Learning Procedure-aware Video Representation from Instructional Videos and Their Narrations

Yiwu Zhong\textsuperscript{1,}\textsuperscript{*}, Licheng Yu\textsuperscript{2}, Yang Bai\textsuperscript{2}, Shangwen Li\textsuperscript{2}, Xueting Yan\textsuperscript{2,}\textsuperscript{†}, Yin Li\textsuperscript{1,\dagger}

\textsuperscript{1}University of Wisconsin-Madison, \textsuperscript{2}Meta AI

yzhong52@wisc.edu, \{lichengyu, yangbai, dylanwen, xyan18\}@meta.com, yin.li@wisc.edu

Abstract

The abundance of instructional videos and their narrations over the Internet offers an exciting avenue for understanding procedural activities. In this work, we propose to learn video representation that encodes both action steps and their temporal ordering, based on a large-scale dataset of web instructional videos and their narrations, without using human annotations. Our method jointly learns a video representation to encode individual step concepts, and a deep probabilistic model to capture both temporal dependencies and immense individual variations in the step ordering. We empirically demonstrate that learning temporal ordering not only enables new capabilities for procedure reasoning, but also reinforces the recognition of individual steps. Our model significantly advances the state-of-the-art results on step classification (\(+2.8%/+3.3\%\) on COIN / EPIC-Kitchens) and step forecasting (\(+7.4\%) on COIN). Moreover, our model attains promising results in zero-shot inference for step classification and forecasting, as well as in predicting diverse and plausible steps for incomplete procedures. Our code is available at https://github.com/facebookresearch/ProcedureVRL.

1. Introduction

Many of our daily activities (e.g. cooking or crafting) are highly structured, comprising a set of action steps conducted in a certain ordering. Yet how these activities are performed varies among individuals. Consider the example of making scrambled eggs as shown in Fig. 1. While most people tend to whisk eggs in a bowl, melt butter in a pan, and cook eggs under medium heat, expert chefs have recommended to crack eggs into the pan, add butter, and stir them under high heat. Imagine a vision model that can account for the individual variations and reason about the temporal ordering of action steps in a video, so as to infer prior missing steps, recognize the current step, and forecast a future step. Such a model will be immensely useful for a wide range of applications including augmented reality, virtual personal assistant, and human-robot interaction.

Understanding complex procedural activities has been a long-standing challenge in the vision community [7, 18, 22, 39, 41, 46]. While many prior approaches learn from annotated videos following a fully supervised setting [13, 27, 69], this paradigm is difficult to scale to a plethora of activities and their variants among individuals. A promising solution is offered by the exciting advances in vision-and-language pre-training, where models learn from visual data (images or videos) and their paired text data (captions or narrations) [30, 43, 54, 70] in order to recognize a variety of concepts. This idea has recently been explored to analyze instructional videos [33, 37], yet existing methods are limited to recognize single action steps in procedural activities.

In this paper, we present a first step towards modeling temporal ordering of action steps in procedural activities by learning from instructional videos and their narrations. Our key innovation lies in the joint learning of a video representation aiming to encode individual step concepts, and a deep probabilistic model designed to capture temporal dependencies and variations among steps. The video representation, instantiated as a Transformer network, is learned by...

\textsuperscript{*}Work done while Yiwu Zhong was an intern at Meta.
\textsuperscript{†}Co-corresponding authors.
matching a video clip to its corresponding narration. The probabilistic model, built on a diffusion process, is tasked to predict the distribution of the video representation for a missing step, given steps in its vicinity. With the help of a pre-trained vision-and-language model [43], our model is trained using only videos and their narrations from automatic speech recognition (ASR), and thus does not require any manual annotations.

Once learned, our model celebrates two unique benefits thanks to our model design and training framework. First, our model supports zero-shot inference given an input video, including the recognition of single steps and forecasting of future steps, and can be further fine-tuned on downstream tasks. Second, our model allows sampling multiple video representations when predicting a missing action step, with each presenting a possibly different hypothesis of the step ordering. Instead of predicting a single representation with the highest probability, sampling from a probabilistic model provides access to additional high-probability solutions that might be beneficial to prediction tasks with high ambiguity or requiring user interactions.

