Panda-70M: Captioning 70M Videos with Multiple Cross-Modality Teachers

Supplementary Material

Table of Contents

A. Details of Semantics-Aware Video Splitting Algorithm

In Section 3.1, we propose a video splitting algorithm to cut a long video into several semantically coherent clips. The algorithm includes two stages, splitting and stitching, for which the details are described in Appendix A.1 and A.2.

A.1. Stage1: Splitting based on Shot Boundary Detection

We first split a long video by PySceneDetect [1]. Specifically, we use ContentDetector with cutscene_threshold of 25 and min_scene_len of 15 frames. Next, we design a two-step post-processing algorithm to handle 1) long videos with complex transitions, such as fade-in and fade-out effects, that cannot be reliably detected by PySceneDetect and 2) unedited footage that does not contain any cut-scenes but has semantic changes within the same clip.

To handle both cases, we propose creating artificial scene cuts each 5 seconds for clips without cut-scene. That is, if a video clip is longer than 5 seconds, we cut out the first 5 seconds as a new clip and recursively apply the same procedure to the remaining part. Since we are only interested in semantically consistent video clips, we extract the ImageBind [25] features of the frames near the beginning or the end. If the features of these two frames are dramatically different we remove that clip. Specifically, given a n-frame video clip C, we extract the features $f(C_A)$ and $f(C_B)$ for the number $0.1 \times n$ and $0.9 \times n$ frames, denoted as C_A and C_B. We only keep the video clips if satisfying $\|f(C_A) - f(C_B)\| \leq 1.0$. As such, we can exclude video clips with transition effects or significant semantics changes within a clip.

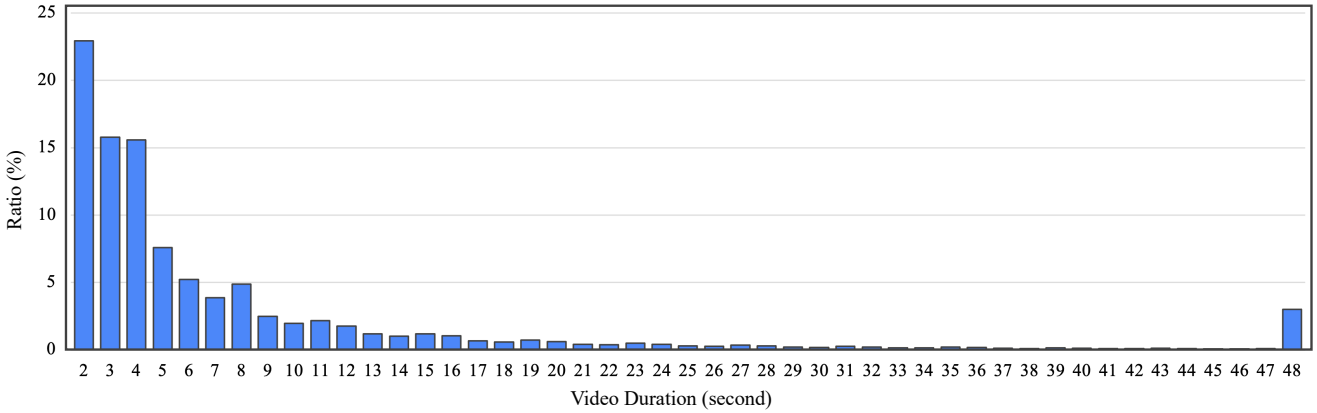


Figure 7. **Distribution of video duration of Panda-70M.**

Table 7. **Overview of 31 teacher models.** 31 teacher models are composed of 6 base models with various weights and input information. Input data includes vision (V), subtitles (S), and metadata (M). Vision data is either a video or a static video frame, depending on the type of base model. Metadata includes the title and the description of a video. For example, V-S-M for MiniGPT-4 means MiniGPT-4 with the inputs of a video frame, subtitles, and metadata.

