Sparse-Dense Side-Tuner for efficient Video Temporal Grounding

David Pujol-Perich, Sergio Escalera, Albert Clapés
Universitat de Barcelona and Computer Vision Center, Barcelona, Spain
{david.pujolperich, sescalera, aclapes}@ub.edu

Supplementary Material

In this supplementary material, we provide additional details and various ablation studies of the main contributions of our work. Concretely. Sec. A describes the implementation details of our proposed SDST architecture and of our tested evaluation setups. Sec. B presents a detailed description of the datasets used in our experiments: QVHighlights, TACoS, and Charades-STA. Sec. C expands on the objective functions used by SDST, covering the losses used for both Highlight Detection (HD), Moment Retrieval (MR), as well as alignment losses critical for effective side-tuning. Sec. D provides the necessary background related to the deformable attention mechanism. We then conduct a series of ablation studies such as Sec. E that evaluates SDST over other relevant baselines under a fair comparison using only InternVideo2-1B features. Sec. F extends the efficiency analysis of the main text, analyzing multiple efficiency metrics like memory, parameters or running time. Sec. G ablates over the training stability of our proposal, and consequently the difficulty of its corresponding optimization. Sec. H extends our comparison against other Parameter-Efficient and Memory-Efficient Fine-Tuning methods. Sec. I examines the impact of parameter sharing across intermediate refinement layers and Sec. J studies the contribution of intermediate features when leveraging InternVideo2-1B. Additionally, Sec. K provides an in-depth analysis of our proposed RDSA mechanism, including the effect of different sampling strategies, the additional CNN-based context-enhancing module, and an extended analysis of the predicted offsets. Sec. L investigates the ordering of the modules in the sparse stream $\mathcal{S}$ and its impact on performance. Finally, Sec. M presents various statistical significance tests, including Friedman and Nemenyi analyses, that confirm the robustness of our results.

A. Implementation details

In this section, we describe the most relevant implementation details of our proposed SDST architecture, a summary of which is also provided in Tab. A. Note that the reported hyperparameters correspond only to the best-performing model. These hyperparameters remain mostly fixed w.r.t. to existing ST works like [8] and only introduce a handful new hyperparameters which we set to 1.0 for simplicity, while we do perform grid search to optimize the learning rate.

Overall, our method is implemented using PyTorch2.0 and CUDA12.8 and runs on a single NVIDIA RTX 6000 GPU with a precision of fp16. Our models are optimized, unless stated otherwise, using AdamW with a learning rate of $1e-4$ and a weight decay of $1e-4$. This follows a step-based schedule, decaying every 20 epochs. We apply linear warmup for the first 2000 iterations with a warmup ratio of 0.001 and clip gradients to a max norm of 35.

Our model operates with a hidden dimension of 256 and leverages a sinusoidal positional encoding. The entire model relies on ReLU non-linearities to enhance the model expressivity. To improve the regularization we use a dropout of 0.5 and incorporate a droppath with a drop probability of 0.25. Our model incorporates various Transformer blocks for instance for cross-modality injection and temporal relation learning, all of which have 8-heads, an attention dropout of 0.0, and an attention output dropout of 0.0. The attention modules are initialized with Xavier. The feedforward modules of the transformer block use a hidden dimension ratio of 4 times the chosen hidden dimension and also leverage a dropout of 0.0 and a Kaiming initialization. Importantly, following standard practices, we always incorporate residual connections to improve the stability of the training and follow a PostNorm strategy [12] that normalizes the input based on a learnable LayerNorm module based on PostNorm.

Architecturally, our model consists of a dense and sparse stream, as well as their respective prediction heads. The dense stream incorporates various Transformer blocks for cross-modality injection and temporal relation learning, all of which have 8-heads, an attention dropout of 0.0, and an attention output dropout of 0.0. The attention modules are initialized with Xavier. The feedforward modules of the transformer block use a hidden dimension ratio of 4 times the chosen hidden dimension and also leverage a dropout of 0.0 and a Kaiming initialization. Importantly, through-

out the Transformer blocks, we normalize the input through a learnable LayerNorm module based on PostNorm. Additionally, one of the key components of our Sparse stream is the use of our novel deformable attention mechanism named RDSA. This first applies a context-enhancing CNN is defined as a 2-layer CNN with a hidden dimension of 256, a learnable LayerNorm, and a non-linearity. Then, after concatenating the left-most, center, and right-most tokens, it uses an MLP to project them to a 64-dimension latent space. This is then used to apply two simple linear projections to compute the 4 different sampled Keys, with their respective attention scores.

For the different prediction heads, we distinguish various different modules. On the one hand, the CLS and Regression heads are defined as a 1 and 3-layer MLPs, respectively. On the other hand, we define the actionness head following previous works like [7], which uses RoiPooling with a Roi size of 16. These roi features are then used for the actionness prediction, applying a 3-layer MLP.

Finally, we make several important training considerations. All the experiments across the different datasets use a batch size of 32 and a minimum video length of 5. FPS is set to 0.5 for QVHighlights and TACoS, and 1.0 for Charades-STA.. We train for 60 epochs on QVHighlights, 50 on Charades-STA, and 150 on TACoS. The number of queries per sample varies on the nature of the dataset. In QVHighlights and Charades-STA, for instance, we define 30 different queries, while for TACoS we use only 5. For Charades-STA, we use a slightly higher learning rate of 2.5×10^{-4} with a decay schedule of 30 epochs.

B. Description of the chosen datasets

To test the effectiveness of our proposed *SDST*, we conduct experiment on three different datasets –i.e., QVHighlights[6], TACoS [10] and Charades-STA [2].