We train our models on a large-scale instructional video dataset collected from YouTube (HowTo100M [38]), and evaluate them on two public benchmarks (COIN [55] and EPIC-Kitchens-100 [10]) covering a wide range of procedural videos and across the tasks of step classification and step forecasting. Through extensive experiments, we demonstrate that (1) our temporal model is highly effective in forecasting future steps, outperforming state-of-the-art methods by a large margin of +7.4% in top-1 accuracy on COIN; (2) modeling temporal ordering reinforces video representation learning, leading to improved classification results (+2.8%/+3.3% for step classification on COIN/EPIC-Kitchens) when probing the learned representations; (3) our training framework offers strong results for zero-shot step classification and forecasting; and (4) sampling from our probabilistic model yields diverse and plausible predictions of future steps.

Contributions. Our work presents the first model that leverages video-and-language pre-training to capture the temporal ordering of action steps in procedural activities. Our key technical innovation lies in the design of a deep probabilistic model using a diffusion process, in tandem with video-and-language representation learning. The result is a model and a training framework that establish new state-of-the-art results on both step classification and forecasting tasks across the major benchmarks. Besides, our model is capable of generating diverse step predictions and supports zero-shot inference.

2. Related Work

Understanding Procedural Activities. Reasoning about procedural activities, including their action steps and the temporal ordering of these steps, has been a central problem in activity recognition. While early works model temporal ordering with stochastic grammars [7, 18, 22, 39, 41, 46], more recent works consider supervised learning to localize steps and predict their ordering by learning from videos with human annotated action steps [8, 10, 13, 27, 55, 69, 71]. To alleviate the burden of costly video annotations, several works propose various forms of weakly supervised settings, with assumptions that the ordered list of steps is given without their temporal boundaries [5, 6, 67, 71], or that the key steps and their ordering remain fixed across all videos [1, 12, 13, 17, 28, 49].

Most of prior methods focus on the tasks of step classification and localization [1, 5, 6, 12, 13, 28, 49, 71]. Others have considered the tasks of step forecasting [48], step verification [42] and procedure planning [67]. Our work also seeks to understand procedural activities. Different from these approaches, our method focuses on learning video representation from videos and their narrations without using human annotations. The resulting video representation can be leveraged for step classification and step forecasting.

Learning from Procedural Videos and Narrations. The success of vision-and-language pre-training has fueled a new line of research that seeks to learn concepts of individual steps from instructional videos and their narrations [19, 36, 50, 62]. For example, Miech et al. [37] propose MIL-NCE to learn representations from instructional videos [38] and their narrations extracted using ASR.

The most relevant work is DistantSup [33], where they propose using distant supervision from a textual knowledge base (wikiHow) [26] to denoise text narrations from ASR. Specifically, DistantSup leverages a pre-trained language model [52] to link step descriptions from wikiHow to text narrations from video ASR results, and thus to create training labels for individual steps in videos. Different from [33], our method models the temporal ordering of steps in procedural activities, thus moves beyond representations of single steps to support temporal reasoning in videos. Further, our method learns from videos and narrations only, with the help of a pre-trained image-language model [43] yet without using a textual knowledge base.

