Characterizing Video Question Answering with Sparsified Inputs

Shiyuan Huang[1], **Robinson Piramuthu**[2], **Vicente Ordonez**[2,3], **Shih-Fu Chang**[1], **Gunnar A Sigurdsson**[2]

[1]Columbia University, [2]Amazon Alexa AI, [3]Rice University

Abstract

In Video Question Answering, videos are often processed as a full-length frame sequence to ensure minimal information loss. Recent works have shown evidence that sparse video inputs are sufficient to maintain high performance. However, they usually discuss single frame selection. In our work, we extend the setting to various input lengths and other modalities, and characterize the task with different input sparsities and provide a tool for doing that. Specifically, we propose a Multi-Gumbel-based sparsification module to adaptively find the best video inputs for the final task. We experiment over public VideoQA benchmarks and analyze how sparsified inputs affect the performance. We have observed only $5.2\%-5.8\%$ loss of performance with only 10% of video lengths. Meanwhile, we also observed the complimentary behaviour between visual and textual inputs, even under highly sparsified settings, and that by adding just 100 words, we can even beat a state-of-the-art. Our work suggest the potential of improving data efficiency for video-and-language tasks.

Introduction

Watching long videos is time-consuming and easily loses user attention. How to efficiently present videos to users is an important and practical problem in various video applications. For example, for home surveillance videos which are usually recorded continuously throughout the day, it is hard for users to capture a moment of package delivery from an hour-long video. More generally speaking, for videos that are not carefully edited (e.g., Youtube videos), they often contain purposeless parts and need pre-processing of content so that users can quickly get meaningful information.

Videos often come from different modalities. Commonly, they are composed of image frame sequences. With the advances of recording devices and editing tools, videos often contain speech (e.g., Youtube videos recorded from user phones) and subtitles (e.g., in movies and TV-shows). It has been shown that leveraging different modalities benefits various video tasks (Yang et al. 2021; Li et al. 2020). However, it is worthwhile noticing that the various modalities in video could be quite noisy and redundant — meaningless utterances, repeating frames, etc. — causing computational inefficiency and distracting model learning. Furthermore, the problem of modality imbalance (Goyal et al. 2017) has been studied, where the unbalanced information across modalities could result in significant bias towards one modality. For example, prior works (Lei et al. 2020a) have shown that in TV-related videos, the major contribution for the various video-language tasks comes from subtitles while the video frames play a negligible role.

In this work, we characterize the VideoQA problem from the perspective of input sparsity. As illustrated in Figure 1, we aim to answer the question: "How much visual/textual signals are sufficient for a task?" For VideoQA specifically,

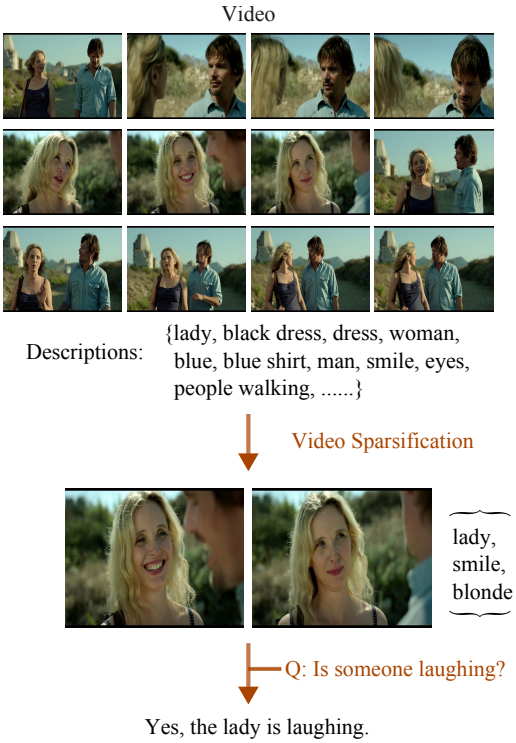

Figure 1: We study the Sparsified VideoQA problem, where we learn to sparsify the original long video into very few inputs for QA. We design a video sparsification process to deal with video of multiple modalities (frames, word and phrase descriptions, etc).

different questions require different amount of video information to give the answer. For example, if the question asks about people, then theoretically the system only needs to look at the moments where people are present. In the literature, there is evidence showing that video action classification can be accomplished with single frame (Wu et al. 2019; Huang et al. 2018). Recently there have also been works that imply sparse uniform sampling of the video is sufficient for video and language tasks (Lei et al. 2021), and an analysis tool which shows that video and language tasks could be achieved by picking one optimal frame (Buch et al. 2022). In this work, we instead move beyond single frame input, and try to characterize the role of videos by learning to select an optimal set of video inputs. We propose a generic framework which learns to drop video inputs while training for the video-and-language task. This framework can be applied to different kind of video modalities, and in our experiments we provide analysis on visual-only (i.e., video frames), text-only (i.e., video subtitles or key words), and visual-textual inputs.

