Supplementary Material for CLOT: Closed Loop Optimal Transport for Unsupervised Action Segmentation

Elena Bueno-Benito, Mariella Dimiccoli
Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain
{ebueno, mdimiccoli}@iri.upc.edu

This supplementary material provides extended analyses and implementation details to support our proposed CLOT paper. It includes sensitivity studies on key parameters, complexity comparisons with ASOT, and additional ablations to isolate the effect of each component. We also provide full hyperparameter settings and extra qualitative results across all datasets to further illustrate the advantages of our approach.

1. Sensitivity Analysis

Effect of K'. K' controls the capacity of the decoder, that is the *maximum* number of predicted segments, the actual number is learned. We set $K' = K + \text{nseg}$, where K is the number of action classes and $\text{nseg} \in \mathbb{Z}$ is a dataset-specific offset. See the results for $\text{nseg} \in [-6, +6]$ in Fig. 1 (first three tables). The straight lines indicate the SOTA. CLOT achieves equal or superior performance in F1 and mIoU across nearly all nseg in BF, 50S(Mid), 50S(Eval), and DA. For MoF, CLOT consistently outperforms or equals the SOTA $\forall$nseg in BF, 50S (Mid), and YTI. In particular, negative nseg can help reduce over-segmentation in presence of background or very unbalanced action durations.

Effect of M. We apply SWD exclusively to the initial frame-to-action matching, as this is where raw visual features are directly compared to action prototypes. At this stage, SW best captures the underlying geometric structure and distributional structure by projecting features onto multiple one-dimensional subspaces. Later stages use refined embeddings where this structure is no longer preserved, making SWD less meaningful. We use $M = P \times d$ projections, where d is the feature dimension and with $M \in [50, 100]$ typically sufficient for convergence (see [1, 2]). As shown in Fig. 1, we observe stable performance across $P \in [1, 6]$.

2. Complexity

CLOT introduces a multi-level cyclic learning mechanism, solving three OT problems instead of one, which could lead to increased computational costs. However, to maintain efficiency, we incorporate an entropy-regularized solver, similar to ASOT [4], which accelerates the OT optimization process while maintaining accuracy. This approach leverages sparsity structures in the cost matrices, enabling faster convergence with a computational complexity of $O(NK)$ per iteration. As a result, CLOT remains scalable to large datasets, efficiently handling videos with thousands of frames while keeping processing times competitive. The overall runtime remains comparable to ASOT, with only a moderate increase in memory usage due to additional attention layers in the decoder. Tab. 1 shows the GFLOPs and per-batch training time, using `flop counter` in PyTorch. Despite CLOT using 3 Optimal Transport (OT) problems and adding architectural complexity, it increases Floating Point Operations (FLOPs) and runtime moderately.

	Time (s)	BF	YTI	50S (Eval)	50S (Mid)	DA
ASOT	2.2	24.7	110.7	27.8	36.0	86.8
CLOT	3.7	48.4	117.7	36.9	62.8	125.8

Table 1. GigaFLOPs and the average training runtime.

3. Additional Ablation Study

Parallel decoding vs. autoregressive decoding To assess the influence of decoding strategies on segmentation quality, we compare our parallel decoder with its autoregressive counterpart, similar to [3] across both activity-level and video-level evaluations (Tab 2). The parallel design, aligned with CLOT's architecture, avoids error propagation by predicting segments simultaneously rather than sequentially. Results indicate consistent improvements across all datasets and metrics, with notable gains in F1 and mIoU, especially for short or overlapping actions, highlighting the benefit of structured segment-level modelling over temporally fragile autoregressive predictions.

