Accurate and Fast Compressed Video Captioning

Yaojie Shen^{1,2*}, Xin Gu^{1,2*}, Kai Xu^{3}, Heng Fan^{4}, Longyin Wen^{3}, Libo Zhang^{1,2†}
^{1} Institute of Software, Chinese Academy of Sciences, Beijing, China
^{2} University of Chinese Academy of Sciences, Beijing, China
^{3} ByteDance Inc., San Jose, USA
^{4} Department of Computer Science and Engineering, University of North Texas, Denton TX, USA

Abstract

Existing video captioning approaches typically require to first sample video frames from a decoded video and then conduct a subsequent process (e.g., feature extraction and/or captioning model learning). In this pipeline, manual frame sampling may ignore key information in videos and thus degrade performance. Additionally, redundant information in the sampled frames may result in low efficiency in the inference of video captioning. Addressing this, we study video captioning from a different perspective in compressed domain, which brings multi-fold advantages over the existing pipeline: 1) Compared to raw images from the decoded video, the compressed video, consisting of I-frames, motion vectors and residuals, is highly distinguishable, which allows us to leverage the entire video for learning without manual sampling through a specialized model design; 2) The captioning model is more efficient in inference as smaller and less redundant information is processed. We propose a simple yet effective end-to-end transformer in the compressed domain for video captioning that enables learning from the compressed video for captioning. We show that even with a simple design, our method can achieve state-of-the-art performance on different benchmarks while running almost 2× faster than existing approaches. Code is available at `https://github.com/acherstyx/CoCap`.

1. Introduction

Video captioning is a representative example of applying deep learning to the fields of computer vision and natural language processing with a long list of applications, such as blind navigation, video event commentary, and human-computer interaction. To generate captions for a video, the model needs to not only identify objects and actions in the video, but also be able to express them accurately in natural language. Despite significant progress, accurate and fast video captioning remains a challenge.

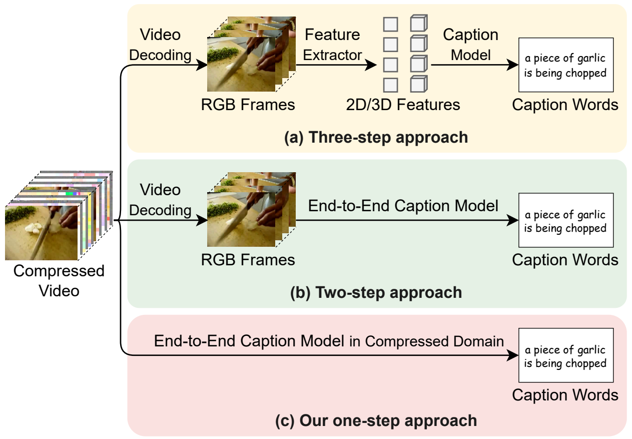

Figure 1. Comparing our method with prior methods for video captioning. Prior works are all based on decoding video frames. The difference between them is that some methods use offline extracted multiple features as input and generate captions, while others directly take dense video frames as input. By avoiding heavy redundant information and offline multiple feature extraction, our method speedup the caption generation process while maintaining high quality results.

Video captioning requires both 2D appearance information, which reflects the objects in the video, and 3D action information, which reflects the actions. The interaction between these two types of information is crucial for accurately captioning the actions of objects in the video. Most of the existing methods [36, 38, 22] are shown in Fig. 1 (the upper branch), mainly including the three-steps: (1) Decoding the video and densely sampling frames. (2) Extracting the 2D/3D features of the video frames offline. (3) Training the model based on these 2D/3D features. In these methods, densely sampled video frames result in significant redundancy, which in turn increases the computation and inference time of the model. This is because the model needs to extract features from each video frame and use all

*The two authors make equal contributions and are co-first authors.
†Corresponding author: Libo Zhang (libo@iscas.ac.cn).

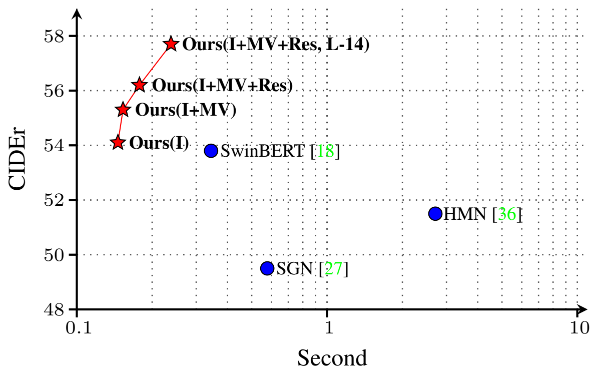

Figure 2. Comparison of model inference speed and CIDEr score on MSRVTT dataset. I, MV and Res refer to I-frame, motion vector and residual respectively. The test is run on 1 Card V100 machine with batch size set to 1.

of these features as input. Furthermore, extracting 2D appearance features, 3D action features, and region features for each video frame requires additional time. To address the speed issue and improve inference speed, some recent works [18, 29] have adopted an end-to-end approach that avoids extracting multiple visual features offline. As shown in Fig. 1 (The middle branch), the flow of their method is as follows: (1) Decoding the video and densely sample frames. (2) Take video frames directly as input and then end-to-end training model. These approaches involve a trainable visual feature extractor, rather than relying on multiple offline 2D/3D feature extractors. For example, SwinBERT [18] uses VidSwin [19] as the trainable feature extractor, while MV-GPT [29] uses ViViT [1]. While these two-steps methods address the time consumption associated with offline feature extraction, they do not alleviate the computational burden and time required to handle the redundancy of information.

To address the above problems, we propose an end-to-end video captioning method based on compressed video. Our work significantly simplifies the video caption pipeline by eliminating time-consuming video decoding and feature extraction steps. As in Fig. 1 (the lower branch), unlike previous methods, we take compressed video information as input and directly output a natural language description of the video. Compressed video is mainly composed of I-frame, motion vector and residual, and there is no redundant information between them, and they are all refined information. Therefore, the model needs less computation to process compressed domain information, and model inference is faster. At the same time, the end-to-end network structure in our proposed method can also avoid the time consumption caused by extracting multiple features. Besides, Our model is better at understanding the content of videos by utilizing the refined information in compressed domain, including the 2D feature from I-frame and the 3D action feature extracted from motion vector and residual.