From our experiments, we demonstrate that with very sparse inputs, the task can still be accomplished pretty well. Specifically, we are able to achieve $5.2\%-5.8\%$ loss of accuracy with only 10% length of the video, which corresponds to only $2-4$ selected frames. Meanwhile, we also observe complimentary behaviour between modalities even with sparsified multi-modal inputs. Our work suggests the potential of improving data efficiency under either single or multi-modal settings for video-and-language tasks.

Approach

In this section, we introduce our token sparsification approach. We first provide a brief preliminary on multi-modal transformers. Then we explain how the learnable token sparsification works. Finally, we explain how we can extend it to multi-modal setting.

Preliminaries — Multi-modal Transformers

To solve for video tasks that involve multi-modal inputs (e.g., VideoQA), current literature usually converts both visual and textual signals into sequence of embeddings, projects them into a common embedding space and applies a multi-modal transformer which takes the sequence concatenation to generate the final output. i.e.,

$$y = \mathcal{M}([v_1, \ldots, v_n; w_1, \ldots, w_m]), \tag{1}$$

where $\mathcal{M}$ is the multi-modal transformer, y is the task output (e.g., predicted answer in text), $\{v_i\}_{i=1}^n$ is the sequence of video frame/segment features, and $\{w_j\}_{j=1}^m$ is the sequence of word embeddings for the text inputs.

To get $\{v_i\}$, the common practice is to span the entire video, and use an off-the-shelf feature extractor to get frame or segment level features (depending on whether the feature extractor is image-based or segment-based). In this way, all the video information is included for the task. However, videos contain a lot of redundant information which is not necessarily useful for the task. For example, if the task is asking about "What the person is wearing", intuitively it only needs one video frame that contains the target person. From this perspective, we aim to design a framework that learns to select only the key information for the task.

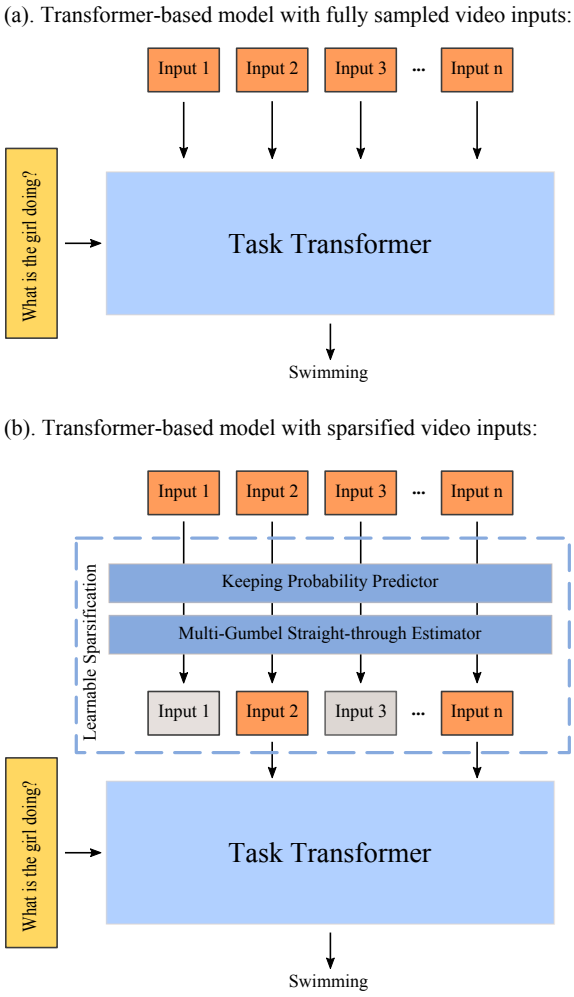

Figure 2: The pipeline of our method. We insert a learnable selection module between the inputs and the task transformer to sparsify the inputs. Our method can be generally applied to single or multi- modal inputs from videos. The entire network is trained end-to-end using the task objective as well as the input sparsity constraints.

Token Sparsification

Observing the architecture of transformers which treat visual and/or textual embeddings as sequence tokens, we propose to learn to sparsify the input tokens during training. Starting with single modality inputs $\{v_i\}_{i=1}^n$, we first generate the keeping probability s_i for each token v_i to estimate how likely v_i should be kept. To ensure s_i is a valid probability value within the range of $[0, 1]$, we place a predictor $f(\cdot) : \mathcal{R}^d \rightarrow \mathcal{R}^2$ to map each token v_i to 2-dim, and apply a Softmax normalization to re-scale the 2-dim vector into $[0, 1]$, and take the first entry as the keeping probability s_i.

