paper code

CenterCLIP: Token Clustering for Efficient Text-Video Retrieval

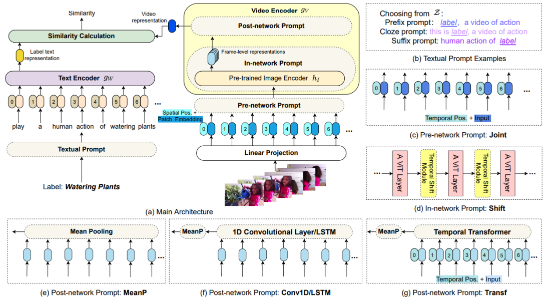
Shuai Zhao
CCAI, Zhejiang University

Linchao Zhu
ReLER Lab, AAII, University of Technology Sydney

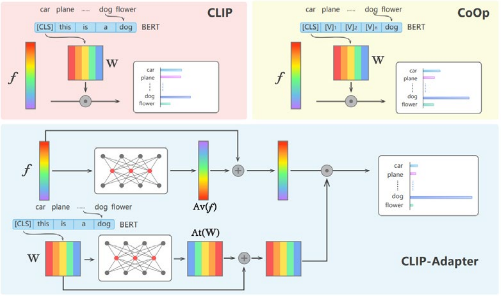
Xiaohan Wang
CCAI, Zhejiang University

Yi Yang
CCAI, Zhejiang University

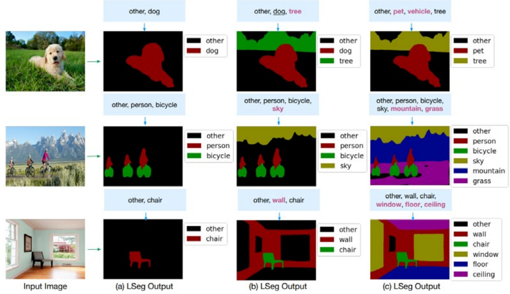
Research Based on CLIP



ActionCLIP[1]

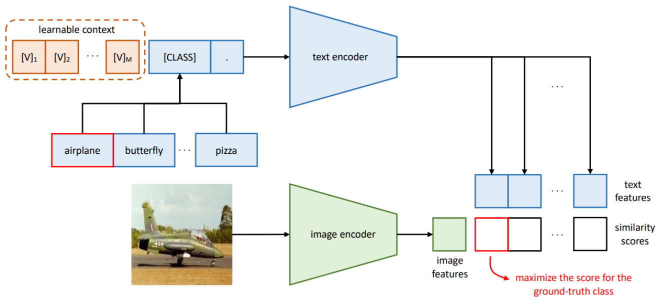


CLIP-Adapter[2]

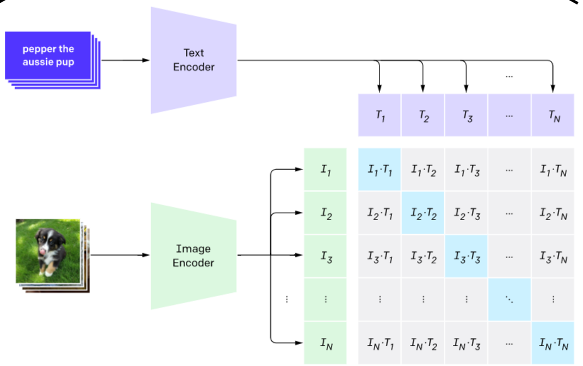


Lseg[3]

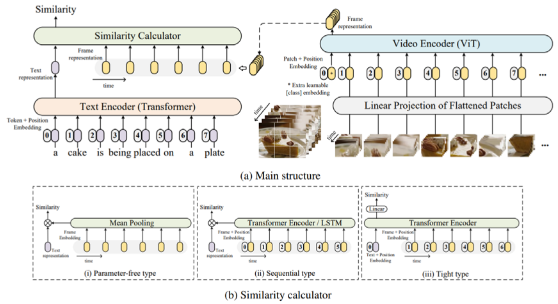
StyleCLIP[4]



CoOp[5]



CLIP

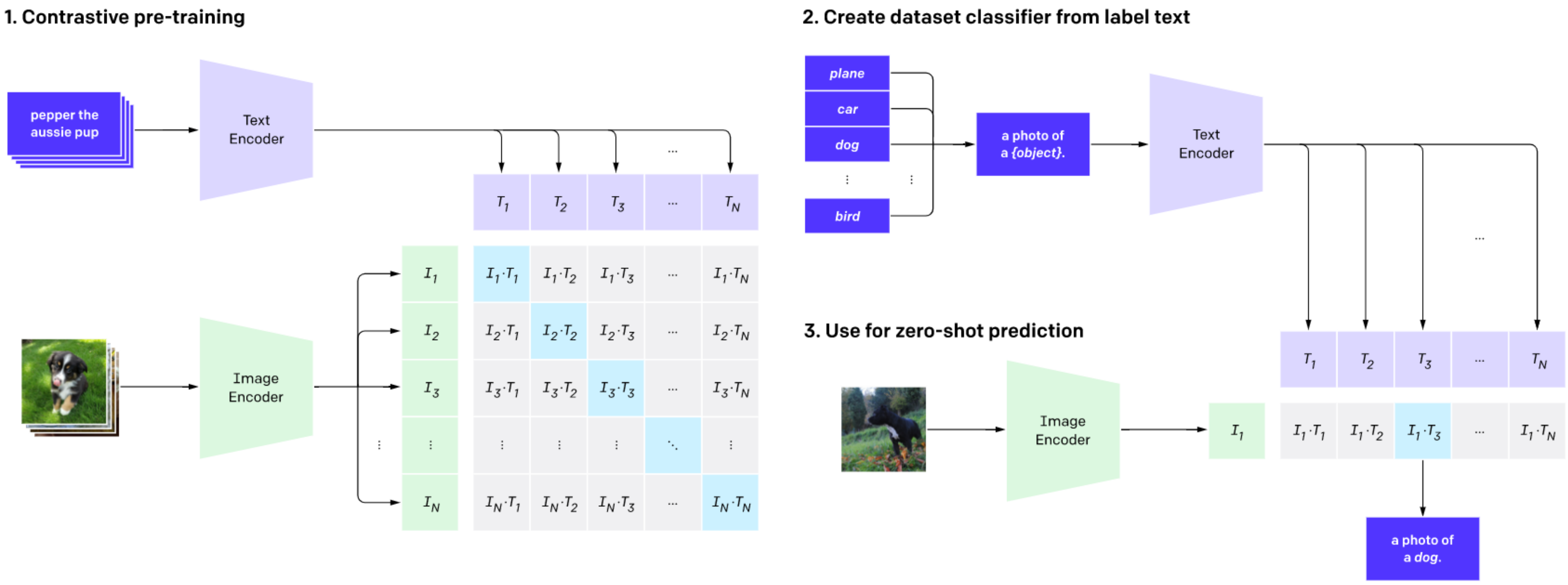


CLIP4clip[6]

CLIP is hot and plays a vital role in many multi-modal research works...

[1] Mengmeng Wang, Jiazheng Xing, Yong Liu. ActionCLIP: A New Paradigm for Video Action Recognition. CoRR abs/2109.08472 (2021)
[2] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, Yu Qiao. CLIP-Adapter: Better Vision-Language Models with Feature Adapters. CoRR abs/2110.04544 (2021)
[3] Boyi Li, Kilian Q. Weinberger, Serge J. Belongie, Vladlen Koltun, René Ranftl. Language-driven Semantic Segmentation. ICLR, 2022
[4] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, Dani Lischinski. StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery. ICCV 2021: 2065-2074
[5] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, Ziwei Liu. Learning to Prompt for Vision-Language Models. CoRR abs/2109.01134 (2021)
[6] Huaishao Luo, Lei Ji, Ming Zhong, Yang Chen, Wen Lei, Nan Duan, Tianrui Li. CLIP4Clip: An Empirical Study of CLIP for End to End Video Clip Retrieval. CoRR abs/2104.08860 (2021).