As shown in Fig. 2, compared with other two-steps and three-steps methods, such as SwinBERT [18], HMN [36] and SGN [27], our method is not only faster, but also has competitive performance. Our model comprises two parts, as depicted in Fig. 4. One part consists of three encoders that extract features and an action encoder that fuses them, while the other part comprises a multimodal decoder that generates video captions. Specifically, we first extract the context feature, motion vector feature and residual feature of the compressed video through I-frame Encoder, Motion Encoder, and Residual Encoder, respectively. The context feature contains information about objects in the video, but action information is missing. In order to extract the action feature of the video, we fuse the motion vector feature, residual feature, and context feature through the action encoder. Then use the context feature and action feature as visual input of the multimodal decoder to generate video captions.

The contributions of this paper are summarized below:

1. We propose a simple and effective transformer that can take compressed video as input and directly generate a video description.

2. Our experimental results demonstrate that our method is nearly 2× further than the fastest existing state-of-the-art method in inference time, while maintaining competitive results on three challenging video captioning datasets, e.g., MSVD, MSRVTT and VATEX.

2. Related Work

Compressed vision task. The main idea of introducing compressed video into current computer vision tasks is to utilizing the motion vector and residual on the compressed domain to avoid fully decode all frames from the video and save the storage space at the same time. Early work mainly base on MPEG-4 video codec [33, 16, 12, 4]. CoViAR [33] proposed a back-tracking technique to trace motion vectors back to I-frame, which works on MPEG-4. MM-ViT [4] proposed a multi-modal transformer to process the I-frame, motion vector, residual and audio in the compressed video. Since the MPEG-4 codec is outdated, other works, e.g., MVCGC [13] and ATTP [14] , is designed to work on other coedcs like H.264 and H.265 to ensure generalizability. Comparing with MPEG-4, H.264 and H.265 allow a more flexible yet complicated compression, which makes it more challenging to learn from compressed domain. MVCGC [13] proposed a self-supervised method to learn video representations by utilizing the mutual information between RGB video frames and motion vectors. ATTP [14] designed a lightweight deep neural network to process the compressed video and achieve real time action recognition on embedded AI devices. Similarly, our work is conducted on H.264 video codec, which is currently one of the most popular video codecs.

Video captioning. Video captioning aims to convert the content of videos into natural language descriptions, which requires the model to understand the objects in the video and the behavior of the objects. Some works focus on the design of the model structure. These methods usually extract features offline, and then models use these features to generate captions by designing different network architectures. HMN [36] proposed a hierarchical modular network that serves as a strong video encoder, which bridges videos and languages. ORG-TRL [38] proposes an object relational graph based encoder, which captures more detailed interaction features to enrich visual representation. SGN [27] designed a semantic grouping network to group video frames with discriminating word phrases of partially decoded caption. Some works explore additional information to help the model generate more accurate video captions. TextKG [9] propose a two-stream network capable of knowledge-assisted video description using knowledge graphs. Univl [20] learns powerful vision-and-language representations by pre-training the models on large-scale datasets, *e.g.*, HowTo100M [21] and WebVid-2M [2]. Some other works focus more on end-to-end video captioning generation. SwinBERT [18] proposed an end-to-end transformer-based model, which takes video frame patches directly as inputs and then uses VidSwin to extract visual features. MV-GPT [29] designed an encoder-decoder model end-to-end to generate the video caption from video frames and transcribed speech directly. We propose an end-to-end video captioning model based on the compressed domain without decoding video frames and extracting features offline, which not only accelerates the generation of captions, but also performs favorably against the state-of-the-art methods.

3. Methods

As mentioned above, our method aims to take the dense information (including I-frame, motion vector and residual) in compressed domain as input to accelerate inference and improve performance for video caption. To this end, we design an end-to-end transformer-based network as shown in Fig. 4. In this section, we first detail the information in the compressed video in Sec. 3.1, then introduce the model network in Sec. 3.2 and 3.3, and finally introduce the training strategy of the model in Sec. 3.4.

3.1. The Structure of Compressed Video

Modern video codecs utilizing the temporal redundancy of successive video frames to compress raw video. As shown in Fig. 3, most modern codecs (*e.g.*, H.264, and H.265) divide video frames into three different types according to their dependencies with other frames: I-frame

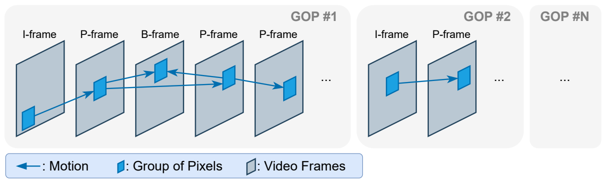

Figure 3. The GOP structure in compressed video. In each GOP, the first frame must be an I-frame, followed by several B/P-frames.

(intra coded frame), P-frame (predictive coded frame) and B-frame (bipredictive coded frame). I-frame is fully encoded independently using intra-prediction without relying on other frames. Other frames like B-frame and P-frame are encoded by referring to the other frames using inter-prediction, which is stored in the form of motion vector. Motion vector describes the movement of a group of pixels from source (reference frames) to destination (current B-frame or P-frame), which contains highly compressed motion information of successive video frames. The difference between P-frame and B-frame is that B-frame could refer to the frames before or after it, while P-frame only refer to the frames before it. Since predicting a frame using neighboring frames could be inaccurate, an additional residual error between the current frame and the prediction is calculated. We denote $\mathcal{I}_I$, $\mathcal{I}_P$ and $\mathcal{I}_B$ as decoded I-frame, P-frame, and B-frame, and $\mathcal{I}_{mv}$ and Δ_{res} as the motion vector and residual of P-/B-frame respectively. In compressed domain, the P-frame and B-frame could be reconstructed by

$$\mathcal{I}_{B/P} = \mathrm{Pred}(\mathcal{I}_{mv}, \mathcal{I}_{ref}) + \Delta_{res} \tag{1}$$

where $\mathcal{I}_{ref}$ is the referenced frame, and Pred is the prediction method to reconstruct current frame based on motion vector and referenced frame. Since the reconstruction process is time consuming, our model takes highly compressed information from compressed domain directly as input to achieve end-to-end video captioning.

