StepFormer: Self-supervised Step Discovery and Localization in Instructional Videos

Supplemental Material

Nikita Dvornik[1] Isma Hadji[1] Ran Zhang[1] Konstantinos G. Derpanis[1,2]
Richard P. Wildes[1,2] Allan D. Jepson[1]
[1]Samsung AI Centre Toronto, [2]York University
{kosta, wildes}@eecs.yorku.ca, {isma.hadji, ran.zhang, allan.jepson}@samsung.com

1. Summary

In this supplemental material, we elaborate on the details of the proposed approach and experimental validation setup. In Section 2, we begin by supplementing the description of data processing provided in Section 3.1 of the main paper with an explicit illustration of the subtitle processing outcome. Next, we define the regularization losses in Section 3. Then, in Section 4, we give a more detailed description of the used baselines. Finally, we present the additional ablation studies in Section 5.

2. Subtitle processing

To train StepFormer, we derive supervision from video subtitles (or narrations), as described in Section 3.1 of the main paper. More precisely, we run the narrations through punctuation [7] and co-reference resolution [6] modules, followed by a dependency parser to discover verb-phrases of the form *verb+(prt)+dobj+(prep+pobj)*. As a result, we transform long subtitle text into an ordered sequence of verb phrases, some of which describe groundable actions and procedure steps occurring in the video. We demonstrate an example of the verb phrase extraction in Figure 1, and additionally highlight the phrases that were matched to step slots during training (in the last epoch). This example, confirms that a subset of the extracted verb phrases indeed contain important information about procedure steps. Notably, most of the relevant verb phrases get selected by Drop-DTW for supervision, as shown in the underlined steps on the right of Figure 1.

3. Regularization losses

As described in Section 3.3 of the main paper, we train StepFormer with verb phrases supervision and employ two extra regularizers, acting on the step slots.

Diversity regularizer. The first regularizer, $\mathcal{L}_{\text{div}}$, enforces diversity among step slots. Precisely, given the step slots $\mathbf{s} \in \mathbb{R}^{K \times d}$ extracted from video $\mathbf{v} \in \mathbb{R}^{N \times d}$ using the transformer, $\mathcal{T}$, i.e., $\mathbf{s} = \mathcal{T}(\mathbf{v})$, the diversity regularizer encourages low cosine similarity among the step slots $\mathbf{s}_i$ as follows:

$$\mathcal{L}_{\text{div}} = \frac{1}{K(K-1)} \sum_{i=1}^{K} \sum_{j \neq i} \cos(\mathbf{s}_i, \mathbf{s}_j), \tag{1}$$

Where K is the number of predicted slots. The diversity regularizer promotes slot diversity and improves performance by removing duplicate slots as validated in Table 3 of the main paper.

Smoothness regularizer. The second regularizer, $\mathcal{L}_{\text{smooth}}$, enforces that step slots attend to video content smoothly. Intuitively, due to natural video continuity, we expect the attention of step slots in the video to change smoothly, and be similar for close frames. Given the video, $\mathbf{v} \in \mathbb{R}^{N \times d}$, and the corresponding step slots, $\mathbf{s} \in \mathbb{R}^{K \times d}$, the attention, $\mathbf{a} \in \mathbb{R}^{N \times K}$, of all steps in the video is defined as follows:

$$\mathbf{a} = \text{softmax}(\cos(\mathbf{v}, \mathbf{s})/\gamma), \tag{2}$$

where $\gamma = 0.03$ is the softmax temperature, softmax is taken along the last dimension, and $\cos(\mathbf{v}, \mathbf{s}) \in \mathbb{R}^{N \times K}$ is a matrix of cosine similarities, *i.e.* $\cos(\mathbf{v}, \mathbf{s})_{ij} = \cos(\mathbf{v}_i, \mathbf{s}_j)$.