Base Model	Type	Weights	Input Information				# of Models
			V	V-S	V-M	V-S-M	
Video-LLaMA [88]	Video VQA	pretrain / finetune	✓	✓	✓	✓	8
VideoChat [38]	Video VQA	7B	✓	✓	✓	✓	4
VideoChat Text [38]	NLP-based Video VQA	-	✓	✓	✓	✓	4
Video-ChatGPT [51]	Video VQA	-	✓	✓	✓	✓	4
BLIP-2 [37]	Image Captioning	opt2.7b / opt6.7b / flant5xl	✓	✗	✗	✗	3
MiniGPT-4 [94]	Image VQA	7B / 13B	✓	✓	✓	✓	8

A.2. Stage2: Stitching based on Semantics Similarity

The first stage introduces many short consecutive clips with the same semantic content. To this end, we propose an additional procedure to merge the clips with the same semantic content. Formally, given two adjacent clips C^1 and C^2 in sequence, we concatenate them into a clip if $\|f(C^1_B) - f(C^2_A)\| \leq 0.6$.

Finally, we perform a post-processing to stabilize the quality and diversity of the video clips with the following steps:

- First, we exclude the clips shorter than 2 seconds or clips that contain only slight motion (*i.e.*, $\|f(C_A) - f(C_B)\| \leq 0.15$). For the videos longer than 60 seconds, we only retain the first 60 seconds.
- Next, we represent each clip by the average of ImageBind features extracted from stage1 (Section A.1) and only keep the video clips that are semantically different (*i.e.*, Euclidean distance > 0.3) from the precedent clips to increase the diversity of the video samples.
- Finally, we trim out the first and last 10% of a video clip as we notice that the beginning and the ending of a clip usually contain unstable camera movement or transition effects.

With the proposed splitting algorithm, we split $3,790,459$ long videos into $70,817,169$ clips with an average clip duration of 8.477 seconds. We plot the distribution of video length in Figure 7.

B. Details of Teacher Captioning Models: Pool, Inference, and Selection

In Section 3.2, we propose to use multiple cross-modality teacher models for captioning. Specifically, we start with a large pool including 31 captioning models. We elaborate on the composition of the model pool and how we implement them for video captioning in Appendix B.1 and B.2 respectively. As running the inference of the models to 70M videos is computationally expensive, we select only 8 models as the representative, based on a human evaluation. We will describe more details about this process in Appendix B.3.

You are given some information about a video and will be asked to summarize the video (or the given video frame).
The subtitles of the video: "(video subtitles)"
Some descriptions of the video: ["(video title)", "(video description)"]
Please faithfully summarize the video (or the video frame) in one sentence.

Figure 8. **Prompt template of the VQA models.**

B.1. Introduction of 31 Captioning Models Pool

The primary reason to utilize cross-modality teacher models is to leverage multimodal data that would benefit video captioning. As such, we consider the base models, including image/video visual-question-answering (VQA) and image captioning models. Specifically, we employ Video-LLaMA [88], VideoChat [38], VideoChat Text [38], Video-ChatGPT [51], BLIP-2 [37], and MiniGPT-4 [94] as the base models. Based on these models, we collect 31 captioning models in total using different weights and input information. We list the summary of all captioning models in Table 7.

B.2. Inference of Cross-Modality Teacher Model for Video Captioning

We list the inference details of each base model as follows:

- **Video-LLaMA** [88] is a video VQA model. We only use the vision branch and do not use the audio one. The model uses Vicuna-7B [18] as the LLM to implement VQA. We use two official weights, including the pretraining weight, which is trained on 2.5M video-text pairs and LLaVA-CC3M [43], and the finetuning weight, which is further finetuned on instruction-tuning data from [38, 43, 94].
- **VideoChat** [38] **and Video-ChatGPT** [51] are video VQA models. We use Vicuna-7B as the LLM and follow the official codebase for the rest of the configuration.
- **VideoChat Text** [38] is a natural-language processing (NLP)-based video VQA model. The model would textualize the video content into video tags, dense captions, and a general caption respectively by three models [45, 56, 77]. As such, users can have a conversation with a chatbot and discuss the video based on the extracted textual content. The original codebase uses ChatGPT-4 [53] as the chatbot, which is, however, not freely released to the public. Thus, we replace it with LLaMA [67] for large-scale captioning.
- **BLIP-2** [37] is a language-image pretraining model. We only use it for image captioning and do not input texts. We use the weights pretraining with different LLMs, including OPT [90] (opt2.7b and opt6.7b) and FlanT5 [19] (flant5xl).
- **MiniGPT-4** [94] is an image VQA model. We use two variants respectively with Vicuna-7B and Vicuna-13B as LLMs.

