

End-to-end Generative Pretraining for Multimodal Video Captioning

Supplementary Material

In this supplementary we first provide additional experimental results and descriptions on the dataset configurations for our pretraining and downstream tasks in Section A and B. Further implementation details for the downstream tasks are described in Section C. We then present more qualitative results in Section D. Finally, we discuss limitations and broader impacts of our method in Section E

A. Additional Experiments

A.1. Ablations on MSR-VTT

We perform additional ablations on MSR-VTT (mirroring Table 1 in the main manuscript) and provide results in Table A. We observe similar trends as on YouCook2 (Table 1), albeit with smaller gaps. We believe this is due to the larger domain gap between HowTo100M and MSR-VTT.

A.2. Impact of Pretraining Dataset Size

Figure A reports performance against dataset size. All four metrics show improvement (almost linear) when the dataset size is doubled. This signifies that our model could improve further by collecting more unlabelled videos for pretraining.

A.3. Open-ended Generative VideoQA

To further investigate our model’s decoding capability, we test our model on the open-ended long-form VideoQA (OL-VideoQA) benchmark [3]. Note that the training set is smaller than the one reported in [3] (26K vs. 53K examples) although we obtained the dataset directly from the authors. We test our model with/without pretraining to show its effectiveness and report the scores in B-1 and WUPS@ α metrics where α is a threshold for word similarity (see [3] for details). In Table B, our model without pretraining (No PT) serves as a strong baseline outperforming almost all the scores of the existing methods despite the fewer training examples used. The pretrained model (MV-GPT) then boosts performances further in all the metrics. Note that the gaps in WUPS@0.0 are relatively small since all soft matches are equally weighted regardless of their semantic similarities.

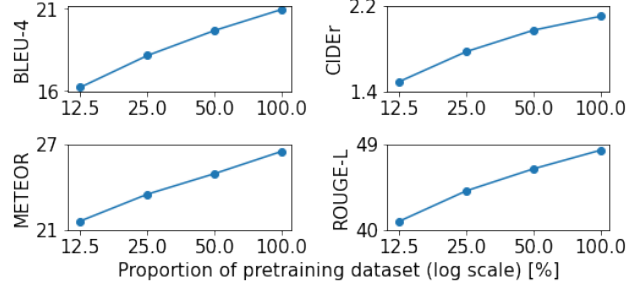


Figure A. Performance changes in four captioning metrics with varying pretraining dataset sizes on YouCook2 for video captioning.

PT Losses	PT parts	B-4	C	M	R-L
No PT	–	45.99	0.48	36.12	61.55
Baseline PT	E	46.47	0.51	36.64	61.82
CoMVT	E	47.02	0.52	37.03	62.19
M-MASS	E+D	47.88	0.56	38.00	63.27
UniVL	E+D	47.17	0.56	37.17	63.53
MV-GPT	E+D	48.92	0.60	38.66	64.00

Table A. Comparisons to existing pretraining losses on MSR-VTT.

Method	Train-set size	Bleu-1	WUPS@0.9	WUPS@0.0
CDMN+	52,604	25.38	34.53	59.20
AHN	52,604	25.81	34.14	59.66
HCSA	52,604	28.83	36.90	61.74
No PT	25,636	42.32	37.94	60.47
MV-GPT	25,636	46.98	40.81	62.09

Table B. Comparisons to SOTA on OL-VideoQA (from [3]).

A.4. Impact of Decoder as a Part of Encoder

As described in the main manuscript and depicted in Figure Bd, we use the pretrained decoder as a part of the encoder for the VideoQA model. To investigate the effectiveness of our decoder when used as a part of an encoder, we compare our model with and without the decoder for VideoQA, and observe a 1.0% and 0.8% gain in accuracy with the decoder on the MSR-VTT-QA and ActivityNet-QA benchmarks respectively.

