 Utterance and Video Fusion

Finally, to take advantage of the fact that utterance and video provide complementary views, we perform multimodal fusion of the results of both of these model varieties. We first adopt a naive method of fusion by taking the union of result sets from best performing utterance-only model and visual detection model. However, we found in evaluations that this degrades the performance, partly due to the differences in video data distribution and domain, as well as the limitation of the predefined set of verbs and nouns in the EpicKitchen dataset. To tackle the limitation of the label set, we compare an “oracle” version by first expanding the predefined verbs and nouns in the EpicKitchen dataset with synonyms and 1-hop siblings with synsets in WordNet. With these, the visual detection results are expanded as above and we filter them with the ground truth annotations (oracle) before they are combined with utterance model predictions.

6. Evaluation

We provide evaluation results on our annotated dataset for both of the two stages: key clip selection and structured procedural extraction. Besides quantitative evaluation and qualitative evaluations, we also analyze the key challenges of this task.

6.1 Extraction Stage 1: Key Clip Selection

In this section, we evaluate results of the key clip selection described in §4. We evaluate using the accuracy, precision, recall and F1 score for the binary classification problem of whether a given clip in the video is a key clip. The results are shown in Table 4. We compare parsing-based heuristic models and supervised neural models, with ablations (model details in Appendix B).

From the experimental results in Table 4, we can see that:

1. Unsupervised heuristic methods perform worse than neural models with training data. This is despite the fact that the dataset used for training neural models has a different data distribution and domain from the test set.
2. Among heuristic methods, pretrained SRL is better than Kiddon et al. [2015] even though the second is trained on transcript text from YouCook2 videos. One possible reason for this is that

<table>
<thead>
<tr>
<th>Model</th>
<th>Acc</th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parsing-based Heuristics</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRL, w/o heur.</td>
<td>25.9</td>
<td>23.4</td>
<td>97.6</td>
<td>37.7</td>
</tr>
<tr>
<td>SRL, w/ heur.</td>
<td>61.2</td>
<td>35.2</td>
<td>81.4</td>
<td>49.1</td>
</tr>
<tr>
<td>Kiddon et al. [2015]</td>
<td>67.3</td>
<td>33.5</td>
<td>42.7</td>
<td>37.6</td>
</tr>
<tr>
<td>Neural Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Visual Only</td>
<td>43.8</td>
<td>27.2</td>
<td>81.0</td>
<td>41.3</td>
</tr>
<tr>
<td>Text Only</td>
<td>76.3</td>
<td>49.0</td>
<td>73.9</td>
<td>60.2</td>
</tr>
<tr>
<td>V+T (Full Model)</td>
<td>77.7</td>
<td>51.0</td>
<td>63.2</td>
<td>60.8</td>
</tr>
</tbody>
</table>

Table 4: Key clip selection results.

7. Notably, this contrasts to our setting of attempting to recognize into an open label set, which upper-bounds the accuracy of any model with a limited label set.
### Table 5: Clip/sentence-level structured procedure extraction results for verbs and arguments.

<table>
<thead>
<tr>
<th>Model</th>
<th>Verbs Exact Match</th>
<th>Verbs Fuzzy</th>
<th>Verbs Partial Fuzzy</th>
<th>Arguments Exact Match</th>
<th>Arguments Fuzzy</th>
<th>Arguments Partial Fuzzy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F1</td>
<td>P</td>
<td>R</td>
<td>F1</td>
</tr>
<tr>
<td>Kiddon et al. [2015]</td>
<td>12.0</td>
<td>10.9</td>
<td>11.4</td>
<td>18.8</td>
<td>17.2</td>
<td>18.0</td>
</tr>
<tr>
<td>SRL w/o heur.</td>
<td>19.4</td>
<td>18.0</td>
<td>18.4</td>
<td>25.3</td>
<td>20.1</td>
<td>23.0</td>
</tr>
<tr>
<td>SRL w/ heur.</td>
<td>38.7</td>
<td>35.7</td>
<td>37.3</td>
<td>45.2</td>
<td>41.6</td>
<td>43.3</td>
</tr>
<tr>
<td>Visual</td>
<td>4.1</td>
<td>3.8</td>
<td>4.0</td>
<td>17.9</td>
<td>17.3</td>
<td>17.6</td>
</tr>
<tr>
<td>Fusion</td>
<td>19.9</td>
<td>18.9</td>
<td>18.9</td>
<td>27.6</td>
<td>25.9</td>
<td>26.2</td>
</tr>
<tr>
<td>Oracle Fusion</td>
<td>42.1</td>
<td>40.9</td>
<td>41.3</td>
<td>47.9</td>
<td>45.3</td>
<td>46.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>R</th>
<th>F1</th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
</table>