To implement cross-modality teacher models for video captioning, we design the algorithms specifically for the models of different modalities. For an image model, given an n-frame video clip, we randomly sample a video frame in-between number $0.3 \times N$ and $0.7 \times N$ frames as the input. For a VQA model, in addition to the visual data, we also input a text prompt that could include additional textual information, such as video title, description, and subtitles, to assist video captioning. Specifically, we use the prompt template in Figure 8 if we would like to include the information of either metadata or subtitles or both for captioning. In contrast, we use a dummy prompt: "*Please faithfully summarize the video (or image) in one sentence.*" if we only input the vision data for captioning.

B.3. Selecting 8 Captioning Models based on a Human Evaluation

Running 31 captioning models on 70M videos requires significant computation resources. Hence, we propose to find a well-performing subset of the models by a two-step algorithm, including a human evaluation and model selection algorithm.

Human evaluation. First, we conduct a user study by showing the output captions of each model to humans. Specifically, we randomly sample 1K video clips and perform the inference of 31 captioning models on each video. Next, the human annotators are asked to select "*every good caption*", where a good caption is defined as: "*the caption cannot contain any wrong information and needs to cover the main action OR all of the main objects presented in the video.*" If none of the captions is a good caption, the annotators are asked to select the "All Bad" option. We randomly shuffle 31 captions to minimize the annotator's bias on the order of the captions. Considering that a human is hard to focus on reading all 31 caption sentences at the same time, we split the captions into three groups. The annotator will see the same video three times with at most 11 captions once. We show the interface of this user study in Figure 9 and plot the results in Figure 10.

Algorithm of model selection. In the second step, we collect a list of 8 captioning models as the representative to reduce the computation for large-scale captioning. Intuitively, one may opt for the models exhibiting the top 8 performance. Nonethe-