Moreover, successive frames are divided into several groups, which is called Groups of Pictures (GOP). GOP is an independent encoding or decoding unit, which means that the frames in a GOP do not refer to any frames on other GOP. Each GOP starts with an I frame, followed by several P-frames or B-frames. For each GOP, we take one I-frame and M B-/P-frames as inputs. The B-/P-frames are uniformly sampled from each GOP, and we only use their motion vector and residual as replacements. Therefore, the visual inputs of our model would be

$$\begin{aligned} X = [&\mathcal{I}_I^{(1)}, \mathcal{I}_{mv}^{(1,1)}, \Delta_{res}^{(1,1)}, \ldots, \mathcal{I}_{mv}^{(M,1)}, \Delta_{res}^{(M,1)}], \\ &\ldots, [\mathcal{I}_I^{(N)}, \mathcal{I}_{mv}^{(1,N)}, \Delta_{res}^{(1,N)}, \ldots, \mathcal{I}_{mv}^{(M,N)}, \Delta_{res}^{(M,N)}] \end{aligned}$$

where N is the number of GOP sampled from each video and M is the total number of P-/B-frames sampled from each GOP. We set N according to the average GOP number, and M is equal to the maximum number of P-/B-frames in each GOP, which is a hyper-parameter during encoding.

3.2. Model Architecture for Compressed Domain

Based on the GOP structure mentioned above, we proposed a transformer based structure to utilizing the dense information from the compressed domain. Fig. 4 (left) shows the main framework of our proposed compressed video transformer. The model takes all information of the compressed video as inputs, including I-frame, motion vector and residual, while maintaining a fast inference speed. Specifically, we use three different Vision Transformers [8] (ViT) as encoder to extract the visual features for I-frame, motion vector and residual. We adopt a pretrained Vision Transformer as the encoder to extract the context feature from the I-frame:

$$\mathcal{F}_{\text{ctx}}^{(n)} = \text{Encoder}_{\text{I}}(\mathcal{I}_I^{(n)}).$$

For each B-frame or P-frame, we get a motion vector and a residual from the compressed domain. We use two lightweight Vision Transformers as encoders to extract features from motion vectors and residuals. The motion and residual features is added together to generate the B-/P-frame features $\mathcal{F}_{\text{BP}}^{(m,n)}$:

$$\mathcal{F}_{\text{BP}}^{(m,n)} = \text{Encoder}_{\text{mv}}(\mathcal{I}_{mv}^{(m,n)}) + \text{Encoder}_{\text{res}}(\Delta_{res}^{(m,n)}).$$

In this way, for each GOP we obtain M B-/P-frame features

$$\mathcal{F}_{\text{BP}}^{(n)} = [\mathcal{F}_{\text{BP}}^{(1,n)}, \dots, \mathcal{F}_{\text{BP}}^{(M,n)}].$$

As motion vector and residual lack fine-grained context information, we use features from motion vector and residual as queries to retrieve the rich context information in RGB frames instead of simply fusing them. We employ action encoder to integrate the object information of I-frame into the action information of motion vector and residual, which takes B-/P-frame features in current GOP $\mathcal{F}_{\text{BP}}^{(n)}$ and the context feature $\mathcal{F}_{\text{ctx}}^{(n)}$ as input to generate the action feature $\mathcal{F}_{\text{act}}^{(n)}$ of current GOP. The action encoder is constructed by N_a sets of alternately stacked self-attention and cross-attention blocks.

Specifically, the workflow of the action encoder is as follows. Firstly, according to the reconstruction process described in Eq. 1, we utilize the self-attention module fuse the temporal representation of successive frames to obtain $\mathcal{F}_{\text{att}}^{(n)}$:

$$\begin{aligned}
&X = \mathcal{F}_{\text{BP}}^{(n)} + \text{Emb}_{\text{p}} + \text{Emb}_{\text{t}},\\
&Q = W_q * X, K = W_k * X, V = W_v * X,\\
&\mathcal{F}_{\text{att}}^{(n)} = \text{SelfAttention}(Q, K, V),
\end{aligned}$$

where Emb_{p} is the positional embeddings, Emb_{t} is the type embeddings, and W_q, W_k, W_v are learnable matrices. The type embeddings are added to distinguish B-frames and P-frames. And then we use the cross-attention to integrate the $\mathcal{F}_{\text{ctx}}^{(n)}$ from I-frame into the $\mathcal{F}_{\text{att}}^{(n)}$ from the motion vector and residual. Finally, the action feature $\mathcal{F}_{\text{act}}^{(n)}$

$$\begin{aligned}
&Q' = W_q' * \mathcal{F}_{\text{att}}^{(n)}, K' = W_k' * \mathcal{F}_{\text{ctx}}^{(n)}, V' = W_v' * \mathcal{F}_{\text{ctx}}^{(n)},\\
&\mathcal{F}_{\text{att}}^{(n)'} = \text{CrossAttention}(Q', K', V'),\\
&\mathcal{F}_{\text{act}}^{(n)} = \text{Mean}(\mathcal{F}_{\text{att}}^{(n)'}),
\end{aligned}$$

where W_q', W_k', W_v' are learnable matrices and $\text{Mean}()$ is a function that calculates the average feature.