QVHighlights: QVHighlights is the only dataset among the three that provides annotations of both MR and HD tasks. Concretely, this comprises 10k YouTube videos of humanly-annotated NLP queries of a vast variety of topics, from daily activities. For convenience, these videos are trimmed to a maximum duration of 150 seconds.

TACoS: TACos is a widely used dataset for MR consisting of only 127 videos of cooking scenes with an average duration of 287 seconds. Overall, this includes 19k sentence-moment pairs. Notice that following previous works from the literature, we adapt this dataset to support our multi-task-based model by generating synthetic saliency annotations. For this, we consider a frame to have a saliency score of 1 if this belongs to the action, and 0 otherwise.

Charades-STA : Charades-STA extends the original Charades dataset, including 10k videos and 16k different sentence-moment annotations that capture a variety of indoor activities, making it a relevant benchmark to evaluate models in everyday human activity understanding.

C. Descriptions of the objective functions

In this section, we describe in greater detail the different objective functions that we used for our proposed SDST.

C.1. Highlight detection loss

Given the dense visual embedding of the final refinement layer $\mathbf{D}^K \in \mathbb{R}^{T \times F}$, we first apply a learnable Adaptive-Pooling mechanism to produce a single aggregated textual embedding $\mathbf{T}^{pool} \in \mathbb{R}^F$ from the original textual representations $\mathbf{T}^K \in \mathbb{R}^{L \times F}$. We then define the per-frame saliency scores $\hat{\mathbf{Y}}^s \in \mathbb{R}^T$ as

$$\hat{\mathbf{Y}}^s = cos\_sim(\mathbf{D}^K, \mathbf{T}^{pool}) = \frac{\sum_{j=1}^{F} \mathbf{D}_j^K \mathbf{T}_j^{pool}}{||\mathbf{D}^K||\ |\mathbf{T}^{pool}||}. \quad (1)$$

where this cosine similarity is computed for each of the visual frames. To ensure that a higher score corresponds to a higher relevance of a given frame w.r.t. the textual embedding $\mathbf{T}^{pool}$ –corresponding to the last layer K– we use a SampledNCE loss, which ranks the positive frames.

C.2. Moment Retrieval losses

Following the standard DETR pipeline, we apply the Hungarian algorithm to obtain a one-to-one matching between the predicted moment boundaries $\mathbf{R}^\ell \in \mathbb{R}^{M \times 2}$ at each intermediate layer ℓ, and the ground-truth (GT) annotations $\mathbf{Y}^m \in \mathbb{R}^{M^* \times 2}$. Note that unless stated otherwise, we refer to the corresponding matches of the ground-truth $\mathbf{Y}_m$ as $\hat{\mathbf{Y}}^m \in \mathbb{R}^{M^* \times 2}$. Below we describe the different objective functions that we apply to these matching embeddings.

Classification loss: The classification loss takes the predicted action probabilities of the M different recurrent decoder queries $\hat{\mathbf{p}} \in \mathbb{R}^{M \times 1}$ and brings the probability of the unmatched proposals to 0, while the rest have a probability of 1. Given the imbalance between matched and unmatched queries, we leverage a FocalLoss

$$\mathcal{L}_{\text{cls}} = -\frac{1}{M} \sum_{m=1}^{M} \alpha(1 - \hat{\mathbf{p}}_m)^\gamma \log(\hat{\mathbf{p}}_m), \quad (2)$$

where $\hat{\mathbf{p}}_m$ is the predicted probability for proposal m , and α and γ are the standard Focal Loss hyperparameters that help address the class imbalance.

Regression losses: We then focus our attention on the actual regression of the boundaries. For this, following previ-

Pipeline component	Module	Field	Value
Architecture	General config	Dropout	0.5
		K	4
		PE	Sinusoidal
		Hidden dimension	256
		Droppath	0.25
		Non-Linearities	ReLU
		FFN ratio	4
		Attention dropout	0.0
		FFN dropout	0.0
		Attention output dropout	0.0
		FFN output dropout	0.0
		PreNorm	No
		Normalization type	LN
		Attention initialization	Xavier
		FFN initialization	Kaiming
	Sparse module	Deformable sampling points	4
	CLS head	Type	MLP
		Depth	1
	Regression head	Type	MLP
		Depth	3
	Actionness head	Type	MLP
		Depth	3
		Roi size	16
		Roi scale	0
Optimization	Optimizer		AdamW
	Learning Rate		1e-4
	Weight Decay		1e-4
	LR Schedule	Type	Step-based
		Decay rate	Every 20 epochs
	Warmup strategy	Type	Linear
		N. iterations	2000
		Ratio	0.001
	Gradient clipping	Max norm	35
Datasets	QVHighlights	Batch size	32
		FPS	0.5
		Min video len	5
		Epochs	60
		Num. queries	30
	Charades-STA	Batch size	32
		FPS	1.0
		Min video len	5
		Epochs	50
		Num. queries	30
		Learning rate	2.5e-4
		Learning rate schedule	30
	TACoS	Batch size	32
		FPS	0.5
		Min video len	5
		Epochs	150
		Num. queries	5

Table A. Summary of the most relevant hyperparameters and implementation details of our model.

ous DETR works [6] we first define an L1 loss given by

$$\mathcal{L}_{\text{L1}} = \frac{1}{M^*} \sum_{i=1}^{M^*} \left| \hat{\mathbf{Y}}_i^m - \mathbf{Y}_i^m \right|, \tag{3}$$

minimizing the absolute error between the predicted and ground-truth segment boundaries. Additionally, we employ an IoU-based loss [7] to maximize the overlap between the predicted and GT action segments:

$$\mathcal{L}_{\text{IoU}} = 1 - \frac{\sum_{i=1}^{M^*} \text{IoU}(\hat{\mathbf{Y}}_i^m, \mathbf{Y}_i^m)}{M^*}, \tag{4}$$

where $\text{IoU}(\hat{\mathbf{Y}}_i^m, \mathbf{Y}_i^m)$ is the intersection-over-union between the predicted and ground-truth segments.

