Cross-task weakly supervised learning from instructional videos

Dimitri Zhukov[1,2] Jean-Baptiste Alayrac[1,3] Ramazan Gokberk Cinbis[4] David Fouhey[5]
Ivan Laptev[1,2] Josef Sivic[1,2,6]

Abstract

In this paper we investigate learning visual models for the steps of ordinary tasks using weak supervision via instructional narrations and an ordered list of steps instead of strong supervision via temporal annotations. At the heart of our approach is the observation that weakly supervised learning may be easier if a model shares components while learning different steps: "pour egg" should be trained jointly with other tasks involving "pour" and "egg". We formalize this in a component model for recognizing steps and a weakly supervised learning framework that can learn this model under temporal constraints from narration and the list of steps. Past data does not permit systematic studying of sharing and so we also gather a new dataset, CrossTask, aimed at assessing cross-task sharing. Our experiments demonstrate that sharing across tasks improves performance, especially when done at the component level and that our component model can parse previously unseen tasks by virtue of its compositionality.

1. Introduction

Suppose you buy a fancy new coffee machine and you would like to make a latte. How might you do this? After skimming the instructions, you may start watching instructional videos on YouTube to figure out what each step entails: how to press the coffee, steam the milk, and so on. In the process, you would obtain a good visual model of what each step, and thus the entire task, looks like. Moreover, you could use parts of this visual model of making lattes to help understand videos of a new task, e.g., making filter coffee, since various nouns and verbs are shared. The goal of this paper is to build automated systems that can similarly learn visual models from instructional videos and in particular, make use of shared information across tasks (e.g., making lattes and making filter coffee).

[1]Inria, France
[2]Département d'informatique de l'Ecole Normale Supérieure, PSL Research University, Paris, France
[3]Now at DeepMind
[4]Middle East Technical University, Ankara, Turkey
[5]University of Michigan, Ann Arbor, MI
[6]CIIRC – Czech Institute of Informatics, Robotics and Cybernetics at the Czech Technical University in Prague

Figure 1. Our method begins with a collection of tasks, each consisting of an ordered list of steps and a set of instructional videos from YouTube. It automatically discovers both where the steps occur and what they look like. To do this, it uses the order, narration and commonalities in appearance across tasks (e.g., the appearance of *pour* in both *making pancakes* and *making meringue*).

The conventional approach for building visual models of how to do things [8, 30, 31] is to first annotate each step of each task in time and then train a supervised classifier for each. Obtaining strong supervision in the form of temporal step annotations is time-consuming, unscalable and, as demonstrated by humans' ability to learn from demonstrations, unnecessary. Ideally, the method should be weakly supervised (i.e., like [1, 18, 22, 29]) and jointly learn *when* steps occur and *what* they look like. Unfortunately, any weakly supervised approach faces two large challenges. Temporally localizing steps in the input videos for each task is hard as there is a combinatorial set of options for the step locations; and, even if the steps were localized, each visual model learns from limited data and may work poorly.

We show how to overcome these challenges by sharing across tasks and using weaker and naturally occurring forms of supervision. The related tasks let us learn better visual models by exploiting commonality across steps as illustrated in Figure 1. For example, while learning about *pour water* in *making latte*, the model for *pour* also depends on *pour milk* in *making pancakes* and the model for *water* also depends on *put vegetables in water* in *making bread and butter pickles*. We assume an ordered list of steps is given per task and that the videos are instructional (i.e., have a natural language narration describing what is being done). As it is often the case in weakly supervised video learning [2, 18, 29], these assumptions constrain the search for when steps occur, helping tackle a combinatorial search space.

We formalize these intuitions in a framework, described in Section 4, that enables compositional sharing across tasks together with temporal constraints for weakly supervised learning. Rather than learning each step as a monolithic weakly-supervised classifier, our formulation learns a component model that represents the model for each step as the combination of models of its components, or the words in each step (e.g., *pour* in *pour water*). This empirically improves learning performance and these component models can be recombined in new ways to parse videos for tasks for which it was not trained, simply by virtue of their representation. This component model, however, prevents the direct application of techniques previously used for weakly supervised learning in similar settings (e.g., DIFFRAC [3] in [2]); we therefore introduce a new and more general formulation that can handle more arbitrary objectives.

Existing instructional video datasets do not permit the systematic study of this sharing. We gather a new dataset, CrossTask, which we introduce in Section 5. This dataset consists of $\sim$4.7K instructional videos for 83 different tasks, covering 374 hours of footage. We use this dataset to compare our proposed approach with a number of alternatives in experiments described in Section 6. Our experiments aim to assess the following three questions: how well does the system learn in a standard weakly supervised setup; can it exploit related tasks to improve performance; and how well can it parse previously unseen tasks.

The paper's contributions include: **(1)** A component model that shares information between steps for weakly supervised learning from instructional videos; **(2)** A weakly supervised learning framework that can handle such a model together with constraints incorporating different forms of weak supervision; and **(3)** A new dataset that is larger and more diverse than past efforts, which we use to empirically validate the first two contributions. We make our dataset and our code publically available[1].

2. Related Work

Learning the visual appearance of steps of a task from instructional videos is a form of action recognition. Most work in this area, e.g., [8, 30, 31], uses strong supervision in the form of direct labels, including a lot of work that focuses on similar objectives [9, 11, 14]. We build our feature representations on top of advances in this area [8], but our proposed method does not depend on having lots of annotated data for our problem.