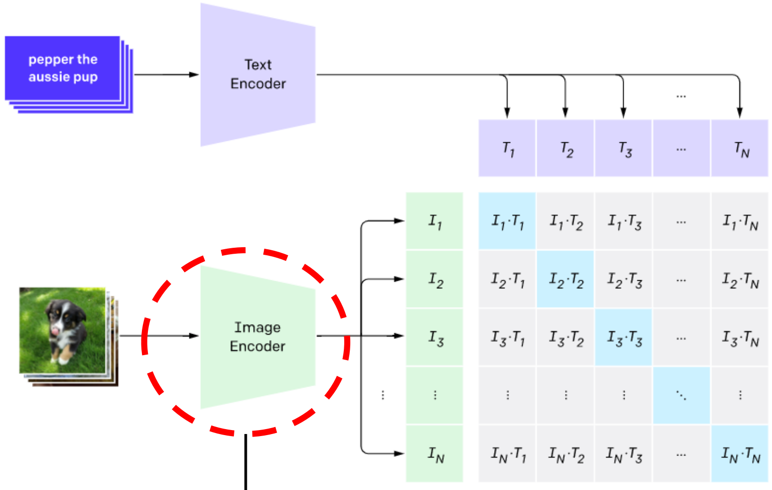
What Is CLIP ?



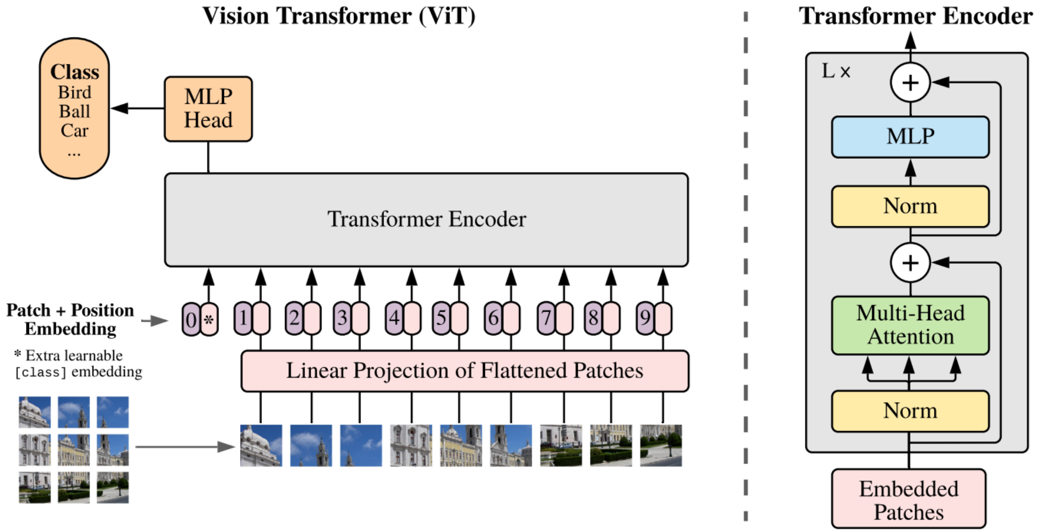
CLIP is a pre-training model consisted of an image encoder and text encoder.
It is pre-trained on **400 million (image, text) pairs** via contrastive learning and shows great generalization performance on over 30 different existing computer vision datasets.

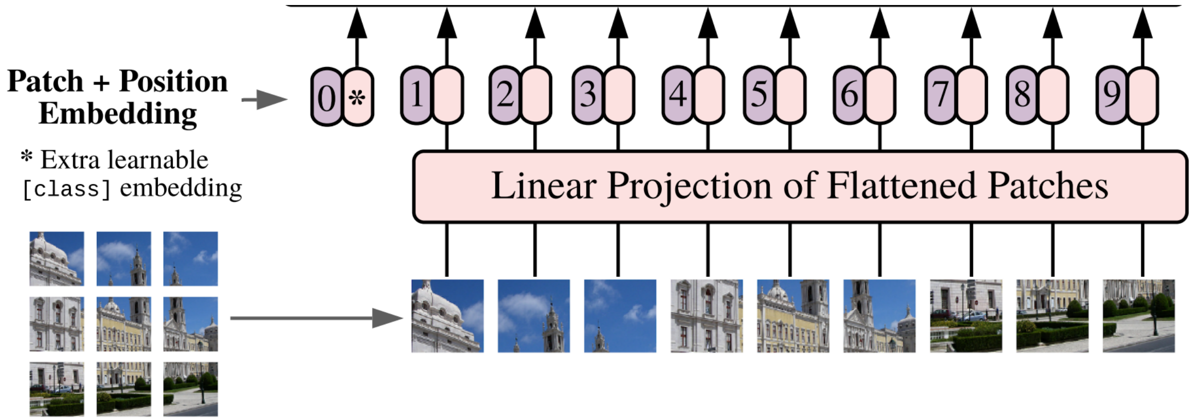
* The picture comes from https://openai.com/blog/clip/

Problem of CLIP When It Comes to Text-Video Retrieval



<u>The image encoder is generally a vision transformer (ViT).</u>





Images will be divided into patches and projected linearly into embedding space. Then they can be processed by the Multi-Head Self-Attention (MHSA) blocks in the transformer. This process is called ***visual tokenization***.

* The picture comes from Alexey Dosovitskiy et al. An image is worth 16x16 words: transformers for image recognition at scale. ICLR 2022.

