DisCo: Towards Distinct and Coherent Visual Encapsulation in Video MLLMs

Supplementary Material

A. Details of Training

In Tab. S1 and Tab. S2, we list the hyper-parameters we adopt for the training of DisCo. In ***Stage 1***, for the ST-LLM [4] based DisCo, since ST-LLM did not adopt a pre-training stage, we set the stage 1 hyper-parameters according to their instruction tuning stage. Specifically, following common MLLM pre-training approaches, we adopt larger batch size and larger learning rates. For InternVideo2 [5] based DisCo, we follow the hyper-parameter setting of their video-text pretraining stage. In ***Stage 2***, we use diverse video conversation data for instruction tuning. For this stage, we follow the hyper-parameter settings of the instruction tuning stage in ST-LLM and InternVideo2, accordingly.

Table S1. Hyperparameter settings for the training of DisCo based on ST-LLM [4] framework.

ST-LLM		
Hyperparameters	Stage 1	Stage 2
input frame	8	8
input resolution	224	224
batch size	512	128
total epochs	1	2
learning rate	1e-4	2e-5
learning rate schedule	cosine decay	

Table S2. Hyperparameter settings for the training of DisCo based on InternVideo2 [5] framework.

InternVideo2		
Hyperparameters	Stage 1	Stage 2
input frame	8	8
input resolution	224	224
batch size	1024	256
total epochs	1	1
learning rate	1e-4	2e-5
learning rate schedule	cosine decay	

B. Details of Semantic Instance Extraction

In the Visual Concept Discriminator (VCD) module, to acquire distinct semantic concepts of training videos, we adopt GPT-4 [1] to extract words or phrases that correspond to specific entities in the video caption. In Fig. S1, we show the prompts we use to guide GPT-4 to perform the extraction of semantic instances. Notably, we find that it is important to add the instruction on requiring GPT not to repeatedly draw the same instances that appear multiple times in the video caption ('Do not include repetitive objects' in Fig. S1). Examples of the extracted instances in Fig. S2. We can see that our approach comprehensively draws out major instances in the caption, without containing repetitive items.

```
# task definition
Given the following video caption, identify only the tangible objects and people that appear.
Separate each item with a semicolon. Focus only on physical items or beings, including their
descriptive details. If no tangible objects are present, respond with 'None'. Do not include
repetitive objects.

# in-context example
Example:
Caption: The video depicts an outdoor setting with a series of events where a person wearing
colorful clothing is seated, playing a set of congas, while another person, dressed in a green
top and white skirt, is standing, dancing to the beat. The background shows a tent and bicycles,
indicating a leisurely, festive atmosphere. The conga player appears focused on their
instrument, and the dancer is energetically moving to the music. There's a dynamic exchange
of musical energy between the two.
Extracted Objects: a seated person; colorful clothing; a set of congas; a standing person; green
top; white skirt; tent; bicycles; instruments

# instruction
Now, find the tangible objects and people with descriptions from the following caption.
Caption: {caption}
Extracted Objects:
```

Figure S1. The prompt we used to guide GPT-4 to perform the semantic instance extraction task.

C. More Ablations

Methods on Semantic Instance Extraction. To verify the necessity of extracting non-overlapping instances in the semantic extraction process, we compare our 'unoverlapped' extraction method with the simple approach of extracting all appeared instances ('overlapped'), even if there are repetitive items. From Tab. S3, we can see that although using our 'unoverlapped' method results in a slight decrease in the average number of instances per video (9.96 v.s. 11.03), our method consistently achieves better performance on all three benchmarks. These results validate the superiority of our semantic instance extraction method, while further consolidating the importance of relieving semantic redundancy in the learning process of visual tokens.

Results on Varied Caption Quality. In the VCD module, DisCo utilizes textual instances extracted from video captions. To explore the influence of caption quality (*e.g.*, length, detailedness) on the final results, we utilize two sets of captions: (1) ShareGPT4o [7] which features high-quality dense captions, and (2) WebVid2M [2] which features short, brief captions. As shown in Tab. S4, the two

Table S3. Ablations on different methods of extracting semantic instances. EgoSchema is validated on *subset*.