We are not the first to try to learn with weak supervision in videos and our work bears resemblances to past efforts. For instance, we make use of ordering constraints to obtain supervision, as was done in [5, 18, 22, 26, 6]. The aim of our work is perhaps closest to [1, 24, 29] as they also use narrations in the context of instructional videos. Among a number of distinctions with each individual work, one significant novelty of our work is the compositional model used, where instead of learning a monolithic model independently per-step as done in [1, 29], the framework shares components (e.g., nouns and verbs) across steps. This sharing improves performance, as we empirically confirm, and enables the parsing of unseen tasks.

In order to properly evaluate the importance of sharing, we gather a dataset of instructional videos. These have attracted a great deal of attention recently [1, 2, 19, 20, 24, 29, 35] since the co-occurrence of demonstrative visual actions and natural language enables many interesting tasks ranging from coreference resolution [19] to learning person-object interaction [2, 10]. Existing data, however, is either not large (e.g., only 5 tasks [2]), not diverse (e.g., YouCookII [35] is only cooking), or not densely temporally annotated (e.g., What's Cooking? [24]). We thus collect a dataset that is: **(i)** relatively large (83 tasks, 4.7K videos); **(ii)** simultaneously diverse (Covering car maintenance, cooking, crafting) yet also permitting the evaluation of sharing as it has related tasks; and **(iii)** annotated for temporal localization, permitting evaluation. The scale, and relatedness, as we demonstrate empirically contribute to increased performance of visual models.

Our technical approach to the problem builds particularly heavily on the use of discriminative clustering [3, 32], or the simultaneous constrained grouping of data samples and learning of classifiers for groups. Past work in this area has either had operated with complex constraints and a restricted classifier (e.g., minimizing the L2 loss with linear model [3, 2]) or an unrestricted classifier, such as a deep network, but no constraints [4, 7]. Our weakly supervised setting requires the ability to add constraints in order to converge to a good solution while our compositional model and desired loss function requires the ability to use an unrestricted classifier. We therefore propose an optimization approach that handles both, letting us train with a compositional model while also using temporal constraints.

Finally, our sharing between tasks is enabled via the composition of the components of each step (e.g., nouns, verbs). This is similar to attributes [12, 13], which have been used in action recognition in the past [23, 33]. Our components are meaningful (representing, e.g., "lemon") but also automatically built; they are thus different than pre-defined semantic attributes (not automatic) and the non-

[1] https://github.com/DmZhukov/CrossTask

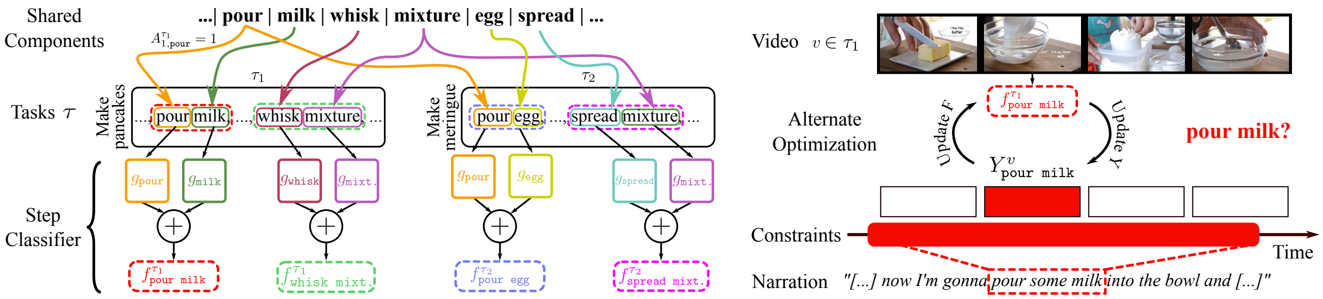

Figure 2. Our approach expresses classifiers for each step of each task in terms of a component model (e.g., writing the *pour milk* as a *pour* and *milk* classifier). We thus cast the problem of learning the steps as learning an underlying set of component models. We learn these models by alternating between updating labels for these classifiers and the classifiers themselves while using constraints from narrations.

semantic attributes (not intrinsically meaningful) as defined in [12]. It is also related to methods that compose new classifiers from others, including [25, 34, 15] among many others. Our framework is orthogonal, and shows how to learn these in a weakly-supervised setting.

3. Overview

Our goal is to build visual models for a set of **tasks** from instructional videos. Each task is a multi-step process such as *making latte* consisting of multiple **steps**, such as *pour milk*. We aim to learn a visual model for each of these steps. Our approach uses **component models** that represent each step in terms of its constituent **components** as opposed to a monolithic entity, as illustrated in Figure 2. For instance, rather than building a classifier solely for *whisk mixture* in the context of *make pancakes*, we learn a set of classifiers per-component, one for *whisk*, *spread*, *mixture* and so on, and represent *whisk mixture* as the combination of *whisk* and *mixture* and share *mixture* with *spread mixture*. This shares data between steps and enables the parsing of previously unseen tasks, which we both verify empirically.

We make a number of assumptions. Throughout, we assume that we are given an ordered list of steps for each task. This list is our only source of manual supervision and is done once per-task and is far less time consuming than annotating a temporal segmentation of each step in the input videos. At training time, we also assume that our training videos contain audio that explains what actions are being performed. At test time, however, we do not use the audio track: just like a person who watches a video online, once our system is shown how to make a latte with narration, it is expected to follow along without step-by-step narrations.

4. Modeling Instructional Videos

We now describe our technical approach for using a list of steps to jointly learn the labels and visual models on a set of narrated instructional videos. This is weakly supervised since we provide only the list of steps, but not their temporal locations in training videos.