Specifically,

$$s_i = < \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \text{Softmax}(f(v_i)) >,$$

where $< \cdot >$ is the inner product function. At inference time, we can treat sparsification as a ranking procedure, whereas we rank tokens according to their keeping probabilities and select top-K tokens.

During training, to facilitate the learnability of the network, we treat sparsification as a top-K sampling procedure. That is, we draw K samples from $\{v_i\}$ from their keeping probabilities as the sparsified inputs to the rest of the model. By default, the sampling process is not differentiable. In order to overcome this non-differentiability, we refer to and extend the Gumbel-Softmax (Jang, Gu, and Poole 2016) trick.

Gumbel-Softmax Straight-through Estimator (Jang, Gu, and Poole 2016) offers a differentiable way for *single* discrete sampling. It first re-parameterizes the sampling distribution by adding Gumbel noise followed by a Softmax normalization, i.e.,

$$s_i^g = \frac{exp((\log s_i + g_i)/\tau)}{\sum_{j=1}^{n} exp((\log s_j + g_j)/\tau)},$$

Then it follows the design of straight-through estimator (Toderici et al. 2015) to get the sample index from $k = argmax_i\{s_i^g\}_{i=1}^n$ and select the corresponding v_k in the forward pass while zeroing-out the rest. In the backward propagation, the gradients of s_i^g are kept for use. We encourage referring to (Jang, Gu, and Poole 2016) for more details and proofs.

Multi-Gumbel Straight-through Estimator. The standard Gumbel-Softmax STE trick only supports single sampling, as it performs along the sequence and only selects one most likely token. In our context, would like to extend it to multi-sampling of most likely tokens. (Kool, Van Hoof, and Welling 2019) introduces a trick to perform sampling without replacement. In our implementation and experiment, we instead modified (Kool, Van Hoof, and Welling 2019) to two variants:

1. **Gumbel-TopK Selection**: select the top-K tokens with higher s_i^g values. And then use straight-through estimator after selection, i.e., we only keep the selected tokens and zero-out the rest, but in the backward pass, we still use the gradients of all s_i^g.
2. **Ratio-controlled Gumbel**: instead of hard selection of K samples, we allow sampling with arbitrary number of tokens while keeping a sparsity ratio constraint in the loss during training. Specifically, we add Gumbel disturbance to s_i:

   $$s_i^g = \frac{exp((\log s_i + g_{i_0})/\tau)}{exp((\log s_i + g_{i_0})/\tau) + exp((\log(1 - s_i) + g_{i_1})/\tau)}$$

   where $g_{i_0}, g_{i_1} \sim Gumbel(0, 1)$ are Gumbel noises. We then keep all the tokens whose perturbed keeping probability $s_i^g > 0.5$. For a human-selected target keeping ratio p, we add a loss constraint on the overall keeping ratio over the batch:

   $$\mathcal{L}_{select} = \frac{1}{B}\sum_{b=1}^{B}(p - \frac{1}{n}\sum_{i=1}^{n}\mathbb{1}[v_i^b \text{ is kept}])^2,$$

   where B is the batch size, and v_i^b is denoted as the tokens within a batch.

The overall training loss generically is the weighted balance between the task loss and the selection loss:

$$\mathcal{L} = \mathcal{L}_{task} + \lambda\mathcal{L}_{select}, \tag{2}$$

where λ is the balancing weight between two loss components, and $\mathcal{L}_{task}$ is the task loss. Note that $\mathcal{L}_{select}$ is not necessary required for the first variant and we can simply set $\lambda = 0$ in that case. The method is generic and task-agnostic. For VideoQA, we refer to (Li et al. 2020) for $\mathcal{L}_{task}$, which essentially is a cross entropy loss between the predicted answer and the ground-truth answer word. During inference, we directly rank $\{s_0\}$ and select the inputs associated with the top-K scores. In this way, we are able to get the sparsified video inputs for the target task.

Sparsified Positional Encoding

Positional encoding is a typical mechanism designed for transformers to encode the order of tokens within a sequence. Typical designs for positional encoding are either fixed, such as sinusoidal function with selected frequencies (Vaswani et al. 2017), or learnable positional embeddings like (Dosovitskiy et al. 2020). In our context, we make use of learnable positional embeddings following (Li et al. 2020; Yang et al. 2021). For a full-length sequence, each token v_i is added to a unique learnable embedding vector p_i that sticks to this position i. After sparsification, tokens are dropped and kept by probability, so the remaining tokens could be at various positions.

