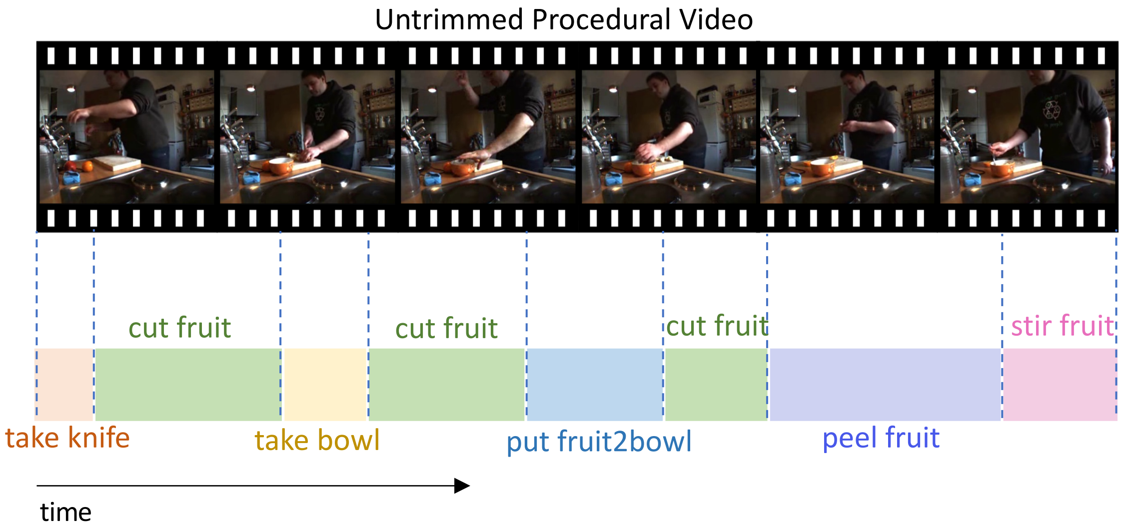
Efficient Temporal Action Segmentation via Boundary-aware Query Voting

Peiyao Wang[1] **Yuewei Lin**[2] **Erik Blasch**[3] **Jie Wei**[4] **Haibin Ling**[1*]

[1]Stony Brook University, [2]Brookhaven National Laboratory,
[3]Air Force Research Laboratory, [4]The City College of New York
{peiyaowang, hling}@cs.stonybrook.edu, ywlin@bnl.gov,
erik.blasch@gmail.com, jwei@ccny.cuny.edu

Task and Challenge

- Task: Temporal Action Segmentation (TAS)



Temporal Action Segmentation aims to allocate an action label to each frame, enabling the detailed analysis of complex activities by identifying specific actions within long-form videos.

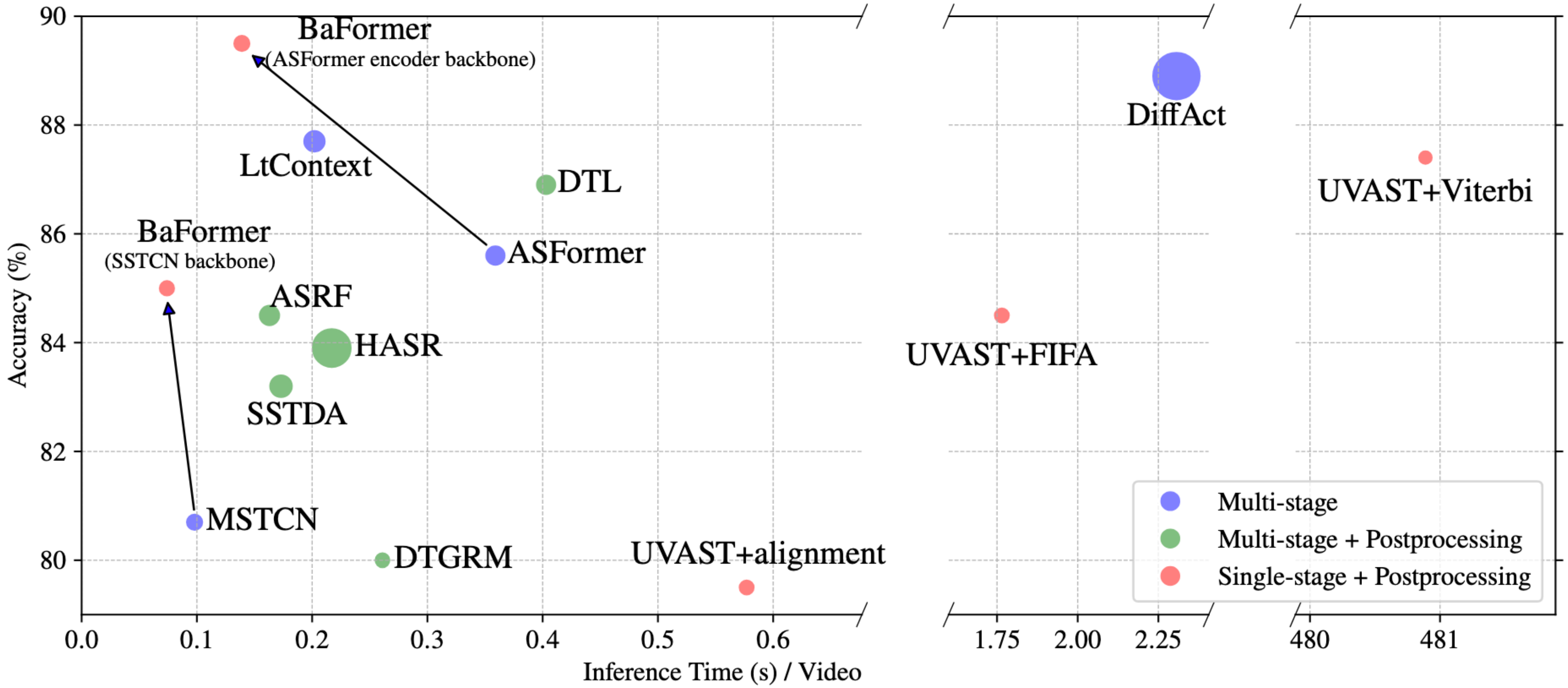
Try to get smooth results, there are models with main trends:

1) *multi-stage model*: stack several models for refinement
2) *Post processing refinement*: global review for refinement

Task and Challenge

- Challenge: High computational cost

Accuracy vs. inference time on 50Salads

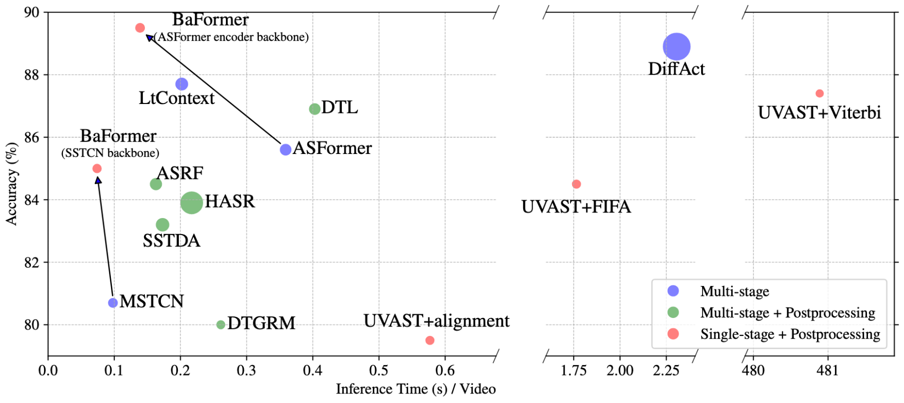


☹ Better performance but with longer inference time

☺ Try to get a trade-off between the efficiency and performance

Task and Challenge

- Challenge: High computational cost

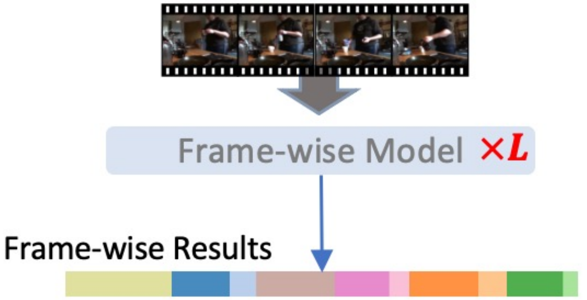


(1) Long-form Input

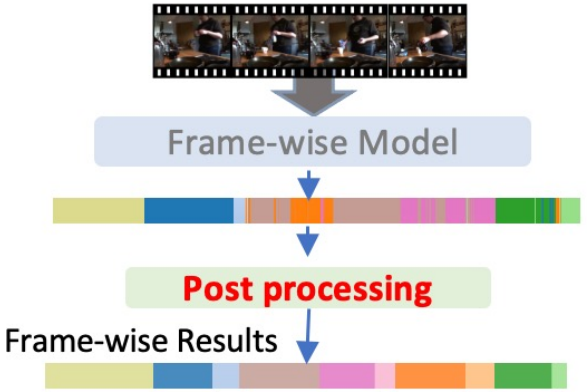
Untrimmed videos in TAS often include tens of thousands of frames

Dataset	video length(min)	#segments per video	segments length(s)
GTEA	1.24	31	2.21
50Salads	6.4	18	36.8
Breakfast	2.3	6.6	15.1

(2) Heavy Model Structure



☹ multi-stage refinement models



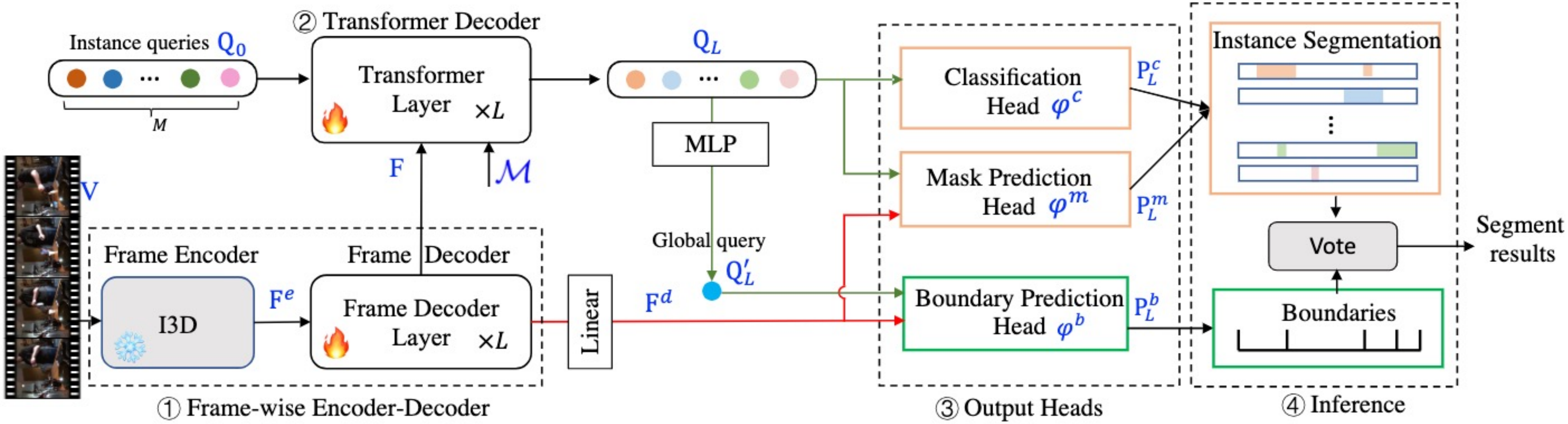
☹ time-consuming post-processing techniques