Comparison to ASOT. ASOT corresponds to our **1st stage** (frame-to-action) without SWD and FD. However, it is not meaningful to treat SWD or FD as independent add-ons to ASOT, as our method redefines the OT formulation

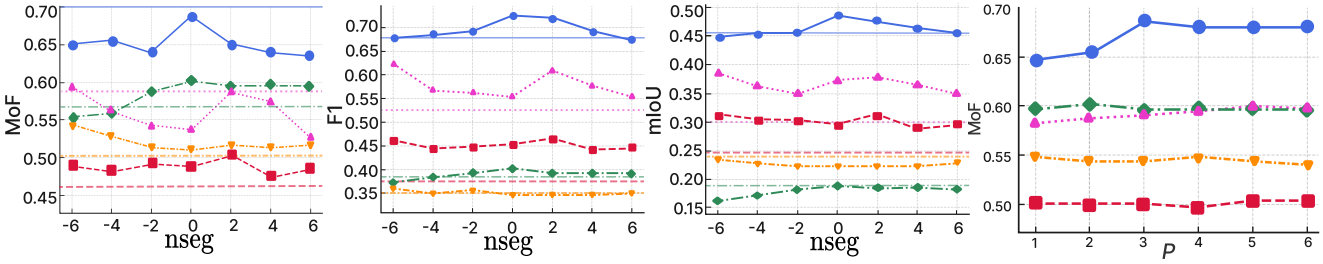

Figure 1. Sensitivity analysis for MoF, IoU and F1 on nseg and and MoF on P.
DA (○, solid), **FS (Mid)** (□, dashed), **FS (Eval)** (△, dotted), **BF** (◇, dash-dot), and **YTI** (▽, custom dash).

Eval	Decoder	Breakfast			YTI			50Salads (Mid)			50Salads (Eval)			DA		
		MoF	F1	mIoU	MoF	F1	mIoU	MoF	F1	mIoU	MoF	F1	mIoU	MoF	F1	mIoU
Activity	Autoregressive	55.6	35.2	16.6	51.5	34.4	20.6	52.3	44.3	30.2	55.9	55.0	32.7	35.3	25.3	13.0
	Parallel	**60.1**	**40.1**	**18.5**	**54.4**	**36.7**	**23.4**	**50.6**	**46.6**	**31.4**	**59.4**	**63.2**	**38.8**	**68.8**	**72.6**	**48.1**
Video	Autoregressive	63.1	53.5	34.9	68.7	60.5	44.2	66.7	60.0	41.2	61.3	62.4	36.4	42.6	30.1	15.8
	Parallel	**66.3**	**55.9**	**37.1**	**69.3**	**60.8**	**48.2**	**69.4**	**63.8**	**45.0**	**64.6**	**69.7**	**42.5**	**73.5**	**75.2**	**52.4**

Table 2. Comparison between Autoregressive and Parallel decoders evaluated at activity and video level across the four datasets.

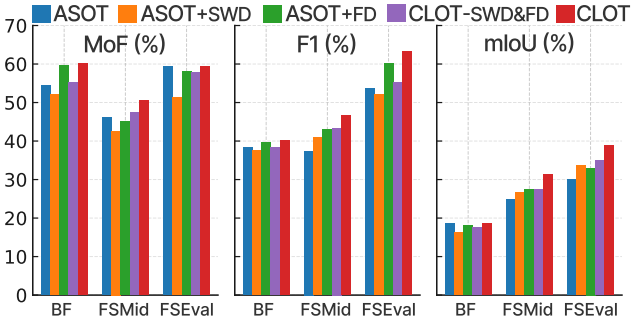

Figure 2. Comparative results, isolating the impact of each component.

itself by integrating both components to enhance representations prior to transport resolution. These are not plug-and-play modules but part of a coherent architectural redesign. In Fig. 2, both components bring measurable benefits over ASOT, while their absence in CLOT underscores the critical role they play in overall performance.

4. Implementation Details

Hyperparameter Settings. In Tab. 3, we provide a summary of the hyperparameter settings used in our experiments. The values are categorized based on their role in the encoder, decoder, and optimal transport (OT) components of the model.

Computing Resources. All experiments were conducted on a single NVIDIA GeForce RTX 3090 GPU (24GB) with CUDA 12.3, providing the necessary computational resources for training and evaluation.

5. Additional Qualitative Results

For a more comprehensive analysis, we present additional qualitative results from all four datasets in Fig. 3. These visualizations illustrate how CLOT effectively refines segmentation boundaries compared to ASOT and other baseline methods.