Problem formulation. We denote the set of narrated instructional videos $\mathcal{V}$. Each video $v \in \mathcal{V}$ contains a sequence of N_v segments of visual features $X^v = (x_1, \dots, x_{N_v})$ as well as narrations we use later. For every task τ we assume to be given a set of videos V_τ together with a set of ordered natural language steps K_τ.

Our goal is then to discover a set of classifiers F that can identify the steps of the tasks. In other words, if τ is a task and k is its step, the classifier f_k^τ determines whether a visual feature depicts step k of τ or not. To do this, we also learn a labeling Y of the training set for the classifiers, or for every video v depicting task τ, a binary label matrix $Y^v \in \{0,1\}^{N_v \times K_\tau}$ where $Y_{tk}^v = 1$ if time t depicts step k and 0 otherwise. While jointly learning labels and classifiers leads to trivial solutions, we can eliminate these and make meaningful progress by constraining Y and by sharing information across the classifiers of F.

4.1. Component Classifiers

One of the main focuses of this paper is in the form of the step classifier f. Specifically, we propose a component model that represents each step (e.g., "pour milk") as a combination of components (e.g., "pour" and "milk"). Before explaining how we formulate this, we place it in context by introducing a variety of alternatives that vary in terms of how they are learned and formulated.

The simplest approach, a **task-specific step model**, is to learn a classifier for each step in the training set (i.e., a model for *pour egg* for the particular task of *making pancakes*). Here, the model simply learns $\sum_\tau K_\tau$ classifiers, one for each of the K_τ steps in each task, which is simple but which permits no sharing.

One way of adding sharing would be to have a **shared step model**, where a single classifier is learned for each unique step in the dataset. For instance, the *pour egg* classifier learns from both *making meringues* and *making pancakes*. This sharing, however, would be limited to exact duplicates of steps, and so while *whisk milk* and *pour milk* both share an object, they would be learned separately.

Our proposed **component model** fixes this issue. We automatically generate a vocabulary of **components** by taking the set of stemmed words in all the steps. These components are typically objects, verbs and prepositions and we combine classifiers for each component to yield our steps. In particular, for a vocabulary of M components, we define a per-task matrix $A^\tau \in \{0,1\}^{K_\tau \times M}$ where $A^\tau_{k,m} = 1$ if the step k involves components m and 0 otherwise. We then learn M classifiers $g_1, \dots, g_M$ such that the prediction of a step f^τ_k is the average of predictions provided by component classifiers

$$f^\tau_k(x) = \sum_m A^\tau_{km} g_m(x) / \sum_m A^\tau_{km}. \tag{1}$$

For instance, the score for *pour milk* is the average of outputs of g_{pour} and g_{milk}. In other words, when optimizing over the set of functions F, we optimize over the parameters of $\{g_i\}$ so that when combined together in step models via (1), they produce the desired results.

4.2. Objective and Constraints

Having described the setup and classifiers, we now describe the objective function we minimize. Our goal is to simultaneously optimize over step location labels Y and classifiers F over all videos and tasks

$$\min_{Y \in \mathcal{C}, F \in \mathcal{F}} \sum_\tau \sum_{v \in \mathcal{V}(\tau)} h(X^v, Y^v; F), \tag{2}$$

where $\mathcal{C}$ is the set of temporal constraints on Y defined below, and $\mathcal{F}$ is a family of considered classifiers. Our objective function per-video is a standard cross-entropy loss

$$h(X^v, Y^v; F) = -\sum_{t,k} Y^v_{tk} \log \left(\frac{\exp\left(f^\tau_k(x^v_t)\right)}{\sum_{k'} \exp(f^\tau_{k'}(x^v_t))} \right). \tag{3}$$

Optimizing (2) may lead to trivial solutions (e.g., $Y^v = 0$ and F outputting all zeros). We thus constrain our labeling of Y to avoid this and ensure a sensible solution. In particular, we impose three constraints:

At least once. We assume that every video v of a task depicts each step k at least once, or $\sum_t Y^v_{tk} \geq 1$.

Temporal ordering. We assume that steps occur in the given order. While not always strictly correct, this dramatically reduces the search space and leads to better classifiers.

Temporal text localization. We assume that the steps and corresponding narrations happen close in time, e.g., the narrator of a *grill steak* video may say "just put the marinated steak on the grill". We automatically compare the text description of each step to automatic YouTube subtitles. For a task with K_τ steps and a video with N_v frames, we construct a $[0,1]^{N_v \times K_\tau}$ matrix of cosine similarities between steps and a sliding-window word vector representations of narrations (more details in supplementary materials [**?**]). Since narrated videos contain spurious mentions of tasks (e.g., "before putting the steak on the grill, we clean the grill") we do not directly use this matrix, but instead find an assignment of steps to locations that maximizes the total similarity while respecting the ordering constraints. The visual model must then more precisely identify when the action appears. We then impose a simple hard constraint of disallowing labelings Y^v where any step is outside of the text-based interval (average length 9s)

4.3. Optimization and Inference

We solve problem (2) by alternating between updating assignments Y and the parameters of the classifiers F.

Updating Y. When F is fixed, we can minimize (2) w.r.t. Y independently for each video. In particular, fixing F fixes the classifier scores, meaning that minimizing (2) with respect to Y^v is a constrained minimization of a linear cost in Y subject to constraints. Our supplemental [**?**] shows that this can be done by dynamic programming.