Contribution

- How to get a trade-off between the efficiency and performance?
  - Reducing the temporal dimension
    - transform the long-form video into a **sparse representation** via Transformer based model
  - Minimize the running time
    - employ a single-stage model : Frame-wise supervision into **segment level** supervision
    - an appropriate post-processing method: **query based voting**

Method

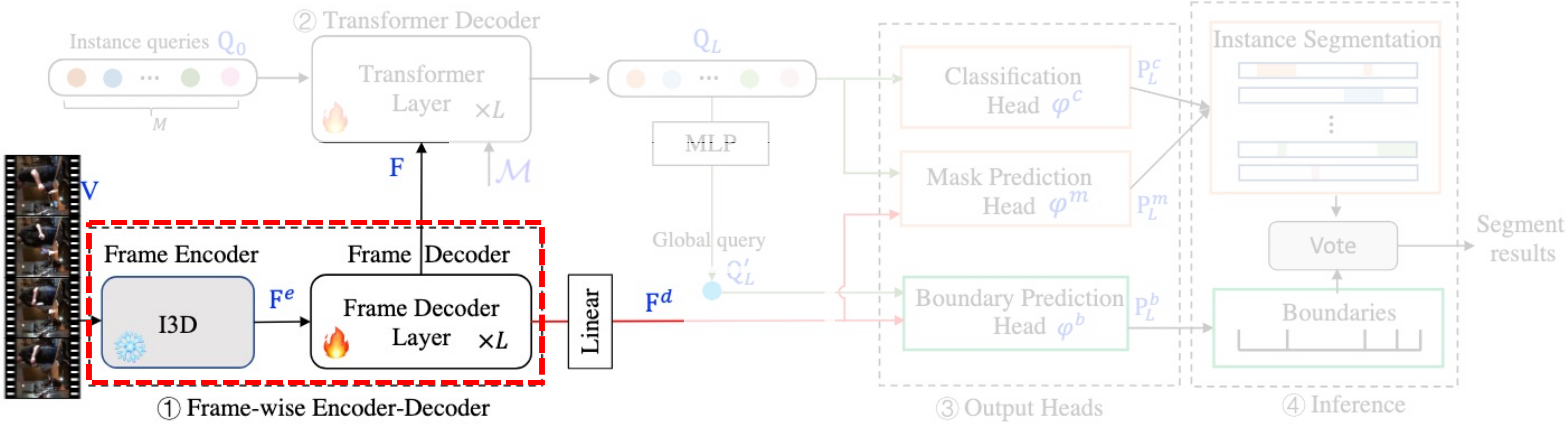
- Framework



Overview of BaFormer architecture. It predicts query classes and masks, along with boundaries from output heads. Although each layer in the Transformer decoder holds three heads, we illustrate the three heads in the last layer for simplicity.

Method

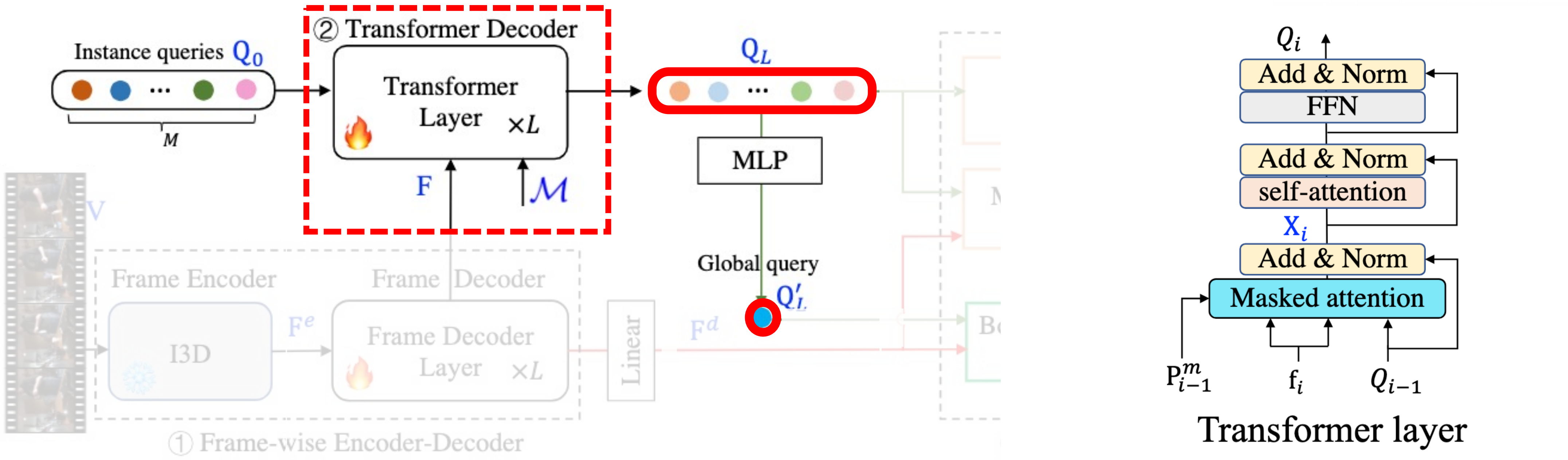
- Framework



Frame-wise Encoder-Decoder: preserve dense information essential for our model’s functionality

