In this supplement, we describe (1) the derivation of our training loss, (2) the implementation details of data pre-processing, our model architecture, and key frame generation, (3) experiment results on the additional benchmark on COIN dataset, and (4) additional ablation studies on open-vocabulary recognition, the effects of using ASR phrases and backbone architecture. For sections, figures, tables, and equations, we use numbers (e.g., Sec. 1) to refer to the main paper and capital letters (e.g., Sec. A) to refer to this supplement.

A. Derivation of Training Loss

Our method aims at minimizing the negative log likelihood $- \log p(Y|X)$ (Eq. 1 in paper). Here, we provide the derivation of its evidence lower bound, as shown in Eq. A, where $x_i$ are video embeddings learned by our video encoder $f(\cdot)$, $y_i$ are text embeddings offered by a pre-trained text encoder $g(\cdot)$ from CLIP [12] that remains fixed during our training. $\{x_i\}$ and $\{y_i\}$ are observed video and text embeddings, while $x_j$ and $y_j$ are the missing (masked) video and text embeddings.

There are three terms in the evidence lower bound, with each one corresponding to a loss in our main paper. First, $p(y_i|x_i)$ is computed by Eq. 6 of the paper, as a softmax over the cosine similarity between an input video embedding and a set of text embeddings. This term corresponds to the loss $L_{XE}$ (Eq. 8). Second, $p(x_j|\{x_i\}_{i \neq j})$ is approximated using a diffusion model that consists of a diffusion process and an reverse diffusion (denoising) process. This term is performed by the loss $L_{MSE}$ (Eq. 9). Third, $p(y_j|x_j)$ seeks to predict text embedding $y_j$ using the masked video embedding $x_j$. It is again calculated by Eq. 6 of the paper. This term corresponds to $L_{MC}$ (Eq. 10).

B. Additional Implementation Details

Data Pre-processing: During pre-training, we used the timestamps of ASR sentences to segment video clips from full videos. For step classification, the video clips are trimmed by human-annotated step boundaries. When evaluating step classification, multi-view augmentation is applied with 3 clips sampled on the temporal dimension. For step forecasting (both training and evaluation), we cropped 68 seconds of video before the target action and uniformly cut it into 8 video clips as the model input. For HowTo100M [10] and COIN dataset [13, 14], we sampled 1 frame per second. For EPIC-Kitchens-100 dataset [2], we sampled 16 frames per second. The text embedding of each verb phrase was the averaged embedding over 28 action prompts.\footnote{https://github.com/openai/CLIP/blob/main/data/prompts.md#kinetics700}

Model Architecture and Hyper-parameters: We adopted TimeSformer architecture \cite{1} for our video encoder. TimeSformer is a Transformer \cite{15} based model that applies attention mechanism over both spatial and temporal dimension. For denoising model, we used Transformer from CLIP’s implementation\footnote{https://github.com/openai/CLIP} with bi-directional attention. In denoising model, we implemented the maximum time level $T$ as 4, maximum length of video sequence as 9, and the number of Transformer layers as 4. For time variable in diffusion model, we first mapped it into vector representation using position embeddings and then added it to the input of Transformer. When calculating the matching score between video and text embedding (Eq. 4 in main paper), we divided the matching score by a temperature $\tau = 0.02$ when computing the softmax.

Details about Future Key Frame Generation: Future key frame generation is posed as text guided image-to-image translation, where the text is provided by our predicted step and the image is from a sampled frame within the current video. Specifically, we use a pre-trained stable diffusion

---

\*Work done while Yiwu Zhong was an intern at Meta.
\*Co-corresponding authors.

1 University of Wisconsin-Madison, 2 Meta AI

yzhong52@wisc.edu, \{lichengyu, yangbai, dylanwen, xyani1\}@meta.com, yin.li@wisc.edu
$$- \log p(Y|X) = - \log(p(y_j|x_j) \cdot p(x_j|\{x_i\}_{i \neq j}) \cdot \prod_i p(y_i|x_i))$$

$$\leq \sum_i - \log(p(y_i|x_i))$$

cross-entropy loss (L_{CE})

$$+ \sum_{t=1}^T \mathbb{E}_{x_t' \sim p(x_t'|x_0, \{x_i\}_{i \neq j})} \left[ \frac{1}{T} \cdot KL(p(x_t'|x_t, x_0, \{x_i\}_{i \neq j})||p_0(x_t'|x_t, \{x_i\}_{i \neq j})) \right]$$

diffusion model loss (L_{SDEdit})

$$+ \mathbb{E}_{x_j \sim p_0(x_j|\{x_i\}, x_j)} [- \log(p(y_j|x_j))]$$

cross-entropy loss (L_{AC})

(A)