In our preliminary experiments, we found that inappropriate positional embedding could harm training convergence, especially when we just leave the positional embeddings as they are in the full sequence scenarios. As a result, in our implementation, we arrange the sparsified tokens $\{v_{k_i}\}_i^K$ into a new sequence $\{v'_i\}_i^K$ (where $v'_i = v_{k_i}$), and assign p_i to v'_i accordingly. We observe a good convergence behaviour and performance from this simple rearrangement.

Multi-modal Token Sparsification

We generalize the standard single modality token sparsification to multi-modal scenarios where videos come with both visual and textual signals. Specifically, our method allows the inputs being any kind of tokenized inputs. For videos that come with multi-channel inputs, e.g., subtitles, we can directly concatenate time-stamped subtitles along with the frames as multi-modal inputs. Then we can perform similar token sparsification on both of them. Specifically, we denote the multi-modal inputs as $\{v_i, w_i\}_{i=1}^n$ (note that we can assume same number of v_i's and w_i's by interpolating or padding, etc). We first get the unified representation from multi-modal inputs by

$$u_1, \ldots, u_n = \mathcal{C}_m([v_1, \ldots v_n; w_1, \ldots, w_n]) \tag{3}$$

where $\mathcal{C}_m$ is a context model (e.g., cross-modal transformer) that exploits both visual v_i and subtitle w_i information. Then we apply the uni-modal sparsification technique introduced

Table 1: Effect of the temperature τ. Smaller τ leads to more exploitation while higher leads to more exploration. We observe that more explorative selection is beneficial for denser inputs.

	VLEP			VIOLIN		
Input Percentage	$\tau = 0.01$	$\tau = 0.1$	$\tau = 0.5$	$\tau = 0.01$	$\tau = 0.1$	$\tau = 0.5$
10%	**60.25**	56.01	58.94	56.25	**60.57**	58.80
30%	60.95	**63.52**	59.13	61.72	57.64	**62.34**
50%	63.05	63.64	**64.30**	65.57	64.48	**66.06**
70%	63.73	65.14	**65.32**	65.94	66.52	**67.06**

Table 2: Effect of the balancing weight λ. λ balances the selection loss and task loss as specificed in eq. 2. We report results on VLEP dataset and observe that $\lambda = 1.0$ yields the best balance. Higher λ might lead to distraction of task, while lower λ might lead to insufficient sparsification. We pick $\lambda = 1.0$ based on the following ablation.

Input Percentage	$\lambda = 0.01$	$\lambda = 0.1$	$\lambda = 1.0$	$\lambda = 10.0$
10%	59.12	59.85	60.25	59.90
70%	65.23	65.32	65.32	65.11

in Sec. on top of $\{u_i\}$ sequence to compute the keeping scores, and sparsify the original multi-modal input sequence $\{v_i, w_i\}$ according to the score, to get the sparsified multi-modal sequence $\{v_{k_i}, w_{k_i}\}_{i=1}^{K}$. Then this sparsified sequence is fed to the task performer $\mathcal{M}$ to compute the outputs (eq. 1).

We also consider another approach where we ease the restriction on time-matched input pairs and operate on each modality regardless of their timestamps. In this setting, we apply different context model on each modality sequence, and then use separate selection loss to constrain the sparsity. We will provide more details on this setting in the Experiment details, where we demonstrate our idea with key-word selection that summarizes the videos.

Experiments

In this section, we provide our experiment results. First we detail our implementation and VideoQA datasets; then we provide VideoQA results under different input sparsities, followed by multi-modal results. Finally, we offer some qualitative visualization to analyze our approach.

Implementation Details

To verify our idea, we experimented on two state-of-the-art video-and-language models VQA-T (Yang et al. 2021) and HERO (Li et al. 2020). HERO considers multi-channel videos where videos come with subtitles as additional channel of inputs. HERO follows a hierarchical transformer architecture to first exploit the information within video modalities and contexts, and then has another task head to operate the task. VQA-T simply consists of two DistillBERT models to deal with video+question inputs and answer candidates, and computes the answer based on embedding similarity. For extracting the video features, we follow (Yang et al. 2021) to use the S3D model pre-trained on Howto100M dataset. For extracting the key word candidates, we use the model offered by (Yang et al. 2021) and the vocabulary from the training split of the dataset to extract the words/phrases based on feature similarity.

Datasets and Metrics

We evaluate our idea on public VideoQA benchmarks including VLEP (Lei et al. 2020b), VIOLIN (Liu et al. 2020) and iVQA (Yang et al. 2021). For VLEP and VIOLIN, we follow (Li et al. 2020) to build our method on top of HERO. VLEP and VIOLIN provide both raw videos and subtitles as inputs. Our selection is then based on the multi-modal inputs. For iVQA, we follow (Yang et al. 2021) to build our method on top of VQA-T. We report VideoQA accuracies across different input sparsity level: 10%, 30%, 50%, 70% and full (100%) inputs.