Method	Avg. Inst	MVBench	STAR	EgoSchema
Overlapped	11.03	67.8	76.0	71.6
Unoverlapped	9.96	**68.2**	**77.7**	**72.2**

Table S4. Ablations on caption quality. We compare the results of adopting two set of captions: WebVid2M with short, sketchy captions and ShareGPT4o with long, detailed captions. 'Avg words' and 'Avg inst.' indicates the average number of words and extracted instances in each caption, respectively.

Method	Avg words	Avg inst.	MVBench	STAR
InternVideo2-HD	-	-	66.3	75.7
InternVideo2-HD+WebVid2M	14.2	3.23	67.8	76.7
InternVideo2-HD+ShareGPT4o	109.3	9.96	**68.2**	**77.7**

caption sources vary a lot in caption length and number of entities. ShareGPT4o captions contain an average of 9.96 instances per sample, while WebVid2M captions could only yield 3.23 instances per sample. Nevertheless, we observe that using both captions could result in a notable performance gain, with 1.9% and 1.5% improvement on MVBench, respectively. This highlighting DisCo's adaptability to different caption types. As the instance number in WebVid2M data is significantly less than ShareGPT4o data, for the training of WebVid2M captions, we decrease the number of tokens used in VCD module from 64 to 24, and decrease the number of token groups from 16 to 6, to reduce the proportion of unmatched visual tokens.

Ablations on Weights of Different Loss Functions. Moreover, in Eq.7, the weights of each loss component are crucial hyperparameters that can largely affect the capability of the trained model. Therefore, in order to decide the best combinations of each hyperparameter, we carry out an ablation in Tab. S5. Experimental results show that the model achieves an overall best performance when setting all weights $\lambda_{vsc}, \lambda_{vsm}, \lambda_{fsc}$ to 1.0.

Comparison with Other Token Compressing Methods. In the area of MLLMs, there have been a series of token compression methods aiming at effectively representing visual features using fewer tokens, which share similarities with DisCo. In Tab. S6, we compare two related works, TokenPacker [3] and DeCo [6], with DisCo. As shown in Tab. S6, by using significantly fewer visual tokens (64 against 400/256), DisCo achieves comparable performance with TokenPacker and DeCo. At the same time, the training and inference time of DisCo largely outcompetes the other two methods, demonstrating the superiority of our visual encapsulation approach.

Table S5. Ablations on the weights different components in the total training loss of DisCo. λ_{vsc}, λ_{vsm}, λ_{ffa} indicates weights for the losses in Eq.7.

λ_{vsc}	λ_{vsm}	λ_{ffa}	MVBench	STAR	EgoSchema
0.5	1.0	1.0	66.9	75.5	70.4
2.0	1.0	1.0	68.0	76.4	71.1
1.0	0.5	1.0	68.1	76.7	71.3
1.0	2.0	1.0	67.4	76.4	69.8
1.0	1.0	0.5	67.8	**78.0**	70.5
1.0	1.0	2.0	66.5	75.4	69.7
1.0	**1.0**	**1.0**	**68.2**	77.7	**72.2**

Table S6. Comparisons between DisCo and two other visual token compression methods in MLLMs, TokenPacker and DeCo. We compare the number of visual tokens, training time per step, inference time per instance, and the accuracy on MVBench.