Method

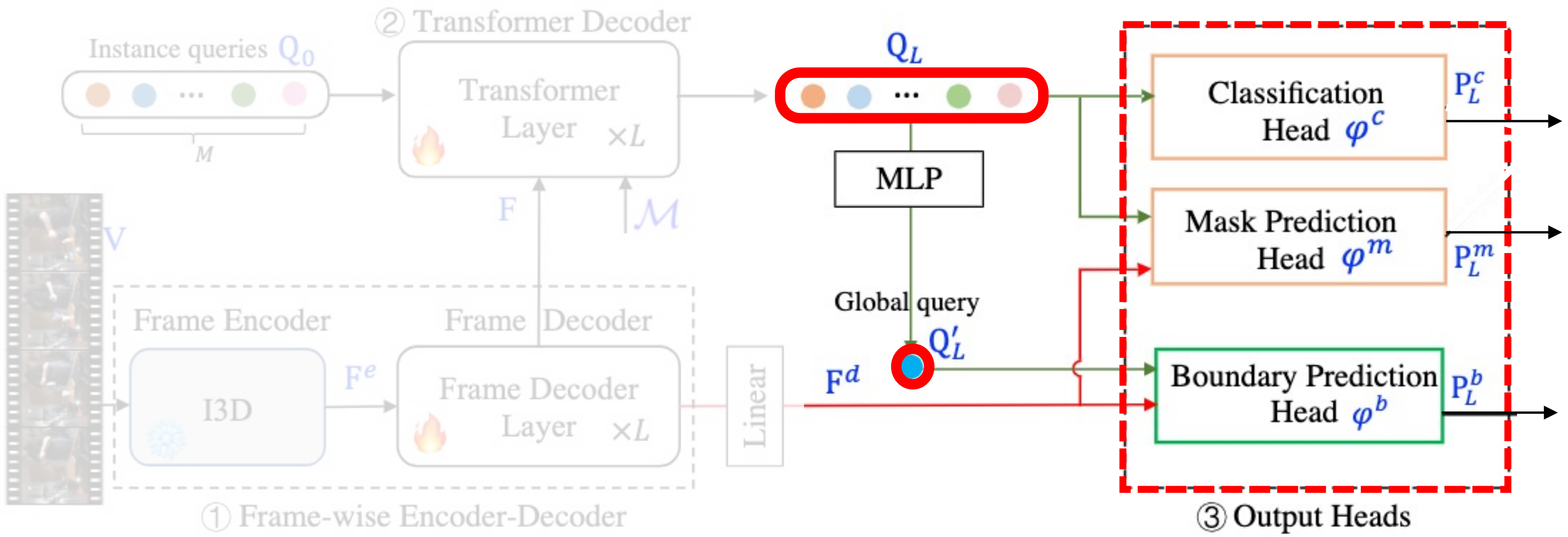
- Framework



Transformer Decoder: compress video sequences into sparse representations via queries