VideoQA Experiments

We present our single modality sparsified VideoQA results here. First, we study the design choices of our two multi-gumbel estimator variants, followed by the comparison between our approach and other token sparsification baselines.

Effect of temperature τ. In our experiments, we found that varying τ could result in very different performance. We elaborate the result in Table 1 with our Gumbel-TopK selection variant, where we choose $\tau = (0.01, 0.1, 0.5)$ for each sparsity level, and fix $\lambda = 1.0$. A smaller τ means the model focuses more on exploitation, while a larger τ makes the model focus more on exploration. We can observe that on both datasets, the model that is more explorative with denser inputs gives a better results; but on sparser inputs, the model tends to stick to exploitation.

Effect of loss balancing weight λ. We also study how the balancing weight λ affects the performance with our ratio-controlled Gumbel estimator. In Table 2, we choose $\lambda = 0.01, 0.1, 1.0$ and 10.0, and then report the results at highly sparsified (10%) and lowerly sparsified (70%) levels on VLEP dataset. We fix $\tau = 0.01$ for 10% level and $\tau = 0.5$ for 70% level. λ has slightly larger impact on highly

Table 3: Comparison of our two Gumbel variants. Overall the first variants perform slightly better. The second variant is superior at highly sparsified level (10%) as it adds more flexibility in individual sparsity levels across different videos.

Input Percentage	10%	30%	50%	70%
Gumbel-TopK Selection	60.25	63.52	64.30	65.32
Ratio-controlled Gumbel	61.43	63.42	63.49	65.01

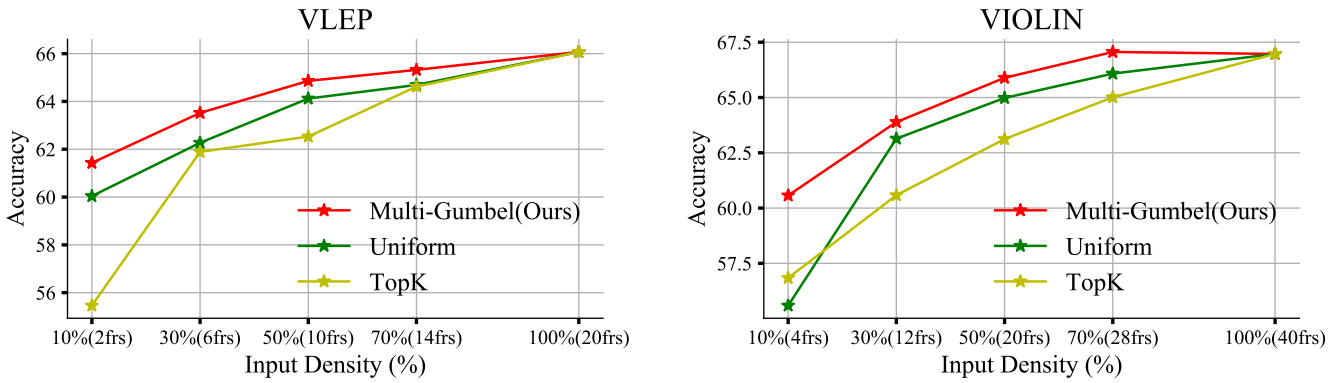

Figure 3: Sparsified VideoQA results on VLEP and VIOLIN datasets. Accuracy at the 100% level refers to the original full input baseline result. We can conclude that learnable sparsification is better than fixed sampling (Uniform), and that stochasitic sampling is better than deterministic selection (TopK). Our Multi-Gumbel estimator achieves the best result overall.

sparsified setting. We observe that $\lambda = 1.0$ yields the best balance and hence choose it for all the other performance.

Comparison of two multi-gumbel variants. We compare the two variants of Gumbel estimator for token sparsification. In Table 3, we compare these two variants at different sparsity levels on VLEP. Our Gumbel-TopK selection variant is better than ratio-controlled Gumbel overall. Ratio-controlled Gumbel is superior at highly sparsified level (10%) as it adds more flexibility in individual sparsity levels across different videos.

Comparison with other sparsification approaches. To our knowledge, no prior work has studied the same topic on VideoQA before, so there is no direct comparison. To validate our approach of Multi-Gumbel Estimator, we define the baselines on our own:

1. Uniform(Fixed): Fixed uniform sampling of inputs w.r.t. different sparsity levels.
2. TopK: During training, directly select inputs with higher keeping probability s_i after softmax step, without noise perturbing.
3. Multi-Gumbel(Ours): Our approach which stochastically sparsifies tokens with Gumbel perturbing, we plot the better result from the two variants we introduced.