3.3. Multimodal Decoder for Video Captioning

The context features $\mathcal{F}_{\text{ctx}}^{(n)}$ and action features $\mathcal{F}_{\text{act}}^{(n)}$ for each GOP are contacted to form the visual representation:

$$\mathcal{V} = [\mathcal{F}_{\text{ctx}}^{(1)}, \mathcal{F}_{\text{act}}^{(1)}, \dots, \mathcal{F}_{\text{ctx}}^{(N)}, \mathcal{F}_{\text{act}}^{(N)}].$$

Then we design a multimodal decoder to predict the video captions based on the visual representation $\mathcal{V}$. The multimodal decoder is composed of N_m masked self-attention modules stacked as shown in Fig. 4 (right) and the workflow is as follows:

$$\begin{aligned}
&\mathcal{T}_{<\text{t}} = \text{Embedding}(Y_{<\text{t}}),\\
&\mathcal{X} = \text{Concat}(\mathcal{V}, \mathcal{T}_{<\text{t}}),\\
&\mathcal{X}' = \mathcal{X} + \text{Emb}_{\text{p}}' + \text{Emb}_{\text{t}}',\\
&Q'' = W_q'' * \mathcal{X}', K'' = W_k'' * \mathcal{X}', V'' = W_v'' * \mathcal{X}',\\
&h_{\text{t}} = \text{MaskedSelfAttention}(Q'', K'', V''),\\
&p(y_{\text{t}}|\mathcal{V}, \mathcal{T}_{<\text{t}}) = \text{softmax}(\text{Linear}(h_{\text{t}})),
\end{aligned}$$

where $Y_{<\text{t}}$ is the words generated in previous $t-1$ steps, $\text{Embedding}()$ is a function that converts one-hot word vectors into word embeddings, Emb_{p}' is the positional embeddings, Emb_{t}' is used to distinguish different modality of inputs, W_q'', W_k'', W_v'' are learnable matrices and y_{t} is the prediction of current step. In the multimodal decoder, position embedding and type embedding is added to distinguish the order and type of features respectively.

3.4. Optimization

We train our model using the cross-entropy loss function. Given the ground-truth indices of previous (t-1) words and the visual representation $\mathcal{V}$, we can get the predictions of the current t-th word y_t^*. After that, the training loss is computed as

$$L = -\textstyle\sum_{t=1}^{l} \log p(y_t^*|y_{:t-1}^*, \mathcal{V}),$$

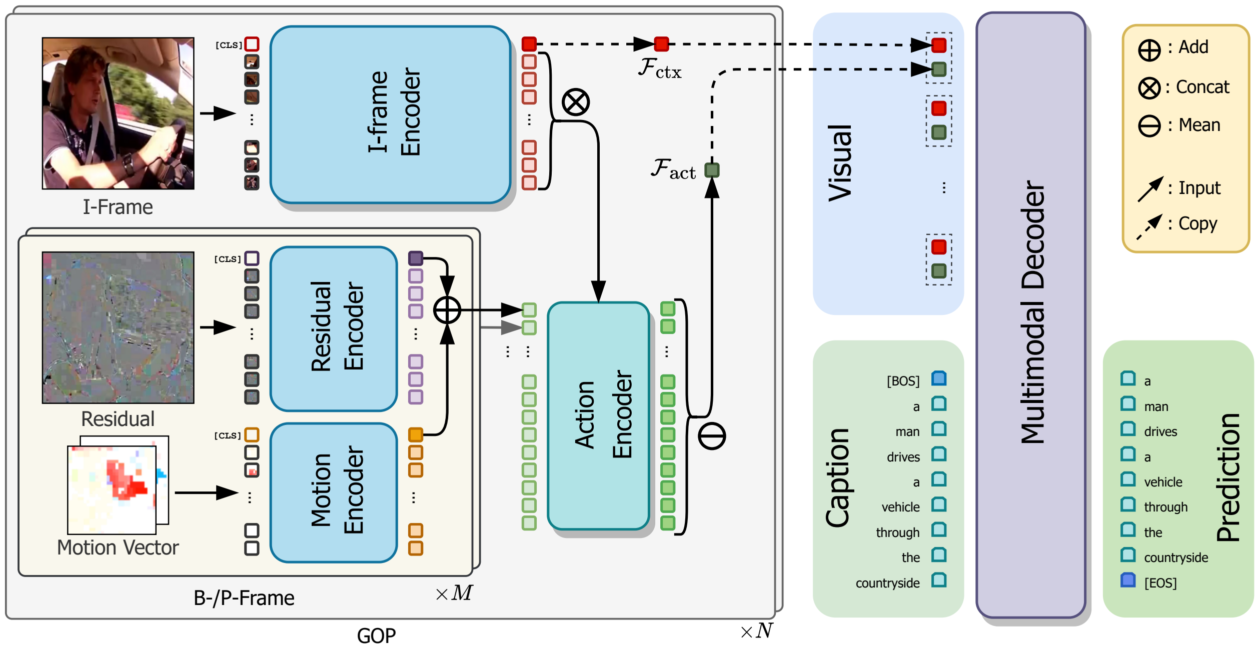

Figure 4. The architecture of our proposed Compressed Video Captioner. *Left*: The Compressed Video Transformer which extract video representation for each GOP. A large visual backbone is used to extract visual representations from I-frame, and two small Vision Transformer is used to extract residual and motion representations from compressed domain. After that, an action encoder is used to fuse the features. *Right*: The Multimodal Decoder. We use a multimodal decoder with causal mask to learn caption.

where $y^*_{1:T}$ is the ground truth sequence and l is the total length of predicted captions. Notably, we add the label smoothing to mitigate overconfidence in implementation.

4. Experiments

4.1. Datasets

MSRVTT [34] is a generic video captioning dataset that comprises $10,000$ video clips, with each clip annotated with 20 captions. On average, each video clip lasts about 15 seconds. The standard split involves the use of $6,513$ clips for training, 497 clips for validation, and $2,990$ clips for testing.

MSVD [3] contains $1,970$ videos, with each video clip having 40 captions. The average duration of each video clip is around 10 seconds. We adopt the standard split, which involves using $1,200$ videos for training, 100 videos for validation, and 670 videos for testing.