Video-and-Language Pre-Training. A relevant topic is video-and-language pre-training, aiming at learning video representation from videos and their paired natural language descriptions [3, 15, 16, 29, 31, 35, 54, 59, 63, 64, 66, 70], often generated from ASR outputs. Despite the latest development in ASR, automatically-transcribed speech from videos can be rather noisy and lacks precise temporal alignment with the visual content. Several recent works seek to address this challenge. VideoCLIP [63] starts from the pre-trained MIL-NCE model and further improves the model by retrieval augmented training with overlapped video-text pairs. Bain et al. [3] collect a less noisy dataset of video alt-
3. Method

We consider the problem of learning video representation for understanding procedural activities from instructional videos and their narrations. An input video is represented as a sequence of N clips \( \{v_1, v_2, ..., v_N\} \). Each \( v_i \) captures a potential action step in the input video, and the time step \( i \) records the temporal ordering of these clips. The video clips \( \{v_i\} \) can be either segmented by using the timestamps of ASR outputs (as we consider during training), or densely sampled from a video following their temporal ordering (as we use during inference). During learning, we further assume that an ordered set of sentences \( \{s_1, s_2, ..., s_N\} \) is associated with the video clips \( \{v_1, v_2, ..., v_N\} \), with each \( s_i \) describing the action step in video clip \( v_i \). These sentences \( \{s_i\} \) can be the output text from ASR, or given by matching the video clips to a text corpus using an external vision language model [43].

Procedural-aware Video Representation. Our goal is to learn video representation that encodes both action step concepts and their temporal dependencies across a range of procedural activities. Our representation consists of (a) a video encoder \( f \) that extracts a representation \( x_i \) from an input clip \( v_i \) (i.e., \( x_i = f(v_i) \)); and (b) a probabilistic model that characterizes the conditional probability \( p(x_j = f(v_j) | \{x_i = f(v_i)\}_{i \neq j}) \forall j \). This design is highly flexible and supports a number of procedural reasoning tasks. \( f \) offers video representation suitable to classify individual steps in a clip, \( p(x_j | \{x_i\}_{i \neq j}) \) models the temporal dependencies among steps, and can be used to predict the video representation of missing steps and further infer their labels.

Method Overview. To learn our representation, we leverage a pre-trained text encoder \( g \) that remains fixed during learning, and extend the idea of masked token modeling, populated in natural language processing [24]. For each input video and its narrations at training time, we randomly sample a clip \( v_j \) from \( \{v_1, v_2, ..., v_N\} \) and mask it...
out. We then train our model to predict the distribution of $x_j = f(v_j)$ from $\{x_i = f(v_i)\}_{i \neq j}$ (i.e., $p(x_j|x_i)_{i \neq j}$), align the expectation of the predicted distribution $E(x_j)$ with the corresponding text embedding $y_j = g(s_j)$, and match all other video representations $\{x_i = f(v_i)\}_{i \neq j}$ to their text embeddings $\{y_i = g(s_i)\}_{i \neq j}$.

Despite the conceptual similarity, our learning is fundamentally different from masked token prediction. Our method seeks to characterize the distribution of $x_j$ instead of predicting the most likely $x_j$, resulting in a more principled approach to capture the temporal dependencies among steps, as well as the new capability of sampling multiple high-probability solutions for $x_j$. Our method is illustrated in Fig. 2. In what follows, we lay out the formulation of our model, and describe its training and inference schemes.

### 3.1. Modeling Action Steps and Their Ordering

Formally, given an input video with its clips $\{v_1, v_2, ..., v_N\}$ and their narrations $\{s_1, s_2, ..., s_N\}$, our method assumes a factorization of $p(Y = \{y_i\}|X = \{x_i\})$ with video representation $x_i = f(v_i)$ (learnable) and text embedding $y_i = f(s_i)$ (pre-trained and fixed).

$$p(Y|X) = p(y_j|x_j) \cdot p(x_j|x_i)_{i \neq j} \cdot \prod_i p(y_i|x_i) \quad \forall j.$$  

$p(y_j|x_j)$ measures the alignment between a video representation $x_j$ and a text embedding $y_j$. $p(x_j|x_i)_{i \neq j}$ characterizes the distribution of a video representation for a missing step given the representations of all other steps, thereby modeling the temporal ordering of steps. Note that our model is not limited to single step prediction and can be readily extended to predict multiple missing steps.