Using oracle key clips

<table>
<thead>
<tr>
<th>Model</th>
<th>Verbs</th>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>Kiddon et al. [2015]</td>
<td>7.0</td>
<td>6.3</td>
</tr>
<tr>
<td>SRL w/o heur.</td>
<td>11.2</td>
<td>11.7</td>
</tr>
<tr>
<td>SRL w/ heur.</td>
<td>22.5</td>
<td>22.9</td>
</tr>
<tr>
<td>Visual</td>
<td>2.4</td>
<td>2.9</td>
</tr>
<tr>
<td>Fusion</td>
<td>11.5</td>
<td>12.0</td>
</tr>
<tr>
<td>Oracle Fusion</td>
<td>24.4</td>
<td>24.7</td>
</tr>
</tbody>
</table>

Using predicted key clips

---

3. Post-processing significantly improves the SRL model, showing that filtering unrelated arguments and incorporating the cooking lexicon helps, especially with reducing false positives.

4. Among neural method ablations, the model using only visual features performs worse than that using only text features. The best model for identifying key clips among proposed baselines uses both visual and text information in the neural model.

Besides quantitative evaluation, we analyzed key clip identification results and found a number of observations. First, background introductions, advertisements for the YouTube channel, etc. can be relatively well classified due to major differences both visually and textually from procedural clips. Second, alignment and grounding between the visual and textual domains is crucial for key clip prediction, yet challenging. For example, the clip with the transcript sentence “add more pepper according to your liking” is identified as a key clip. However, it is in fact merely a suggestion made by the speaker about an imaginary scenario, rather than a real action performed and thus should not be regarded as a key procedure.

### 6.2 Extraction Stage 2: Structured Procedure Extraction

In this stage, we perform key clip-level evaluation for structured procedural knowledge extraction by matching the ground truth and predicted structures with both exact match and two fuzzy scoring strategies. To better show how stage 1 performance affects the whole pipeline, we evaluate on both ground truth (oracle) and predicted key clips. Similarly to the evaluation of key clip selection, we compare the parsing-based methods ($\S$ 5.1), as well as purposing the action detection results from video signals for our task. Besides, we compare utterance-only and video-only baselines with our naive multi-modal fusion method. We also compare given a oracle during fusion for filtering, how much will incorporating visual signal help to provide reference for better multimodal fusion in the future.

We evaluate with respect to precision, recall and the F1 measure. Similarly to the evaluation method used for SRL [Carreras and Márquez, 2004], precision ($P$) is the proportion of verbs or arguments predicted by a model which are correct, i.e. $TP/#predicted$ where $TP$ is the number of
true positives. Recall (R) is the proportion of correct verbs or arguments which are predicted by a model, i.e. $TP/\#\text{gold}$. The key here is how to calculate $TP$ and we propose 3 methods to calculate them: exact match, fuzzy matching, and partial fuzzy matching. The first is straight forward, we count true positives if and only if the predicted phrase is an exact string match in the gold phrases. However, because our task lies in the realm of open phrase extraction without predefined labels, it is unfairly strict to count only the exact string matches as $TP$. Also by design, the gold extraction results cannot always be found in the original transcript sentence (refer to §3.2), so we are also unable to use token-based metrics as in sequence tagging [Sang and De Meulder, 2003], or span-based metrics as in some question answering tasks [Rajpurkar et al., 2016]. Thus for the second metric we call “fuzzy”, we leverage edit distance to enable fuzzy matching and assign a “soft” score for $TP$. In some cases, the two strings of quite different lengths will hurt the fuzzy score due to the nature of edit distance, even though one string is a substring of another. To get around this, we propose a third metric, “partial fuzzy” to get the score of the best matching substring with the length of the shorter string in comparison (see ). Note that this third metric will bias towards shorter, correct phrases and thus we should have a holistic view of all 3 metrics during the evaluation. Details of two fuzzy metrics are described in Appendix D. Table 5 illustrates evaluation results:

1. Argument extraction is much more challenging compared to verb extraction, according the results: arguments contain more complex types of phrases (e.g. objects, location, time, etc.) and are longer in length. It is hard to identify complex arguments with our current heuristic or unsupervised baselines and thus the need for better supervised or semi-supervised models.

