Learning and Verification of Task Structure in Instructional Videos

Medhini Narasimhan[1,2], Licheng Yu[2], Sean Bell[2], Ning Zhang[2], Trevor Darrell[1]

[1]UC Berkeley, [2]Meta AI

https://medhini.github.io/task_structure

Abstract

Given the enormous number of instructional videos available online, learning a diverse array of multi-step task models from videos is an appealing goal. We introduce a new pre-trained video model, VideoTaskformer, focused on representing the semantics and structure of instructional videos. We pre-train VideoTaskformer using a simple and effective objective: predicting weakly supervised textual labels for steps that are randomly masked out from an instructional video (masked step modeling). Compared to prior work which learns step representations locally, our approach involves learning them globally, leveraging video of the entire surrounding task as context. From these learned representations, we can verify if an unseen video correctly executes a given task, as well as forecast which steps are likely to be taken after a given step. We introduce two new benchmarks for detecting mistakes in instructional videos, to verify if there is an anomalous step and if steps are executed in the right order. We also introduce a long-term forecasting benchmark, where the goal is to predict long-range future steps from a given step. Our method outperforms previous baselines on these tasks, and we believe the tasks will be a valuable way for the community to measure the quality of step representations. Additionally, we evaluate VideoTaskformer on 3 existing benchmarks—procedural activity recognition, step classification, and step forecasting—and demonstrate on each that our method outperforms existing baselines and achieves new state-of-the-art performance.

1. Introduction

Picture this, you're trying to build a bookshelf by watching a YouTube video with several intricate steps. You're annoyed by the need to repeatedly hit pause on the video and you're unsure if you have gotten all the steps right so far. Fortunately, you have an interactive assistant that can guide you through the task at your own pace, verifying each step as you perform it and interrupting you if you make a mistake. A composite task such as "*making a bookshelf*" involves multiple fine-grained activities such as "*drilling holes*" and "*adding support blocks*." Accurately categorizing these activities requires not only recognizing the individual steps that compose the task but also understanding the task structure, which includes the temporal ordering of the steps and multiple plausible ways of executing a step (e.g., one can beat eggs with a fork or a whisk). An ideal interactive assistant has both a high-level understanding of a broad range of tasks, as well as a low-level understanding of the intricate steps in the tasks, their temporal ordering, and the multiple ways of performing them.

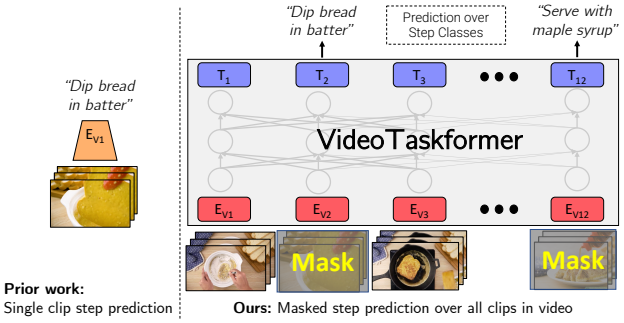

Figure 1: Prior work [5, 4] learns step representations from single short video clips, independent of the task, thus lacking knowledge of task structure. Our model, VideoTaskformer, learns step representations for masked video steps through the global context of all surrounding steps in the video, making our learned representations aware of task semantics and structure.

As seen in Fig. 1, prior work [4, 5] models step representations of a single step independent of the overall task context. This might not be the best strategy, given that steps for a task are related, and the way a step is situated in an overall task may contain important information about the step. To address this, we pre-train our model VideoTaskformer, with a masked modeling objective that encourages the step representations to capture the *global context* of the entire video. Prior work lacks a benchmark for detecting

*Work done while an intern at Meta AI and a graduate student at UC Berkeley. Correspondence to medhini@google.com