Actionness losses: As described in the Sec. 3.4, our NMS post-processing first considers the CLS score, which measures the probability of a predicted segment to be matched to a GT. As shown by [7], this is not enough as another pillar to an effective post-processing is having an estimate of the regression quality. For this, we also define the actionness scores $\hat{\mathbf{Y}}^a \in \mathcal{R}^M$ as the maximum overlap of every query with any of the GT. In other words, for each of the learnable recurrent query embeddings we compute the maximum IOU with any of the GT actions. Then, we apply an L1 loss to regress this score which we can then leverage during inference.

$$\mathcal{L}_{act} = \frac{1}{M} \sum_{i=1}^{M} \left| \hat{\mathbf{Y}}_i^a - max_j^{M^*}(IOU(\mathbf{R}_i^\ell, \mathbf{Y}_j^m)) \right| \tag{5}$$

where $\hat{\mathbf{Y}}_i^a$ is the predicted actionness score of the i-th recurrent decoder query, and $max_{j=1}^{M^*}(IOU(\mathbf{R}_i^\ell, \mathbf{Y}_j^m))$ is the maximum overlap of the i-th query w.r.t. any of the GT actions.

C.3. Alignment losses

One critical aspect to guarantee an effective side-tuning is the inclusion of alignment losses which bring the visual and textual latent space closer in a semantically meaningful way. This is particularly important because, although the backbone has been pre-trained to ensure some degree of alignment, adapting it to a new domain such as VTG inevitably introduces domain shifts and noise. Without this alignment losses, this would hence significantly degree the quality of the features, and thus hinder the final performance. In our work we address this issue by introducing two contrastive losses [8] that enforce video-query consistency at different levels of intermediate representation: 1) video-level alignment 2) layer-wise alignment. Notably, these losses are applied to all the intermediate layers independently.

C.3.1. Video-level contrastive loss

At a given level ℓ, this loss enforces similarity between action-relevant frames and their corresponding textual query embedding. Specifically, it takes the embeddings $\mathbf{V}^\ell$ and the pooled textual embedding $\mathbf{T}^{pool}$ of that level, and pulls the positive frames –i.e., belonging to the action– closer while pushing away the negatives. Interestingly, for a given j-th frame, we consider the negatives to be all the other j-th frames of the remaining batch elements, at the same refinement level ℓ. Thereafter, we enforce our objective via an InfoNCE loss:

$$\mathcal{L}_{video\_cal} = InfoNCE(\mathbf{V}^\ell, \mathbf{T}^{pool}) \tag{6}$$

where InfoNCE [9] maximizes the similarity between the correct video-text pairs while promoting its separation from unrelated samples.

C.3.2. Layer-wise contrastive loss

This loss is similar to the previous one but operates across layers instead of the batch. This is, this ensures that the same frame-query pair learns different representations at two distinct levels ℓ and ℓ'. This promotes that these representations are not redundant, and thus add complementary information to the model. To be more specific, following [8] we define the negatives at level ℓ as the same frame-embedding but corresponding to a different intermediate layer ℓ'.

$$\mathcal{L}_{video\_cal} = InfoNCE(\mathbf{V}^\ell, \mathbf{T}^{pool}). \tag{7}$$

C.4. Inference

During inference, we apply a soft NMS post-processing to filter out redundant action predictions. This algorithm sorts the proposals based on a confidence score, which in our case we define as the square root of the product of the class probabilities and actionness scores

$$\hat{\mathbf{C}} = \sqrt{\hat{\mathbf{p}} \cdot \hat{\mathbf{Y}}^a}. \tag{8}$$

This prioritizes a high classification confidence together with a high localization confidence.

D. Background on the deformable attention mechanism

The Vanilla Attention mechanism is the core component of Transformers, one of the most popular architectures in the community at the time of this writing. The attention mechanism can be defined as:

$$\mathbf{Q} = \mathbf{X}_\mathcal{Q}\mathbf{W}_\mathcal{Q}\ ,\ \mathbf{K} = \mathbf{X}_\mathcal{K}\mathbf{W}_\mathcal{K},\ \ \mathbf{V} = \mathbf{X}_\mathcal{V}\mathbf{W}_\mathcal{V} \tag{9}$$

$$\mathbf{S} = \frac{\sigma(\mathbf{Q}\mathbf{K}^T)}{\sqrt{d_k}}\mathbf{V} \tag{10}$$

Here σ is a softmax activation, and $\mathbf{X}_\mathcal{Q}, \mathbf{X}_\mathcal{K}$ and $\mathbf{X}_\mathcal{V}$ define the inputs to the query, keys and values projection matrices $\mathbf{W}_\mathcal{Q}, \mathbf{W}_\mathcal{K}, \mathbf{W}_\mathcal{V}$, respectively. In the self-attention case, $\mathbf{X}_\mathcal{Q} = \mathbf{X}_\mathcal{K}$ while for cross-attention, $\mathbf{X}_\mathcal{Q} \neq \mathbf{X}_\mathcal{K}$.