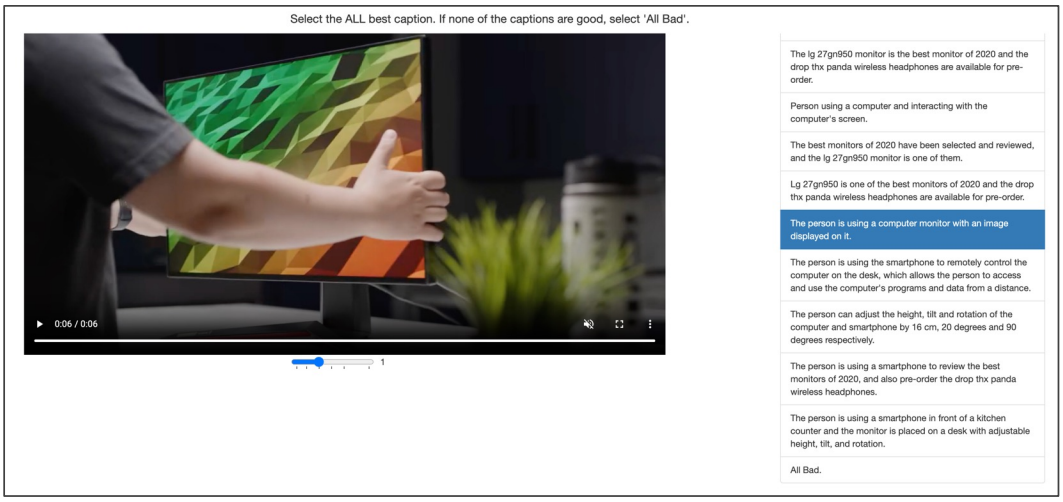


Figure 9. **Screenshot of the user study interface.**

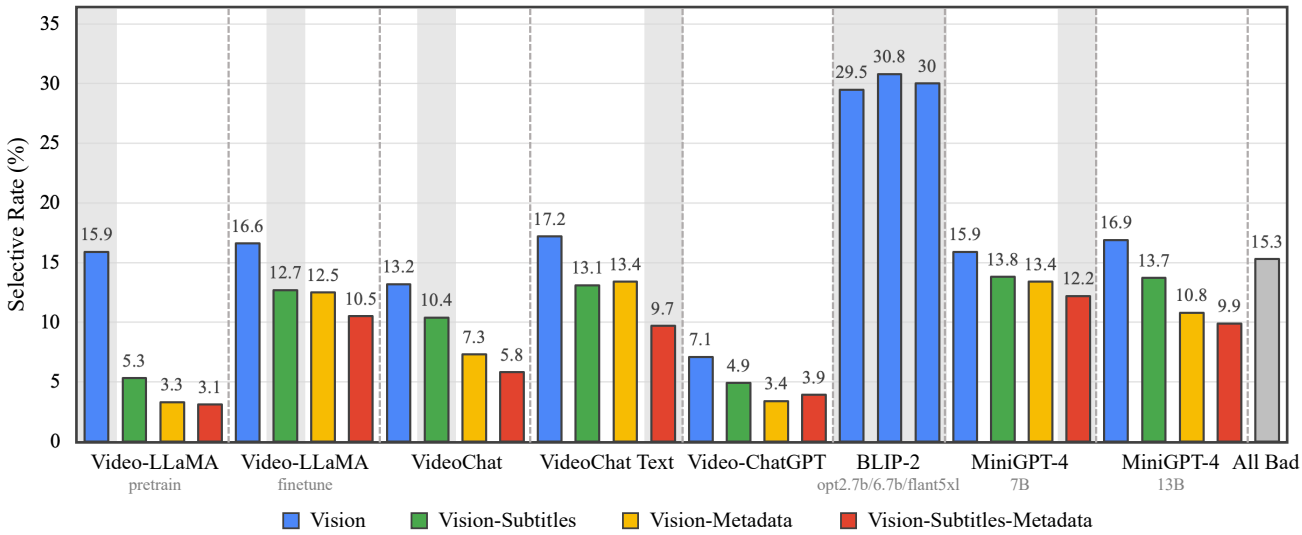


Figure 10. **Ratio of an individual captioning model to predict a good caption.** Each bar represents an individual model and is colored by its input information. We highlight the 8 selected teacher models with gray. Note that we also report the ratio of "All Bad" at rightmost.

less, such behavior does not align with the philosophy of our captioning algorithm. Specifically, our algorithm utilizes multiple cross-modality models to cover good captioning on various types of videos and only retrieves one best caption as the annotation for each video (as described in Section 3.3). Accordingly, we propose to use the set of models that can jointly cover a good caption for most video samples. The algorithm starts by selecting the best-performing model (*i.e.*, BLIP-2 with opt6.7b). Next, we only consider the videos that the previously selected model(s) cannot generate a good caption and then greedily find the model that performs best on those videos. We recursively collect the models under this mindset until we make the list of 8 captioning models. The 8 selected models are highlighted in Figure 10.

Additional findings. From Figure 10, we can also observe that a single captioning model can predict a good caption for at most $\mathbf{30.8\%}$ of the videos. In comparison, all 31 captioning can jointly predict at least one good caption for $\mathbf{84.7\%}$ of the videos (based on the "All Bad" ratio of 15.3%). This fact supports our motivation to use multiple cross-modality teacher models to jointly predict the captions for a video. Last but not least, according to our statistics, using 8 selected teacher captioning models can jointly predict a good caption for $\mathbf{76.8\%}$ of the videos which shows comparable performance with all 31 models while significantly reducing the computational requirements.