To define the regularizer on the attention vectors, we draw inspiration from the video representation learning literature [3]. In particular, we require that, for a given attention vector $\mathbf{a}_i \in \mathbb{R}^K$ (at frame i), the attention vectors $\mathbf{a}_k$ of the neighboring frames, *i.e.* $\|i - k\| \leq M$, are similar, and the attention vectors, $\mathbf{a}_j$ of distant frames, *i.e.* $\|i - j\| > M$, are dissimilar, where M is the size of the positive neighborhood. We implement this requirement using the MIL-NCE contrastive loss on the attention according to

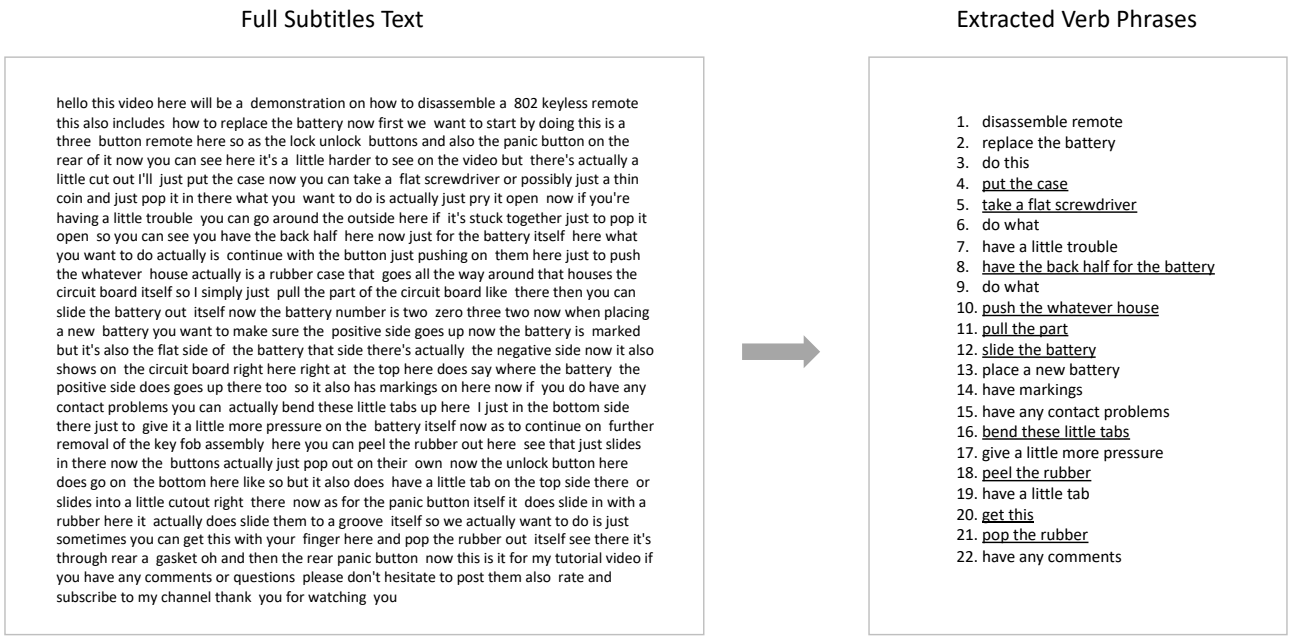

Figure 1. **Verb-phrase extraction from subtitles.** (left) Original video subtitles. (right) Verb phrases extracted from the subtitles. The underlined verb phrases are the ones chosen by Drop-DTW for step slot supervision at end of training.

Method	Unsup. Segmentation				Zero-shot Localization			
	F1	Prec.	Rec.	MoF	IoU	Prec.	Rec.	MoF
Ours (24 slots)	27.9	21.4	40.5	**45.3**	21.6	30.9	44.2	66.4
Ours (48 slots)	26.5	20.7	39.8	40.6	**24.2**	**34.1**	40.1	67.7
Ours (2 layers)	23.2	18.4	34.1	39.6	14.9	24.9	26.8	**68.1**
Ours (4 layers)	27.8	21.4	41.2	43.6	20.8	28.9	38.8	67.4
Ours (8 layers)	26.6	22.0	39.7	42.5	22.7	32.4	**45.5**	66.7
Ours (6 layers, 32 slots)	**28.3**	**22.1**	**42.0**	41.9	23.7	32.9	43.1	67.1

Table 1. Ablation study of StepFormer's training and inference components on CrossTask.