Problem of CLIP When It Comes to Text-Video Retrieval

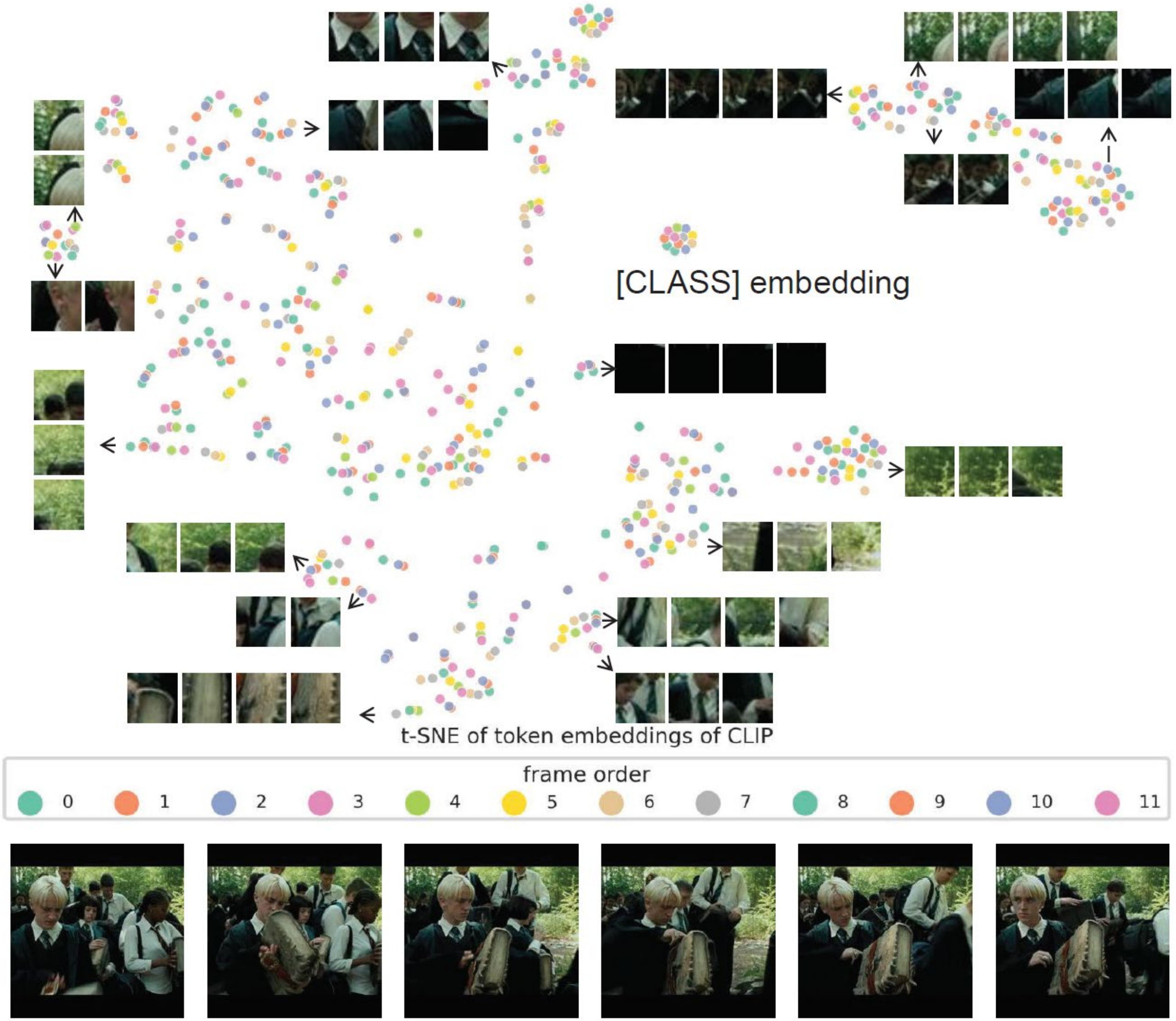


Figure. t-SNE visualization of video token embeddings of CLIP. The shown similar image patches within a cluster are from different temporal frames in the same video

Continuous frames in a video are similar.
In this case, visual tokenization process in the vision transformer of CLIP produces many homogeneous tokens.

Visual tokenization of a video usually produce hundreds or even thousands of frame tokens. Redundant tokens significantly increase computation costs and hinders the deployment of video retrieval models in web applications.

Motivation

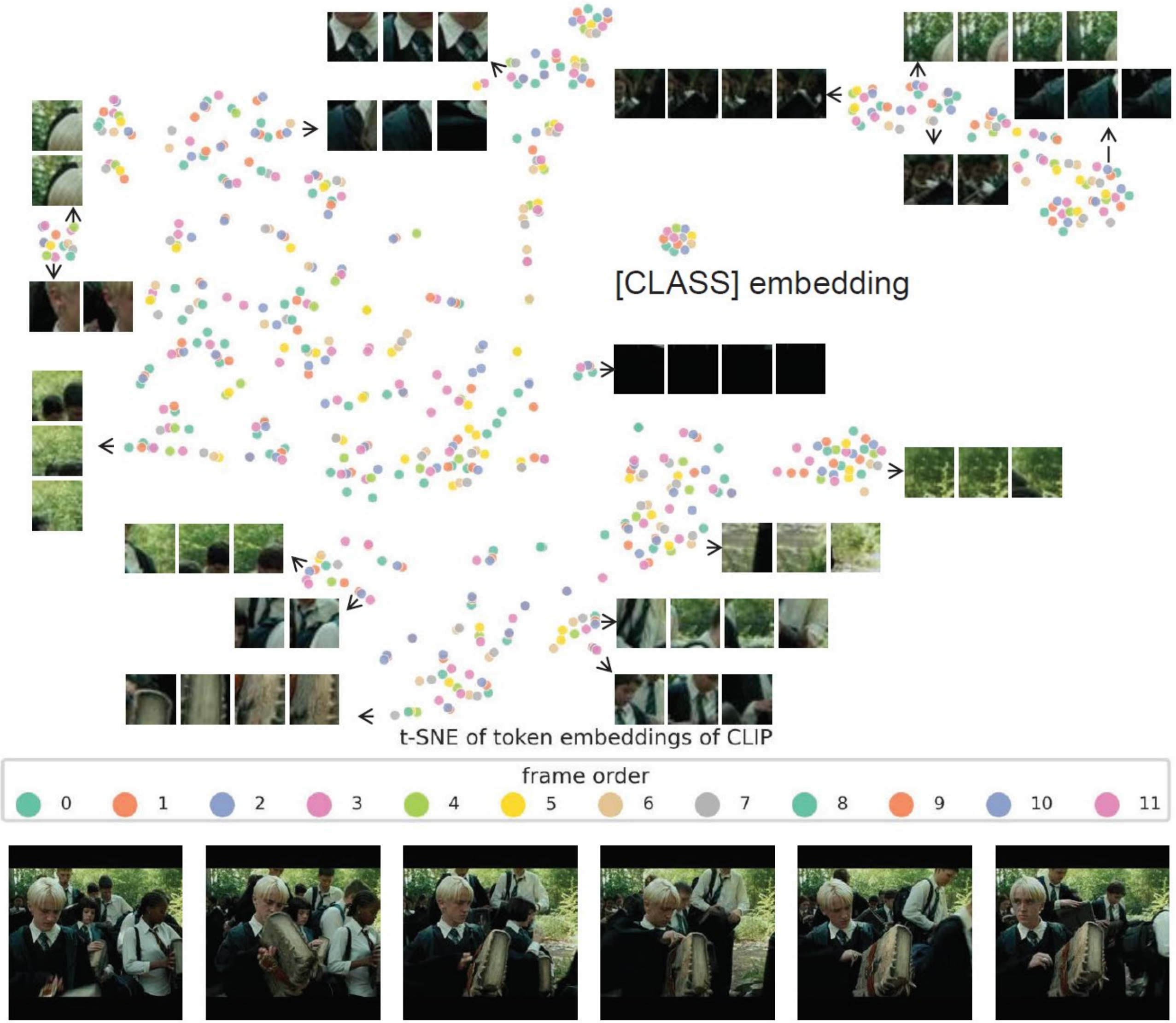


Figure. t-SNE visualization of video token embeddings of CLIP. The shown similar image patches within a cluster are from different temporal frames in the same video

The video tokens are redundant, *can we find a small set of tokens to represent the video*?