We show the accuracy vs. density curve in Figure 3. We can see that our Multi-Gumbel approach module is able to achieve the best performance across different sparsity levels. Compared to learnable selection, fixed uniform sampling is weaker as it does not contain any form of task adaptive selection. A direct TopK selection training performs weaker than training with stochastic sampling, as we observe that the deterministic selection tends to a local optimal choice, while our stochastic Multi-Gumbel approach gives more flexibility of by adding noises while learning. One noticeable observation is that, at 10% level, which corresponds to very few frames (2 frames for VLEP and 4 frames for VIOLIN), the performance is still quite good. It implies the potential of accomplishing the task with very few inputs.

Multi-modal Sparsification Results on iVQA

In the multi-modal experiments, we would like to study the relation between visual and textual modalities under a controlled input setting. In order to do that, we extend our learnable selection module to the multi-modal setting following Section to generate key frames and key words from the original video inputs. We first get a pool of candidate inputs from the raw video. The candidate frames are directly sampled from the videos, while the candidate key words are extracted using CLIP-based model, which finds the closest words or phrases using nearest embedding matching. We use all the phrases and words from the iVQA training set as the vocabulary dictionary to choose words from. To better demonstrate the results, we use the format of few-word or few-frame inputs. For visual frame inputs, we process with the same method as before. For textual inputs, we treat 5-word as one unit. 5-word/sec is the average reading speed for adults, which consumes similar attention from watching a frame. So 5-word and one single frame could be thought of as equivalent in consuming user attention. We combine the units into a sequence, then apply the same selection method for word selection. For multi-modal setting, we concatenate the frames and word units as a multi-modal sequence and select from both. We fix $\tau = 0.1$ for training the models.

Table 4: VideoQA results on iVQA. We apply our approach on the state-of-the-art method (Yang et al. 2021). We consider multi-modal sparsification where we sparsify both visual (i.e., frames) and textual (i.e., words) inputs. Compared to single-modality, multi-modal performance is stronger at different sparsification levels. With additional extracted words, we also outperform the state-of-the-art result on iVQA (last column).

		Visual (Snippets)				
		0	1 snippet	2 snippets	5 snippets	20 snippets
Textual (Words)	0	14.6 (Q-only)	28.65	30.24	31.26	35.43 (Yang et al. 2021)
	5 words	17.5	28.68	30.31	31.70	35.43
	10 words	18.22	29.87	31.43	31.88	36.01
	25 words	20.14	30.16	31.59	32.03	36.09
	100 words	26.75	31.47	32.11	33.21	**36.42**

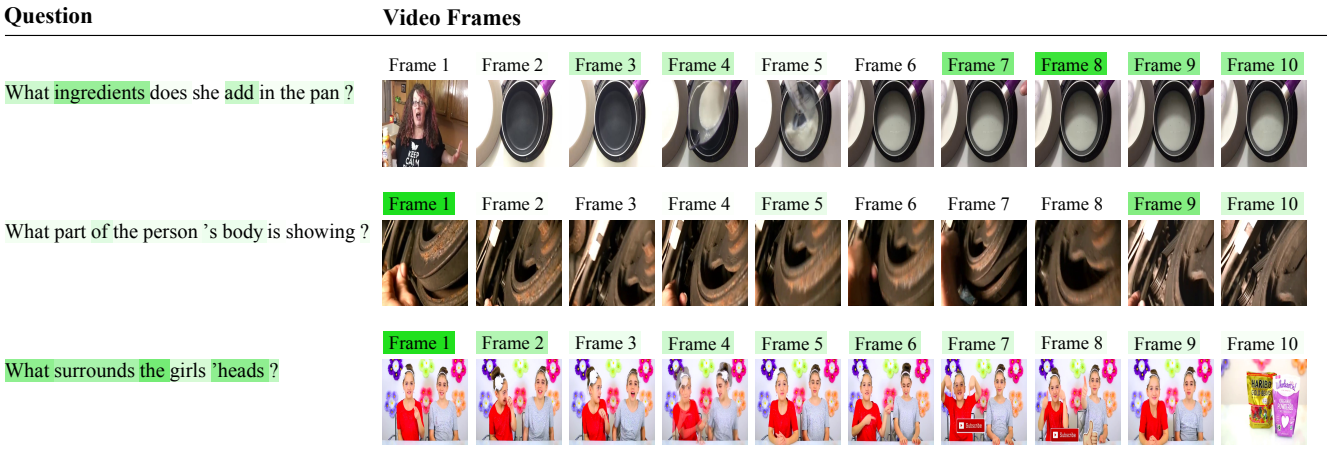

Figure 4: Frame importance visualization. Darker color means the corresponding word/frame is of more importance to predict the answer. We can see that the model is able to discard some repetitive frames or frames that are not relevant.