Method

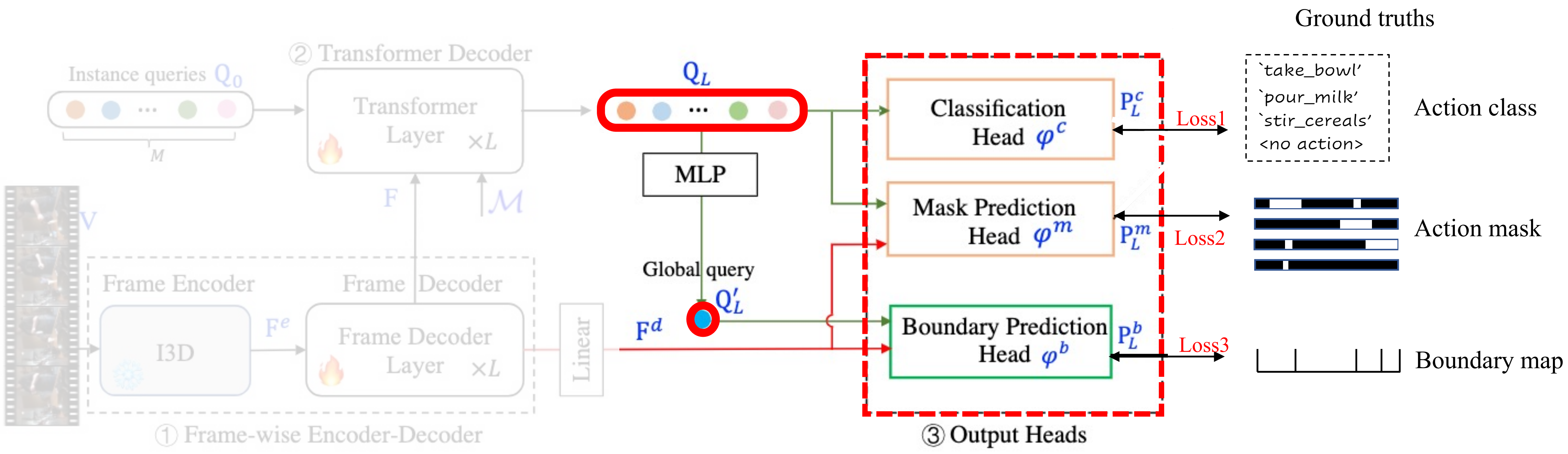
- Framework



Output Heads: generate query classes, query masks, and class-agnostic boundaries

Method

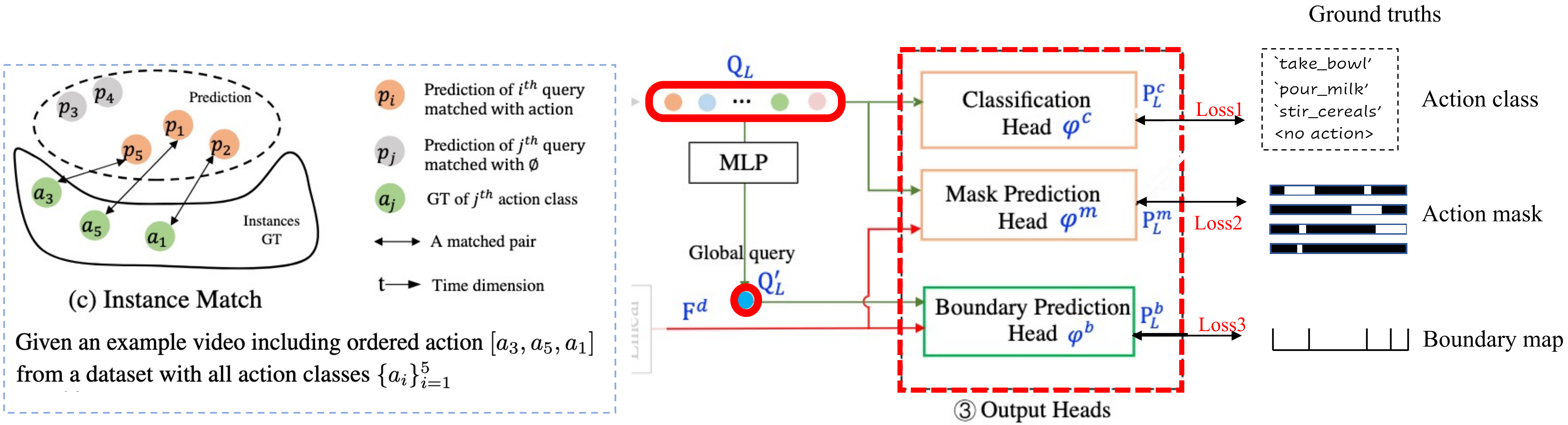
- Framework



Training: Match the outputs of queries with action class-mask pairs, the apply losses

Method

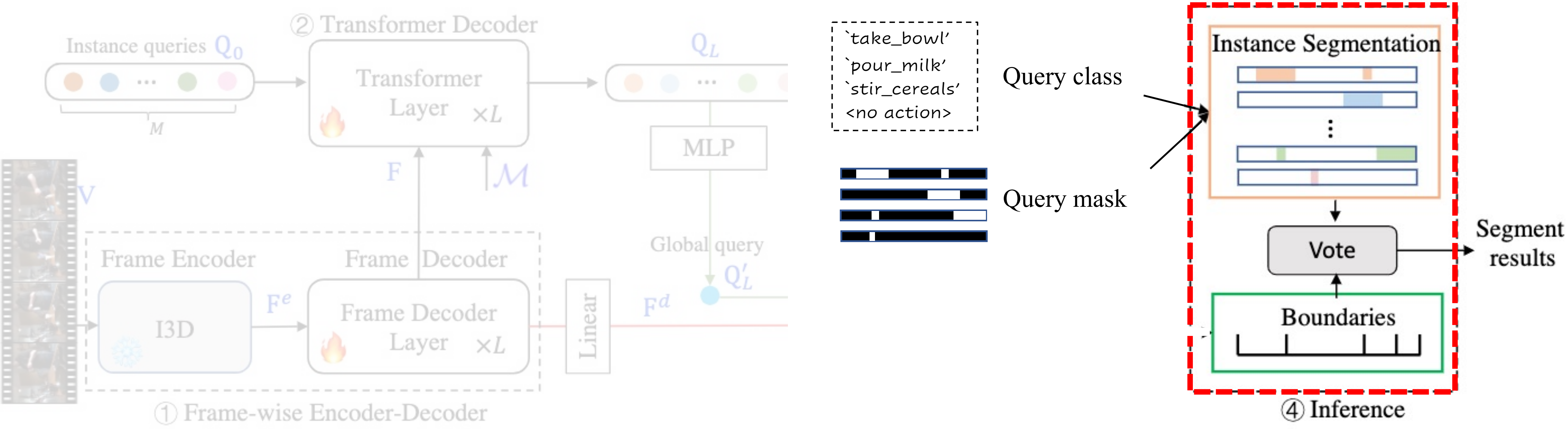
- Framework



Training: Match the outputs of queries with action class-mask pairs, the apply losses

Method

- Framework



Inference: derive the ultimate segmentation outcomes

Method

- Framework

Algorithm 1: Boundary-aware Query Voting

Input: Probability of query class–mask pairs: $\{(\mathbf{p}_i^c, \mathbf{p}_i^m)\}_{i=1}^{M}$, where $\mathbf{p}_i^c \in \mathbb{R}^{K+1}$, $\mathbf{p}_i^m \in \mathbb{R}^T$; Boundary probability: $\mathbf{P}^b = \{p_t^b\}_{t=1}^{T}$, where $p_t^b \in \mathbb{R}$ is the boundary probability in the t^{th} frame.

Output: Frame-wise segmentation: $\mathbf{S} \in \mathbb{R}^T$.