Updating F. When Y is fixed, our cost function reduces to a standard supervised classification problem. We can thus apply standard techniques for solving these, such as stochastic gradient descent. More details are provided below and in the supplemental material [**?**].

Initialization. Our objective is non-convex and has local minima, thus a proper initialization is important. We obtain such an initialization by treating all assignments that satisfy the temporal text localization constraints as ground-truth and optimizing for F for 30 epochs, each time drawing a random sample that satisfies the constraints.

Inference. Once the model has been fit to the data, inference on a new video v of a task τ is simple. After extracting features, we run each classifier f on every temporal segment, resulting in a $N_v \times K_\tau$ score matrix. To obtain a hard labeling, we use dynamic programming to find the best-scoring labeling that respects the given order of steps.

4.4. Implementation Details

Networks: Due to the limited data size and noisy supervision, we use a linear classifier with dropout for regularization. Preliminary experiments with deeper models did not yield improvements. We use ADAM [21] with the learning rate of 10^{-5} for optimization. *Features:* We represent each video segment x_i using RGB I3D features [8] (1024D), Resnet-152 features [16] (2048D) extracted at each frame and averaged over one-second temporal windows, and audio features from [17] (128D). *Components:* We obtain the dictionary of components by finding the set of unique stemmed words over all step descriptions. The total number of components is 383. *Hyperparameters:* Dropout and the learning rate are chosen on a validation data set.

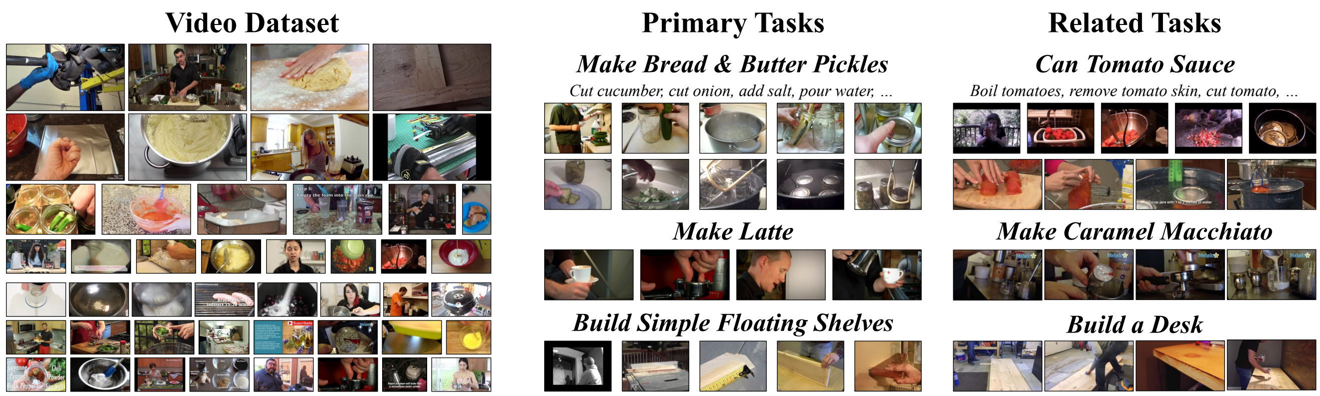

Figure 3. Our new dataset, used to study sharing in a weakly supervised learning setting. It contains primary tasks, such as *make bread and butter pickles*, as well as related tasks, such as *can tomato sauce*. This lets us study whether learning multiple tasks improves performance.

Table 1. A comparison of CrossTask with existing instructional datasets. Our dataset is both large and more diverse while also having temporal annotations.

	Num. Vids	Total Length	Num. Tasks	Not only Cooking	Avail. Annots
[2]	150	7h	5	✓	Windows
[29]	1.2K+85	100h	17	✓	Windows
[35]	2K	176h	89	✗	Windows
[24]	180K	3,000h	✗	✗	Recipes
CrossTask	4.7K	375h	83	✓	Windows

5. CrossTask dataset

One goal of this paper is to investigate whether sharing improves the performance of weakly supervised learning from instructional videos. To do this, we need a dataset covering a diverse set of interrelated tasks and annotated with temporal segments. Existing data fails to satisfy at least one of these criteria and we therefore collect a new dataset (83 tasks, 4.7K videos) related to cooking, car maintenance, crafting, and home repairs. These tasks and their steps are derived from wikiHow, a website that describes how to solve many tasks, and the videos come from YouTube.

CrossTask dataset is divided into two sets of tasks to investigate sharing. The first is **primary tasks**, which are the main focus of our investigation and the backbone of the dataset. These are fully annotated and form the basis for our evaluations. The second is **related tasks** with videos gathered in a more automatic way to share some, but not all, components with the primary tasks. One goal of our experiments is to assess whether these related tasks improve the learning of primary tasks, and whether one can learn a good model only on related tasks.

5.1. Video Collection Procedure

We begin the collection process by defining our tasks. These must satisfy three criteria: they must entail a sequence of physical interactions with objects (unlike e.g., *how to get into a relationship*); their step order must be deterministic (unlike e.g., *how to play chess*); and they must appear frequently on YouTube. We asked annotators to review the tasks in five sections of wikihow to get tasks satisfying the first two criteria, yielding $\sim$ 7K candidate tasks, and manually filter for the third criteria.

We select 18 primary tasks and 65 related tasks from these 7K candidate tasks. The primary tasks cover a variety of themes (e.g., auto repair to cooking to DIY) and include *building floating shelves* and *making latte*. We find 65 related tasks by finding related tasks for each primary task. We generate potential related tasks for a primary task by comparing the wikiHow articles using a TF-IDF on a bag-of-words representation, which finds tasks with similar descriptions. We then filter out near duplicates (e.g., *how to jack up a car* and *how to use a car jack*) by comparing top YouTube search results and removing candidates with overlaps, and manually remove a handful of irrelevant tasks.