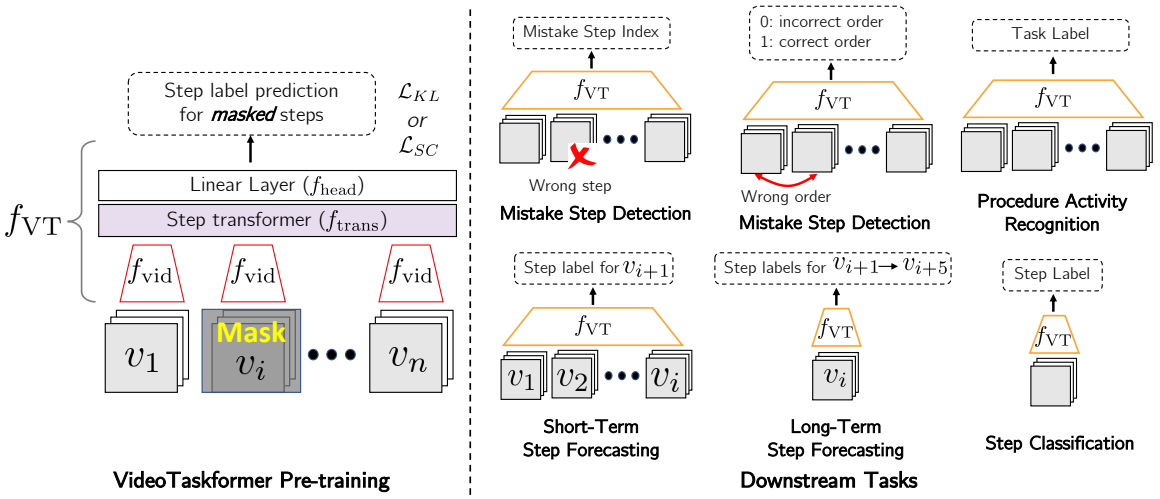

Figure 2: **VideoTaskformer Pre-training (Left).** VideoTaskformer f_{VT} learns step representations for the masked out video clip v_i, while attending to the other clips in the video. It consists of a video encoder f_{vid}, a step transformer f_{trans}, and a linear layer f_{head}, and is trained using weakly supervised step labels. **Downstream Tasks (Right).** We evaluate step representations learned from VideoTaskformer on 6 downstream tasks.

mistakes in videos, which is a crucial component of verifying the quality of instructional video representations. We introduce a mistake detection task and dataset for verifying if the task in a video is executed correctly—i.e. if each step is executed correctly and in the right order.

Additionally, we evaluate representations learned by VideoTaskformer on three existing benchmarks: step classification, step forecasting, and procedural activity recognition on the COIN dataset. Our experiments show that learning step representation through masking pre-training objectives improves the performance on the downstream tasks. We will release code, models, and the mistake detection dataset and benchmark to the community.

2. Learning Task Structure through Masked Modeling of Steps

Our approach for pre-training VideoTaskformer is outlined in Fig. 2. Our framework consists of two steps: pre-training and fine-tuning. During pre-training, VideoTaskformer is trained on weakly labeled data on the pre-training task. For fine-tuning, VideoTaskformer is first initialized with the pre-trained parameters, and a subset of the parameters is fine-tuned using labeled data from the downstream tasks. Each downstream task yields a separate fine-tuned model.

We extend masked language modeling techniques used in BERT [2] and VideoBERT [8] to learn step representations for instructional videos. While BERT and VideoBERT operate on language and visual tokens respectively, VideoTaskformer operates on clips corresponding to steps in an instructional video. By predicting weakly supervised natural language step labels for masked-out clips in the input video, VideoTaskformer learns semantics and long-range temporal interactions between the steps in a task.

Masked Step Modeling. Let $V = \{v_1, \ldots, v_K\}$ denote the visual clips corresponding to K steps in video V. The task for pre-training is to predict categorical natural language step labels for the masked-out steps. While we do not have ground truth step labels, we use the weak supervision procedure proposed by [5] to map each clip v_i to a distribution over step labels $p(y_i \mid v_i)$ by leveraging the noisy ASR annotations associated with each clip. The distribution $p(y_i \mid v_i)$ is a categorical distribution over a finite set of step labels Y. More details are provided in the Supplemental.