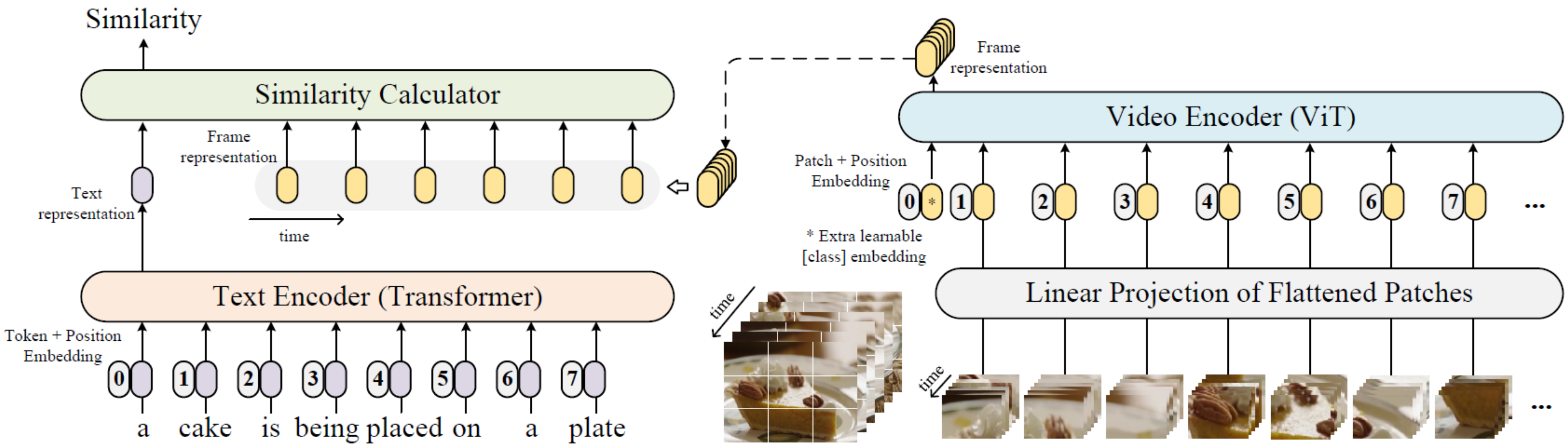
This small set of video should be discriminative enough for the video representation learning.

In the left picture, data points are naturally clustered, *can we use tokens corresponds to cluster centers to represent the video*?

The answer is absolutely **YES**! Center tokens contain most valuable information of the video.

Method

- **the base framework – following CLIP4clip**



The goal of the model is to learn a function f, which can score the similarity of the video v_i and text t_i. The whole model is consisted of a video encoder h and a text encoder g.

$$f(v_i, t_i) = h(v_i)^T g(t_i).$$

Generally, the video encoder and text encoder are both transformers.

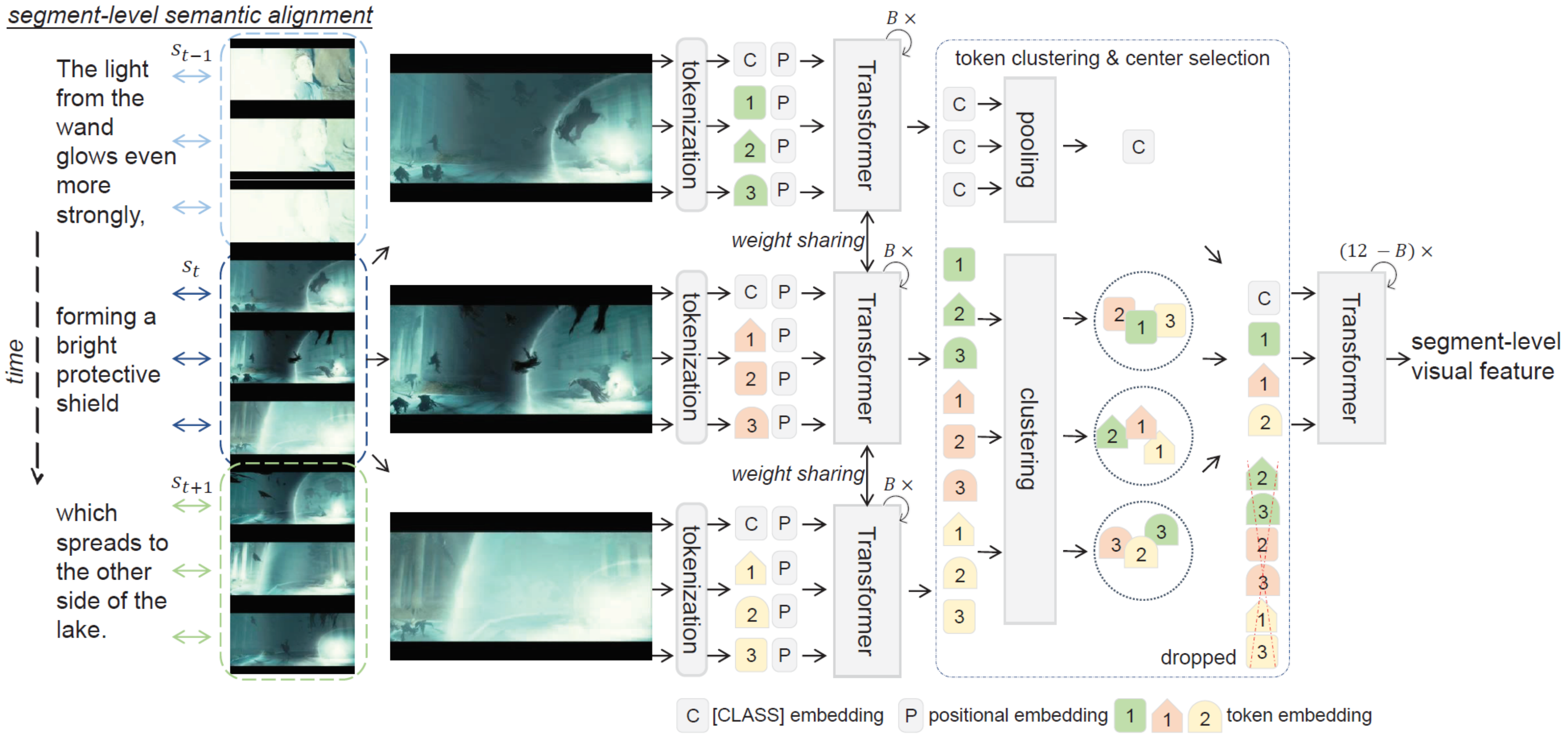
Learning objectives:

$$\mathcal{L}_{v2t} = -\frac{1}{N}\sum_i^N \log \frac{\exp(h(v_i)^T g(t_i)/\tau)}{\sum_{j=1}^N \exp(h(v_i)^T g(t_j)/\tau)}, \qquad \mathcal{L}_{t2v} = -\frac{1}{N}\sum_i^N \log \frac{\exp(g(t_i)^T h(v_i)/\tau)}{\sum_{j=1}^N \exp(g(t_i)^T h(v_j)/\tau)}, \qquad \mathcal{L} = \frac{1}{2}(\mathcal{L}_{v2t} + \mathcal{L}_{t2v}),$$

* The picture comes from Huaishao Luo, et al. CLIP4Clip: An Empirical Study of CLIP for End to End Video Clip Retrieval. CoRR abs/2104.08860 (2021).