References

[1] Ishan Deshpande, Yuan-Ting Hu, Ruoyu Sun, Ayis Pyrros, Nasir Siddiqui, Sanmi Koyejo, Zhizhen Zhao, David Forsyth, and Alexander G. Schwing. Max-sliced wasserstein distance and its use for gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[2] Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and Gustavo Rohde. Generalized sliced wasserstein distances. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019. 1

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017. 1

[4] Ming Xu and Stephen Gould. Temporally consistent unbalanced optimal transport for unsupervised action segmentation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1, 3

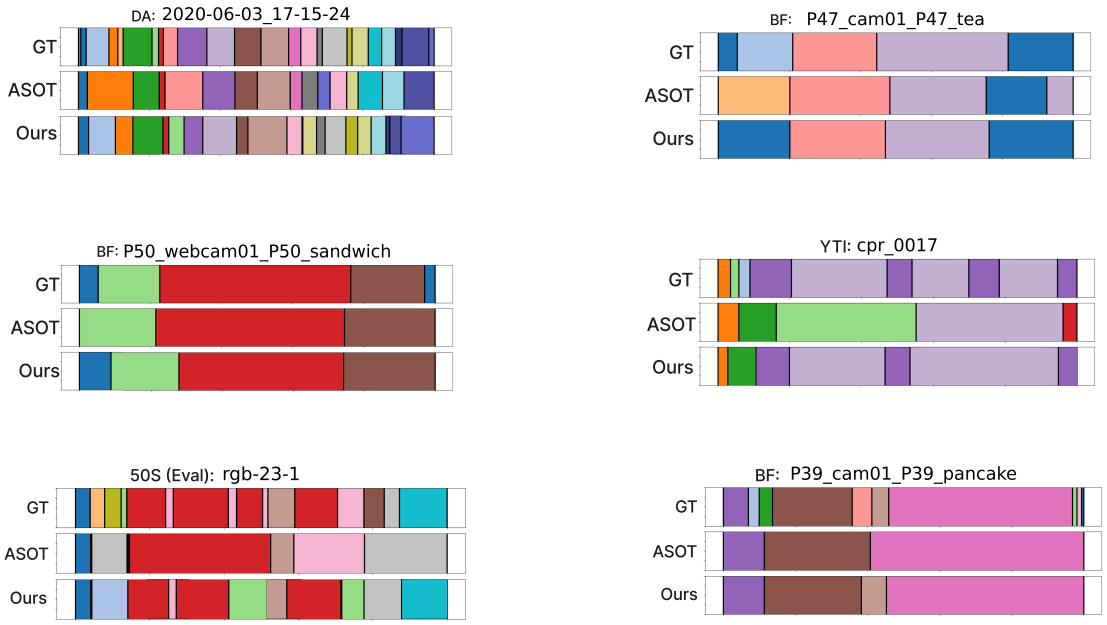

Figure 3. **Qualitative results.** We display the ground-truth (GT), the results of CLOT (Ours), and ASOT [4]. Results from different datasets and activities for comparison.

Hyperparameter	**Value**
Learning Rate	0.001 (BF and DA), 0.005 (YTI and 50S)
Batch Size	2
Epochs	15(BF), 20 (YTI), 30(50S), 70 (DA)
Optimizer	Adam
Loss Function	Cross-Entropy
Dropout Rate	0.5
Weight Decay	$1e^{-4}$
Activation Function	ReLU
Decoder: heads	8
Decoder: dropout	0.5(BF), 0.2(YTI), 0.1(50S-Mid), 0.2(50S-Eval), 0.5(DA)
Decoder: Num of layer	2(BF), 2(YTI), 1(50S-Mid), 4(50S-Eval), 3(DA)
$nseg$	0(BF), -6(YTI), 2(50S-Mid), -6(50S-Eval), 0(DA)
P	2(BF), 2(YTI), 1(50S-Mid), 4(50S-Eval), 3(DA)
ρ	01.5(BF), 0.2(YTI), 0.1(50S-Mid), 0.1(50S-Eval), 0.25(DA)
λ	0.1(BF), 0.08(YTI), 0.11(50S-Mid), 0.2(50S-Eval), 0.16(DA)
Nr	0.04(BF), 0.02(YTI), 0.1(50S-Mid), 0.1(50S-Eval), 0.25(DA)

Table 3. Hyperparameter Settings