VATEX [32] is a large-scale dataset which contains about $41,250$ video clips. The duration of each video clip is between 10 seconds, and 10 English captions are manually annotated per clip. We use the official training set for training and evaluate the results using the public test set.

4.2. Evaluation Metrics

To evaluate the effectiveness of our approach, we use the standard metrics for video captioning: BLEU@4 (B4) [23], METEOR (M) [7], ROUGE (R) [17], and CIDEr (C) [31]. Each metric provides a unique perspective on the quality of the generated captions. BLEU@4 evaluates sentence fluency, METEOR assesses semantic accuracy, ROUGE measures word order, and CIDEr evaluates the degree to which the caption conveys key information. By considering these different metrics, we can comprehensively evaluate the performance of our model.

4.3. Implementation Details

Our model is implemented using PyTorch, and to read motion vectors and residuals from the compressed video, we utilize the x264 library in FFmpeg. Before training and testing, the videos are resized to 240 on its smallest edge and compressed using the H.264 codec with KeyInt set to 60. For each video, we fixedly sampled 8 GOPs, each of which contains 1 I-frame, 59 motion vectors, and 59 residuals. The size of the I-frame and residual is $3 * 224 * 224$, and the size of the motion vector is $4 * 56 * 56$. We use Adam with initial learning rate of 1e-4, β_1=0.9, $\beta 2$=0.999 and the warmup strategy is adopted in the training. The maximum length of the caption sentence is set to 22, which contains two special tokens, *e.g.*, [CLS] token and [EOS] token. The feature dimension in each block is set to 768, and the number of heads in multi-head architecture is set to 12 for all layers. The batch size is set to 64 and the training epochs to 20. The I-frame encoder has 12 layers and is

Method	Decoding	E2E	Features			MSVD				MSRVTT			
			2D Appearance	3D Action	Object Detection	B4	M	R	C	B4	M	R	C
SAAT [39]	✓	-	IncepResnetV2	C3D	-	46.5	33.5	69.4	81.0	39.9	27.7	61.2	51
STG-KD [22]	✓	-	ResNet101	I3D	FasterRCNN	52.2	36.9	73.9	93.0	40.5	28.3	60.9	47.1
PMI-CAP [5]	✓	-	IncepResnetV2	C3D	-	54.6	36.4	-	95.1	42.1	28.7	-	49.4
ORG-TRL [38]	✓	-	IncepResnetV2	C3D	FasterRCNN	54.3	36.4	73.9	95.2	43.6	28.8	62.1	50.9
OpenBook [37]	✓	-	IncepResnetV2	C3D	-	-	-	-	-	42.8	29.3	61.7	52.9
SGN [27]	✓	-	ResNet101	C3D	-	52.8	35.5	72.9	94.3	40.8	28.3	60.8	49.5
MGRMP [6]	✓	-	IncepResnetV2	C3D	-	55.8	36.9	74.5	98.5	41.7	28.9	62.1	51.4
HMN [36]	✓	-	IncepResnetV2	C3D	FasterRCNN	59.2	37.7	75.1	104	43.5	29	62.7	51.5
UniVL [20]	✓	-		S3D		-	-	-	-	42.2	28.8	61.2	49.9
SwinBERT [18]	✓	✓		VidSwin		58.2	41.3	77.5	120.6	41.9	29.9	62.1	53.8
MV-GPT [29]	✓	✓		ViViT		-	-	-	-	48.9	38.7	64	60
Ours	-	✓		CLIP		55.9	39.9	76.8	113.0	43.1	29.8	62.7	56.2
Ours(ViT/L14)	-	✓		CLIP		**60.1**	**41.4**	**78.2**	**121.5**	**44.4**	**30.3**	**63.4**	**57.2**

Table 1. Comparison with state-of-the-art methods on the test split of MSVD and MSRVTT. Decoding means decoding video frames, and E2E means end-to-end training without offline feature extraction. For a fair comparison, we gray out models that pre-train on large-scale datasets.

	B4	M	R	C
NITS-VC [30]	20.0	18.0	42.0	24.0
VATEX [32]	28.4	21.7	47	45.1
ORG-TRL [38]	32.1	22.2	48.9	49.7
Support-set [24]	32.8	24.4	49.1	51.2
SwinBERT [18]	**38.7**	**26.2**	**53.2**	**73**
VideoCoCa [35]	39.7	-	54.5	77.8
Ours	31.4	23.2	49.4	52.7
Ours(ViT/L14)	35.8	25.3	52.0	64.8

Table 2. Comparison with state-of-the-art methods on the test split of VATEX. For a fair comparison, we gray out models that pre-train on large-scale datasets.

initialized with pre-trained weights from the CLIP [25] visual encoder, while the other encoders and the multimodal decoder are randomly initialized. The layers for the motion encoder, residual encoder and action encoder are 2, 2 and 1, respectively. Lastly, we set the hyperparameters M, N, N_a, and N_m to 60, 8, 2 and 2.

4.4. Performance Comparison with SOTA Methods

In order to verify the effectiveness of the method, we evaluated the proposed model against state-of-the-art methods on three public benchmark datasets.

MSVD dataset. The evaluation results on the MSVD dataset are reported in Table 1 (left). We conducted experiments using two sizes of the I-frame encoder, namely $B/16$ and $L/14$, with the results reported in the article based on $B/16$, unless otherwise stated. Our method using the $L/14$ I-frame encoder achieves the best performance on all metrics, with only SwinBERT [18] performing better than our method using $B/16$. Our approach stands out by being able to directly utilize compressed domain information and extract visual features in real-time. The result shows that our model can efficiently extract information from the refined compressed domain information.

MSRVTT dataset. In the MSRVTT benchmark, our method outperforms other approaches in all metrics, as shown in Table 1 (right). Specifically, both the based on $B/16$ model and based on $L/14$ model achieve higher scores compared to other methods. In particular, our method achieves a CIDEr score of 56.2 / 57.2, which represents a significant improvement of $+2.4$ / $+3.4$. This result demonstrates that our approach can generate captions with higher semantic accuracy than other methods based on video decoding [31]. CIDEr is particularly effective at capturing human consensus, which makes our achievement in this metric even more impressive.