We define steps and their order for each task by examining the wikiHow articles, beginning with the summaries of each step. Using the wikiHow summary itself is insufficient, since many articles contain non-visual steps and some steps combine multiple physical actions. We thus manually correct the list yielding a set of tasks with 7.4 steps on average for primary tasks and 8.8 for related tasks.

We then obtain videos for each task by searching YouTube. Since the related tasks are only to aid the primary tasks, we take the top 30 results from YouTube. For primary tasks, we ask annotators to filter a larger pool of top results while examining the video, steps, and wikiHow illustrations, yielding at least 80 videos per task.

5.2. Annotations and Statistics

Task localization annotations. Since our focus is the primary tasks, annotators mark the temporal extent of each primary task step independently. We do this for our 18 primary tasks and make annotations publically available[1].

Dataset. This results in a dataset containing 2763 videos

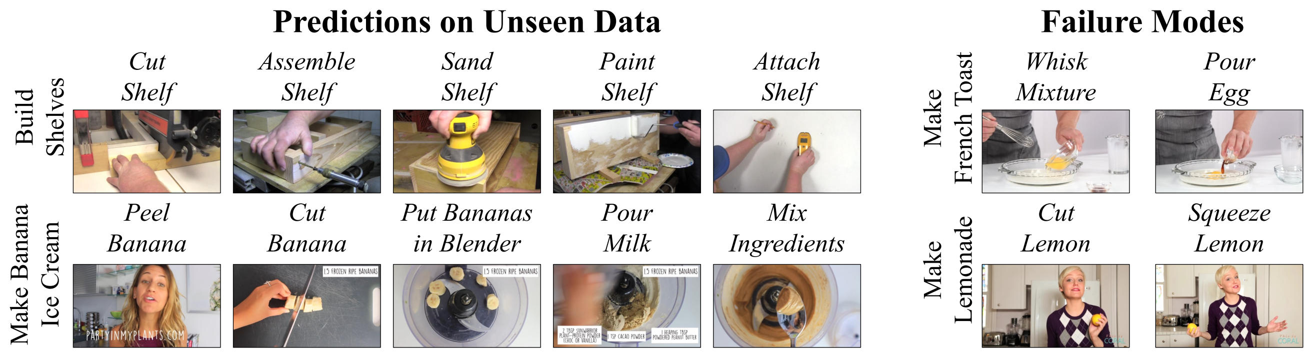

Figure 4. Predictions on unseen data as well as typical failure modes. Our method does well on steps with distinctive motions and appearances. Failure modes include (top) features that cannot make fine-grained distinctions between e.g., egg and vanilla extract; and (bottom) models that overreact to particular nouns, preferring a more visible lemon over a less visible lemon actually being squeezed.

of 18 primary tasks comprising 213 hours of video; and 1950 videos of 65 related tasks comprising 161 hours of video. We contrast this dataset with past instructional video datasets in Table 1. Our dataset is simultaneously large while also having precise temporal segment annotations.

To illustrate the dataset, we report a few summary statistics about the primary task videos. The videos are quite long, with an average length of 4min 57sec, and depict fairly complex tasks, with 7.4 steps on average. Less complex tasks include *jack up a car* (3 steps); more complex ones include *pickle cucumbers* or *change tire* (11 steps each).

Challenges. In addition to being long and complex, these videos are challenging since they do not precisely show the ordered steps we have defined. For instance, in *add oil to car*, 85% of frames instead depict background information such as shots of people talking or other things. This is not an outlier: on average 72% of the dataset is background. On the other hand, on average 31% of steps are not depicted due to variances in procedures and omissions (*pickle cucumber* has 48% of steps missing). Moreover, the steps do not necessarily appear in the correct order: to estimate the order consistency, we compute an upper bound on performance using our given order and found that the best order-respecting parse of the data still missed 14% of steps.

6. Experiments

Our experiments aim to address the following three questions about cross-task sharing in the weakly-supervised setting: **(1)** Can the proposed method use related data to improve performance? **(2)** How does the proposed component model compare to sharing alternatives? **(3)** Can the component model transfer to previously unseen tasks? Throughout, we evaluate on the large dataset introduced in Section 5 that consists of primary tasks and related tasks. We address (1) in Section 6.1 by comparing our proposed approach with methods that do not share and show that our proposed approach can use related tasks to improve performance on primary asks. Section 6.2 addresses (2) by analyzing the performance of the model and showing that it outperforms step-based alternatives. We answer (3) empirically in Section 6.3 by training only on related tasks, and show that we are able to perform well on primary tasks.

6.1. Cross-task Learning

We begin by evaluating whether our proposed component model approach can use sharing to improve performance on a fixed set of tasks. We fix our evaluation to be the 18 primary tasks and evaluate whether the model can use the 65 related tasks to improve performance.

Metrics and setup. We evaluate results on 18 primary tasks over the videos that make up the test set. We quantify performance via *recall*, which we define as the ratio between the number of correct step assignments (defined as falling into the correct ground-truth time interval) and the total number of steps over all videos. In other words, to get a perfect score, a method must correctly identify one instance of each step of the task in each test video. All methods make a single prediction per step, which prevents the trivial solution of assigning all frames to all actions.