This mechanism, despite very extended in the community these days, presents several important pitfalls like its quadratic complexity or its slow convergance. This motivated the proposal of various efficient attention mechanisms to attain similar performance while improving its efficiency. In this regard, we highlight the Deformable attention mechanism, proposed by [15], inspired by the previous works on the deformable convolution. This mechanism attains a considerable efficiency boost with respect of the vanilla attention [12] by limiting the amount of attendable keys to a (small) predefined set of key tokens. More formally, the key of this module is the lack of explicit interaction between queries and keys, which are defined as

$$\mathbf{Q} = \mathbf{X}_\mathcal{Q}\mathbf{W}_\mathcal{Q}^{def} \ , \ \mathbf{K} = \mathbf{X}_\mathcal{K}\mathbf{W}_\mathcal{K}^{def}, \tag{11}$$

where $\mathbf{W}_\mathcal{Q}^{def}$, $\mathbf{W}_\mathcal{K}^{def}$ are two linear projections. The deformable attention thus avoids computing the $\mathbf{QK}^T$ similarity matrix, and instead employs an offset and attention-score predictors, $\mathcal{G}_\Delta$ and $\mathcal{G}_\mathbf{A}$ to select and weight –only based on the queries– a small subset P of selectable keys. The output of this mechanism, the weighted aggregation of the selected keys, namely $\mathbf{S}$, is computed as follows

$$\Delta = \mathcal{G}_\Delta(\mathbf{Q}) \in \mathbb{R}^{M\times P}, \ \mathbf{A} = \mathcal{G}_\mathbf{A}(\mathbf{Q}) \in \mathbb{R}^{M\times P}, \tag{12}$$

$$\mathbf{S} = \sum_{p=1}^{P} (\mathbf{A}_{:,p} \ \mathbf{K}[\mathbf{c} + \mathbf{w} \odot \Delta_{:,p}]) \in \mathbb{R}^{M\times F} \tag{13}$$

where $\mathcal{G}_\Delta$, $\mathcal{G}_\mathbf{A}$ are often modeled as CNNs and $x[y]$ is the bilinear sampling of x on index(es) y. This can be seen as a learnable method to identify, given a specific query, a small subset of key tokens to attend to, as well as their respective weight –necessary to compute their weighted average. The cornerstone of its efficiency boost is thus the fact that it does not require *looking* at all the keys to make that decision –unlike Vanilla Attention– but, instead, it is just a result of a simple projection layer of the query itself.

E. Detailed ablation using only InternVideo2-1B features

In order to guarantee a fair comparison between our proposed SDST and the rest of the tested baselines, in Tab. B we follow the work of [3] and test the relevant baselines –on 3 independent seeds– using only the InternVideo2-1B features. Note that we evaluate only the relevant baselines capable of doing MR and HD. Observe that our method substantially outperforms the other existing side-tuning method for VTG [8]. Concretely, it improves its MR capabilities by a 3.82% mAP, while it improves HD by a 2.21% mAP and 2.1% HIT@1. Similarly to our previous observations, our method outperforms the rest of the methods and remains competitive with the SG-DETR and even improves it in the two different HD metrics. We deem this to be particularly noteworthy given that our method poses only 27.3% of the trainable parameters of SG-DETR.

Method	MR-mAP			HD ≥ Very Good	
M-DETR	60.20 ± 0.55	34.43 ± 0.43	35.40 ± 0.41	40.31 ± 0.21	63.89 ± 0.62
UniVTG	63.51 ± 0.25	38.83 ± 0.26	37.78 ± 0.16	42.68 ± 0.09	69.34 ± 0.23
QD-DETR	67.78 ± 0.29	46.40 ± 0.26	45.52 ± 0.15	41.82 ± 0.07	68.06 ± 0.24
CG-DETR	69.86 ± 0.21	49.35 ± 0.28	48.69 ± 0.17	42.72 ± 0.07	69.87 ± 0.15
TR-DETR	70.08 ± 0.15	49.20 ± 0.50	47.99 ± 0.42	43.43 ± 0.16	71.13 ± 0.25
R2-Tuning*	71.40 ±0.330	53.786 ±0.684	51.49 ±0.358	41.72 ±0.085	69.52 ±0.472
SG-DETR	**73.52 ± 0.05**	**57.91 ± 0.13**	**55.64 ± 0.20**	43.91 ± 0.14	71.47 ± 0.73
Ours	73.20 ± 0.226	56.76± 0.53	55.31± 0.23	**43.93 ± 0.063**	**71.62± 0.348**

Table B. Evaluation of a set of representative baselines when leveraging InternVideo2-1b features, evaluated on QVHighlights *val* split. **Bold** stands for the best and underline for the second-best.

Method	Charades-ST			TACoS		
	R1@0.5	R1@0.7	mIoU	R1@0.5	R1@0.7	mIoU
R^2-Tuning	68.2	46.26	58.14	38.02	25.27	35.36
SG-DETR	70.2	49.5	59.1	**44.7**	29.9	40.9
FlashVTG	70.3	49.9	–	41.8	24.7	37.6
Ours	**72.0**	**52.6**	**61.2**	44.5	**32.3**	**42.2**

Table C. Comparison of multiple representative baselines on Charades-STA and TACoS datasets when leveraging InternVideo2-1b features. **Bold** stands for the best and underline for the second-best.

Additionally, in Tab. C we also show the analysis of the two remaining considered datasets, these being Charades-STA and TACoS. Similarly, in these two scenarios, our method improves the second-best performing works –i.e., FlashVTG for Charades-STA and SG-DETR for TACoS– in all the metrics but R1@0.5 on TACoS where it incurs in a marginal degradation.

F. Additional efficiency analysis

In this paper, we normally study the efficiency of our model w.r.t existing related works based only on the number of parameters. In Tab. D we extend this analysis to the training memory and the running time. For simplicity purposes, we limit this study to the QVHighlights dataset.