2. Heuristic SRL methods perform better than the unsupervised segmentation model even though it is trained on our corpus. This demonstrates the generality of SRL models, but the heuristics applied at the output of SRL models still improve the performance by reducing false positives.

3. The visual-only method performs the worst, mainly because of the domain gap between visual detection model outputs and our annotated verbs and arguments. Other reasons include: the closed label set predefined in EpicKitchen; challenges in domain transferring from closed to open extraction; different video data distribution between EpicKitchen (for training) and our dataset (YouCook2, for testing); limited performance of video detection model itself.

4. Naive multimodal fusion leads to a performance drop to below the utterance-only model. However, filtering visual outputs with the oracle annotations before merging with the utterance-only output outperforms single-modality models. This indicates a path forward for fusion strategies, though this is not sufficient for handling the complexity of arguments. To get a phrase for open extraction, we need more than just object detection.

There are two key challenges we see moving forward:

**Verb extraction**: We find that verb ellipsis is common in transcripts. The transcript text contains sentences where key action “verbs” do not have verb part-of-speech in the sentence. For example, in the sentence “give it a flip ...” with the annotation (“flip”, “pancake”), the model detects “give” as the verb rather than “flip”. Currently all our baselines are highly reliant on a curated lexicon for verb selection and thus such cases will get filtered out. How to deal with such cases with general verbs like make, give, do remains challenging and requires extracting from the contexts.

**Argument Extraction**: Speech-to-text errors are intrinsic in automatically acquired transcripts and cause problems during parsing that cascade. Examples are that “add flour” being recognized as “add flower” and “sriracha sauce” being recognized as “sarrah cha sauce” causing wrong extraction outputs. Coreference and ellipsis are also challenging and hurting current benchmark performance,
as our baselines do not tackle any of these explicitly. Visual co-reference and language grounding [Huang et al., 2018, 2017] provides a feasible method for us to tackle these cases in the future.

7. Related Work

Text-based procedural knowledge extraction. Procedural text understanding and knowledge extraction [Chu et al., 2017, Park and Motahari Nezhad, 2018, Kiddon et al., 2015, Jermsurawong and Habash, 2015, Liu et al., 2016, Long et al., 2016, Maeta et al., 2015, Malmaud et al., 2014, Artzi and Zettlemoyer, 2013, Kuehne et al., 2017] has been studied for years on step-wise textual data such as WikiHow. Chu et al. [2017] extracted open-domain knowledge from how-to communities. Recently Zhukov et al. [2019] also studied to adopt the well-written how-to data as weak supervision for instructional video understanding. Unlike existing work on action graph/dependency extraction [Kiddon et al., 2015, Jermsurawong and Habash, 2015], our approach differs as we extract knowledge from the visual signals and transcripts directly, not from formal imperative recipe texts.

Instructional video understanding. Unlike existing tasks for learning from instructional video [Zhou et al., 2018c, Tang et al., 2019, Alayrac et al., 2016, Song et al., 2015, Sener et al., 2015, Huang et al., 2016, Sun et al., 2019b,a, Plummer et al., 2017, Shi et al., 2019, Palaskar et al., 2019], visual-linguistic reference resolution [Huang et al., 2018, 2017], visual planning [Chang et al., 2019], joint learning of object and actions [Zhukov et al., 2019, Richard et al., 2018, Gao et al., 2017, Damen et al., 2018b], pretraining joint embedding of high level sentence with video clips [Sun et al., 2019b, Miech et al., 2019], and multimodal reading comprehension with RecipeQA [Yagcioglu et al., 2018], our task proposal requires explicit structured knowledge extraction.

Visual procedure learning. In addition to closely related work (§3) there is a wide literature [Zhou et al., 2018b,c, Alayrac et al., 2016, Ushiku et al., 2017, Nishimura et al., 2019, Tang et al., 2019, Alayrac et al., 2016, Huang et al., 2016, Shi et al., 2019, Ushiku et al., 2017] that aims to predict dense procedural captions given the video, which are the most similar works to ours. Zhou et al. [2018c] extracted temporal procedures and then generated captioning for each procedure. Sanabria et al. [2018] proposes a multimodal abstractive summarization for how-to videos with either human labeled or speech-to-text transcript. Alayrac et al. [2016] also introduces an unsupervised step learning method from instructional videos. Inspired by cross-task sharing [Zhukov et al., 2019], which is a weakly supervised method to learn shared actions between tasks, fine grained action and entity are important for sharing similar knowledge between various tasks. We focus on structured knowledge of fine-grained actions and entities. Visual-linguistic coreference resolution [Huang et al., 2018, 2017] is among one of the open challenges for our proposed task.