#### Matching Image and Text Representations

Our model matches the video representation $x_i$ and text embedding $y_i$ in a learned vector space, such that the alignment between them can be measured by cosine similarity. We will later instantiate this definition into a more tractable form for learning. Yet it suffices to notice that $p(y_j|x_j)$ does not involve additional learnable parameters given $x_i$ and $y_i$.

#### Modeling Step Ordering with Diffusion Process

The key challenge lies in the modeling of $p(x_j|x_i)_{i \neq j}$, as the video representation $x_i$ is at least of a few hundred dimensions. To this end, we propose to model $p(x_j|x_i)_{i \neq j}$ using a diffusion process [51, 53] conditioned on observed video representations $\{x_i\}_{i \neq j}$. Here we briefly describe diffusion process in the context of our model, and refer the readers to recent surveys for more technical details [9, 65].

Specifically, we assume a diffusion process that gradually adds noise to the input $x_j$ over $t \in [0, 1, ..., T]$ steps.

$$p(x_j^T|x_j^0) = \prod_{t=1}^{T} p(x_j^t|x_j^{t-1}),$$

$$p(x_j^t|x_j^{t-1}) = \mathcal{N}(\alpha x_j^{t-1}, (1 - \alpha_t)I).$$

where $\alpha_t$ are constant hyper-parameters. The reverse diffusion (denoising) process is parameterized with

$$p(x_j^0|x_j^t) = p(x_j^T|x_j^t) \prod_{i=t}^{T} p(x_j^{i-1}|x_j^i, \{x_i\}_{i \neq j}).$$  

With sufficiently large $T$, $p(x_j^T|x_j^t) \sim \mathcal{N}(0, I)$, i.e., a standard Gaussian noise that is independent of $\{x_i\}_{i \neq j}$. The denoising process is approximated by $p_\theta(x_j^0|x_j^t, \{x_i\}_{i \neq j})$ using a neural network with parameters $\theta$ such that

$$p_\theta(x_j^0|x_j^t, \{x_i\}_{i \neq j}) = \mathcal{N}(\mu_\theta(x_j^t, \{x_i\}_{i \neq j}), \Sigma_\theta(x_j^t, \{x_i\}_{i \neq j}))$$  

In practice, we follow Ho et al. [20] and Tevet et al. [57] to directly predict $x_j^0$ by using a denoising model $h$. With slight abuse of the symbols, we denote

$$\tilde{x}_j^0 = h(x_i^j, \{x_i\}_{i \neq j}, t).$$

$h$ is realized using a Transformer network with the embedding of step $t$ as part of its inputs. Once learned, one can sample from $\mathcal{N}(0, I)$ and apply $h$ through the denoising process to predict $x_j$ based on $\{x_i\}_{i \neq j}$.

### 3.2. Learning from Videos and Their Narrations

Our training approximately maximizes the likelihood of Eq. 1 given a set of training videos and their narrations.

#### Pseudo Labels from CLIP

It is straightforward to directly align video representations to the embeddings of corresponding ASR text. Doing so, however, faces the challenges of low-quality ASR text and imprecise alignment between video and ASR sentences. To address these challenges, we propose to create pseudo labels by leveraging a pre-trained image-language model (e.g., CLIP [43]).

Specifically, we first create a pool of step descriptions in the form of verb phrases (e.g., “add water”, “wear gloves”) parsed from ASR sentences [50], with their embeddings as $\{y_i^{1:K}\}$. Then a trained CLIP model is applied to link each video clip with verb phrases, by matching the averaged visual features across frames with the language embeddings of verb phrases. The resulting matching scores are used as our training target.

Our pseudo labeling instantiates the matching $p(y_i|x_i)$ between video representation and text embedding using

$$p(y_i|x_i) = \text{softmax} \left( \frac{x_j^T y_i}{\tau \|x_i\| \|y_i\|} \right),$$

where $y_i$ is selected from pool of verb phrases, i.e. $y_i \in \{y_i^{1:K}\}$, and $\tau$ is the pre-defined temperature. The matching problem thus is converted into a “classification” problem, making the training feasible.