Let $M \subseteq [1, \ldots, K]$ denote some subset of clip indices (where each index is included in M with some masking probability r, a hyperparameter). Let $V_{\setminus M}$ denote a partially masked-out sequence of clips: the same sequence as V except with clips v_i masked out for all $i \in M$.

Let f_{VT} represent our VideoTaskformer model with parameters θ. f_{VT} is composed of a video encoder model f_{vid} which encodes each clip v_i independently, followed by a step transformer f_{trans} operating over the sequence of clip representations, and finally a linear layer f_{head} (which includes a softmax). The input to the model is an entire video (of size $K \times L \times H \times W \times 3$) and the output is of size $K \times S$ (where S is the output dimension of the linear layer).

For the downstream tasks, we extract step-aware representations using f_{VT} by feeding an unmasked video V to the model. We then extract the intermediate outputs of f_{trans} (which are of size $K \times D$, where D is the output embedding size).

To predict step labels for masked-out steps at pre-

Downstream Model	Base Model	Pre-training Supervision	Pre-training Dataset	Acc (%)
Transformer	S3D [6]	Unsupervised: MIL-NCE on ASR	HT100M	28.1
Transformer	SlowFast [3]	Supervised: action labels	Kinetics	25.6
Transformer	TimeSformer [1]	Supervised: action labels	Kinetics	34.7
LwDS: Transformer	TimeSformer [1]	Unsupervised: k-means on ASR	HT100M	34.0
LwDS: Transformer w/ KB Transfer	TimeSformer	Distant supervision	HT100M	39.4
VideoTF (SC; fine-tuned) w/ KB Transfer	TimeSformer	Unsupervised: NN on ASR	HT100M	35.1
VideoTF (SC; linear-probe) w/ KB Transfer	TimeSformer	Distant supervision	HT100M	39.2
VideoTF (DM; linear-probe) w/ KB Transfer	TimeSformer	Distant supervision	HT100M	40.1
VideoTF (SC) w/ KB Transfer	TimeSformer	Distant supervision	HT100M	41.5
VideoTF (DM) w/ KB Transfer	**TimeSformer**	Distant supervision	**HT100M**	**42.4**

Table 1: Accuracy of different methods on the **short-term step forecasting** dataset.

Downstream Model	Base Model	Pre-training Supervision	Pre-training Dataset	Acc (%)
Transformer (ASR text) w/ Task label	MPNet			39.0
Transformer	SlowFast [3]	Supervised: action labels	Kinetics	15.2
Transformer	TimeSformer [1]	Supervised: action labels	HT100M	17.0
Transformer w/ Task label	TimeSformer [1]	Supervised: action labels	HT100M	40.1
LwDS: Transformer w/ Task label	TimeSformer	Distant supervision	HT100M	41.3
VideoTF (DM)	TimeSformer	Distant supervision	HT100M	40.2
VideoTF (DM) w/ Task label	**TimeSformer**	Distant supervision	**HT100M**	**46.4**

Table 2: Accuracy of different methods on the **long-term step forecasting** dataset.

training time, we consider two training objectives: (1) step classification, and (2) distribution matching.

Step classification loss. We use the outputs of f_{VT} to represent an S-dimensional prediction distribution over steps, where $S = |Y|$. We form the target distribution by placing all probability mass on the best textual step description y_i^* for each clip v_i according to the weak supervision process. That is,

$$y_i^* = \underset{y \in Y}{\operatorname{argmax}}\, p(y \mid v_i). \tag{1}$$

We calculate the cross entropy between the predicted and target distributions for each masked out clip, yielding the following expression:

$$-\log([f_{\text{VT}}(V_{\setminus M})]_j) \tag{2}$$

where j is the index of y_i^* in Y, i.e., such that $y_i^* = Y_j$. To get the final training objective for a single masked video $V_{\setminus M}$, we sum over all indices $i \in M$, and minimize with respect to θ.