Our results are shown in Table 4. For single-modality inputs, we similarly observe an increasing performance trend with increasing number of inputs. Even with very few inputs, the VideoQA performance is very close to the upper bound from dense inputs. We can also observe a boost of performance from increasing density of inputs on both modalities at sparsified levels, which validates the effectiveness of our sparsification techniques. In general, the visual inputs perform stronger than textual inputs, which is mainly due to the fact that visual signals are much more informative. On the other hand, we can still observe an increase of performance from adding even very few multi-modal inputs. For example, adding only 5-word to the visual snippet could still get some performance gains. This implies the complimentary manner from different modalities from the perspective from strictly controlled inputs. Noticeably, as an intermediate output from our learnable selection, we can get a few-frame and few-word summarization of the original video, which is human-interpretable. We provide more examples and analysis in the following section to demonstrate this advantage.

Qualitative Analysis

Here we provide visualizations on the selected frame and/or key words from iVQA dataset. For illustration purposes, we present the result for single frame selection and 10-word extraction in Figure 5, along with their associated questions and the predicted answer. Answer in green color means the system correctly predicts the answer, while red color means the system predicts the wrong answer and the ground truth is in parenthesis. In the successful cases, the sparsified output is able to capture an appropriate figure for the topic, and the texts also contain words related to the answer, which leads to the correct answer. In the first failure case, even though the selected frame contains information related to the answer "shirt", the textual component is a distraction, and the system generates an answer more related to the key words which are closely describing pipes. In the second failure case, the generated key frame and words are both irrelevant to the question. This is probably because the question itself is asking something minor (since most of the video contents are about the architectures and surroundings) while the model is trained to get information that is of major interest for the overall dataset and task.

Additionally, we analyze the token importance using the tool provided by (Chefer, Gur, and Wolf 2021) which calculates the importance score of each input token w.r.t. to the task prediction. In Figure 4, we provide some visualization examples where question words and video frame inputs are