Initialize $\mathbf{S} \in \mathbb{R}^T$ with all zeros

$\mathbf{C} \leftarrow \{\text{cls}_i | \text{cls}_i = \text{argmax}(\mathbf{p}_i^c[:K])\}_{i=1}^{M}$

$\mathbf{B} = \{b_i\}_{i=1}^{N_b} \leftarrow \text{sort}\big(\{1, T\} \bigcup \{t | (p_t^b > p_{t-1}^b) \& (p_t^b > p_{t+1}^b), 1 < t < T\}\big)$

for $i = 1, 2, ..., N_b - 1$ **do**

  for $j = 1, 2, ..., M$ **do**

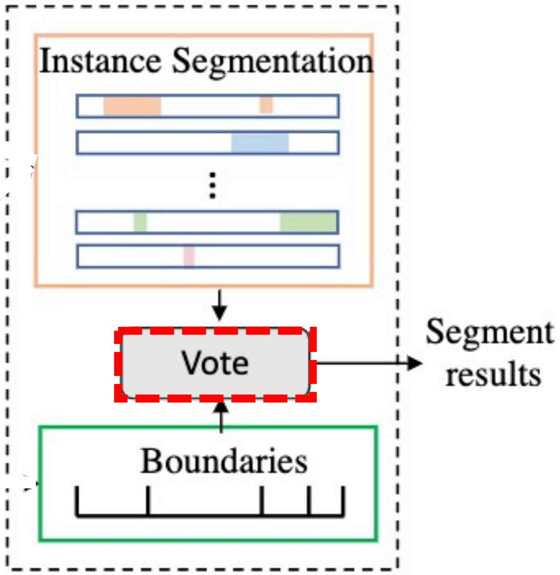
    $w_{ij} = \sum \mathbf{p}_j^m[b_i : b_{i+1}]$

  end

  $k = \text{argmax}_j(\{w_{ij}\}_{j=1}^{M})$

  $\mathbf{S}[b_i : b_{i+1}] = \text{cls}_k$

end



④ Inference

Inference: derive the ultimate segmentation outcomes

Experiments

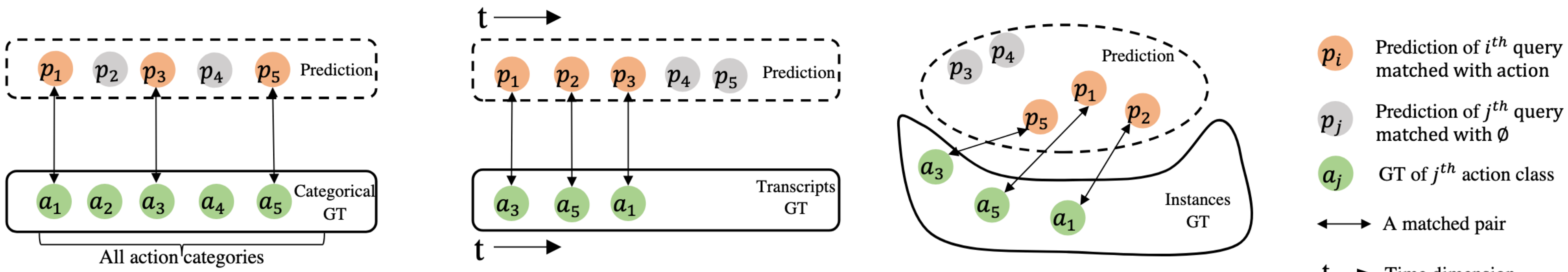
- Comparison with state-of-the-art methods

S	Method	Yr	Time (s)	FLOP (G)	Param (M)	GTEA F1@{10,25,50}			GTEA Edit	GTEA Acc.	50Salads F1@{10,25,50}			50Salads Edit	50Salads Acc.	Breakfast F1@{10,25,50}			Breakfast Edit	Breakfast Acc.
Multiple	MSTCN [15]	2019	0.094	4.59	0.80	85.8	83.4	69.8	79.0	76.3	76.3	74.0	64.5	67.9	80.7	52.6	48.1	37.9	61.7	66.3
	SSTDA [7]	2020	0.173	9.37	0.80	90.0	89.1	78.0	86.2	79.8	83.0	81.5	73.8	75.8	83.2	75.0	69.1	55.2	73.7	70.2
	BCN [43]	2020	0.152	73.54	12.77	88.5	87.1	77.3	84.4	79.8	82.3	81.3	74.0	74.3	84.4	68.7	65.5	55.0	66.2	70.4
	HASR [1]	2021	0.217	29.02	19.17	90.9	88.6	76.4	87.5	78.7	86.6	85.7	78.5	81.0	83.9	74.7	69.5	57.0	71.9	69.4
	DTGRM [41]	2021	0.261	3.75	0.73	87.8	86.6	72.9	83.0	77.6	79.1	75.9	66.1	72.0	80.0	68.7	61.9	46.6	68.9	68.3
	ASRF [21]	2021	0.163	7.43	1.30	89.4	87.8	79.8	83.7	77.3	84.9	83.5	77.3	79.3	84.5	74.3	68.9	56.1	72.4	67.6
	Gao *et al* [17]	2021	-	-	-	89.9	87.3	75.8	84.6	78.5	80.3	78.0	69.8	73.4	82.2	74.9	69.0	55.2	73.3	70.7
	ASFormer [45]	2021	0.359	6.66	1.13	90.1	88.8	79.2	84.6	79.7	85.1	83.4	76.0	79.6	85.6	76.0	70.6	57.4	75.0	73.5
	UARL [6]	2022	-	-	-	92.7	91.5	82.8	88.1	79.6	85.3	83.5	77.8	78.2	84.1	65.2	59.4	47.4	66.2	67.8
	DTL[44]	2022	0.403	6.66	1.13	-	-	-	-	-	87.1	85.7	78.5	80.5	86.9	78.8	74.5	62.9	77.7	75.8
	RTK [22]	2023	-	-	-	91.2	90.6	83.4	87.9	80.3	87.4	86.1	79.5	81.4	85.9	76.9	72.4	60.5	76.1	73.3
	LtContext [2]	2023	0.202	8.31	0.66	-	-	-	-	-	89.4	87.7	82.0	83.2	87.7	77.6	72.6	60.1	77.0	74.2
	DiffAct [32]	2023	2.306	43.94	1.21	**92.5**	**91.5**	**84.7**	**89.6**	**82.2**	**90.1**	**89.2**	**83.7**	**85.0**	**88.9**	**80.3**	**75.9**	**64.6**	**78.4**	**76.4**
	KARI [18]	2023	-	-	-	-	-	-	-	-	85.4	83.8	77.4	79.9	85.3	78.8	73.7	60.8	77.8	74.0
Single	UVAST† [3]	2022	0.577	3.86	1.27	77.1	69.7	54.2	90.5	62.2	86.2	81.2	70.4	83.9	79.5	76.7	70.0	56.6	77.2	68.2
	UVAST [3]	2022	480.888	3.06	1.10	**92.7**	**91.3**	81.0	**92.1**	80.2	89.1	87.6	81.7	83.9	87.4	76.9	71.5	58.0	77.1	69.7
	UVAST‡ [3]	2022	1.765	3.86	1.27	82.9	79.4	64.7	90.5	69.8	88.9	87.0	78.5	83.9	84.5	76.9	71.5	58.0	77.1	69.7
	BaFormer	–	0.139	4.54	1.63	92.0	**91.3**	**83.5**	88.7	**83.0**	**89.3**	**88.4**	**83.9**	**84.2**	**89.5**	**79.2**	**74.9**	**63.2**	**77.3**	**76.6**