Distribution matching loss. For this objective, we treat the distribution of step labels $p(y_i \mid v_i)$ from weak supervision as the target distribution for each clip v_i. We then compute the KL Divergence between the prediction distribution $f_{\text{VT}}(V_{\setminus M})$ and the target distribution $p(y_i \mid v_i)$ as follows:

$$\sum_{j'=1}^{S} p(Y_{j'} \mid v_i) \log \frac{p(Y_{j'} \mid v_i)}{[f_{\text{VT}}(V_{\setminus M})]_{j'}} \tag{3}$$

We sum over all $i \in M$ and minimize with respect to θ. Following [5], we use only the top-k steps in $p(y_i \mid v_i)$ and set the probability of the remaining steps to 0. Lin *et al.* [5] show that the distribution matching loss results in a slight improvement over step classification loss. For VideoTaskformer, we find both objectives to have similar performance and step classification outperforms distribution matching on some downstream tasks. We use f_{VT} as a feature extractor (layer before softmax) to extract step representations for new video segments.

Downstream Tasks. To show that the step representations learned by VideoTaskformer capture task structure and semantics, we evaluate the representations on 6 downstream tasks—3 new tasks which we introduce (mistake step detection, mistake ordering detection, and long-term step forecasting) and 3 existing benchmarks (step classification, procedural activity recognition, and short-term step forecasting). We describe the dataset creation details for our 3 new benchmarks in the Supplemental.

Mistake Detection. A critical aspect of step representations that are successful at capturing the semantics and structure of a task is that, from these representations, *correctness* of task execution can be verified. We consider two axes of correctness: content (what steps are portrayed in the video) and ordering (how the steps are temporally ordered). We introduce 2 new benchmark tasks to test these aspects of correctness.

• **Mistake step detection.** The goal of this task is to identify which step in a video is incorrect. Each input consists of a video $V = \{v_1, \ldots, v_K\}$ with K steps. V is identical to some unaltered video V_1 that demonstrates a correctly executed task, except that step v_j (for some randomly selected $j \in [1, \ldots, K]$) is replaced with a random step from a different video V_2. The model needs to predict the index j of the incorrect step in the video.

Downstream Model	Base Model	Pre-training Supervision	Pre-training Dataset	Mistake Detection	
				Step	**Order**
Transformer (ASR text) w/ Task label	MPNet [7]			34.2	33.4
Transformer w/ Task Label	SlowFast [3]	Supervised: action labels	Kinetics	28.6	26.1
Transformer w/ Task label	TimeSformer [1]	Supervised: action labels	HT100M	36.0	34.7
LwDS: Transformer	TimeSformer	Distant supervision	HT100M	17.1	11.2
LwDS: Transformer w/ Task Label	TimeSformer	Distant supervision	HT100M	37.6	31.8
VideoTF (SC)	TimeSformer	Distant supervision	HT100M	20.1	15.4
VideoTF (DM) w/ Task label	TimeSformer	Distant supervision	HT100M	40.8	34.0
VideoTF (SC; fine-tuned) w/ Task label	**TimeSformer**	Distant supervision	**HT100M**	**41.7**	**35.4**

Table 3: Accuracy of different methods on the **mistake step and ordering detection** test dataset.

• **Mistake ordering detection.** In this task, the goal is to verify if the steps in a video are in the correct temporal order. The input consists of a video $V = \{v_1, \ldots, v_K\}$ with K steps. The steps are randomly permuted with a 50% probability. The model needs to predict whether the steps are ordered correctly or are permuted.

Step Forecasting. Here we test the network's capabilities in anticipating future steps given one or more past clips of a video.

• **Short-term forecasting.** Consider a video $V = \{v_1, \ldots, v_n, v_{n+1}, \ldots v_K\}$ where v_i denotes a step, and V has step labels $\{y_1, \ldots, y_K\}$, where $y_i \in Y$, the finite set of all step labels in the dataset. Short-term forecasting involves predicting the step label y_{n+1} given the previous n segments $\{v_1, \ldots, v_n\}$ [5].