VATEX dataset. Our method is evaluated on a large-scale dataset, as shown in Table 2. We achieve the second-best results on all metrics, falling behind SwinBERT [18]. Our approach involves extracting visual features using three Vision Transformer encoders, while the I-frame encoder is initialized with the pre-trained CLIP [25] model on LAION-400M [28]. In contrast, SwinBERT uses the VidSwin backbone [19], which is pre-trained on the Kinetic-600 dataset [15]. It is worth noting that LAION-400M is a large image-text dataset, while Kinetics-600 is a video-text dataset, and VATEX dataset is a subset of Kinetics-600 videos. SwinBERT outperforms our method on VATEX due to its backbone pre-trained on Kinetics-600.

Method	Data Type	Inference Time ↓			CIDEr ↑
		Feature Extraction	Model Time	Total	
SGN	RGB Video Frames	303 ms	275 ms	578 ms	49.5
HMN	RGB Video Frames	2,710 ms	108 ms	2,818 ms	51.5
SwinBERT	RGB Video Frames	339 ms		339 ms	53.8
Ours	I-frame	146 ms		**146 ms**	54.1
Ours	I-frame+MV	153 ms		153 ms	55.3
Ours	I-frame+MV+Res	178 ms		178 ms	**56.2**

Table 3. A detailed comparison of speed with other methods on the test split of the MSRVTT dataset. During the test, the model is running on a NVIDIA Tesla V100 GPU and the batch size is set to 1. The time cost is computed on the overall MSRVTT test split.

4.5. Speed Comparison with the SOTA Methods

To evaluate the speed of our method, we compared it to three representative methods, namely SGN [27], HMN [36], and SwinBERT [18], as reported in Table 3. SGN is a three-step method that first decodes video frames and densely sample, then extracts the 2D appearance and 3D action features based on ResNet101 [11] and C3D [10] (consuming 303 ms) offline, and finally uses the visual features as the input of the model (consuming 275 ms). Therefore, the total time for SGN to generate a video caption is 578 ms. HMN achieves the best results among the three-steps models, but it is relatively slow as it requires offline region feature extraction based on Faster RCNN [26] (consuming $2,520$ ms), leading to its total time of $2,818$ ms. SwinBERT, on the other hand, is an end-to-end method that does not extract multiple features offline, using only 339 ms.

Compared to these methods, our proposed method does not require a dense sampling of video frames or the extraction of multiple features offline. As shown in Table 3, our baseline method only considers the I-frame of the entire video, achieving a CIDEr score of 54.1 and a total time of 146 ms. By integrating the motion vector, we improved the CIDEr to 55.3, demonstrating that the action information in the motion vector helps the model generate captions. Furthermore, by incorporating residual information, the CIDEr score is further improved by 0.9 to reach 56.2. Although considering three inputs increases our total inference time, our speed is still nearly 2 times faster than SwinBERT, 3 times faster than SGN, and 15 times faster than HMN.

4.6. Ablation Study

Impact of input information. To evaluate the effectiveness of different input information in our method, we conducted several experiments on the MSRVTT dataset, as shown in Table 4. To investigate the role of I-frame, motion vector, and residual, we first experimented with using only one of them. As shown in Table 4, using only I-frame, motion vector, or residual achieved CIDEr scores of 54.1, 19.4, and 13.0, respectively. This indicates that the model can directly use I-frame instead of motion vector and residual. By jointly using I-frame and motion vector and fusing their information through the action encoder, we achieved a CIDEr score of 55.3. Similarly, using I-frame and residual achieved a score of 54.9. This demonstrates that motion vector and residual can help the model generate more accurate captions. The performance of the model is further improved by inputting all three types of information, achieving a CIDEr score of 56.2, an improvement of 1.7. Removing the action encoder from the proposed method resulted in a slight drop in CIDEr scores, from 56.2 to 54.3. This demonstrates that the action encoder can help the model integrate the object information of I-frame into the action information of motion vector and residual.

Input			Module	B4	M	R	C
$\mathcal{I}_I$	$\mathcal{I}_{mv}$	Δ_{res}	En_A				
✓	-	-	-	41.6	29.7	62.3	54.1
-	✓	-	-	27.3	21.6	52.6	19.4
-	-	✓	-	23.9	20.5	51.0	13.0
✓	✓	-	✓	43.4	29.9	62.6	55.3
✓	-	✓	✓	42.2	30.0	62.5	54.9
✓	✓	✓	-	42.1	**30.1**	62.4	54.3
✓	✓	✓	✓	**43.1**	29.8	**62.7**	**56.2**

Table 4. Ablation study of different input on the test subset of MSRVTT. The $\mathcal{I}_I$, $\mathcal{I}_{mv}$ and Δ_{res} mean decoded I-frame, motion vector and residual respectively. And the En_A means the action encoder.

KeyInt (M)	GOP Nums (N)	Inference Time	B4	M	R	C
250	2	153 ms	39.6	28.7	60.8	49.5
60	2	**131 ms**	41.6	29.3	61.7	52.4
60	4	139 ms	42.8	**29.9**	62.6	55.3
60	8	178 ms	**43.1**	29.8	**62.7**	**56.2**
60	10	187 ms	42.7	29.8	62.6	55.5

Table 5. Ablation study of GOP numbers on MSRVTT test subset.