Method

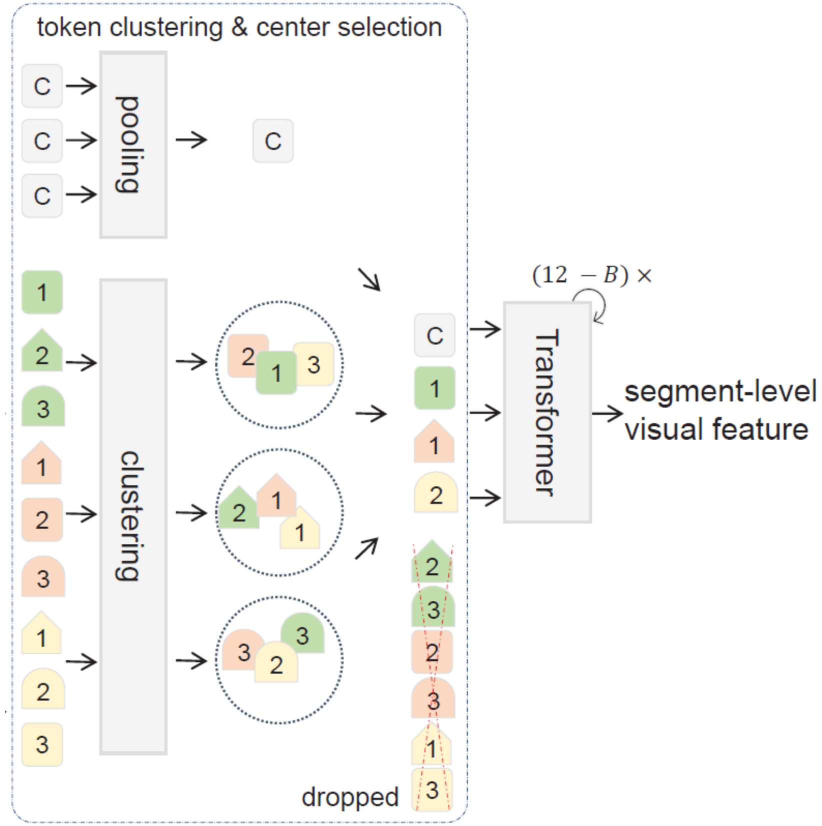
- **Multi-segment Token Clustering**



We adopt a multi-segment clustering strategy in the vision transformer as the neighbor frames are more likely to be similar. Here, the video is divided into three segments and each contains three frames. Clustering is performed independently on tokens of each segment, and center tokens of all clusters from one segment are selected and concatenated into a new sequence. Via attention on this new sequence, the visual model is able to learn features that contain segment-level video semantics.

Method

- **Multi-segment Token Clustering**



The similarity score of the video v_i and text t_i now becomes:

$$f(v_i, t_i) = \frac{1}{S}\sum_{j=1}^{S} h(s_i^j)^T g(t_i).$$

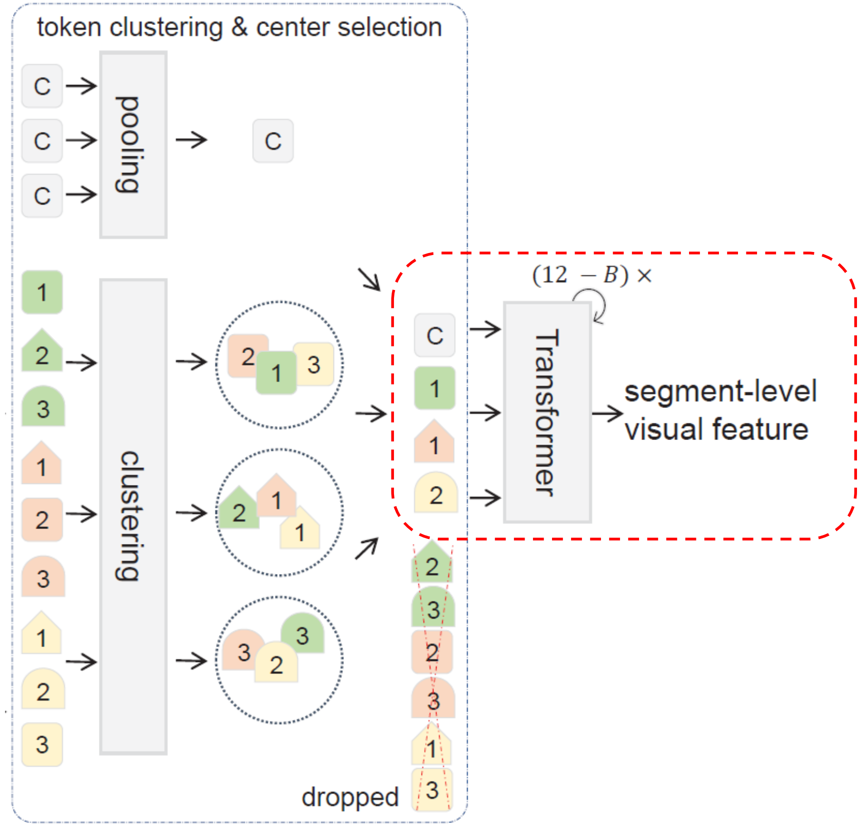
Here ***S*** is the number of segments $\{s_i^1, s_i^2, \ldots, s_i^S\}$ in a video v_i, which contains frames $\{v_i^1, v_i^2, \ldots v_i^{|v_i|}\}$. $|v_i|$ is the number of frames. Then each segments contains $|v_i|/S$ frames and $L|v_i|/S$ tokens, where L is the number of tokens per frames.

When input frame size is 224, for ViT-B/32, $L = 49$; for ViT-B/16, $L = 196$. The amount of the video tokens is $L|v_i|$, it can be up to 1000+ in some case.

By only reserving the center tokens after clustering, we save a lot of memory and computation cost.

Method

- **Multi-segment Token Clustering**



The similarity score of the video v_i and text t_i now becomes:

$$f(v_i, t_i) = \frac{1}{S} \sum_{j=1}^{S} h(s_i^j)^T g(t_i).$$

Via attention among tokens from different frames in a video segment, out method can learn segment-level visual representations.
This help the model to achieve segment-level semantic alignment.

SIGIR 2022 Madrid

Method

- **two instances of clustering method – k-medoids++**

Given a set of tokens $\{x_1, \ldots, x_m\} \in \mathbb{R}^d$, playing normal k-means on these tokens:

1. Initialize cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^d$ randomly;
2. For every i, set $p_i := \arg\min_j \|x_i - \mu_j\|_2^2$;
3. For every j, set $\mu_j := \frac{\sum_{i=1}^{m} \mathbf{1}\{p_i=j\}x_i}{\sum_{i=1}^{m} \mathbf{1}\{p_i=j\}}$; $\mathbf{1}\{\cdot\}$ equals to 1 if and only if the inner condition is true;
4. Repeat step 2 and 3 until convergence.

Random initialization is not suitable for text-video retrieval – a deterministic initialization method:

Algorithm 1: KKZ initialization for k-means [23]

Input: tokens $\{x_1, \ldots, x_m\} \in \mathbb{R}^d$, cluster number K
Output: centroids $\{\mu_1, \mu_2, \ldots, \mu_K\} \in \mathbb{R}^d$

$\mu_1 \leftarrow \arg\max_{x_i} \|x_i\|_2$;
for $i \leftarrow 2$ **to** K **do**
  for $j \leftarrow 1$ **to** m **do**
    for $k \leftarrow 1$ **to** i **do**
      `// calculate distance between` x_j `and` μ_k
      $d_{j,k} \leftarrow \mathtt{distance}(x_j, \mu_k)$;
    end
    $d_j \leftarrow \min_k d_{j,k}$;
  end
  $\mu_i \leftarrow \arg\max_{x_j} d_j$;
end

SIGIR 2022
Madrid

Method

- **two instances of clustering method – spectral clustering**

Data clusters may not be spherical in the high-dimension space. Spectral clustering maybe better in such cases.

1. Construct similarity graph. Let W be its weighted adjacency matrix, D be the degree matrix;
2. Compute normalized Laplacian $L_{sym} = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$;
3. Compute the first K eigenvectors $\mu_1, \ldots, \mu_k$ of L_{sym} which correspond to the first K least eigenvalues;
4. Let $U = [\mu_1, \ldots, \mu_k]$; Normalize each row of U to have norm of 1, generally, ℓ_2 norm is used;
5. Consider each row of U as a new data point, apply k-means to these data points.

Directions of eigenvectors also matter for some distance metric. We align the direction of eigenvectors with the major direction of data points.