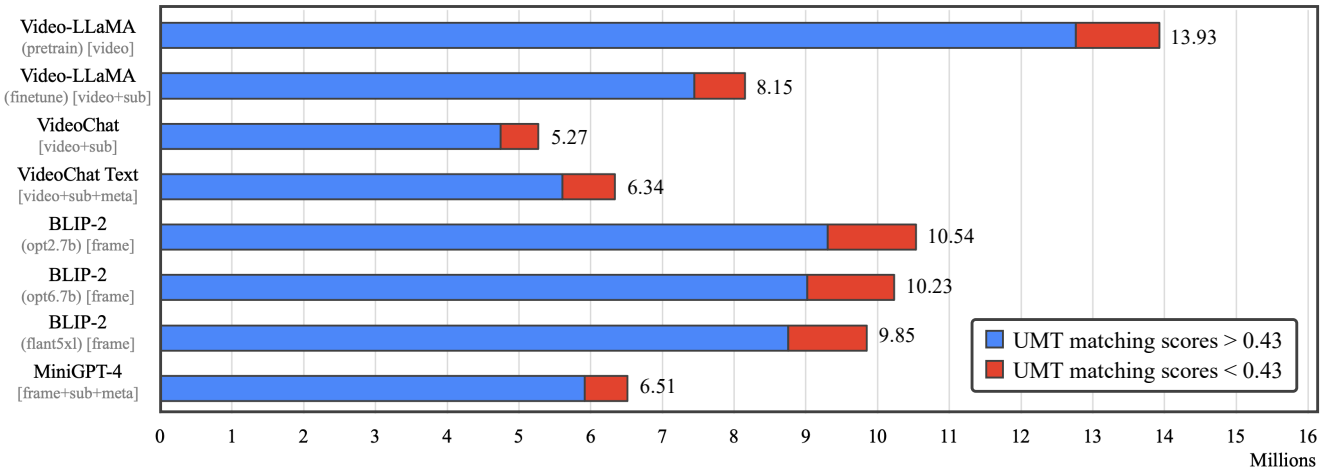


Figure 11. **Distribution of the source teacher models of the captions in Panda-70M.**

C. Details of Fine-Grained Video-to-Text Retrieval: Dataset, Training, and Inference

In Section 3.3, we mention that the available generic retrieval models [25, 39] cannot pick the best caption from 8 candidates predicted by our teacher models. The main reason is that all of the candidate captions are highly relevant to the video sample and require the model to discern subtle distinctions within each caption for optimal performance. To better perform our "fine-grained" retrieval task, we first annotate a subset of video samples by manually selecting the best caption as detailed in Appendix C.1. Next, we finetune Unmasked Teacher [39] (UMT) and run the inference of the model on all video samples respectively in Appendix C.2 and C.3.

C.1. Collection of Dataset

We randomly sample 100K video samples from our dataset and ask human annotators to select "*the best caption*" for each video. At the beginning of the task, the annotator will read the task description as follows:

"You are presented with a short video clip and a set of textual summaries that describe this clip. Choose the textual summary that is the most faithful and descriptive of the content of the video clip. Imagine you are talking on the phone with your friend and you need to describe the video to him."

Note that this task is different from the user study in Appendix B.3, where a human is asked to select "*every good caption*". But, we also randomly shuffle the captions and provide an "All Bad" option if all of the captions contain wrong information. We filter out $12,064$ videos with the "All Bad" option selected and split the dataset into $86,131$ and $1,805$ videos for training and validation. We plot the selective rate of each teacher model on the validation set in Figure 3 (blue bar).

C.2. Finetuning of Retrieval Model

We finetune Unmasked Teacher [39] as the text retrieval model on the training set. We use the larger model configuration, consisting of ViT-L/16 [21] and BERTlarge [20], and initialize the model with the weights pretrained on 25M image-text and video-text pairs. We follow the original codebase and only use the video-text contrastive (VTC) and video-text matching (VTM) loss functions for finetuning. For VTC, we implement hard negative mining [16, 35] which guides the model focusing on distinguishing the selected caption (*i.e.*, the positive sample) and the other 7 captions (*i.e.*, the hard negative samples). Specifically, we set the training weights of the positive and hard negatives as 1 while the weights of other negatives (*i.e.*, captions from other videos) as 0.01. For the training videos, we randomly sample 12 video frames and apply $\mathrm{RandomResizedCrop}$ transformation with scale $[0.5, 1.0]$ to get the video with the resolution of 224×224px. We use the AdamW [46] optimizer with a learning rate of $2e^{-5}$, $\beta = [0.9, 0.999]$, and a weight decay of 0.02. We set the batch size of 32 and last the training for 10 epochs. The model is funtuned on 8 Nvidia A100 GPUs (80GB).