G. Study of the inherent optimization difficulty

Undeniably, existing SOTA models –e.g., [3, 8]– accumulate a considerable number of losses and components. And unfortunately, ours is no exception. While this represents a considerable opportunity for future studies trying to create more compact models, in this section we aim to extend the ablation from Tab.7 of the main text where we justified the necessity of the various model components of the sparse stream $\mathcal{S}$. Concretely, first of all, we focus on the need for the different proposed losses –with the exception of those

	# Params (M)	Memory (GB)	Runtime (it/s)
Moment-DETR	4.8	1.54	7.45
R2-Tuning	2.7	2.4	5.55
TR-DETR	7.9	1.76	4.75
HL-CLIP	2.0	22.98	0.64
Llava-MR	17.0	$\approx 80 \times 8$	-
MR.Blip	19.0	$\approx 80 \times 8$	-
SG-DETR	15.0	-	-
Flash-VTG	10.9	2.3	5.2
Ours	4.1	3.4	4.16

Table D. Efficiency summary of a set of representative models evaluated on QVHighlight with InternVideo2-1b features, and a batch size of 32.

$\mathcal{L}_1$	$\mathcal{L}_{IOU}$	$\mathcal{L}_{align}$	$\mathcal{L}_{act}$	$\mathcal{L}_{cls}$	MR R1@0.5	MR R1@0.7	MR mAP@0.5	MR mAP@0.75	MR mAP	HD mAP	HD HIT@1
	✓	✓	✓	✓	71.68	58.58	72.28	56.28	55.0	43.52	71.35
✓		✓	✓	✓	73.94	59.55	72.07	55.21	54.25	43.35	71.26
✓	✓		✓	✓	71.55	58.13	71.55	55.21	54.25	43.68	69.48
✓	✓	✓		✓	73.1	59.61	73.87	57.01	55.6	43.29	71.00
✓	✓	✓	✓		70.52	57.23	69.88	54.82	52.95	42.28	68.65
✓	✓	✓	✓	✓	**73.68**	**60.90**	**73.52**	**57.42**	**55.60**	**44.00**	**72.00**

Table E. Importance of the main losses of our model when evaluated on QVHighlight val with InternVideo2-1b features.

Perm. #	MR R1@0.5	MR R1@0.7	MR mAP@0.5	MR mAP@0.7	MR mAP	HD mAP	HD HIT@1
1	73.03	59.10	72.78	56.31	55.17	43.70	70.58
2	73.94	59.35	73.92	57.26	55.87	44.30	72.58
3	72.19	57.87	72.43	56.22	54.72	44.03	71.74
4	72.97	58.77	73.13	56.84	55.58	44.52	71.68
Chosen	73.68	60.90	73.52	57.42	55.60	44.00	72.00
Mean±Std	73.16 ± 0.61	59.19 ± 0.98	73.15 ± 0.52	56.81 ± 0.48	55.38 ± 0.40	44.11 ± 0.27	71.71 ± 0.65

Table F. Performance across different permutations with main retrieval and detection metrics.

that are indispensable to solve the inherent task, like the saliency-related losses. For this, find in Tab. E.

In terms of difficulty of optimization and parameter search, we highlight the considerable robustness in terms of hyperparameter choice. Concretely, as specified in Sec. A, the vast majority of the hyperparameters are kept consistent with previous relevant works like [8]. The newly introduced hyperparameters were set to 1.0 for simplicity. Nevertheless, this does not guarantee the robustness to this hyperparameter choice. Consequently we propose the following experiment: We randomly sample 4 additional different configurations –i.e., all the loss weights– defining a range of $[0.25, 2]$ for $\lambda_{L1}, \lambda_{IOU}, \lambda_{act}$ and λ_{cls} and from $[0.1, 0.5]$ for $\lambda_{sal}, \lambda_{align\_video}$ and λ_{align_layer}. Observe in Tab. F the results of each of these configurations –defined in Tab. G– and observe that our model does not deviate significantly given these new random permutations. In fact, these more robust performance metrics –not requiring cherry picking the best configuration– would still rank equally in the overall ranking from Tab.1.

Perm #	λ_{L1}	λ_{IOU}	λ_{sal}	λ_{align_video}	λ_{align_layer}	λ_{act}	λ_{cls}
1	1.47	1.91	0.11	0.18	0.42	0.84	1.27
2	1.43	0.41	0.17	0.33	0.37	0.3	0.29
3	1.81	0.92	0.36	0.42	0.23	0.63	0.64
4	0.4	1.24	0.31	0.1	0.16	1.13	0.64
Chosen	1	1	0.1	0.1	0.1	1	1

Table G. The randomly chosen 4 different loss weight configurations and the final chosen configuration.

Method	#Params (M)	Memory (GB)	MR R1@0.5	MR R1@0.7	MR mAP	HD mAP	HD HIT@1
w/o Tuning	2.70	2.35	66.97	51.10	46.19	41.45	67.23
E^3VA [13]	2.57	2.96	68.97	53.16	47.68	41.04	68.13
LoSA [4]	6.40	**2.39**	72.13	58.32	53.73	41.82	68.19
LST [11]	**2.04**	2.49	70.32	55.55	50.59	41.53	69.48
R^2-Tuning[8]	2.70	2.44	70.84	55.35	51.30	41.64	69.74
Ours	4.10	3.40	**73.68**	**60.90**	**55.60**	**44.00**	**72.00**

Table H. Performance comparison of different tuning methods on QVHighlights *val* split. **Bold** stands for the best.

H. Comparison with other PEFT and MEFT methods

In this section, we compare our proposed SDST against other relevant PEFT methods when leveraging InternVideo2-1B features for QVHighlights *val* split. Importantly, we note that we were unable to evaluate relevant methods based on Adapters, LORA, or Prompt-based, due to severe computational limitations. Specifically, these methods require full backpropagation through the frozen backbone, exceeding the memory capacity of our NVIDIA RTX 6000. This underscores the importance of MEFT methods like ST. Furthermore, [8] shows that these memory-expensive alternatives underperform over ST for VTG, allowing us to safely restrict the scope of this ablation to *w/o Tuning*, and to relevant ST baselines –i.e., E^3VA [13], LoSA [4], LST [11] and R^2-Tuning [8]. Among these, only R^2-Tuning is naturally suitable for a multi-modal setup like ours. Consequently, for a fair comparison, we made minimal modifications to adapt the other baselines to our setting.