Algorithm 2: sign flip for SVD [5]

Input: $L \in \mathbb{R}^{m \times m}$, truncated singular value decomposition (U, Σ, V) of L, $U = [u_1, \ldots, u_K] \in \mathbb{R}^{m \times K}$

Output: $U' = [u'_1, \ldots, u'_K]$ with appropriate signs

for $k \leftarrow 1$ **to** K **do**

  $Y = L - \sum_{i=1, i \neq k}^{K} \sigma_i u_i v_i^T$;

  /* sign(·) return the sign of input, $Y_{\cdot,j}$ denote the j-th column of Y */

  $s_k = \sum_{j=1}^{m} \mathrm{sign}(u_k^T Y_{\cdot,j})(u_k^T Y_{\cdot,j})^2$;

end

for $k \leftarrow 1$ **to** K **do**

  $u'_k = \mathrm{sign}(s_k) u_k$;

end

Experiments

- **datasets**

MSVD, MSR-VTT, LSMDC, ActivityNet

- **Metric**

Recall at rank $\mathbb{K}$, R@$\mathbb{K}$, higher is better
Median rank, MdR, lower is better
Mean rank, MnR, lower is better

- **Setting of our method – CenterCLIP**

We use $(B_a - S, K)$ to represent the setting. It means we perform token clustering right after the a-th transformer block, the number of temporal segments is S, and the number of clusters/centers are constant K.

Experiments

- **MSVD**

Method	MeM. GB	Speed ms	Text → Video R@1↑	R@5↑	R@10↑	MdR↓	MnR↓	Video → Text R@1↑	R@5↑	R@10↑	MdR↓	MnR↓
CE [29]	-	-	19.8	49.0	63.8	6	-	-	-	-	-	-
TT-CE+ [10]	-	-	25.4	56.9	71.3	4	-	27.1	55.3	67.1	4	-
Frozen in Time [2]	-	-	33.7	64.7	76.3	3	-	-	-	-	-	-
CLIP zero-shot	-	-	37.0	64.1	73.8	3	-	59.9	85.2	90.7	1	-
CLIP4clip (meanP) [32]	20.8	24.4	46.2	76.1	84.6	2	10.0	56.6	79.7	84.3	1	7.6
CLIP4clip (seqTransf)	-	-	45.2	75.5	84.3	2	10.3	62.0	87.3	92.6	1	4.3
baseline (CLIP4clip (meanP), ViT-B/32)	20.8	24.4	45.9	74.9	84.7	2	10.4	51.0	76.3	82.2	1	9.1
CenterCLIP (k-medoids++, $\mathcal{B}_6 - 4, 49$)	15.0	22.9	47.6	77.5	86.0	2	9.8	54.2	78.4	84.9	1	7.6
CenterCLIP (k-medoids++, $\mathcal{B}_6 - 3, 49$)	14.2	22.9	47.3	76.8	85.6	2	9.9	57.9	83.6	90.5	1	5.2
CenterCLIP (spectral, $\mathcal{B}_6 - 4, 49$)	14.9	40.8	47.4	76.5	85.2	2	9.7	62.7	88.1	92.8	1	4.1
CenterCLIP (spectral, $\mathcal{B}_6 - 3, 49$)	14.2	43.6	47.3	76.9	86.0	2	9.7	63.5	86.4	92.6	1	3.8
baseline (CLIP4clip (meanP), ViT-B/16)	25.7	59.6	49.6	79.5	88.0	2	8.6	62.7	83.9	89.4	1	6.1
CenterCLIP (k-medoids++, $\mathcal{B}_6 - 4, 160$)	17.6	86.5	50.6	80.3	88.4	1	8.4	68.4	90.1	95.0	1	3.0

Table 1: Results on MSVD. MeM. is the average GPU memory cost when training on 2 and 8 Tesla V100 GPUs for ViT-B/32 and ViT-B/16, respectively. Speed is the inference time per video during evaluation on a Tesla V100 GPU.

significant improvement, SOTA performance, 32% reduction in memory (ViT-B/32), 6% gain of speed (ViT-B/32)

Experiments

- **ActivityNet**

Method	MeM. GB	Speed ms	Text → Video R@1↑	R@5↑	R@10↑	MdR↓	MnR↓	Video → Text R@1↑	R@5↑	R@10↑	MdR↓	MnR↓
FSE [67]	-	-	18.2	44.8	-	7.0	-	16.7	43.1	-	7.0	-
CE [29]	-	-	18.2	47.7	-	6.0	12.1	17.7	46.6	-	6.0	24.4
CMGSD [15]	-	-	24.2	56.3	-	4.0	-	24.6	56.8	-	4.0	-
MMT [13]	-	-	28.7	61.4	-	3.3	16.0	28.9	61.1	-	4.0	17.1
TT-CE+ [10]	-	-	23.5	57.2	-	4.0	-	23.0	56.1	-	4.0	-
T2VLAD [56]	-	-	23.7	55.5	-	4.0	-	24.1	56.6	-	4.0	-
SSB [38]	-	-	29.2	61.6	-	3.0	-	28.7	60.8	-	2.0	-
CLIP zero-shot	-	-	21.7	46.0	59.6	7.0	39.7	17.9	40.8	54.2	8.0	43.3
CLIP4clip (meanP) [32]	25.0	82.0	40.5	72.4	-	2.0	7.4	42.5	74.1	85.8	2.0	6.6
CLIP4clip (seqTransf)	-	-	40.5	72.4	-	2.0	7.5	41.4	73.7	85.3	2.0	6.7
baseline (CLIP4clip (meanP), ViT-B/32)	25.0	82.0	41.8	73.9	84.7	2.0	7.3	42.8	73.8	85.3	2.0	6.9
CenterCLIP (k-medoids++, $\mathcal{B}_6 - 15, 49$)	16.8	71.3	43.9	75.3	85.2	2.0	7.0	44.2	75.0	86.1	2.0	6.8
CenterCLIP (k-medoids++, $\mathcal{B}_6 - 12, 49$)	16.2	70.4	43.5	75.0	85.9	2.0	6.9	44.5	75.3	86.0	2.0	6.7
CenterCLIP (spectral, $\mathcal{B}_6 - 20, 49$)	17.7	162	43.5	75.1	85.4	2.0	6.9	44.1	75.1	86.0	2.0	6.7
CenterCLIP (spectral, $\mathcal{B}_6 - 15, 49$)	16.8	174	43.9	74.6	85.8	2.0	6.7	44.5	75.7	86.2	2.0	6.5
CenterCLIP (k-medoids++, $\mathcal{B}_6 - 15, 160$, ViT-B/16)	23.0	419	46.2	77.0	87.6	2.0	5.7	46.7	77.1	88.0	2.0	5.5

Table 2: Results on ActivityNet. MeM. is the average GPU memory cost when training on 8 and 32 Tesla V100 GPUs. Baseline with ViT-B/16 OOM on 32 Tesla V100 GPUs. Speed is the inference time per video during evaluation on a Tesla V100 GPU.

significant improvement, SOTA performance, 35% reduction in memory (ViT-B/32), 14% gain of speed (ViT-B/32)