We run experiments 20 times, each time making a train set of 30 videos per task and leaving the remaining 1863 videos for test. We report the average. Hyperparameters are set for all methods using a fixed validation set of 20 videos per primary task that are never used for training or testing.

Baselines. Our goal is to examine whether our sharing approach can leverage related tasks to improve performance on our primary task. We compare our method to its version without sharing as well as to a number of baselines. *(1) Uniform:* simply predict steps at fixed time intervals. Since this predicts steps in the correct order and steps often break tasks into roughly equal chunks, this is fairly well-informed prior. *(2) Alayrac'16:* the weakly supervised learning method for videos, proposed in [1]. This is similar in spirit to our approach except it does not share and optimizes a L2-criterion via the DIFFRAC [3] method. *(3) Richard'18:* the weakly supervised learning method [27] that does not rely on the

Table 2. Weakly supervised recall scores on test set (in %). Our approach, which shares information across tasks, substantially and consistently outperforms non-sharing baselines. The standard deviation for reported scores does not exceed 1%.

	Make Kimchi Rice	Pickle Cucumber	Make Banana Ice Cream	Grill Steak	Jack Up Car	Make Jello Shots	Change Tire	Make Lemonade	Add Oil to Car	Make Latte	Build Shelves	Make Taco Salad	Make French Toast	Make Irish Coffee	Make Strawberry Cake	Make Pancakes	Make Meringue	Make Fish Curry	Average
Supervised	19.1	25.3	38.0	37.5	25.7	28.2	54.3	25.8	18.3	31.2	47.7	12.0	39.5	23.4	30.9	41.1	53.4	17.3	31.6
Uniform	4.2	7.1	6.4	7.3	**17.4**	7.1	14.2	9.8	3.1	10.7	22.1	5.5	9.5	7.5	9.2	9.2	19.5	5.1	9.7
Alayrac'16 [1]	**15.6**	10.6	7.5	14.2	9.3	11.8	17.3	13.1	**6.4**	12.9	27.2	9.2	15.7	8.6	16.3	13.0	23.2	7.4	13.3
Richard'18 [27]	7.6	4.3	3.6	4.6	8.9	5.4	7.5	7.3	3.6	6.2	12.3	3.8	7.4	7.2	6.7	9.6	12.3	3.1	6.7
Task-Specific Step-Based	13.2	17.6	19.3	19.3	9.7	12.6	30.4	16.0	4.5	19.0	29.0	9.1	29.1	**14.5**	22.9	29.0	32.9	7.3	18.6
Proposed	13.3	**18.0**	**23.4**	**23.1**	16.9	**16.5**	**30.7**	**21.6**	4.6	**19.5**	**35.3**	**10.0**	**32.3**	13.8	**29.5**	**37.6**	**43.0**	**13.3**	**22.4**
Gain from Sharing	0.2	0.4	4.1	3.8	7.2	3.9	0.3	5.6	0.1	0.6	6.3	0.9	3.2	-0.7	6.6	8.7	10.1	6.0	3.7

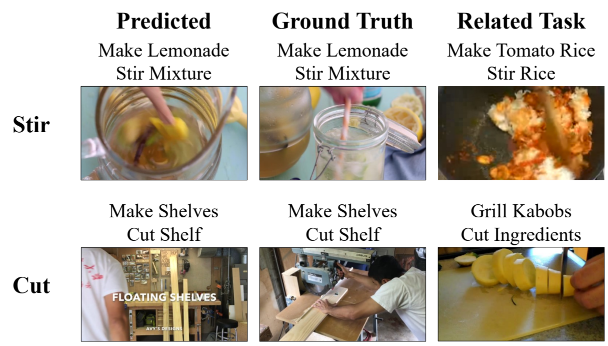

Figure 5. Components that share well and poorly: while stir shares well between steps of tasks, cut shares poorly when transferring from a food context to a home improvement context.

known order of steps. *(4) Task-Specific Steps:* Our approach trained independently for each step of each task. In other words, there are separate models for *pour egg* in the contexts of *making pancakes* and *making meringue*. This differs from Alayrac in that it optimizes a cross-entropy loss using our proposed optimization method. It differs from our full proposed approach since it performs no sharing. Note, that the full method in [1] includes automatic discovery of steps from narrations. Here, we only use the visual model of [1], while providing the same constraints as in our method. This allows for a fair comparison between [1] and our method, since both use the same amount of supervision. At test time, the method presented in [27] has no prior about which steps are present or the order in which they occur. To make a fair comparison, we use the trained classifier of the method in [27], and apply the same inference procedure as in our method.

Qualitative results. We illustrate qualitative results of our full method in Figure 4. We show a parses of unseen videos of *Build Shelves* and *Make Banana Ice Cream* and failure modes. Our method can handle well a large variety of tasks and steps but may struggle to identify some details (e.g., vanilla vs. egg) or actions.

Quantitative results. Table 2 shows results summarized across steps. The uniform baseline provides a strong lower bound, achieving an average recall of 9.7% and outperforming [27]. Note, however, that [27] is designed to adress a different problem and cannot be fairly compared with other methods in our setup. While [1] improves on this (13.3%), it does substantially worse than our task-specific step method (18.6%). We found that predictions from [1] often had several steps with similar scores, leading to poor parse results, which we attribute to the convex relaxation used by DIFFRAC. This was resolved in the past by the use of narration at test time; our approach does not depend on this.

Our full approach, which shares across tasks, produces substantially better performance (22.4%) than the task-specific step method. More importantly, this improvement is systematic: the full method improves on the task-specific step baseline in 17 tasks out of 18.