Observe in Tab. H that all these baselines poses a comparable number of trainable parameters, with the exception of LoSA [4] which has a slightly higher count. Similarly, all these methods show a very efficient memory usage, which as mentioned before, contrasts with other PEFT alternatives. We find that while all these tested baselines considerably improve the *w/o Tuning* on MR, they perform quite similarly in terms of HD. Overall, our proposed SDST improves all these methods, with a especially significant boost on HD.

In Tab. I we include the homologous analysis for Charades-STA and TACoS datasets which shows similar results.

	R1@0.5	R1@0.7	mIOU	R1@0.5	R1@0.7	mIOU
w/o Tuning	67.69	45.56	57.98	34.22	21.82	32.51
E^3VA [13]	66.13	45.11	56.23	38.77	26.02	36.15
LoSA [4]	67.69	45.16	57.42	38.54	24.49	35.71
LST [11]	68.2	46.26	58.14	38.02	25.57	35.26
R^2-Tuning[8]	69.25	46.67	58.69	39.54	27.37	36.27
Ours	**72.00**	**52.60**	**61.20**	**44.50**	**32.30**	**42.20**

Table I. Performance comparison of different tuning methods for MR on Charades-STA and TACoS. **Bold** stands for the best.

Shared	MR					HD		Params
	R1@0.5	R1@0.7	mAP@0.5	mAP@0.75	mAP	mAP	HIT@1	
✓	**73.68**	**60.90**	**73.52**	**57.42**	**55.60**	**44.0**	**72.00**	4.10 M
	71.23	57.1	71.62	55.44	54.1	43.82	70.45	12.43 M

Table J. Ablation of the effect of using shared vs unshared parameters on QVHighlights *val* split. **Bold** stands for the best.

I. Parameter sharing in SDST

Tab. J compares the performance of SDST with and without parameter sharing. This is, we evaluate if creating independent SG side-tuners for each of the $K = 4$ different intermediate layers results in an improved performance on the MR and HD tasks. Observe that this is not the case. Despite the additional 8.32M parameters, unsharing the different side-tuning modules in fact results in a performance degradation in all the tested metrics. We hypothesize that sharing the same alignment module (see Eq.2) with the subsequent $\mathcal{L}_{align}$ loss, promotes that 1) embeddings share a unique latent space while 2) different layers focus on different semantics. This allows the sharing of the remaining modules, which we observed contributes to stabilizing the optimization, and thus, improve performance.

J. Extended ablation on the use of intermediate InternVideo2-1B features

In Sec. 6.1 we propose an ablation study to showcase the importance of the different refinement steps in performance, as well as quantify the effect of using intermediate layers instead of using the last-layer features only. For completeness, in Tab. K we present the complete ablation with all the evaluated metrics that further support our findings and insights.

K. Study of deformable attention

Action-length-based analysis: In Sec. 6.2 we expose the empirical benefits of our proposed RDSA method over the standard CA [12], Deformable CA [15] and even decoder query initialization mechanisms like [14]. To complement these results, in Tab. L we disentangle the MR performance according to the action length –i.e., short, middle, and long actions. This comparison indicates that one of the core limitations of the standard CA module is its almost complete inability to correctly identify short actions. Observe that its performance over short actions degrades by 14.33% and 15.07 w.r.t..the Deformable CA and RDSA, respectively. We also observe that our method is especially effective at predicting long actions, improving by 2.84% mAP w.r.t. the Deformable CA.

K	Interm?	MR					HD	
		R1@0.5	R1@0.7	mAP@0.5	mAP@0.75	mAP	mAP	HIT@1
1	✓	68.00	54.13	68.72	51.23	48.72	43.66	68.26
2	✓	71.94	57.48	71.98	55.15	53.64	43.49	69.61
3	✓	72.84	58.19	72.92	56.0	54.69	43.85	69.94
4	✓	73.68	60.90	73.52	57.42	55.60	44.0	72.00
5	✓	72.39	58.77	71.71	55.35	54.34	43.7	71.87
1		68.00	54.13	68.72	51.23	48.72	43.66	68.26
2		73.23	59.29	72.75	56.61	55.06	44.18	71.68
3		73.61	59.42	73.27	56.41	55.32	43.69	70.84
4		70.84	57.03	72.02	54.27	54.6	44.29	71.55
5		70.00	57.03	70.62	54.72	53.54	43.27	68.52

Table K. Ablation of the effect of refining over multiple refinement levels –i.e., last k– and of the use of intermediate versus last-layer features. Results correspond to QVHighlights *val* split

Att. strat.	mAP short	mAP middle	mAP long	mAP
Stand. CA	3.31	45.92	51.17	42.72
Def.CA	17.64	57.68	56.92	54.27
Ours	**18.38** (+0.74)	**58.15** (+0.47)	**59.76** (+2.84)	**55.60** (+1.33)

Table L. Performance comparison of different attention strategies across different video lengths from the QVHighlights *val* split. We include the absolute difference between our method and the second-best performing baseline –i.e., Def. CA. **Bold** stands for the best.

Effect of the CNN and the sampling points: In this section, we are also interested in providing further insights necessary for a deep understanding of RDSA. To be more specific, we focus our attention on two important aspects of this module, namely the points that are sampled to form the alternative query embeddings, and the additional CNN module to gain local context (see Eq. 10).

