output embedding size).

To predict step labels for masked-out steps at pre-training time, we consider two training objectives: (1) step classification, and (2) distribution matching. We describe them below in the context of Masked Step Modeling.

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,

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

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 [13], 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.* [13] 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.

3.2. 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 Sec. 4.

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. More specifically, 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 goal of the task is to predict the index j of the incorrect step in the video.
- **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. There is a 50% probability that V is identical to some (correctly ordered) video $V_1 = \{v_1^1, \ldots, v_K^1\}$, and there is a 50% probability that the steps are randomly permuted. That is, $v_i = v_{\pi_i}^1$ for some random permutation π of indices $[1, \ldots, K]$. The goal of the task is to predict whether the steps are ordered correctly or are permuted.

Step Forecasting. As another way to evaluate how learned step representations capture task structure, we test the capabilities of our model in anticipating future steps given one or more 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 $\{v_1, \ldots, v_n\}$ [13].
- **Long-term step forecasting.** We introduce the challenging task of 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-