Model	DeCo	TokenPacker	**DisCo**
Token No.	400	256	**64**
Train time(s/step)	6.9	6.4	**4.6**
Inference time(s/it)	1.52	1.33	**1.11**
MVBench Acc.	68.1	67.6	**68.2**

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv:2303.08774*, 2023. 1

[2] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *ICCV*, 2021. 1

[3] Wentong Li, Yuqian Yuan, Jian Liu, Dongqi Tang, Song Wang, Jie Qin, Jianke Zhu, and Lei Zhang. Tokenpacker: Efficient visual projector for multimodal llm. *arXiv:2407.02392*, 2024. 2

[4] Ruyang Liu, Chen Li, Haoran Tang, Yixiao Ge, Ying Shan, and Ge Li. St-llm: Large language models are effective temporal learners. In *ECCV*, 2024. 1

[5] Yi Wang, Kunchang Li, Xinhao Li, Jiashuo Yu, Yinan He, Guo Chen, Baoqi Pei, Rongkun Zheng, Jilan Xu, Zun Wang, et al. Internvideo2: Scaling video foundation models for multimodal video understanding. In *ECCV*, 2024. 1

[6] Linli Yao, Lei Li, Shuhuai Ren, Lean Wang, Yuanxin Liu, Xu Sun, and Lu Hou. Deco: Decoupling token compression from semantic abstraction in multimodal large language models. *arXiv:2405.20985*, 2024. 2

[7] Ruohong Zhang, Liangke Gui, Zhiqing Sun, Yihao Feng, Keyang Xu, Yuanhan Zhang, Di Fu, Chunyuan Li, Alexander Hauptmann, Yonatan Bisk, et al. Direct preference optimization of video large multimodal models from language model reward. *arXiv:2404.01258*, 2024. 1

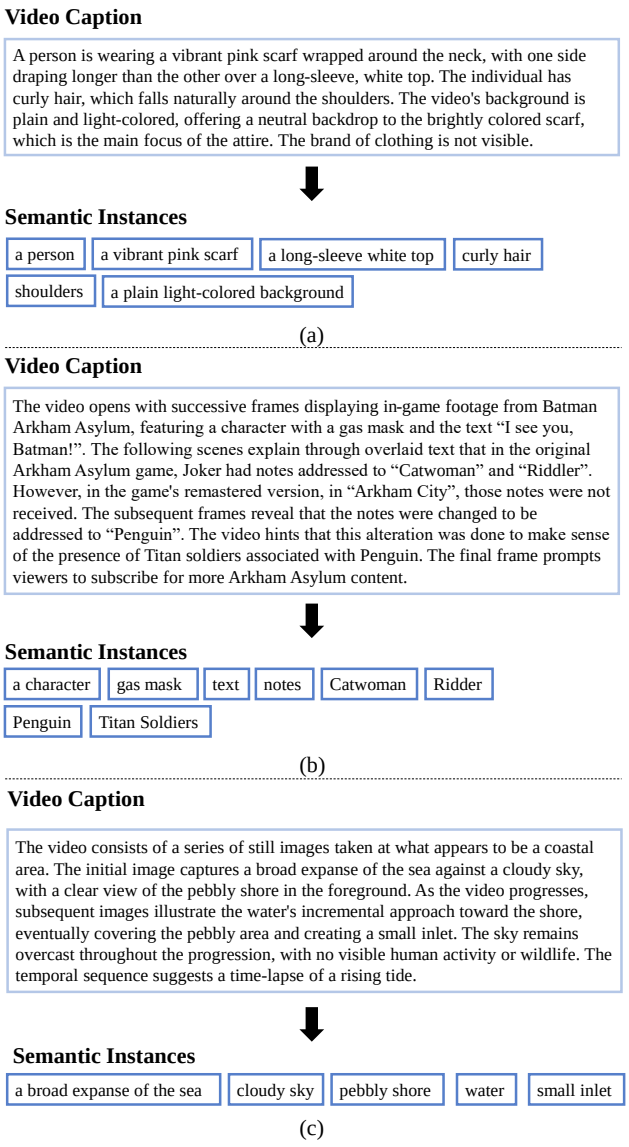

Figure S2. Examples of the semantic instance extraction process. Through our carefully designed prompts, the extracted instances do not undergo redundancy, while fully cover the major entities in the video caption.