8. Conclusions & Open Challenges

We propose a multimodal open procedural knowledge extraction task, present a new evaluation dataset, produce benchmarks with various methods, and analyze the difficulties in the task. Meanwhile we investigate the limit of existing methods and many open challenges for procedural knowledge acquisition, including: testing supervised settings (e.g. through cross-validation); to better deal with cases of coreference and ellipsis in visual-grounded languages; exploit cross-modalities of information with more robust models using unsupervised or semi-supervised learning paradigm; construct action graphs with dependencies between procedures to enable reasoning and machine execution; incorporate human-in-the-loop teaching in automatic procedural knowledge learning.
References


Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. Learning video representations using contrastive bidirectional transformer, 2019a.


Appendix A. Comparison with existing datasets

There are publicly available datasets related to understanding instructional videos:

- AllRecipes [Kiddon et al., 2015] (AR). The authors collected 2,456 recipes from AllRecipes website\(^8\). The sentences in the dataset are mostly simple imperative English describing concise steps to make a given dish, where the first word is usually the verb describing the action. The ingredient list information is also available. In contrast, our task seeks to extract procedural information from more noisy, oral and erroneous languages in real life video context.

- YouCook2\(^9\) [Zhou et al., 2018b] (YC2). The procedure steps for each video are annotated with temporal boundaries in the video and described by human-written imperative English sentences. However, this dataset does not contain more fine-grained annotations in a structured form.

- HowTo100M\(^10\) [Miech et al., 2019]. This is a large scale how-to videos dataset, searched on YouTube using the task taxonomy on WikiHow\(^11\) as a source. However, it does not contain any annotations although the domain is more general.

- CrossTask\(^12\) [Zhukov et al., 2019] (CT). Based on HowTo100M, this dataset is used for weakly supervised learning with 18 tasks fully labeled and 65 related tasks unlabeled. Although the dataset is annotated in a structured way by separating verbs and objects, the label space is closed with predefined sets of verbs and objects. The dataset also does not allow multiple verbs or objects to be extracted for a single segment.

- COIN\(^13\) [Tang et al., 2019]. This contains instructional (how-to) videos, in a closed taxonomy of tasks and steps. The authors annotated time spans of steps in a video with pre-defined steps, however the biggest drawback is that it is unstructured and closed domain.

- How2\(^14\) [Sanabria et al., 2018]. This dataset annotates ground truth transcript text to help abstractive summarization, a very different task than ours of structured data extraction.

- HAKE\(^15\) [Li et al., 2019]. Human Activity Knowledge Engine (HAKE) is a large-scale knowledge base of human activities, built upon existing activity datasets, and supplies human instance action labels and corresponding body part level atomic action labels. However, HAKE uses closed activity and part state classes. It also does not contain videos of activities accompanied with narrative transcripts.

- TACOS\(^16\) [Regneri et al., 2013]. This dataset considers the problem of grounding sentences describing actions in visual information extracted from videos in kitchen settings. The dataset

---

8. [https://www.allrecipes.com/](https://www.allrecipes.com/)
9. [http://youcook2.eecs.umich.edu/](http://youcook2.eecs.umich.edu/)
10. [https://www.di.ens.fr/willow/research/howto100m/](https://www.di.ens.fr/willow/research/howto100m/)
11. [https://www.wikihow.com](https://www.wikihow.com)
12. [https://github.com/DmZhukov/CrossTask](https://github.com/DmZhukov/CrossTask)
13. [https://coin-dataset.github.io](https://coin-dataset.github.io)
14. [https://github.com/srvk/how2-dataset](https://github.com/srvk/how2-dataset)
15. [http://hake-mvig.cn](http://hake-mvig.cn)
contains expert annotations of low level activity tags, with a total of 60 different activity labels with numerous associated objects, and sequences of NL sentences describing actions in the kitchen videos. This dataset also does not support open extraction and the videos are provided using human annotated caption sentences, rather than transcript texts with noise.