We illustrate some qualitative examples of steps benefiting and least benefiting from sharing in Figure 5. Typically, sharing can help if the component has distinctive appearance and is involved in a number of steps: steps involve stirring, for instance, have an average gain of 15% recall over independent training because it is frequent (in 30 steps) and distinctive. Of course, not all steps benefit: *cut shelf* is harmed (47% independent $\rightarrow$ 28% shared) because *cut* mostly occurs in cooking tasks with dissimilar contexts.

Verifying optimizer on small-scale data. We now evaluate our approach on the smaller 5-task dataset of [1]. Since here there are no common steps across tasks, we are able to test only the basic task-specific step-based version. To make a fair comparison, we use the same features, ordering constraints, as well as constraints from narration for every K as provided by the authors of [1], and we evaluate using the F1 metric as in [1]. As a result, the two formulations are on par, where [1] versus our approach result in 22.8% versus 21.8% for K=10 and 21.0% versus 21.1% for K=15, respectively. While these scores are slightly lower compared to those obtained by the single-task probabilistic model in

Table 3. Average recall scores on the test set for our method when changing the sharing settings and the model.

	Unshared Primary	Shared Primary	Shared Primary + Related
Step-based	18.6	18.9	19.8
Component-based	**18.7**	**20.2**	**22.4**

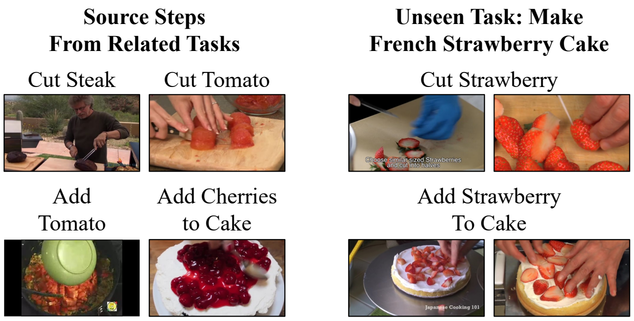

Figure 6. Examples of identified steps for an unseen task. While the model has not seen these steps and objects e.g., strawberries, its knowledge of other components leads to reasonable predictions.

Sener [28] (25.4% at K=10 and 23.6% at K=15), we are unable to compare using our full cross-task model on this dataset. Overall, these results verify the effectiveness of our optimization technique.

6.2. Experimental Evaluation of Cross-task Sharing

Having verified the framework and the role of sharing, we now more precisely evaluate how sharing is performed to examine the contribution of our proposed compositional model. We vary two dimensions. The first is the granularity, or at what level sharing occurs. We propose sharing at a component level, but one could share at a step level as well. The second is what data is used, including (i) independently learning primary tasks; (ii) learning primary tasks together; (iii) learning primary plus related tasks together.

Table 3 reveals that increased sharing consistently helps and component-based sharing extracts more from sharing than step-based (performance increases across rows). This gain over step-based sharing is because step-based sharing requires exact matches. Most commonality between tasks occurs with slight variants (e.g., *cut* is applied to steak, tomato, pickle, etc.) and therefore a component-based model is needed to maximally enable sharing.

6.3. Novel Task Transfer

One advantage of shared representations is that they can let one parse new concepts. For example, without any modifications, we can repeat our experiments from Section 6.1 in a setting where we never train on the 18 tasks that we test on but instead on the 65 related tasks. The only information given about the test tasks is an ordered list of steps.

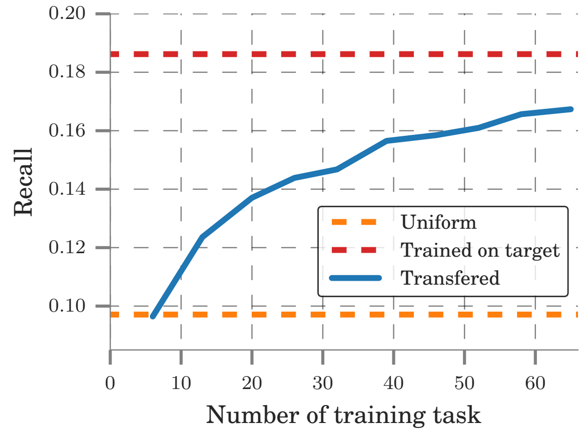

Figure 7. Recall while transferring a learned model to unseen tasks as a function of the number of tasks used for training. Our component model approaches training directly on these tasks.

Setup. As in Section 6.1, we quantify performance with recall on the 18 primary tasks. However, we train on a subset of the 65 related tasks and never on any primary task.

Qualitative results. We show a parse of steps of *Make Strawberry Cake* in Figure 6 using all related tasks. The model has not seen *cut strawberry* before but has seen other forms of cutting. Similarly, it has seen *add cherries to cake*, and can use this step to parse *add strawberries to cake*.

Quantitative results. Figure 7 shows performance as a function of the number of related tasks used for training. Increasing the number of training tasks improves performance on the primary tasks, and does not plateau even when 65 tasks are used.

7. Conclusion

We have introduced an approach for weakly supervised learning from instructional videos and a new CrossTask dataset for evaluating the role of sharing in this setting. Our component model has been shown ability to exploit common parts of tasks to improve performance and was able to parse previously unseen tasks. Future work would benefit from improved features as well as from improved versions of sharing.