For this, in Tab. M we ablate over three possible sampling strategies. The first samples only the center frame of the action, the second samples both the left and right-extremum of the action boundaries, and the later samples all these 3 embeddings. Note that as described in Sec. 3.3.4, these embeddings are sampled based on the predicted action reference, and after concatenation, are used as alternative query embeddings for a Deformable Self-Attention mechanism.

In this regard, our ablation indicates that the RDSA benefits the most from the extremum embeddings when also incorporating a CNN module. This indicates that the CNN is effectively gathering context of the neighborhood of the current action boundaries, providing critical information for the offset prediction, and thus, of *where the model should*

Sampling points	CNN	MR					HD	
		R1@0.5	R1@0.7	mAP@0.5	mAP@0.75	mAP	mAP	HIT@1
c		71.94	58.39	71.37	55.35	53.57	43.59	71.23
	✓	70.97	57.03	71.45	55.18	54.01	43.44	70.58
l-r		72.26	57.16	71.77	55.03	53.55	43.26	70.71
	✓	72.32	57.74	72.31	55.91	54.36	43.28	71.81
l-c-r		72.13	58.77	71.54	54.81	53.94	43.25	71.68
	✓	**73.68**	**60.90**	**73.53**	**57.42**	**55.60**	**44.00**	**72.00**

Table M. Ablation of the effect of different sampling strategies like center-sampling (c), left-most and right-most action-boundary sampling (l) and (r), respectively. We also quantify the importance of our additional CNN module for enhanced context learning. Results correspond to QVHighlights *val* split. **Bold** stands for the best.

look to further refine the predicted segments.

We also observe that the center embeddings are necessary even though they seem to play a lesser role in the overall performance. Interestingly, the use of a CNN in fact harms the effectiveness of these embeddings. We conjecture that by definition, the center embeddings tend to be *surrounded* by very action-like embeddings. Thus, the local neighborhood does not necessarily provide useful information, and might even cause learning instabilities or aggravate the overfitting.

In short, these experiments show the importance of using the 3 proposed sampled embeddings, and the overall positive impact that the additional CNN module has on gathering information on the local neighborhoods of the action boundaries.

Extended analysis on the predicted offsets: Finally, we extend the analysis provided in Sec. 6.2 that sheds light on where the predicted offsets point to. Concretely, in Fig. A we additionally depict a similar analysis for a head that initializes the heads near the center of the action –i.e., 0. Observe that in this case, similar to our previous observations, we find that the original Deformable CA [15] keeps the offsets closer to the original initialization, suggesting a lack of proper understanding of the input video. Our method, in contrast, learns to point the offsets over the frames closer to the left-most boundary. Also, we find that in our model learns to *look* more left as the model processes deeper levels.

L. Study of the ordering of the different modules of the sparse stream

One important aspect to consider is the ordering of the 4 different modules of the sparse stream. For this, in Tab. N we evaluate multiple relevant combinations. Note that we avoid ablating over the final FFN module due to computational limitations. In this regard, Tab. N indicates that it is beneficial to include the CA module to gain textual context as early as possible.

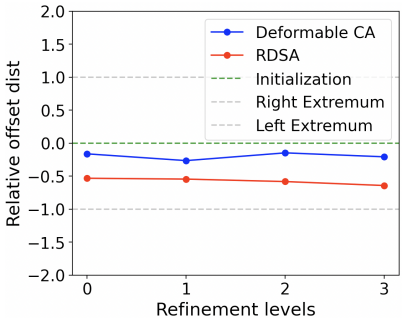

Figure A. Attention-weighted offset distances across refinement levels over head 2 when evaluating QVHighlights *val* split.

Mod. permutation	MR					HD	
	R1@0.5	R1@0.7	mAP@0.5	mAP@0.75	mAP	mAP	HIT@1
SA-CA-Def-FFN	70.77	56.13	71.54	54.66	53.81	43.36	70.39
SA-Def-CA-FFN	70.97	57.29	71.30	54.72	53.74	43.06	69.87
Def-SA-CA-FFN	71.81	57.61	71.61	55.05	53.91	43.16	70.19
Def-CA-SA-FFN	72.77	58.71	72.78	56.46	54.98	44.04	71.35
CA-Def-SA-FFN	72.58	59.42	72.78	57.26	54.94	43.46	70.77
CA-SA-Def-FFN	**73.68**	**60.90**	**73.52**	**57.42**	**55.60**	**44.00**	**72.00**

Table N. Ablation on the importance of the ordering of the components in the sparse stream when evaluated on QVHighlights *val* split.

M. Ablation statistical significance tests

In this section, we aim to assess if the performance of our proposed SDST (ours) significantly differs from the other relevant baselines. For our main results, we were unable to establish a fair comparison with the other baselines across various seeds, given that for instance QVHighlights *test* has a limited number of submissions. Consequently, we focus on the statistical study of the different model rankings across all the 3 studied datasets and their respective metrics. Concretely, we carry out two primary statistical tests: the Friedman test [1] and Nemenyi's test [5].