• **Long-term step forecasting.** Given a single step v_i in a video $V = \{v_1, \ldots, v_K\}$ with step labels $\{y_1, \ldots, y_K\}$, the task is to predict the step labels for the next 5 steps, i.e. $\{y_{i+1}, y_{i+2}, \ldots, y_{i+5}\}$. This task is particularly challenging since the network receives very little context—just a single step—and needs to leverage task information learned during training from watching multiple different ways of executing the same task.

Procedural Activity Recognition. The goal of this task is to recognize the procedural activity (i.e., task label) from a long instructional video. The input to the network is all the K video clips corresponding to the steps in a video, $V = \{v_1, \ldots, v_K\}$. The task is to predict the video task label $t \in \mathcal{T}$ where $\mathcal{T}$ is the set of all task labels for all the videos in the dataset.

Step Classification. In this task, the goal is to predict the step label $y_i \in Y$ given the video clip corresponding to step v_i from a video $V = \{v_1, \ldots, v_K\}$. No context other than the single clip is given. Therefore, this task requires fine-grained recognition capability, which would benefit from representations that contain information about the context in which a step gets performed.

For all of the above tasks, we use the step and task label annotations as supervision. We show the "zero-shot" performance of VideoTaskformer by keeping the video model f_{vid} and the transformer layer f_{trans} fixed and only fine-tuning a linear head f_{head} on top of the output representations. Additionally, we also show fine-tuning results where we keep the base video model f_{vid} fixed and fine-tune the final transformer f_{trans} and the linear layer f_{head} on top of it. The network is fine-tuned using cross-entropy loss with supervision from the step labels for all downstream tasks.

We provide implementation details of our method and describe the datasets and evaluation metrics in the supplemental.

3. Experiments

We evaluate VideoTaskformer (VideoTF) and compare it with existing baselines on 6 downstream tasks: step classification, procedural activity recognition, step forecasting, mistake step detection, mistake ordering detection, and long-term forecasting. Here we report results on mistake detection 3 and forecasting tasks Tab. 2 and 1. The datasets, metrics, baselines, ablations, and additional results are included in the Supplemental.

For short-term forecasting in Tab. 1, we achieve a 3% improvement over LwDS and our unsupervised pre-training using NN with ASR outperforms previous unsupervised methods. We also note that linear-probe outperforms baselines in Tab. 1. VideoTF achieves a strong improvement of 5% over LwDS on the long-term forecasting task 2, 4% on mistake step detection 3, and 4% on mistake ordering detection 3. Adding task labels improves performance on all three tasks.

4. Conclusion

In this work, we introduce a new video model, VideoTaskformer, for learning contextualized step representations through masked modeling of steps in instructional videos. We also introduce 3 new benchmarks: mistake step detection, mistake order detection, and long term forecasting. We demonstrate that VideoTaskformer improves performance on 6 downstream tasks, with particularly strong improvements in detecting mistakes in videos and long-term forecasting. Our method opens the possibility of learning to execute a variety of tasks by watching instructional videos; imagine learning to cook a complicated meal by watching a cooking show.

Acknowledgements. We would like to thank Suvir Mirchandani for his help with experiments and paper writing. This work was supported in part by DoD including DARPA's LwLL, PTG and/or SemaFor programs, as well as BAIR's industrial alliance programs.

References

[1] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *International Conference on Machine Learning (ICML)*, 2021. 3, 4

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2019. 2

[3] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. SlowFast networks for video recognition. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 3, 4

[4] Jie Lei, Linjie Li, Luowei Zhou, Zhe Gan, Tamara L Berg, Mohit Bansal, and Jingjing Liu. Less is more: ClipBERT for video-and-language learning via sparse sampling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1

[5] Xudong Lin, Fabio Petroni, Gedas Bertasius, Marcus Rohrbach, Shih-Fu Chang, and Lorenzo Torresani. Learning to recognize procedural activities with distant supervision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2, 3, 4

[6] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncurated instructional videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[7] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. MPNet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 4

[8] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. VideoBERT: A joint model for video and language representation learning. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 2