Appendix B. Neural Selection Model

Figure 5 presents the overall detailed structure of the neural selection model for combining utterance and video information for key clip selection.

**Sentence token encoding** Each input clip is accompanied with a sentence \( S = \{t_1, \ldots, t_k\} \) which has \( k \) tokens. We use a pre-trained BERT [Devlin et al., 2018] model as the encoder and extract the sentence representation \( s \).

**Video frame features** For each clip we uniformly sample \( T = 10 \) frames and use an ImageNet-pretrained [Deng et al., 2009] ResNet50 [He et al., 2016] to extract the feature vector of each frame as \( X = \{x_1, \ldots, x_T\} \).

**Attention-based frame encoding** To model the interaction between the encoded sentence and the feature of each frame, we adopt an attention-based method. We first calculate the attention weight \( a_s \) by a tensor product of sentence feature \( s \) with each video frame \( x_i \) followed by a softmax layer. Then we perform a weighted sum on all frame features to get \( \text{Attn}(s, X) \).

**Visual-utterance fusion** Finally, we fuse the extracted transcript features \( s \) with the attended video features \( \text{Attn}(s, X) \) by a tensor product and flatten it into a vector. Then we use a non-linear activation layer to map these features into a real number, which represents the probability of the clip being a key clip.

**Experiment details** In the presented experiments, we use a pre-trained BERT [Devlin et al., 2018] model\(^{17}\) to extract the continuous representation of each sentence. During fine-tuning, the model is optimized by Adam optimizer [Kingma and Ba, 2014] with the starting learning rate of \( 1e^{-4} \). The model is trained in a supervised fashion with a separate key clip/sentence classification dataset that is not related to YouCook2. This auxiliary dataset will also be publicly released. All of them are general domain instructional videos harvested from YouTube. Human annotators labeled whether it is a key clip when given a video clip-sentence pair. In the end, we have 1,034 videos (40,146 pairs) for training the classification model. We split the dataset into two subset as 772 videos (28,519 pairs) and 312 videos (11,627 pairs) for training and validation (hyper-parameter tuning) respectively. The testing set is our proposed dataset with key clips and sentences annotated annotated.

\(^{17}\)https://github.com/hanxiao/bert-as-service
Appendix C. SRL Argument Filtering

The argument types that we deem to not contribute as the procedural knowledge for completing the task and filter out include: ARG0 (usually refers to the subject, usually a person), AM-MOD (modal verb), AM-CAU (cause), AM-NEG (negation marker), AM-DIS (discourse marker), AM-REC (reciprocal), AM-PNC/PRP (purpose), AM-EXT (extent), and R-ARG* (in-sentence references).

Appendix D. Fuzzy Matching and Partial Fuzzy Matching

**Fuzzy matching** Denote the Levenshtein distance between string $a$ and string $b$ as $d(a, b)$. We then define a normalized pairwise score between 0 to 1 as $s(a, b) = d(a, b) / \max\{|a|, |b|\}$ Given a set of $n$ predicted phrases $X = \{x_1, \ldots, x_n\}$ and a set of $m$ ground truth phrases $G = \{g_1, \ldots, g_m\},$ we can find a set of $\min(n, m)$ string pairs between predicted $X$ and ground truth $G,$ as $M = \{(x_i, g_j)\}$ that maximizes the sum of scores $\sum_{(x_i, g_j) \in M} s(x_i, g_j).$ This assignment problem can be solved efficiently with Kuhn-Munkres [Munkres, 1957] algorithm\(^\text{18}\). Since this fuzzy pairwise score is normalized, it can be regarded as a soft version for calculating $TP = \max \sum_{(x_i, g_j) \in M} s(x_i, g_j).$

**Partial Fuzzy matching** The only difference from “fuzzy” matching is that the scoring function now follows the “best partial” heuristic that assuming the shorter string $a$ is length $|a|,$ and the longer string $b$ is length $|b|,$ we now calculate the score between shorter string and the best “fuzzy” matching length-$|a|$ substring.

$$s(a, b) = \max\{d(a, t)\} / |a|, \quad t \in \text{substring of } b, \quad |t| = |a|, \quad |a| < |b|$$

Both fuzzy metric implementations are based on FuzzyWuzzy\(^\text{19}\).

\(^{18}\) [http://software.clapper.org/munkres/]

\(^{19}\) [https://github.com/seatgeek/fuzzywuzzy]