M.1. Friedman Test

The Friedman test is a non-parametric statistical test to test if k different variables are part of the same population. Concretely, we apply this test to study if given a set of various models, their rankings differ significantly across different

Method	QVHighlights(test)							Charades-ST			TACoS		
	MR					HD							
	R1		mAP			$\geq Very good$		R1			R1		
	@0.5	@0.7	@0.5	@0.75	Avg.	mAP	HIT@1	@0.5	@0.7	mIOU	@0.5	@0.7	mIOU
Moment-DETR	9	9	9	9	9	7	6	9	9	–	9	9	9
QD-DETR	7	7	7	7	7	5	4	8	8	–	7	7	7
UniVTG	8	8	8	8	8	6	5	7	7	6	8	8	8
CG-DETR	5	6	5	6	6	4	2	6	6	5	5	6	5
BAM-DETR	6	5	5	5	5	–	–	5	4	4	4	3	3
R2-Tuning	4	4	4	4	4	3	3	4	5	3	6	4	6
SG-DETR†	1	1	1	1	1	1	1	3	4	2	1	2	2
Flash-VTG†	3	3	2	3	3	–	–	2	2	–	3	5	4
Ours†	2	2	3	2	2	2	1	1	1	1	2	1	1

Table O. Comparison with the SOTA on QVHighlights *test* and *val*. Also note that for comparability purposes, none of these results rely on pre-training. † indicates that the method uses InternVideo2 backbone (comparable to ours).

datasets and metrics. We define the null hypothesis of the Friedman test as *all models perform similarly*, hence implying that there are no significant differences in the rankings across datasets/metrics. Mathematically, the Friedman statistic χ_F^2 is given by

$$\chi_F^2 = \frac{12}{N \cdot k \cdot (k+1)} \sum_{i=1}^{k} \left(R_i - \frac{N(k+1)}{2} \right)^2, \quad (14)$$

where N is the number of datasets and their respective metrics. K is the number of tested models, and R_i is the sum of ranks for model i across the different datasets and metrics.

In our case, the Friedman test yielded a statistic of $\chi_F^2 = 5.640$ with a p-value of 0.933. This is greater than 0.05, the threshold that is typically employed to determine the statistical significance. Hence, we can reject the null hypothesis, and conclude that there is no significant difference between the rankings of the models evaluated on all datasets. In other words, we observe that the performance of the various evaluated baselines, including our SDST, perform consistently across different datasets and metrics.

M.2. Pairwise Nemenyi's Test

In this second statistical significance test we are interested in a more fine-grained analysis that might allow us to determine if our proposed method performs significantly better than the remaining considered baselines –especially of the R2-Tuning and the SG-DETR. For this, we proceed with a pairwise comparison using the Nemenyi's test. More in detail, for each pair of models, the Nemenyi test statistic is calculated based on their respective rank differences across the various datasets and metrics. We present the obtained p-values in Tab. P. These results indicate that our method (SDST) performs statistically better than all the other baselines with the exception of SG-DETR which performs statistically on par. This matches our previous observations and certifies that our method attains statistically equivalent performance to SOTA while using only 27% of its respective parameter count.

Comparison w.r.t. SDST	p-value	Statistically different
Moment-DETR	0.0012	✓
QD-DETR	0.0013	✓
UniVTG	0.0011	✓
CG-DETR	0.0013	✓
BAM-DETR	0.0013	✓
R2-Tuning	0.0013	✓
SG-DETR	0.7926	
Flash-VTG	0.0011	✓

Table P. Nemenyi's significance test across the various pair-wise comparisons w.r.t. to our proposed SDST.

These results indicate that for Ours vs SG-DETR, the p-value is 0.7926, meaning there is no statistically significant difference between the two methods. In contrast, for comparisons between Ours and the other models, the p-values are all below 0.05, suggesting that Ours is significantly better than the other models.

References

[1] Milton Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association*, 32(200):675–701, 1937. 8

[2] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *Proceedings of the IEEE international conference on computer vision*, pages 5267–5275, 2017. 2

[3] Aleksandr Gordeev, Vladimir Dokholyan, Irina Tolstykh, and Maksim Kuprashevich. Saliency-guided detr for moment retrieval and highlight detection. *arXiv preprint arXiv:2410.01615*, 2024. 5

[4] Akshita Gupta, Gaurav Mittal, Ahmed Magooda, Ye Yu, Graham W Taylor, and Mei Chen. Losa: long-short-range

adapter for scaling end-to-end temporal action localization. *arXiv preprint arXiv:2404.01282*, 2024. 6, 7

[5] Peter J Huber. Fifth berkeley symposium on mathematical statistics and probability. *University of California*, 1967. 8

[6] Jie Lei, Tamara L Berg, and Mohit Bansal. Detecting moments and highlights in videos via natural language queries. *Advances in Neural Information Processing Systems*, 34:11846–11858, 2021. 2, 4

[7] Xiaolong Liu, Qimeng Wang, Yao Hu, Xu Tang, Shiwei Zhang, Song Bai, and Xiang Bai. End-to-end temporal action detection with transformer. *IEEE Transactions on Image Processing*, 31:5427–5441, 2022. 2, 4

[8] Ye Liu, Jixuan He, Wanhua Li, Junsik Kim, Donglai Wei, Hanspeter Pfister, and Chang Wen Chen. r^2-tuning: Efficient image-to-video transfer learning for video temporal grounding. *arXiv preprint arXiv:2404.00801*, 2024. 1, 4, 5, 6, 7

[9] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 4

[10] Anna Rohrbach, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Manfred Pinkal, and Bernt Schiele. Coherent multi-sentence video description with variable level of detail. In *Pattern Recognition: 36th German Conference, GCPR 2014, Münster, Germany, September 2-5, 2014, Proceedings 36*, pages 184–195. Springer, 2014. 2

[11] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. Lst: Ladder side-tuning for parameter and memory efficient transfer learning. *Advances in Neural Information Processing Systems*, 35:12991–13005, 2022. 6, 7

[12] 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*, 30, 2017. 1, 5, 7

[13] Dongshuo Yin, Xueting Han, Bin Li, Hao Feng, and Jing Bai. Parameter-efficient is not sufficient: Exploring parameter, memory, and time efficient adapter tuning for dense predictions. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 1398–1406, 2024. 6, 7

[14] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022. 7

[15] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 5, 7, 8