$$\mathcal{L}_{\text{smooth}} = -\log \frac{1}{L} \sum_{i=1}^{L} \frac{\sum_{j \in \mathcal{P}_i} f(\mathbf{a}_i, \mathbf{a}_j)}{\sum_{l \in \mathcal{I}} f(\mathbf{a}_i, \mathbf{a}_l)}, \tag{3}$$

Where $\mathcal{I}$ is the set of L frame indices sampled at random, $\mathcal{P}_i$ is a subset of $\mathcal{I}$ indices that form a positive pair with i, *i.e.* lie in the positive neighborhood of M frames, and $f(x, z) = \exp(\cos(x, z)/\gamma)$, where $\gamma = 0.03$ is a scaling temperature.

4. Baselines

To verify the effectiveness of StepFormer, we compare our model to three baselines: Kukleva et al. [2], Elhamifar et al. [1], and Shen et al. [5]. We consider all the baselines as weakly-supervised, as they use information about the video task label during training. Kukleva et al. [2] and Elhamifar et al. [1] train a model purely from video; thus, we use these baselines only in unsupervised step localization (Section 4.2 of the main paper). More similar to StepFormer, Shen et al. [5] extract prototypes from a video on the fly and supervises them with text subtitles. In principle, their prototypes should follow the temporal order and be alignable with text features, similar to our step slots. Hence, we compare StepFormer to Shen et al. also in the zero-shot step localization setup (Section 4.3 of the main paper). While Elhamifar et al. directly uses video task labels for supervision, Kukleva et al. and Shen et al. use such labels implicitly, *i.e.* they train a separate model for each task using only the videos that belong to that task. In this work, we adapt the methods of Kukleva et al. and Shen et al. to completely unsupervised training (*i.e.* without video labels), by merging all tasks (and their videos) into a single dataset-level task, thereby not revealing the task-specific labels during training. Unlike StepFormer, all the baseline methods must be trained and tested on the same dataset (as they learn task-specific step prototypes), and cannot generalize to new data. For direct comparison with StepFormer, we train all the baselines using the same video features [4], and fix the same training and testing splits in every dataset.

5. Ablation study

In this section, we complement Section 4.4 of the main paper with additional ablations of StepFormer components on the CrossTask dataset. Specifically, we vary the number of transformer decoder layers and the number of output step slots used to describe a video, and report the results in Table 1. First, as the table demonstrates, increasing the number of transformer layers to 6, *i.e.* the default value used in the main paper, helps improve the perfor-

mance. However, further increasing the number of layers actually hurts. We attribute this effect to optimization difficulty of larger models. Second, using 32 step slots to solve unsupervised step localization seems to be optimal for unsupervised step discovery and localization. However, to solve zero-shot step localization, more step slots seem to work better. We attribute the increased zero-shot step localization performance with 48 slots to the improved text-to-slot matching step, as 48 slots offer more freedom in the matching. Nevertheless, we elect to use the setup consisting of 6 layers and 32 step slots as it offers a good compromise in terms of performance on both target tasks.

References

[1] Ehsan Elhamifar and Dat Huynh. Self-supervised multi-task procedure learning from instructional videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2

[2] Anna Kukleva, Hilde Kuehne, Fadime Sener, and Jurgen Gall. Unsupervised learning of action classes with continuous temporal embedding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[3] Hanwen Liang, Niamul Quader, Zhixiang Chi, Lizhe Chen, Peng Dai, Juwei Lu, and Yang Wang. Self-supervised spatiotemporal representation learning by exploiting video continuity. In *AAAI Conference on Artificial Intelligence*, 2022. 1

[4] Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Jason Li, Taroon Bharti, and Ming Zhou. UniVL: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*, 2020. 2

[5] Yuhan Shen, Lu Wang, and Ehsan Elhamifar. Learning to segment actions from visual and language instructions via differentiable weak sequence alignment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[6] spaCy. Industrial-strength natural language processing. https://spacy.io/. 1

[7] Ottokar Tilk and Tanel Alumäe. Bidirectional recurrent neural network with attention mechanism for punctuation restoration. In *INTERSPEECH*, 2016. 1