Table 6: Performance on GTEA, 50Salads, and Breakfast datasets. In terms of running time, BaFormer outperforms all methods except MSTCN. As for accuracy, BaFormer achieves comparable or better results. UVAST†, UVAST, and UVAST‡ represent UVAST with alignment decoder, Viterbi, and FIFA. All FLOPs and running time are evaluated on 50Salads using the official codes in a consistent environment. We omit the running time and FLOPs on GTEA and Breakfast for simplicity as they are proportional to video length.

Experiments

- Different matching strategies



(a) Ordered Class Match (b) Transcript Match (c) Instance Match

Figure 5: Different matching strategies. Given an example video including ordered action $[a_3, a_5, a_1]$ from a dataset with all action classes $\{a_i\}_{i=1}^{5}$, (a) and (b) are fixed matching, while (c) is dynamic matching.

Match	#Q	FLOP (G)	Time (s)	Para (M)	F1 @{10, 25, 50}			Edit	Acc.
Ordered Class	19	3.74	0.136	1.49	88.1	87.0	83.5	82.7	87.9
Transcript	26	4.23	0.144	1.63	56.3	55.1	48.2	54.5	59.8
Instance†	26	4.23	0.144	1.63	85.3	84.6	79.9	79.8	86.1
Instance	100	4.45	0.139	1.63	**89.3**	**88.4**	**83.9**	**84.2**	**89.5**
$\Delta_{\text{Instance}-\text{Ordered-class}}$		+0.71	+0.003	+0.14	+1.2	+1.4	+0.4	+1.5	+1.6
$\Delta_{\text{Instance}-\text{Transcript}}$		+0.22	-0.005	+0.14	+33.0	+33.3	+35.7	+29.7	+29.7

Table 1: Comparative analysis of matching strategies on 50Salads. (#Q: number of queries.)

Experiments

- How well would our approach perform if we had perfect boundaries?

Method	**Time(s)**	**F1@{10,25,50}**			**Edit**	**Acc.**
NMS	0.138	89.1	88.4	**84.0**	83.8	89.1
peak	0.139	**89.3**	88.4	83.9	**84.2**	**89.5**
$\Delta_{\text{peak}-\text{NMS}}$	+0.001	+0.2	0	-0.1	+0.4	+0.4

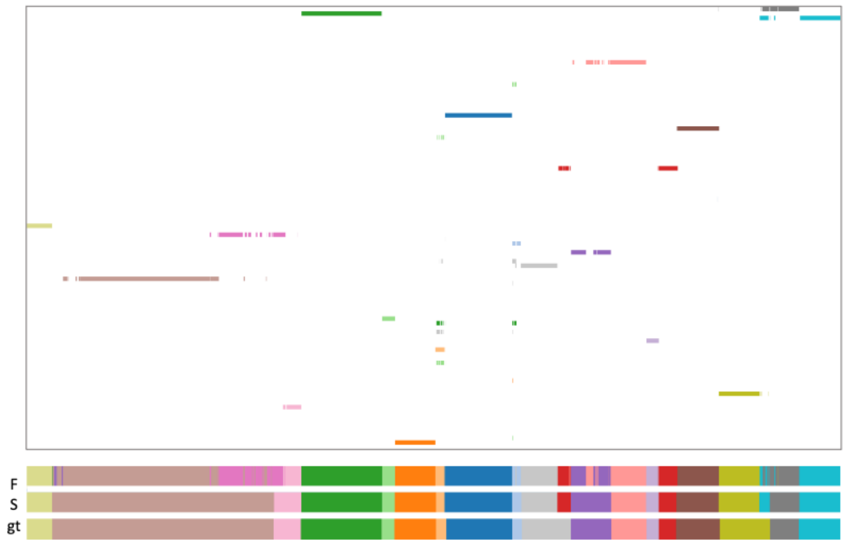
Table 4: Different strategies on boundary generation on 50Salads.