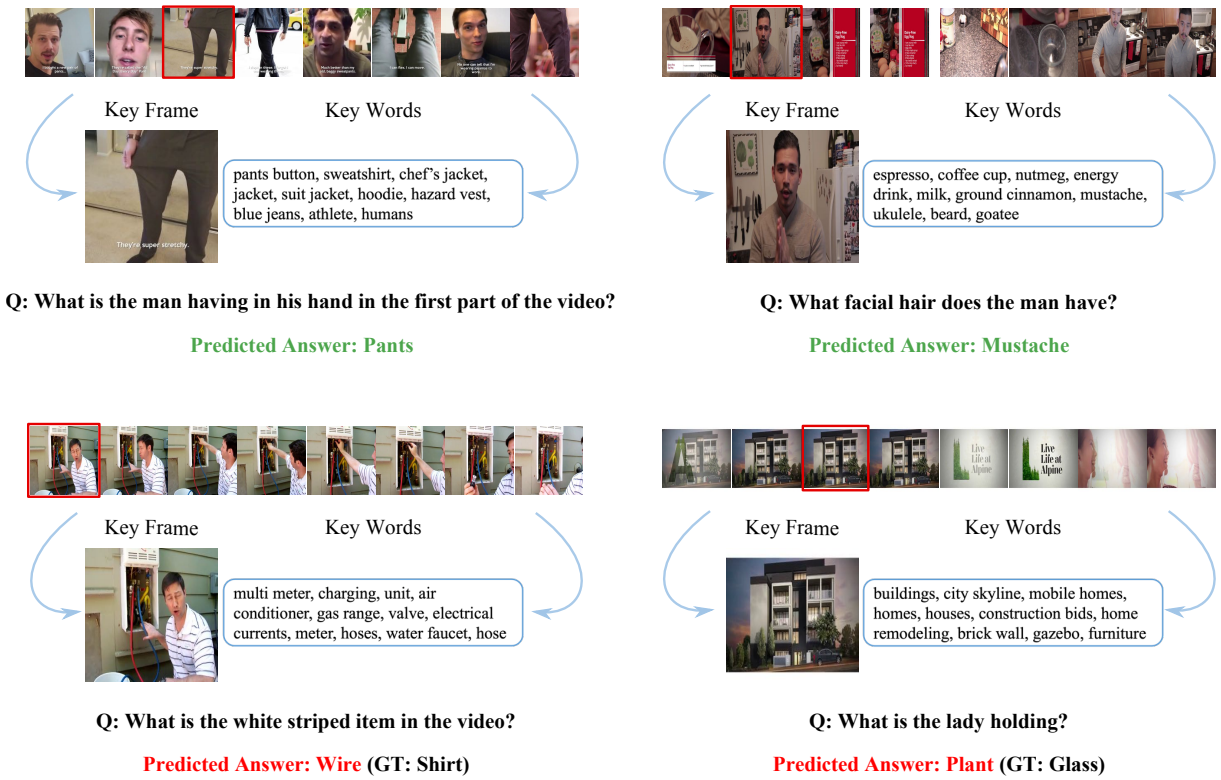

Figure 5: Qualitative examples on iVQA videos. Single frame and ten-word summary is generated from the original video for Video Question Answering task. First two examples demonstrate successful cases where both visual and textual signals signals are able to capture the question-relevant information. The last two examples show some failure cases where visual and/or textual signals are distracted from the question.

highlighted according to their importance scores. For illustration purposes, we only sample 10 frames in each sample. Words or frames that are of darker green color means they contribute more to the prediction. From the given examples, we can see that not every video frame is of significant importance. The model is able to discard frames that do not contain any useful information for the question (e.g., in the second example, only the frames showing the fingers are contributing). On the other hand, in the example where the scene is relatively stable, we can also observe that the model focuses mostly on one of these similar frames (as in the third example), while the rest seems to be diverging. These observations show the potential of dropping unnecessary video inputs to improve the efficiency, which validates our motivation.

Conclusion

In this work, we characterize VideoQA from the perspective of sparsified inputs. We propose learnable sparsification module to adaptively select the task-specific inputs, which are of get multi-lengths and can be extended to multi-modalities. We experimentally analyze our idea on current VideoQA benchmarks, where we observe a fair performance even with very few inputs, and a complimentary performance between modalities. Our work shows the potential of improving data efficiency under various video data types.

References

Buch, S.; Eyzaguirre, C.; Gaidon, A.; Wu, J.; Fei-Fei, L.; and Niebles, J. C. 2022. Revisiting the "Video" in Video-Language Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chefer, H.; Gur, S.; and Wolf, L. 2021. Transformer Interpretability Beyond Attention Visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 782–791.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Goyal, Y.; Khot, T.; Summers-Stay, D.; Batra, D.; and Parikh, D. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *CVPR*.

Huang, D.-A.; Ramanathan, V.; Mahajan, D.; Torresani, L.; Paluri, M.; Fei-Fei, L.; and Niebles, J. C. 2018. What makes a video a video: Analyzing temporal information in video understanding models and datasets. In *Proceedings of the*

IEEE Conference on Computer Vision and Pattern Recognition, 7366–7375.

Jang, E.; Gu, S.; and Poole, B. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Kool, W.; Van Hoof, H.; and Welling, M. 2019. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In *International Conference on Machine Learning*, 3499–3508. PMLR.

Lei, J.; Li, L.; Zhou, L.; Gan, Z.; Berg, T. L.; Bansal, M.; and Liu, J. 2021. Less is More: ClipBERT for Video-and-Language Learningvia Sparse Sampling. In *CVPR*.

Lei, J.; Yu, L.; Berg, T. L.; and Bansal, M. 2020a. TVQA+: Spatio-temporal grounding for video question answering. In *ACL*.

Lei, J.; Yu, L.; Berg, T. L.; and Bansal, M. 2020b. What is More Likely to Happen Next? Video-and-Language Future Event Prediction. In *EMNLP*.

Li, L.; Chen, Y.-C.; Cheng, Y.; Gan, Z.; Yu, L.; and Liu, J. 2020. HERO: Hierarchical Encoder for Video+Language Omni-representation Pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2046–2065. Online: Association for Computational Linguistics.

Liu, J.; Chen, W.; Cheng, Y.; Gan, Z.; Yu, L.; Yang, Y.; and Liu, J. 2020. VIOLIN: A Large-Scale Dataset for Video-and-Language Inference. In *CVPR*.

Toderici, G.; O'Malley, S. M.; Hwang, S. J.; Vincent, D.; Minnen, D.; Baluja, S.; Covell, M.; and Sukthankar, R. 2015. Variable rate image compression with recurrent neural networks. *arXiv preprint arXiv:1511.06085*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wu, Z.; Xiong, C.; Ma, C.-Y.; Socher, R.; and Davis, L. S. 2019. Adaframe: Adaptive frame selection for fast video recognition. In *CVPR*.

Yang, A.; Miech, A.; Sivic, J.; Laptev, I.; and Schmid, C. 2021. Just Ask: Learning To Answer Questions From Millions of Narrated Videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 1686–1697.