C.3. Inference of Retrieval Model on Panda-70M

With the finetuned UMT, we automatically retrieve the best caption as the annotation for all 70M videos. We illustrate the distribution of the finetuned UMT's selection in Figure 11 and the caption length in Figure 12. We also plot the word cloud of the randomly sampled 100K caption annotations in Figure 13 to highlight the rich content within the annotated captions.

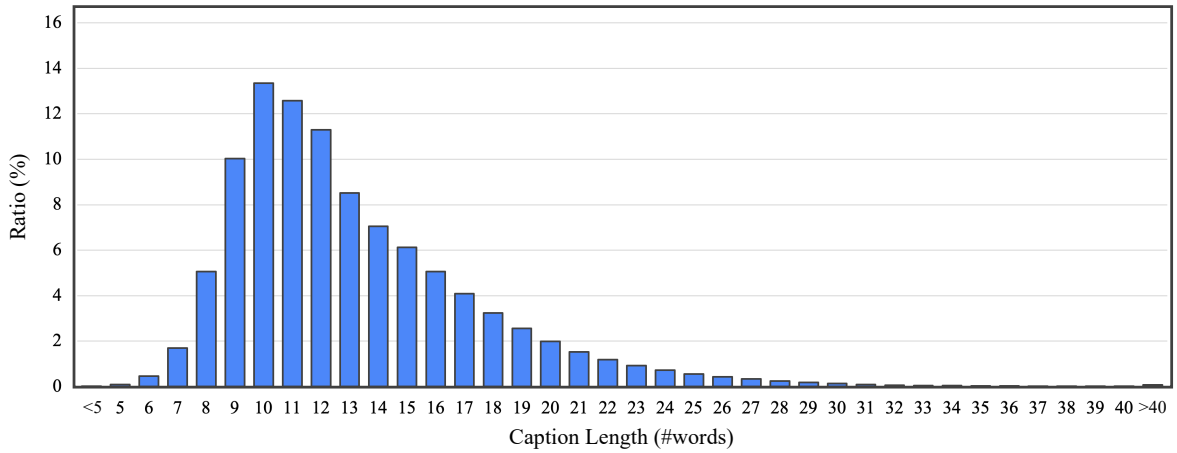


Figure 12. **Distribution of caption length of Panda-70M.**

Figure 13. **Word cloud of 100K caption samples in Panda-70M.**

In addition to the retrieval result, UMT also predicts a matching score for the video-text pair. In practice, we find the score is highly correlated to the alignment of the contents within the video-text pair. A score higher than 0.43 usually represents a strong association between the video and the caption. Numerically, 89.6% of the samples in Panda-70M have matching scores higher than 0.43.

D. Details of Student Captioning Model: Architecture and Training

D.1. Model Architecture

Figure 4 shows the architecture of the student captioning model. The model includes a vision branch and a text branch for additional subtitle and metadata inputs.

The vision branch shares the same design as Video-LLaMA [88]. Specifically, given an 8-frame video with the resolution of 224×224px, a visual encoder first individually encodes each video frame into multiple frame-level features with the dimension of 32×768. The visual encoder is composed of a frozen pretrained visual encoder, including a ViT-G/14 from EVA-CLIP [22] and a Q-former [37]. Subsequently, the temporal fusion module aggregates multiple frame-level features into a single 32×768 video representation. The module includes a position embedding layer to inject temporal information into video frames and a video Q-former to fuse the frame-level features. Finally, the model projects the video representation into a 32×4096 feature by a linear layer.