#### Learning Objective and Training Loss

Our training minimizes an evidence upper bound of the negative log likelihood $\log p(Y|X)$ in Eq. 1. The detailed derivation of
Step Classification

The first term $L_{XE}$ seeks to match observed video representation $\{x_t\}$ to their text embeddings $\{y_t\}$, given by

$$L_{XE} = \frac{1}{N-1} \sum_i H(p_i', p(y_i|x_i)),$$

where $H(\cdot, \cdot)$ is the cross entropy, and $p_i'$ are soft targets given by CLIP matching scores. $p(y_i|x_i)$ defined in Eq. 6 measures the similarity between $x_i$ and $y_i$.

The second term $L_{MSE}$ comes from the Kullback–Leibler (KL) divergence within our diffusion model, and is computed as

$$L_{MSE} = \mathbb{E}_{x_0 \sim \mathcal{P}(x_0^0|\{v_i\}_i, x_{j0})} \left[ \|x_0^0 - \hat{x}_j^0\|_2^2 \right].$$

Note that unlike a standard diffusion model, our model directly predicts $\hat{x}_j^0$. This term is applied at each step $t$.

The third term $L_{MC}$ is derived from matching the predicted video representation $\hat{x}_j^0$ to its text embedding $y_j$.

$$L_{MC} = \mathbb{E}_{x_0 \sim \mathcal{P}(x_0^0|\{v_i\}_i, x_{j0})} \left[ H(p_j', p(y_j|x_j^0)) \right].$$

where $p_j'$ are again soft targets given by CLIP model, and $\hat{x}_j^0$ is denoised from a sampled noise. During training, we adopt Monte Carlo estimation for $\mathbb{E}_p[-\log p(y_j|\hat{x}_j^0)]$, by minimizing $-\log p(y_j|\hat{x}_j^0)$ for each sampled $\hat{x}_j^0$. We attach this term at each step $t$.

A critical design choice lies in $p(y_j|x_j^0)$. $p(y_j|x_j^0)$ is simplified into a score function between a video representation and a finite set of text embeddings (defined using verb phrases). This allows us to reach our loss terms without worrying about global normalization constant as commonly encountered in energy-based models. Indeed, $H(p_j', p(y_j|x_j^0))$ can be interpreted as providing guidance by matching video to text embeddings. This term thus resembles the key idea of classifier guidance, which has shown to be helpful for learning diffusion models [11].

### 3.3. Model Inference

Once trained, our model offers a procedure-aware representation with two key components. First, the video encoder $f(\cdot)$ serves as a feature extractor for any input video clips. Second, the diffusion model, represented as its denoising model $h(\cdot)$, captures the temporal dependencies among steps. Our representation naturally supports a number of tasks. Here we demonstrate how our model can be used for step classification and step forecasting.

**Step Classification.** An input video clip $v$ can be encoded using $f(\cdot)$. The video representation $x = f(v)$ can be directly compared to the text embeddings [43], so as to support zero-shot step classification. Alternatively, an additional classifier can be attached on top of $f$ and further fine-tuned to recognize the action step in input clip.

**Step Forecasting.** A future video clip feature $x_j$ can be sampled from the diffusion model by drawing from a Gaussian distribution and denoising using $h(\cdot)$. The predicted $x_j$ can be further classified into action steps. This prediction can be done using again Monte Carlo estimation given by

$$\mathbb{E}_{x_j \sim \mathcal{P}(x_j|\{v_i\}_i, x_j)} \left[ p(y_j|x_j) \right].$$

Specifically, a noise $x_T$ is first sampled from Gaussian distribution and our denoising model gradually denoises it. At each step $t$, the denoising model $h$ takes a noisy $x_t^j$, predicts clip feature $\hat{x}_j^0$, and diffuses it to $x_t^{j-1}$ based on the sampled noise $x_T^j$, as demonstrated in Eq. 2. After $T$ iterations, the predicted clip feature at $t = 0$ is used to match text embeddings. By sampling noises for multiple times, we can estimate the most likely $y_j$.