Acknowledgements. This work was supported in part by the MSR-Inria joint lab, the Louis Vuitton ENS Chair on Artificial Intelligence, ERC grants LEAP No. 336845 and ACTIVIA No. 307574, the DGA project DRAAF, CIFAR Learning in Machines & Brains program, the European Regional Development Fund under the project IMPACT (reg. no. CZ.02.1.01/0.0/0.0/15003/0000468), the TUBITAK Grant 116E445 and a research fellowship by the Embassy of France. We thank Francis Bach for helpful discussions about the optimization procedure.

References

[1] J.-B. Alayrac, P. Bojanowski, N. Agrawal, I. Laptev, J. Sivic, and S. Lacoste Julien. Unsupervised learning from narrated instruction videos. In *CVPR*, 2016. 1, 2, 6, 7

[2] J.-B. Alayrac, J. Sivic, I. Laptev, and S. Lacoste-Julien. Joint discovery of object states and manipulation actions. In *ICCV*, 2017. 2, 5

[3] F. Bach and Z. Harchaoui. DIFFRAC: A discriminative and flexible framework for clustering. In *NIPS*, 2007. 2, 6

[4] P. Bojanowski and A. Joulin. Unsupervised learning by predicting noise. In *ICML*, 2017. 2

[5] P. Bojanowski, R. Lajugie, F. Bach, I. Laptev, J. Ponce, C. Schmid, and J. Sivic. Weakly supervised action labeling in videos under ordering constraints. In *ECCV*, 2014. 2

[6] P. Bojanowski, R. Lajugie, E. Grave, F. Bach, I. Laptev, J. Ponce, and C. Schmid. Weakly-supervised alignment of video with text. In *ICCV*, 2015. 2

[7] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep Clustering for Unsupervised Learning of Visual Features. In *ICCV*, 2018. 2

[8] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 1, 2, 4

[9] D. Damen, H. Doughty, G. Maria Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. Scaling egocentric vision: The EPIC-KITCHENS dataset. In *ECCV*, 2018. 2

[10] D. Damen, T. Leelasawassuk, O. Haines, A. Calway, and W. Mayol-Cuevas. You-do, i-learn: Discovering task relevant objects and their modes of interaction from multi-user egocentric video. In *BMVA*, 2014. 2

[11] K. Fang, T.-L. Wu, D. Yang, S. Savarese, and J. J. Lim. Demo2vec: Reasoning object affordances from online videos. In *CVPR*, 2018. 2

[12] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009. 2, 3

[13] V. Ferrari and A. Zisserman. Learning visual attributes. In *NIPS*, 2007. 2

[14] D. F. Fouhey, W. Kuo, A. A. Efros, and J. Malik. From lifestyle vlogs to everyday interactions. In *CVPR*, 2018. 2

[15] S. Guadarrama, N. Krishnamoorthy, G. Malkarnenkar, S. Venugopalan, R. Mooney, T. Darrell, and K. Saenko. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *ICCV*, 2013. 3

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[17] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, , and K. Wilson. Cnn architectures for large-scale audio classification. In *ICASSP*, 2017. 4

[18] D.-A. Huang, L. Fei-Fei, and J. C. Niebles. Connectionist temporal modeling for weakly supervised action labeling. In *ECCV*, 2016. 1, 2

[19] D.-A. Huang, J. J. Lim, L. Fei-Fei, and J. C. Niebles. Unsupervised visual-linguistic reference resolution in instructional videos. In *CVPR*, 2017. 2

[20] D.-A. Huang, V. Ramanathan, D. Mahajan, L. Torresani, M. Paluri, L. Fei-Fei, and J. C. Niebles. Finding "it": Weakly-supervised reference-aware visual grounding in instructional video. In *CVPR*, 2018. 2

[21] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[22] H. Kuehne, A. Richard, and J. Gall. Weakly supervised learning of actions from transcripts. In *CVIU*, 2017. 1, 2

[23] J. Liu, B. Kuipers, and S. Savarese. Recognizing human actions by attributes. In *CVPR*, 2011. 2

[24] J. Malmaud, J. Huang, V. Rathod, N. Johnston, A. Rabinovich, and K. Murphy. What's cookin'? Interpreting cooking videos using text, speech and vision. In *NAACL*, 2015. 2, 5

[25] I. Misra, A. Gupta, and M. Hebert. From Red Wine to Red Tomato: Composition with Context. In *CVPR*, 2017. 3

[26] A. Richard, H. Kuehne, and J. Gall. Weakly supervised action learning with rnn based fine-to-coarse modeling. In *CVPR*, 2017. 2

[27] A. Richard, H. Kuehne, and J. Gall. Action sets: Weakly supervised action segmentation without ordering constraints. In *CVPR*, 2018. 6, 7

[28] F. Sener and A. Yao. Unsupervised learning and segmentation of complex activities from video. In *CVPR*, 2018. 8

[29] O. Sener, A. Zamir, S. Savarese, and A. Saxena. Unsupervised semantic parsing of video collections. In *ICCV*, 2015. 1, 2, 5

[30] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014. 1, 2

[31] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, 2013. 1, 2

[32] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In *NIPS*, 2004. 2

[33] B. Yao, X. Jiang, A. Khosla, A. L. Lin, L. Guibas, and L. Fei-Fei1. Human action recognition by learning bases of action attributes and parts. In *ICCV*, 2011. 2

[34] M. Yatskar, V. Ordonez, L. Zettlemoyer, and A. Farhadi. Commonly uncommon: Semantic sparsity in situation recognition. In *Proceedings of the CVPR*, 2017. 3

[35] L. Zhou, X. Chenliang, and J. J. Corso. Towards automatic learning of procedures from web instructional videos. In *AAAI*, 2018. 2, 5