For the text branch, given a prompt with an arbitrary length, the model first tokenizes the prompt and embeds each token into a feature vector with 4096 length by a pretrained embedding layer [18]. Considering that the number of token embedding might be large for a longer prompt and the information of the prompt might not well align with the video content, we then design a text Q-Former to extract a fixed and shorter length of text embedding and at the same time, better bridge the feature

of the input video and text prompt. Specifically, the text Q-Former takes the inputs of the 32×4096 video representation as the queries and multiple token embedding as the key and value. The module then outputs a 32×4096 text representation. Finally, we combine the multimodal inputs by concatenating the text and video representations in sequence to get a 64×4096 feature and input it to the LLM to predict the video caption.

D.2. Training Details

The training data includes a video-caption pair and additional text information (*i.e.*, the metadata and subtitles). For the video data, we randomly sample 8 frames and apply the same video reading algorithm as in Appendix C.2. For the text branch, we embed the extra text information into the prompt. To learn a captioning model that can take both video-only and video-text inputs, we drop part of the text inputs at random. Formally, we use the prompt template in Figure 8 and employ the metadata or/and subtitles information with the probability of 0.5 (the sampling for metadata and subtitles are independent).

We use the AdamW [46] optimizer. The learning rate is initialized as $1e^{-6}$ and linearly warmed up to $1e^{-4}$ within the first 2,500 steps and gradually decreased to $5e^{-5}$ based on cosine annealing strategy [47]. We set $\beta = [0.9, 0.99]$ and use a weight decay of 0.05. We train the model on the whole Panda-70M with a batch size of 48 and last the training for 300K steps. The model is trained on 48 Nvidia A100 GPUs (80GB).

E. Visualization of Panda-70M Dataset

In the following subsections, we visualize video-text pairs in Panda-70M by category.

E.1. Category: Animal

"A person is holding a long haired dachshund in their arms."

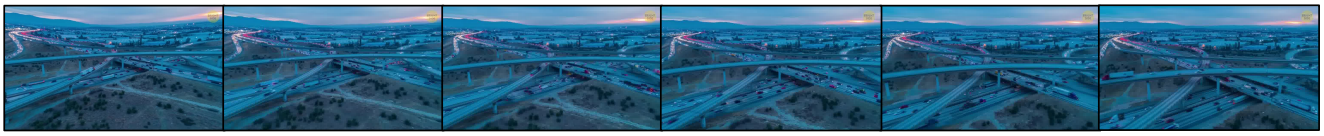
"A group of dolphins are swimming in the ocean."

"A rhino and a lion are fighting in the dirt."

"A cat laying on a rug with a leash around its neck."

E.2. Category: Scenery

"There is a beach with waves and rocks in the foreground, and a city skyline in the background."



"An aerial view of a freeway intersection at dusk."

"The waves are crashing on the beach and the water is foamy."

"There is a field of reeds blowing in the wind against a cloudy sky."

E.3. Category: Food

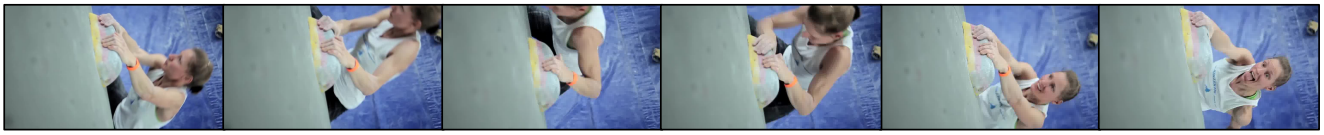
"Someone is frying dough balls in a pan with oil."

"A person is using a chef's knife to chop fresh parsley on a wooden cutting board."

"A person is making a pie crust on a table."

"There are sausages cooking on a grill, and a person is using tongs to turn them over."

E.4. Category: Sports Activity



"A female gymnast is practicing her skills on a climbing wall."

"A group of young girls are playing soccer on a green grass field with a goal in the background."

"A man paddles a canoe on a wave in the ocean."

"A skateboarder performs a trick in a skate park."