Boundary	**F1@{10,25,50}**			**Edit**	**Acc.**
Predict	89.3	88.4	83.9	84.2	89.5
GT	91.8	91.8	90.2	88.3	95.9
$\Delta_{\text{GT}-\text{Predict}}$	+2.5	+3.4	+6.3	+4.1	+6.4

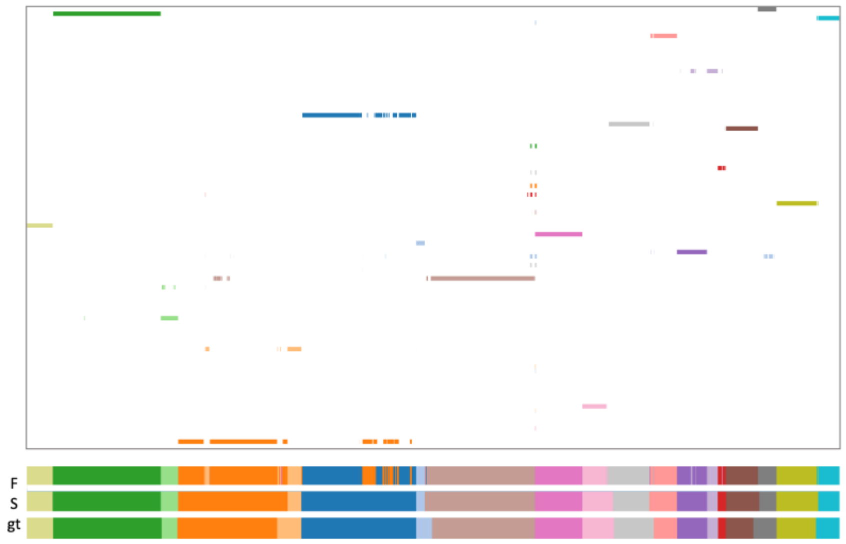
Table 5: Performance with predicted or ground-truth boundaries on 50Salads.

BaFormer yields more promising results by higher-quality class-agnostic boundaries

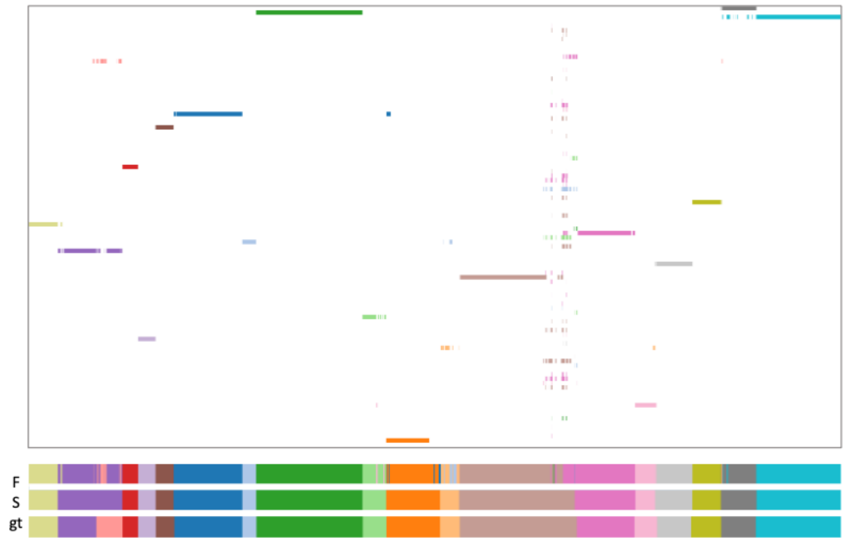
Visualization



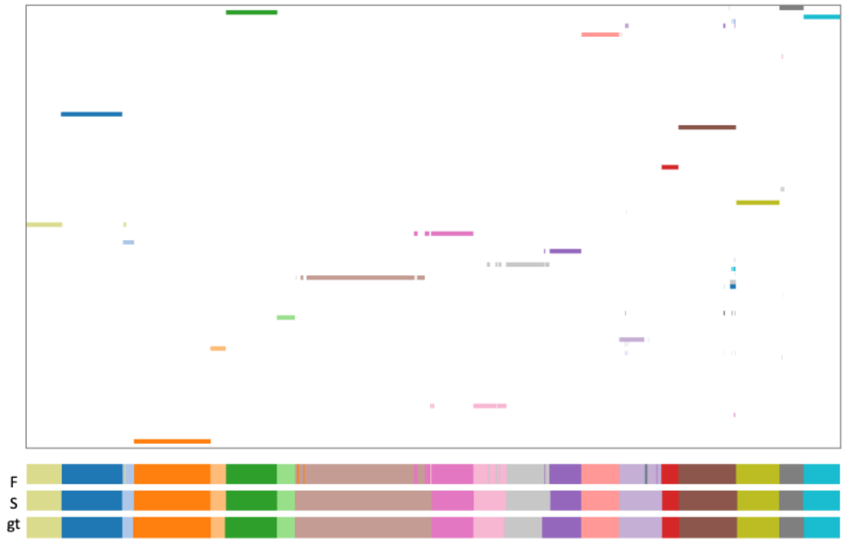
(a) “rgb-03-1” in 50Salads



(b) “rgb-07-2” in 50Salads



(c) “rgb-22-2” in 50Salads



(d) “rgb-25-2” in 50Salads

Instance segmentation, frame-wise results (F), voting results (S), and ground truth (gt)

Conclusion

- we introduce BaFormer, a novel boundary-aware, query-based approach for efficient temporal action segmentation.
- BaFormer employs a one-step strategy. It simultaneously predicts the query-wise class and mask, while yielding global boundary prediction for segment proposals.
- We apply query-based voting for segment proposal classification.
- BaFormer offers a unique perspective for addressing TAS challenges by integrating grouping and classification techniques

Stony Brook University

Brookhaven National Laboratory

AFRL AIR FORCE RESEARCH LABORATORY

The City College of New York

NEURAL INFORMATION PROCESSING SYSTEMS

Efficient Temporal Action Segmentation via Boundary-aware Query Voting

Thank you!

https://github.com/peiyao-w/BaFormer