However, sampling can be costly. In practice, to obtain top-1 prediction for missing steps, we adopt approximate inference, where the sampled noise is replaced with a fixed zero vector, corresponding to peak in the Gaussian distribution. Our empirical results validate that approximate inference achieves a very close performance as the expectation over multiple sampled noises.

### 4. Experiments and Results

In this section, we first introduce datasets, evaluation protocols and implementation details. Then we demonstrate our results on step forecasting and step classification benchmarks. Finally, we show our qualitative results and conduct ablations to study our model components.

**Datasets.** For *pre-training*, we consider HowTo100M dataset [38] with 130K hours of YouTube tutorial videos. The videos cover various daily tasks, such as foods, housework, vehicles, etc. We use a language parser [50] to extract the verb phrases from ASR sentences of these videos and keep 9,871 most frequent verb phrases. For *fine-tuning*, we train our model on COIN dataset [55, 56] and EPIC-Kitchens-100 dataset [10], respectively. COIN has 476 hours of YouTube videos covering 180 tasks, such as dishes, vehicles, housework, etc. Human annotators summarize 778 unique steps in total (e.g., “stir the egg”), and annotate the temporal boundary and the category of each step in all videos. EPIC-Kitchens-100 dataset [10] has 100 hours of egocentric videos, capturing daily activities in kitchen. Each action in the videos is annotated with an action label and a noun label. There are 97/300 unique actions/nouns in total. We use the human annotations in COIN and EPIC-Kitchens-100 to evaluate the model performance.

**Evaluation Protocols.** Our evaluation considers zero-shot and fine-tuning settings for step classification and step fore-
casting on COIN and EPIC-Kitchens-100 datasets. Zero-shot setting indicates that no human annotation is used during pre-training. The pre-trained model is directly tested on the evaluation dataset. Fine-tuning setting further fine-tunes the pre-trained model using human annotations of action steps. For a fair comparison, we follow the same fine-tuning schemes as previous work in respective benchmarks.

**Implementation Details.** We adopted TimeSformer [4] as our video encoder, and used the Transformer [58] from CLIP’s text encoder as denoising model. We set the maximum step $T$ to 4, maximum length of video sequence as 9, and the number of Transformer layers as 4. We used a trained CLIP model (ViT-B/16) to create pseudo labels and encode step descriptions. Following DistantSup [33], for pre-training we used SGD for 5 epochs and then AdamW [34] for 25 epochs with 128 videos in a batch. For fine-tuning, we used AdamW for 15 epochs with batch size of 64. Temperature $\tau$ was set to 0.02. Additional implementation details can be found in Supplement.

### 4.1. Step Forecasting

**Setup.** We follow the benchmark in DistantSup [33] to evaluate step forecasting on COIN, where top-1 accuracy is reported. Given a video with previous steps, the model anticipates the category of next single step (e.g., “stir the egg”). This task thus requires explicit modeling of the temporal ordering among steps. We only fine-tune the diffusion model while keeping the video encoder frozen.

**Results.** Table 1 compares results of our method with a series of baselines. The closest competitor is DistantSup [33] in L5, which learns from ASR text and an external textual knowledge base [26] using the same video backbone (TimeSformer [4]). We also include other baselines reported in DistantSup, e.g., SlowFast [14], TimeSformer [4], and S3D [61] from L1 to L4, where models are supervised using human-annotated action labels or video ASR text. Our model significantly outperforms all baselines by at least 7.4% for the fine-tuning setting (e.g., 46.8% in L8 vs. 39.4% in L5). Further, we consider a strong baseline for the zero-shot setting by re-purposing CLIP model [43] to match the input video with the descriptions of all step candidates. Comparing L7 and L8, our model outperforms this variant of CLIP by a clear margin (11.3% vs. 9.4%).