Experiments

- **MSR-VTT**

Method	MeM. GB	Speed ms	Text → Video R@1↑	R@5↑	R@10↑	MdR↓	MnR↓	Video → Text R@1↑	R@5↑	R@10↑	MdR↓	MnR↓
ActBERT [69]	-	-	8.6	23.4	33.1	36	-	-	-	-	-	-
JSFusion [63]	-	-	10.2	31.2	43.2	13	-	-	-	-	-	-
HowTo100M [35]	-	-	14.9	40.2	52.8	9	-	-	-	-	-	-
CE [29]	-	-	20.9	48.8	62.4	6	-	20.6	50.3	64.0	5.3	-
MMT [13]	-	-	26.6	57.1	69.6	4	24.0	27.0	57.5	69.7	3.7	21.3
T2VLAD [56]	-	-	29.5	59.0	70.1	4	-	31.8	60.0	71.1	3	-
AVLnet [43]	-	-	27.1	55.6	66.6	4	-	28.5	58.6	71.6	3	-
TT-CE+ [10]	-	-	29.6	61.6	74.2	3	-	32.1	62.7	75.0	3	-
CLIP zero-shot	-	-	31.2	53.7	64.2	4	-	27.2	51.7	62.6	5	-
CLIP4clip (meanP) [32]	20.8	24.4	43.1	70.4	80.8	2	16.2	43.1	70.5	81.2	2	12.4
CLIP4clip (seqTransf)	-	-	44.5	71.4	81.6	2	15.3	42.7	70.9	80.6	2	11.6
baseline (CLIP4clip (meanP), ViT-B/32)	20.8	24.4	43.0	70.7	80.6	2	16.2	43.1	70.8	80.6	2	11.4
CenterCLIP (k-medoids++, $B_6 - 4, 49$)	15.0	22.9	43.6	71.4	81.2	2	15.3	42.9	70.4	80.8	2	10.8
CenterCLIP (k-medoids++, $B_6 - 3, 49$)	14.2	22.9	44.0	70.7	81.4	2	15.7	42.9	71.4	81.7	2	11.1
CenterCLIP (spectral, $B_6 - 4, 49$)	14.9	40.8	43.6	71.7	80.6	2	15.4	43.5	72.1	82.2	2	11.1
CenterCLIP (spectral, $B_6 - 3, 49$)	14.2	43.6	44.2	71.6	82.1	2	15.1	42.8	71.7	82.2	2	10.9
baseline (CLIP4clip (meanP), ViT-B/16)	25.7	59.6	45.6	71.2	80.9	2	15.2	43.2	72.5	80.7	2	10.9
CenterCLIP (k-medoids++, $B_6 - 4, 160$)	17.6	86.5	48.4	73.8	82.0	2	13.8	47.7	75.0	83.3	2	10.2

(b) Training on training-9K

Table 3: Results on MSR-VTT. MeM. is the average GPU memory cost when training on 2 and 8 Tesla V100 GPUs for ViT-B/32 and ViT-B/16, respectively. Speed is the inference time per video during evaluation on a Tesla V100 GPU.

significant improvement, SOTA performance

Experiments

SIGIR 2022 Madrid

- **LSMDC**

Method	MeM. GB	speed ms	Text → Video R@1↑	R@5↑	R@10↑	MdR↓	MnR↓	Video → Text R@1↑	R@5↑	R@10↑	MdR↓	MnR↓
JSFusion [63]	-	-	9.1	21.2	34.1	36.0	-	12.3	28.6	38.9	20.0	-
CE [29]	-	-	11.2	26.9	34.8	25.3	96.8	-	-	-	-	-
MMT [13]	-	-	12.9	29.9	40.1	19.3	75.0	12.3	28.6	38.9	20.0	76.0
Frozen in Time [2]	-	-	15.0	30.8	39.8	20.0	-	-	-	-	-	-
TT-CE+ [10]	-	-	17.2	36.5	46.3	13.7	-	17.5	36.0	45.0	14.3	-
CLIP zero-shot	-	-	15.1	28.3	35.8	31.0	132	7.5	18.4	25.1	58.0	151
CLIP4clip (meanP) [32]	20.8	24.4	20.7	38.9	47.2	13.0	65.3	20.6	39.4	47.5	13.0	56.7
CLIP4clip (seqTransf)	-	-	22.6	41.0	49.1	11.0	61.0	20.8	39.0	48.6	12.0	54.2
baseline (CLIP4clip (meanP), ViT-B/32)	20.8	24.4	20.1	40.2	48.4	12.0	57.1	21.2	39.3	48.4	12.0	50.8
CenterCLIP (k-medoids++, $\mathcal{B}_6 - 6, 49$)	16.4	23.9	21.9	41.1	50.7	10.0	55.6	21.1	41.2	50.2	10.0	48.7
CenterCLIP (k-medoids++, $\mathcal{B}_6 - 4, 49$)	15.0	22.9	21.7	39.8	49.8	11.0	54.8	21.4	40.3	50.8	10.0	48.4
CenterCLIP (spectral, $\mathcal{B}_6 - 6, 49$)	16.4	40.8	21.6	40.9	49.3	11.0	57.2	20.6	39.5	48.8	12.0	51.4
CenterCLIP (spectral, $\mathcal{B}_6 - 4, 49$)	15.0	43.6	21.4	39.7	49.4	11.0	55.9	19.5	39.9	48.0	12.0	50.1
baseline (CLIP4clip (meanP), ViT-B/16)	25.7	59.6	24.1	45.0	55.1	8	51.1	22.5	42.9	53.5	9	45.1
CenterCLIP (k-medoids++, $\mathcal{B}_6 - 4, 160$)	17.6	86.5	24.2	46.2	55.9	8	47.3	24.5	46.4	55.8	7	41.3

Table 4: Results on LSMDC. MeM. is the average GPU memory cost when training on 2 and 8 Tesla V100 GPUs for ViT-B/32 and ViT-B/16, respectively. Speed is the inference time per video during evaluation on a Tesla V100 GPU.

