Supplementary Material

In the following section we provide further details regarding the implementation and the empirical results reported in the main paper.

A Pseudo-Label Generation

A.1 Timestamp label smoothing

As timestamps are located arbitrarily within action segments, it is reasonable to assume that adjacent temporal locations are associated with the same action (on average). Hence, when solving the sparse system of linear equations described at Equation (5) in the main paper, we perform timestamp label smoothing on the timestamp vector $\mathbf{y}_a$ by applying a linear decay function to each non-zero value over a specified window size.

In Table 1 we quantify the impact of the timestamp label smoothing and demonstrate a modest improvement in performance at mid-values.

Dataset	Smoothing window size				
	0	5	10	20	30
Accuracy					
Breakfast	70.8	70.9	71.0	**71.0**	70.9
50Salads	80.5	80.6	**80.7**	80.5	80.2
GTEA	78.3	78.9	**78.6**	77.0	74.6
F1@10					
Breakfast	96.5	96.5	96.6	**96.7**	96.6
50Salads	99.5	99.5	**99.5**	99.5	99.4
GTEA	97.3	97.5	**96.0**	95.9	94.0

Table 1: The impact of timestamp label smoothing.

A.2 Stand-alone random walk parameters

We evaluate the effectiveness of the proposed random walk method in generating dense pseudo-labels from timestamps. We examine the following hyper-parameter ranges:

- Similarity method: $\{cosine, euclidean\}$
- Sharpening method: $\{exp, power\}$
- Sharpening factor $\beta \in \{10, 20, 30, 40\}$
- Laplacian neighborhood size $K \in \{0, 5, 15, 30\}$
- Timestamp label smoothing size $L \in \{0, 5, 10, 20, 30\}$
- Temporal prior weight $\gamma \in \{1e-2, 1e-3, 1e-4\}$

When we consider the cosine similarity for the edge weights, we use the following sharpening method:

$$\tilde{w}_{i,j} \triangleq \left(\mathbf{f}_i \cdot \mathbf{f}_j / ||\mathbf{f}_i|| \cdot ||\mathbf{f}_j||\right)^{\beta}.$$

We evaluate the performances of the above configurations over the validation set (averaged over multiple cross validation folds). The best configuration is as follows: we use $euclidean$ distance as our similarity method with exp sharpening (as presented in Section 3.2 in the main paper). We set $\beta = 30$, $K = 15$, $\gamma = 1e-3$ and $L = 10$ for 50Salads/GTEA and $L = 20$ for Breakfast. The same set of hyper-parameters is also used in Section 4.3.1 when measuring the impact of the random walk parameters and for the rest of the reported RWS training.

B Unified Random Walk

B.1 Training loss weights

When training RWS we scanned the following ranges of α (truncated Laplacian smoothing weight) and δ (confidence weight): $\{0, 0.025, 0.05, 0.075, 0.1, 0.15, 0.2\}$. We report results for the best performing models over the validation folds. See Table 2 for the best performing loss weights over the three datasets.

Dataset	α	δ
Breakfast	0.15	0.1
50Salads	0.075	0.075
GTEA	0.15	0.1

Table 2: The different losses weights for the three datasets.

B.2 Impact of the random walk refinement mechanism

The proposed unified training objective, RWS, utilizes three random walk use-cases. In Table 3 we present the

impact of the refinement mechanism. As seen, the refinement is an important addition to the model performance improving on average the F1, edit score and accuracy by 0.9, 0.5 and 1 points respectively.

Model	$F1@\{10, 25, 50\}$	Edit	Acc
	Breakfast		
RWS w/o	69.2 63.1 45.3	70.5	58.0
RWS	70.9 64.7 44.8	71.1	60.2
	50Salads		
RWS w/o	75.1 71.2 54.2	67.8	69.6
RWS	76.7 72.8 55.5	69.3	70.0
	GTEA		
RWS w/o	80.6 73.4 55.9	76.9	58.9
RWS	80.9 74.1 56.3	76.2	59.3

Table 3: Comparing RWS with and without the random walk refinement mechanism.