Impact of GOP numbers. GOP is a fundamental unit in compressed video that affect the compression rate. A larger GOP size results in fewer GOP numbers and commonly higher compression rates. In video codec (*e.g.* FFmpeg), the GOP size is determined by the KeyInt parameter. To

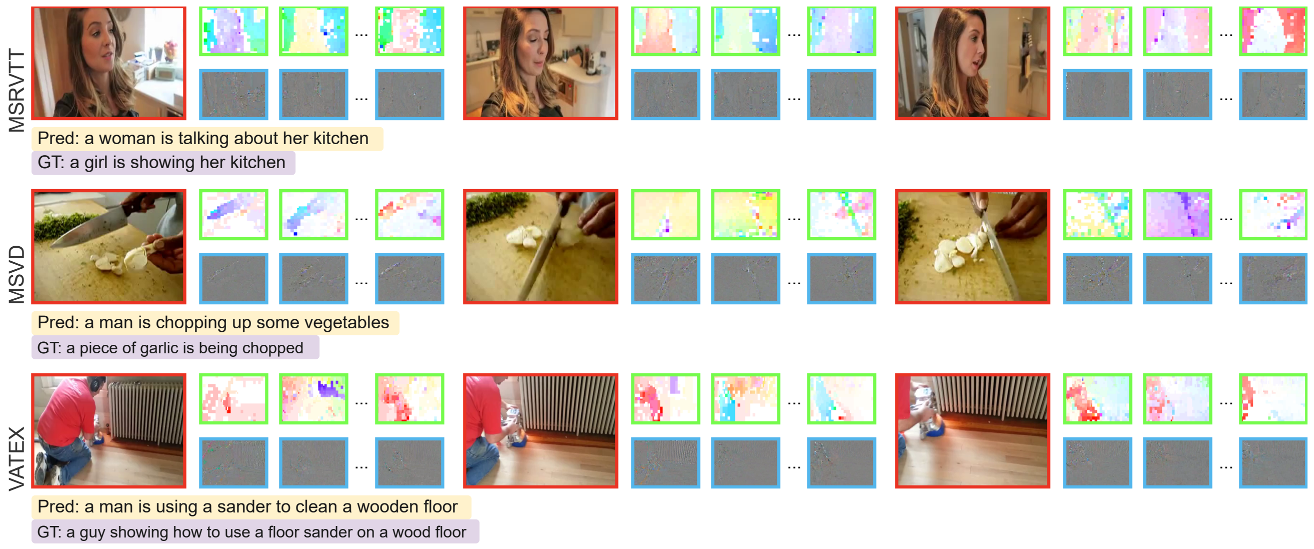
MSRVTT
Pred: a woman is talking about her kitchen
GT: a girl is showing her kitchen
MSVD
Pred: a man is chopping up some vegetables
GT: a piece of garlic is being chopped
VATEX
Pred: a man is using a sander to clean a wooden floor
GT: a guy showing how to use a floor sander on a wood floor

Figure 5. Qualitative results on the MSRVTT, MSVD and VATEX dataset. We show the input of our model, which is in compressed domain. The red, green and blue borders indicate I-frame, motion vector and residual, respectively.

En_I	En_M	En_R	En_C	De_M	CIDEr
12	2	2	1	2	56.2
24	2	2	1	2	57.2
12	4	4	2	2	55.2
12	2	2	1	4	54.9
12	4	4	2	4	55.4

Table 6. Ablation study about module layers on the MSRVTT test subset. En_I, En_M, En_R, En_A and De_M refer to the I-frame encoder, motion encoder, residual encoder, action encoder and multimodal decoder of the model respectively.

investigate the impact of GOP size on our video caption model, we experimented with different GOP numbers and KeyInts, as shown in Table 5. Comparing KeyInt values of 250 and 60, we observed that a smaller GOP size led to better model performance (49.5 CIDEr vs 52.4 CIDEr). By sampling different GOP numbers under the same KeyInt, the best performance is achieved by setting GOP size to 8 and KeyInt to 60. While the performance is improved with more GOPs, yet speed is decreased due to increased computation as more information is included.

Impact of model layers. To investigate the impact of different model layers on our proposed method, we conducted an ablation study on the MSRVTT test subset, as shown in Table 6. Giving that I-frame contains more complex information, we design a deep encoder with more layers for I-frame, while using a shallow encoder for motion vector and residual. Our results show that the performance of the model improves with an increase in the number of layers in the I-frame encoder (56.2 CIDEr to 57.2 CIDEr). However, adding more layers to other modules did not result in further improvements in model performance.

4.7. Qualitative Results

As shown in Fig. 5, we present the qualitative results of our proposed method on three datasets (*e.g.*, MSVD, MSRVTT, and VATEX). Specifically, we visualize the input I-frame, motion vector, and residual and compare the predicted description to the ground truth. Our method consistently produces semantically consistent descriptions that closely align with the ground truth across all three datasets. Furthermore, the results demonstrate a superior ability to capture motion behavior in the videos.

5. Conclusion

In this paper, we introduce an end-to-end transformer-based model for video captioning that takes compressed video as input to eliminate redundant information. Our proposed method is evaluated on three challenging datasets and demonstrates that our proposed method is not only fast, but also competitive in performance with SOTA. In the future, we plan to further improve our method in two ways: (1) Add additional modalities such as audio, text, and knowledge graphs to enhance the quality of the generated captions. (2) Pre-train the model on a large-scale dataset to further boost the overall performance in compressed domain.

Acknowledgement

Libo Zhang was supported by Youth Innovation Promotion Association, CAS (2020111). Heng Fan and his employer received no financial support for research, authorship, and/or publication of this article. This work was done during internship at ByteDance Inc.