SOTA performance

SIGIR 2022 Madrid

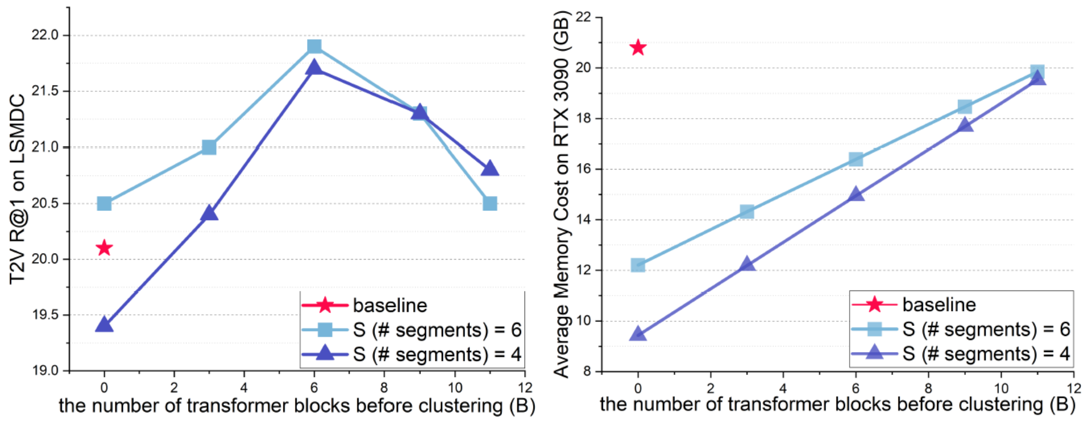
Experiments

- **more baselines**

Method	Mem.	T↔V	R@1↑	R@5↑	R@10↑	MdR↓	MnR↓
MSR-VTT (train on training-7K)							
pooling ($B_6-6,49$)	16.39	T→V	41.9	66.6	76.7	2	18.7
		V→T	40.2	65.6	75.8	2	14.4
pooling ($B_6-4,49$)	14.95	T→V	40.6	65.6	75.8	2	17.5
		V→T	40.6	67.3	77.3	2	14.6
sparse sampling ($B_6-6,49$)	16.39	T→V	42.6	68.4	78.4	2	17.6
		V→T	41.6	68.3	77.5	2	12.8
sparse sampling ($B_6-4,49$)	14.95	T→V	42.3	69.1	78.6	2	17.6
		V→T	40.3	66.7	77.0	2	13.5
token shift [68]	20.77	T→V	42.5	68.5	79.6	2	16.4
		V→T	43.3	70.1	80.8	2	12.2
temporal shift [54]	20.77	T→V	34.2	61.8	73.7	3	21.9
		V→T	31.5	61.8	72.2	3	18.2
CenterCLIP ($B_6-6,49$)	16.39	T→V	43.3	69.9	78.6	2	17.7
		V→T	41.8	68.9	77.3	2	12.7
CenterCLIP ($B_6-4,49$)	14.95	T→V	43.7	71.3	80.8	2	16.9
		V→T	41.8	70.2	79.8	2	11.8
LSMDC							
sparse sampling ($B_6-6,49$)	16.39	T→V	20.6	38.7	48.6	12.0	59.8
		V→T	20.4	37.9	45.6	13.5	54.2
token shift [68]	20.77	T→V	21.4	42.3	50.2	10.0	55.6
		V→T	21.7	41.6	50.1	10.0	49.6
CenterCLIP ($B_6-4,49$)	14.95	T→V	21.7	39.8	49.8	11.0	54.8
		V→T	21.4	40.3	50.8	10.0	48.4
ActivityNet							
token shift [68]	24.98	T→V	42.0	73.6	84.8	2.0	7.3
		V→T	42.5	74.3	85.2	2.0	7.0
CenterCLIP ($B_6-15,49$)	16.75	T→V	43.9	75.3	85.2	2.0	7.0
		V→T	44.2	75.0	86.1	2.0	6.8

Table 5: Comparison with strong token selection baselines.

- **place of performing token clustering**



(a) T2V R@1 on LSMDC **(b) Memory cost on RTX 3090**

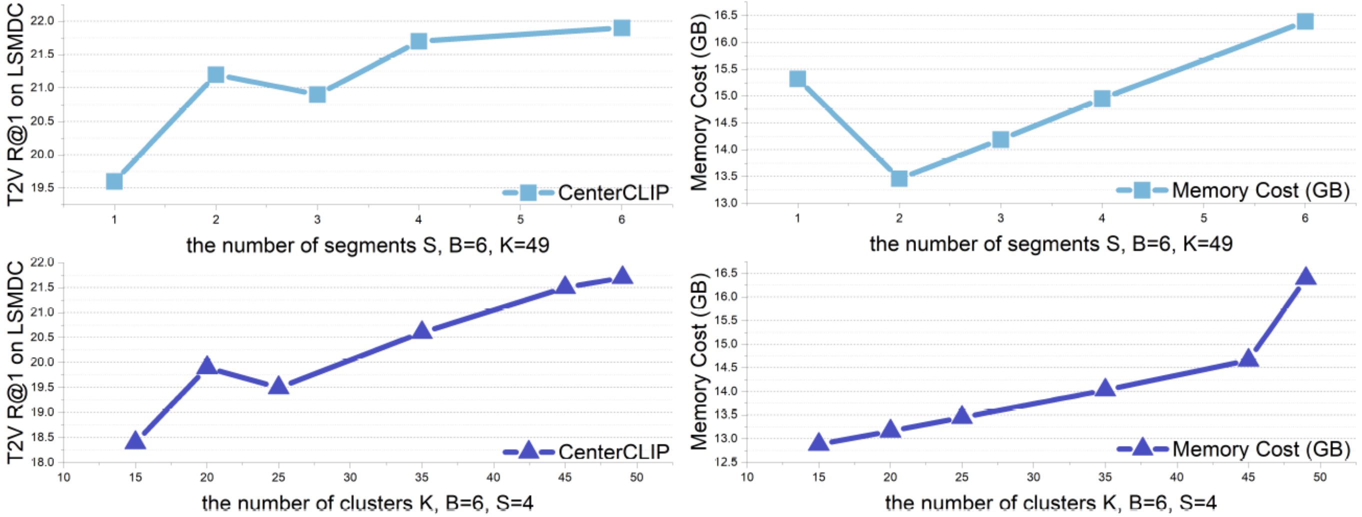
Figure 3: Influence of places of token clustering.

Method	Mem.	T↔V	R@1↑	R@5↑	R@10↑	MdR↓	MnR↓
CenterCLIP ($B_4-6,49$), ($B_8-3,49$)	13.52	T→V	21.0	42.7	51.9	9.0	57.0
		V→T	20.9	41.8	51.5	9.0	50.7
CenterCLIP ($B_6-4,49$)	16.39	T→V	21.7	39.8	49.8	11.0	54.8
		V→T	21.4	40.3	50.8	10.0	48.4
CenterCLIP ($B_6-6,49$)	14.95	T→V	21.9	41.1	50.7	10.0	55.6
		V→T	21.1	41.2	50.2	10.0	48.7

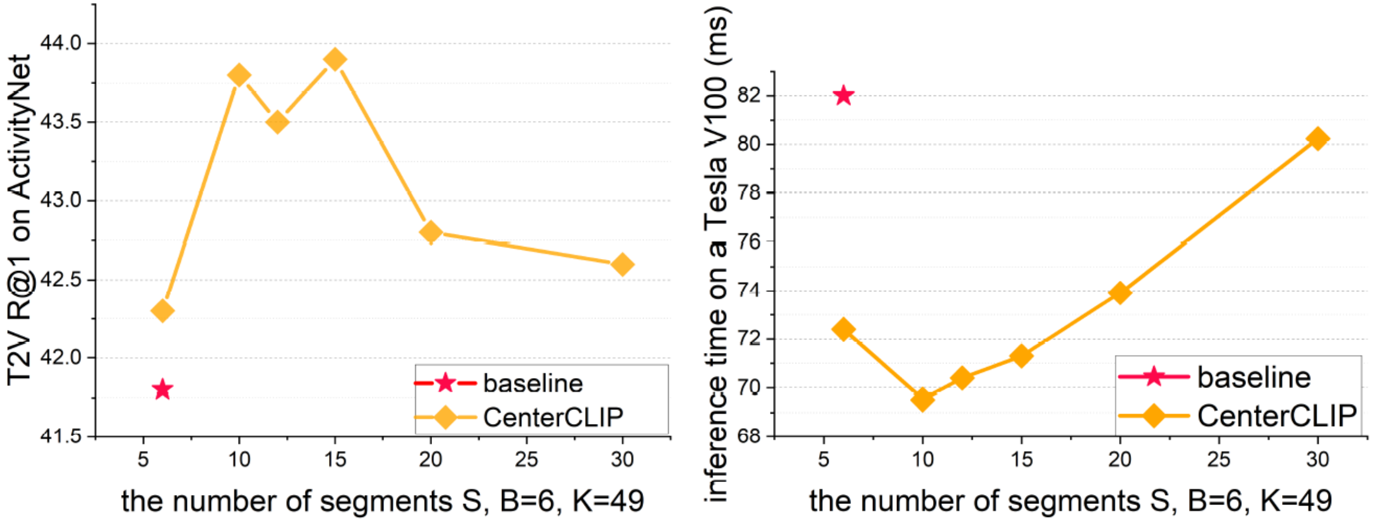
Table 6: Performing clustering twice on LSMDC.

Experiments

- **influence of cluster number *K* and segment *S***



(a) T2V R@1 on LSMDC **(b) Memory cost on RTX 3090**



(c) T2V R@1 on ActivityNet **(d) Inference time**

More ablations can be found in the paper.

Experiments

- **visualization**

The clustering algorithm reserves the most representative tokens.

SIGIR 2022
Madrid

CIRCULO DE BELLAS ARTES

The End ! Thank You !

paper

code