<table>
<thead>
<tr>
<th>Model</th>
<th>Supervision</th>
<th>Pretraining</th>
<th>Dataset</th>
<th>Top-1 Acc. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TSN (RGB+Flow) [13]</td>
<td>Supervised: action labels</td>
<td>Kinetics</td>
<td>73.4*</td>
</tr>
<tr>
<td>2</td>
<td>S3D [16]</td>
<td>Unsupervised: ASR w. MIL-NCE [9]</td>
<td>HT100M</td>
<td>70.2*</td>
</tr>
<tr>
<td>4</td>
<td>TimeSformer [1]</td>
<td>Supervised: action labels</td>
<td>Kinetics</td>
<td>83.5</td>
</tr>
<tr>
<td>5</td>
<td>ClipBERT [5]</td>
<td>Supervised: captions</td>
<td>COCO+VG</td>
<td>65.4</td>
</tr>
<tr>
<td>6</td>
<td>VideoCLIP [17]</td>
<td>Unsupervised: ASR</td>
<td>HT100M</td>
<td>72.5</td>
</tr>
<tr>
<td>8</td>
<td>DistantSup [7]</td>
<td>Unsupervised: ASR + wikiHow</td>
<td>HT100M</td>
<td>88.9</td>
</tr>
<tr>
<td>9</td>
<td>Ours</td>
<td>Unsupervised: ASR</td>
<td>HT100M</td>
<td><strong>90.8</strong></td>
</tr>
</tbody>
</table>

Table A. Procedural activity classification on COIN dataset. * indicates the model is fully fine-tuned on COIN dataset.

model and employ SDEdit [8]. SDEdit adds noise to the sampled input video frame, and then denoises the resulting image using stable diffusion model and the text of our predicted step, in order to generate a future video frame.

C. Additional Benchmarks

C.1. Procedural Activity Classification

We follow the benchmark in DistantSup [7] to evaluate procedural activity recognition on COIN with top-1 accuracy reported. Given a video that has recorded multiple steps, the model classifies the entire video into an activity category (e.g., “make coffee”). Similar to step forecasting, we only fine-tune the diffusion model to predict activity category, with the frozen video encoder as a feature extractor.

In Table A, we compare our model with a series of baselines as in DistantSup [7], such as SlowFast [3], TimeSformer [1] and S3D [16]. These baselines are pre-trained by either human-annotated action labels or video ASR sentences. Our closest competitor is DistantSup [7] which learns individual action concepts by leveraging an external text knowledge base (wikiHow). Our model clearly outperforms all baseline models by a large margin (e.g., +1.9 over DistantSup in L8). Our experimental results suggest that our order pre-training approach, which captures the order among steps, can also improve the recognition of the entire sequence of steps, even if it was not designed for this task.

D. Additional Ablation Studies

We present additional ablation studies on our model. The experiment settings follow the ablation study in the main paper, unless otherwise noticed.

Can our model identify open-vocabulary step concepts? Part of our learning objective is to match the video representations with text embeddings. Such a design allows

---

3https://github.com/CompVis/stable-diffusion

Table B. Visualization of step concepts. We show COIN steps (left) and the step descriptions in pre-training (HowTo100M) that have common verb/noun (right).
To the step concepts considered in pre-training and supports open-vocabulary step recognition. We conjecture that our model has learned the components from similar phrases (e.g., “fry chicken” and “lay eggs” shown in Table B), by learning to project video embeddings into the semantic space defined by the text embeddings of CLIP.

**Are ASR phrases sufficient to learn step concepts?** We propose to use the step phrases parsed from video ASR sentences for learning step concepts. The latest work DistantSup [7] found that external text corpus for procedure activities (e.g., wikiHow [4]) can largely reduce the noise in ASR sentences. In this section, we explore using wikiHow sentences to pre-train our model.

In Table C, we compare our model with a variant pre-trained using wikiHow sentences, following [7]. Our results demonstrate that ASR phrases are sufficient to achieve competitive results across tasks and settings (e.g., +0.7/+0.9 for step forecasting across zero-shot and fine-tuning settings). In other word, our model only requires ASR phrases generated from audio transcriptions of videos, without the need of an external text corpus describing the procedural activities as in [7].

**Backbone Architecture of Video Encoder.** In Table D, we study the effects of backbone architectures for our video encoder. We replace the default backbone TimeSformer with MViT-S [6] which is also a widely-used architecture for video encoders. We slightly increase the frame sampling rate of MViT-S from the default value of 4 to 6 so that the encoder can take a longer video (e.g., on COIN, the average duration of a step is 14 seconds). TimeSformer consistently outperforms MViT-S across tasks (e.g., +4.1 on step classification). We conjecture that TimeSformer, which samples 8 frames from consecutive 256 frames, is better suited for recognizing actions with long durations, such as COIN steps. Conversely, MViT-S, which samples 16 frames from consecutive 96 frames, may perform better for recognizing actions with short durations and high-speed motion.

### Table C. Ablation study on different sources of step descriptions. Top-1 accuracy (%) on COIN dataset is reported. All models are pre-trained on a subset of HowTo100M dataset, defined by [1, 7].

<table>
<thead>
<tr>
<th>Source</th>
<th>Zero-shot Classification</th>
<th>Zero-shot Forecasting</th>
<th>Fine-tuning Classification</th>
<th>Fine-tuning Forecasting</th>
</tr>
</thead>
<tbody>
<tr>
<td>wikiHow sentences</td>
<td>11.6 8.3</td>
<td>48.6 38.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASR phrases</td>
<td>11.8 9.0</td>
<td>47.8 38.9</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table D. Ablation study on the different architectures of video encoder.** All models are pre-trained on HowTo100M dataset.

<table>
<thead>
<tr>
<th>Source</th>
<th>Zero-shot Classification</th>
<th>Zero-shot Forecasting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours (TimeSformer)</td>
<td>16.6 11.3</td>
<td></td>
</tr>
<tr>
<td>Ours (MViT-S)</td>
<td>12.5 9.0</td>
<td></td>
</tr>
</tbody>
</table>
References