A unique property of our model is its ability to output multiple, potentially different predictions. We further evaluate the upper bound of our results by assuming an oracle ranking function that always selects the correction prediction from 5 outputs sampled from our model (Ours (oracle-5)). This oracle further improves the top-1 accuracy from 11.3% to 14.7% in L9, suggesting that our model is able to produce diverse predictions for step forecasting.

### 4.2. Step Classification

**Setup.** Besides step ordering, we also evaluate step classification on COIN and EPIC-Kitchens-100 datasets, where a model is tasked to classify a trimmed video clip into one of the step categories. For COIN, we follow DistantSup [33] to only fine-tune the additional linear layer on top of the pretrained video encoder. For EPIC-Kitchens-100, we fully fine-tune the video encoder, following [2, 25, 33]. We report the accuracy of step classification on COIN, and that of verb, noun, and action on EPIC-Kitchens-100.

**Results.** Table 2 summarizes the results on COIN. We consider baselines as in DistantSup from L1 to L8 (e.g., SlowFast [14], VideoCLIP [63]), where models are trained using either action labels or video ASR text. To support zero-shot inference, we re-implement a model variant (DistantSup†) described in [33]. This model is pre-trained to match video embeddings with language embeddings and thus supports recognizing arbitrary step descriptions in L9. We also report the results of CLIP [43], which creates the pseudo labels for our pre-training in L10. As shown, our model consistently outperforms all the other methods by a clear margin under different settings. For example, ours outperforms CLIP by 18.8% in zero-shot setting (16.6% in L11 vs. 14.8% in L10), and outperforms DistantSup by 2.8% (56.9% in L11 vs. 54.1% in L8) in fine-tuning setting.

Table 3 presents our results on EPIC-Kitchens-100. While TimeSformer [4] in L8 and DistantSup [33] in L9 use the same video encoder architecture as ours, our model...
4.3. Predicting Diverse Future Steps

One of the defining characteristics of our model is that it allows us to sample multiple predictions of video representation corresponding to a future step. This leads to an interesting question about the diversity of the predictions, as partially evaluated in our prior experiments. Here we present further demonstration of this capability by visualizing the step forecasting results, and more interestingly, using these results to generate future video frames.

Fig. 3 presents the visualization for zero-shot step forecasting and key frame generation. In this setting, our model is pre-trained without any human annotation and is directly tested for step forecasting. We show multiple predictions sampled from our diffusion model. Further, we demonstrate that the text description of predicted step can be used to generate the key frames by leveraging the stable diffusion model [45]. To keep the generated images visually consistent with the input video, we let stable diffusion model take one input video frame and the description of predicted step as input and generate an image.

As shown in Fig. 3, our model is capable of forecasting multiple, reasonable next steps (e.g., “flatten the dough”, “bake pizza”), based on which credible future frames can be generated. These results suggest that our model not only predicts meaningful video representations of individual steps, but also captures the variations in step ordering.

4.4. Ablation Studies

We conduct ablation study on COIN, including step classification/forecasting with zero-shot/fine-tuning setting. Additional ablation results can be found in Supplement.

Does modeling of temporal order help? In Table 4, we conduct a comparison on two different pre-training tasks: (1) pre-training by only matching video representations to text embeddings of the verb phrases, and (2) pre-training by our method that combines matching and temporal order modeling. In comparison to pre-training using matching only, our method significantly improves the performance for both zero-shot and fine-tuning settings and across step classification and step forecasting tasks. For example, zero-shot step classification is improved from 13.7% to 16.6%. Our results after fine-tuning attains a major gain of +3.6% and +1.8% for step classification and step forecasting, respectively. Importantly, our method also enables zero-shot step forecasting by predicting future video representations.
Diverse predictions and generated key frames for next step

- flatten the dough
- pour some salt to the garlics
- bake cookies
- bake pizza
- prepare seasonings and side dishes
- put the ingredients into the bowl
- put the ingredients into the bowl
- put the ingredients into the bowl
- prepare seasonings and side dishes

Figure 3. Visualization of zero-shot step forecasting and key frame generation. Without using any human annotation during training, our trained model is directly evaluated on COIN dataset [56]. Given a video recording previous steps (left), our model is capable of forecasting multiple reasonable predictions and each predicted step is further used for key frame generation (right). We adopt stable diffusion [45] for key frame generation, taking inputs as a text description of step and a sampled frame from input video.