E.5. Category: Vehicles

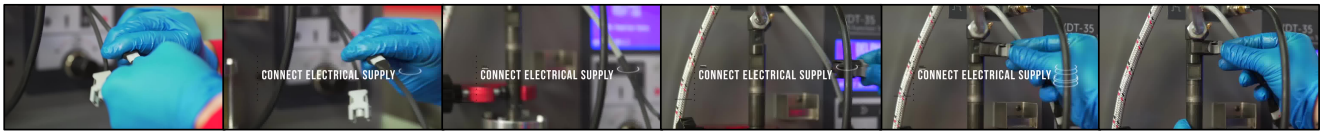
"An orange dodge challenger parked in front of a house."

"It is a rally car driving on a dirt road in the countryside, with people watching from the side of the road."

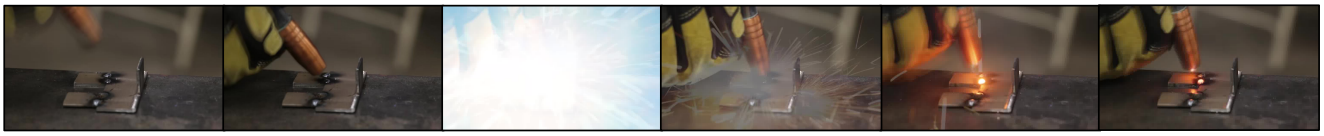
"A remote control monster truck is driving on rough terrain."

"A blue off-road truck is driving on a sand dune and jumping into the air."

E.6. Category: Tutorial and Narrative



"A person in blue gloves is connecting an electrical supply to an injector."



"A person is welding a piece of metal using a welding torch, and the metal is glowing red hot."

"A person is using an electric drill to make a hole in a piece of cardboard."

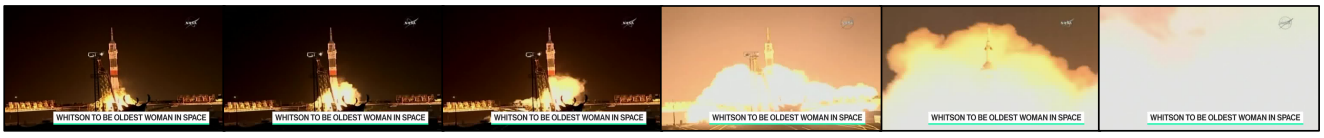
"A person is making a green clay model of a monster using different tools."

E.7. Category: News and TV Shows

"The columns of the temple of mars ultor in rome, italy are surrounded by trees and buildings."

"A large pile of lava blocking a road."

"Two men hugging each other in front of a trophy."

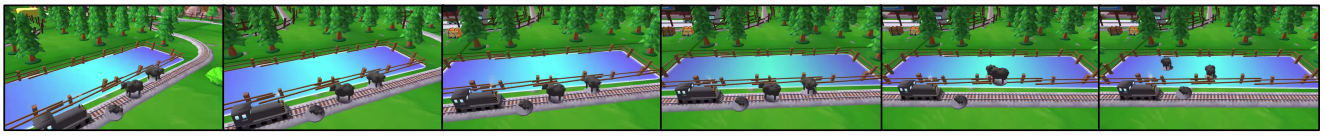


"A rocket launches into space on the launch pad."

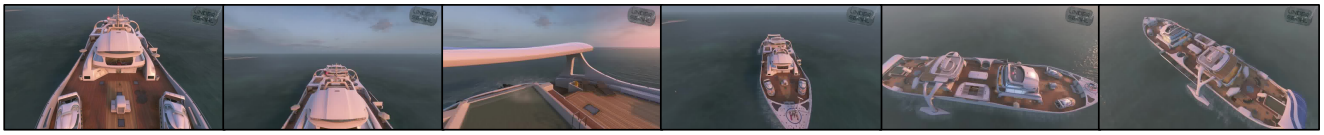
E.8. Category: Gaming and 3D Rendering

"A man in a spartan armor kneeling down."

"A screenshot of a minecraft game showing a snowy landscape."



"A 3d rendering of a zoo with animals and a train."



"The luxury yacht is sailing on calm waters with a beautiful sunset in the background."