References

[1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lucic, and Cordelia Schmid. Vivit: A video vision transformer. In *IEEE/CVF International Conference on Computer Vision*, pages 6816–6826, 2021. 2

[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 *IEEE/CVF International Conference on Computer Vision*, pages 1708–1718, 2021. 3

[3] David L. Chen and William B. Dolan. Collecting highly parallel data for paraphrase evaluation. In *Annual Meeting of the Association for Computational Linguistics*, 2011. 5

[4] Jiawei Chen and Chiu Man Ho. Mm-vit: Multi-modal video transformer for compressed video action recognition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1910–1921, 2022. 2

[5] Shaoxiang Chen, Wenhao Jiang, Wei Liu, and Yu-Gang Jiang. Learning modality interaction for temporal sentence localization and event captioning in videos. In *European Conference on Computer Vision*, pages 333–351, 2020. 6

[6] Shaoxiang Chen and Yu-Gang Jiang. Motion guided region message passing for video captioning. In *IEEE/CVF International Conference on Computer Vision*, pages 1523–1532, 2021. 6

[7] Michael J. Denkowski and Alon Lavie. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, 2014. 5

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 4

[9] Xin Gu, Guang Chen, Yufei Wang, Libo Zhang, Tiejian Luo, and Longyin Wen. Text with knowledge graph augmented transformer for video captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18941–18951, 2023. 3

[10] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6546–6555, 2018. 7

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 7

[12] Lianghua Huang, Yu Liu, Bin Wang, Pan Pan, Yinghui Xu, and Rong Jin. Self-supervised video representation learning by context and motion decoupling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13886–13895, 2021. 2

[13] Yuqi Huo, Mingyu Ding, Haoyu Lu, Nanyi Fei, Zhiwu Lu, Ji-Rong Wen, and Ping Luo. Compressed video contrastive learning. *Advances in Neural Information Processing Systems*, 34:14176–14187, 2021. 2

[14] Yuqi Huo, Xiaoli Xu, Yao Lu, Yulei Niu, Mingyu Ding, Zhiwu Lu, Tao Xiang, and Ji-rong Wen. Lightweight action recognition in compressed videos. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 337–352. Springer, 2020. 2

[15] Ang Li, Meghana Thotakuri, David A Ross, João Carreira, Alexander Vostrikov, and Andrew Zisserman. The ava-kinetics localized human actions video dataset. *arXiv preprint arXiv:2005.00214*, 2020. 6

[16] Jiapeng Li, Ping Wei, Yongchi Zhang, and Nanning Zheng. A slow-i-fast-p architecture for compressed video action recognition. In *ACM International Conference on Multimedia*, pages 2039–2047, 2020. 2

[17] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Annual Meeting of the Association for Computational Linguistics*, pages 74–81, 2004. 5

[18] Kevin Lin, Linjie Li, Chung-Ching Lin, Faisal Ahmed, Zhe Gan, Zicheng Liu, Yumao Lu, and Lijuan Wang. Swinbert: End-to-end transformers with sparse attention for video captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17949–17958, 2022. 2, 3, 6, 7

[19] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3192–3201, 2022. 2, 6

[20] Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Jason Li, Taroon Bharti, and Ming Zhou. Univl: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*, 2020. 3, 6

[21] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. HowTo100M: Learning a text-video embedding by watching hundred million narrated video clips. In *IEEE/CVF International Conference on Computer Vision*, pages 2630–2640, 2019. 3

[22] Boxiao Pan, Haoye Cai, De-An Huang, Kuan-Hui Lee, Adrien Gaidon, Ehsan Adeli, and Juan Carlos Niebles. Spatio-temporal graph for video captioning with knowledge distillation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10867–10876, 2020. 1, 6

[23] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Annual Meeting of the Association for Computational Linguistics*, pages 311–318, 2002. 5

[24] Mandela Patrick, Po-Yao Huang, Yuki Asano, Florian Metze, Alexander Hauptmann, Joao Henriques, and Andrea Vedaldi. Support-set bottlenecks for video-text representation learning. *arXiv preprint arXiv:2010.02824*, 2020. 6

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763, 2021. 6

[26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region

proposal networks. *Advances in Neural Information Processing Systems*, 28, 2015. 7

[27] Hobin Ryu, Sunghun Kang, Haeyong Kang, and Chang D. Yoo. Semantic grouping network for video captioning. In *Association for the Advancement of Artificial Intelligence*, pages 2514–2522, 2021. 2, 3, 6, 7

[28] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021. 6

[29] Paul Hongsuck Seo, Arsha Nagrani, Anurag Arnab, and Cordelia Schmid. End-to-end generative pretraining for multimodal video captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17959–17968, 2022. 2, 3, 6

[30] Alok Singh, Thoudam Doren Singh, and Sivaji Bandyopadhyay. NITS-VC system for vatex video captioning challenge 2020. *arXiv preprint arXiv:2006.04058*, 2020. 6

[31] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4566–4575, 2015. 5, 6

[32] Xin Wang, Jiawei Wu, Junkun Chen, Lei Li, Yuan-Fang Wang, and William Yang Wang. Vatex: A large-scale, high-quality multilingual dataset for video-and-language research. In *IEEE/CVF International Conference on Computer Vision*, 2019. 5, 6

[33] Chao-Yuan Wu, Manzil Zaheer, Hexiang Hu, R Manmatha, Alexander J Smola, and Philipp Krähenbühl. Compressed video action recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6026–6035, 2018. 2

[34] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. MSR-VTT: A large video description dataset for bridging video and language. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5288–5296, 2016. 5

[35] Shen Yan, Tao Zhu, Zirui Wang, Yuan Cao, Mi Zhang, Soham Ghosh, Yonghui Wu, and Jiahui Yu. Video-text modeling with zero-shot transfer from contrastive captioners. *arXiv preprint arXiv:2212.04979*, 2022. 6

[36] Hanhua Ye, Guorong Li, Yuankai Qi, Shuhui Wang, Qingming Huang, and Ming-Hsuan Yang. Hierarchical modular network for video captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17939–17948, 2022. 1, 2, 3, 6, 7

[37] Ziqi Zhang, Zhongang Qi, Chunfeng Yuan, Ying Shan, Bing Li, Ying Deng, and Weiming Hu. Open-book video captioning with retrieve-copy-generate network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9837–9846, 2021. 6

[38] Ziqi Zhang, Yaya Shi, Chunfeng Yuan, Bing Li, Peijin Wang, Weiming Hu, and Zheng-Jun Zha. Object relational graph with teacher-recommended learning for video captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13275–13285, 2020. 1, 3, 6

[39] Qi Zheng, Chaoyue Wang, and Dacheng Tao. Syntax-aware action targeting for video captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13093–13102, 2020. 6