<table>
<thead>
<tr>
<th>Model</th>
<th>Pre-training task</th>
<th>Order modeling</th>
<th>Zero-shot (top-1 acc. %)</th>
<th>Fine-tuning (top-1 acc. %)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Step classification</td>
<td>Step forecasting</td>
</tr>
<tr>
<td>Ours</td>
<td>Matching</td>
<td>–</td>
<td>13.7</td>
<td>No zero-shot ability</td>
</tr>
<tr>
<td>Ours</td>
<td>Matching + Ordering</td>
<td>Mask</td>
<td>16.9</td>
<td>10.6</td>
</tr>
<tr>
<td>Ours</td>
<td>Matching + Ordering</td>
<td>Diffusion</td>
<td>16.6</td>
<td>11.3</td>
</tr>
</tbody>
</table>

Table 4. Ablation study of pre-training tasks and order modeling. Our proposed order pre-training not only enables zero-shot forecasting, but also significantly improves zero-shot step classification and the fine-tuning results across evaluation tasks. Our diffusion models further improves mask modeling, especially on fine-tuning step forecasting.

These results suggest that our procedure-aware pre-training can effectively facilitate the learning for both video representation and step ordering in procedure activities.

**Masked Prediction vs. Diffusion Model.** We explore a model variant using the mask prediction, sharing similar spirit as the well-known Masked Language Modeling in BERT [24]. Specifically, this variant is trained to recover the video embeddings of masked video clips so that it can match to the assigned verb phrases. Our diffusion model largely outperforms the results of this variant, especially on the step forecasting (+0.7% and +3.6% for zero-shot and fine-tuning, respectively, as in Table 4). This result indicates that our diffusion model is a more suitable way to capture the variation inside the step ordering.

**Approximate Inference.** In Table 5, we validate that our approximate inference with a single zero-vector can achieve close empirical results as the Monte Carlo estimation (e.g., within 0.1% difference). Monte Carlo estimation computes the weighted average of multiple sampled predictions (e.g., from 5 sampled noises). We run the experiment for 5 times and the variation is small (e.g., ±0.03%). Further, if we assume an oracle ranking function to pick the correct one from sampled predictions, our results can be further boosted by 3.4% on average, suggesting diverse predictions from our model and ample room to improve.

<table>
<thead>
<tr>
<th>Model</th>
<th>Inference Type</th>
<th>Top-1 Acc. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours</td>
<td>Approximation</td>
<td>11.33</td>
</tr>
<tr>
<td>Ours</td>
<td>Expectation</td>
<td>11.34 ± 0.03</td>
</tr>
<tr>
<td>Ours</td>
<td>Oracle</td>
<td>14.73 ± 0.13</td>
</tr>
</tbody>
</table>

Table 5. Ablation study of inference schemes. “Approximation” uses a zero-vector as noise, “Expectation” adopts Monte Carlo estimation, and “Oracle” further assumes an oracle ranking function to pick the correct prediction derived from multiple noises.

### 5. Conclusion

In this work, we presented a model and a training frame work for learning procedure-aware video representation from a large-scale dataset of instructional videos and their narrations, without the need for human annotations. The key strength of our model lies in the joint learning of a video encoder capturing concepts of action steps, as well as a diffusion model reasoning about the temporal dependencies among steps. We demonstrated that our model achieves strong results on step classification and forecasting in both zero-shot and fine-tuning settings and across COIN and EPIC-Kitchens-100 datasets. We believe our work provides a solid step towards understanding procedural activities. We hope that our work will shed light on the broader problem of video-language pre-training.
References