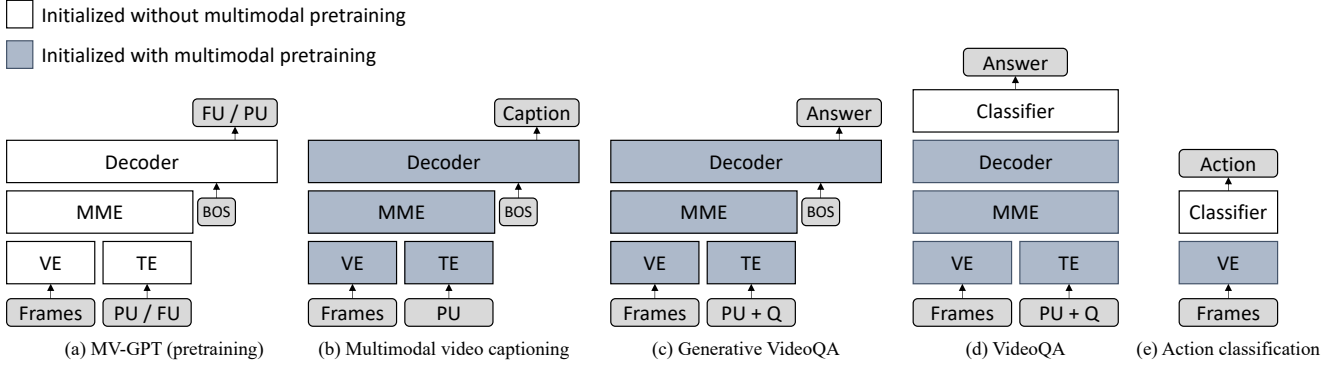


Figure B. **Overview of pretraining (a) and finetuning (b-d).** Each figure shows the network architecture (white and blue boxes) and the inputs and outputs (grey boxes). Blue boxes represent modules pretrained by our framework whereas white boxes are initialized without our multimodal pretraining. **VE**: Visual Encoder. **TE**: Text Encoder. **MME**: Multimodal Encoder. **PU**: Present Utterances. **FU**: Future Utterance. **Q**: Question.

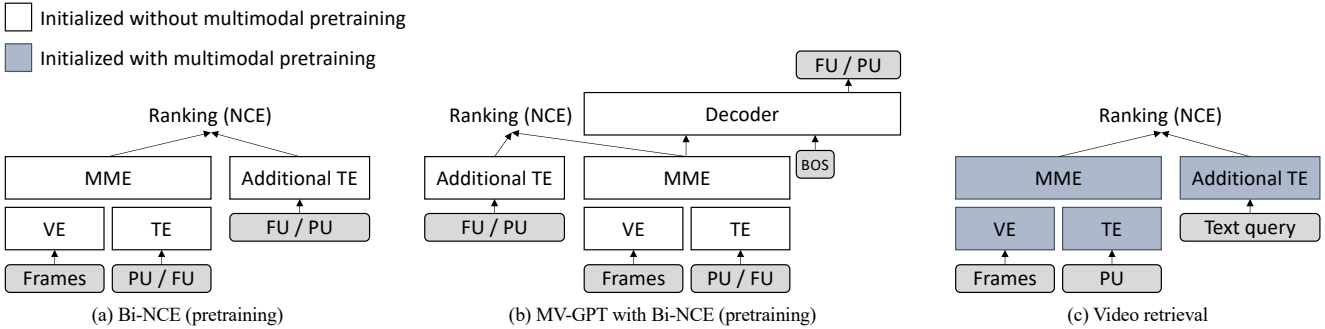


Figure C. **Overview of pretraining with baseline Bi-NCE losses (a) and MV-GPT with Bi-NCE (b), and finetuning for video retrieval (c).** Each figure shows the network architecture (white and blue boxes) and the inputs and outputs (grey boxes). Blue boxes represent modules that are initialized with the pretrained weights whereas white boxes are trained from scratch. We use NCE losses to train scores for the matching pairs of a multimodal video and a text. Note that we use an additional text encoder to compute the target text embedding. **VE**: Visual Encoder. **TE**: Text Encoder. **MME**: Multimodal Encoder. **PU**: Present Utterances. **FU**: Future Utterance.

B. Datasets

B.1. Pretraining Dataset Preparation

We prepare our pretraining dataset following [10] and extract triplets (F, U, W) of the video frames F , the present utterance U , and the future utterance W , from the videos in HowTo100M [8]. We obtained transcribed speech using the YouTube ASR API ¹, however these are noisy. To respect licensing terms, videos that have been removed from YouTube since the dataset was originally created are not used. We then divide these videos into shorter video clips. The duration of video clips is determined as follows: we start with a single ASR sentence and then iteratively expand the length of the video clip backwards by adding previous sentences until the segment is longer than 5 seconds. Each video clip therefore contains full sentences (no sentences

¹YouTube Data API. <https://developers.google.com/youtube/v3/docs/caption>

are cut-off mid way). This process results in 53.5M training examples. Since we focus on the pretraining approach, we keep only 7.5K examples as a small validation split.

B.2. Datasets for Non-generative Tasks

In addition to the datasets used for multimodal video captioning, which are described in the main manuscript, we make use of the following datasets for the experiments on the non-generative video understanding tasks.

MSR-VTT [13] is commonly adopted for video retrieval. We follow the standard splits for retrieval [14] containing 9K and 1K examples in train and test sets, respectively.

MSRVTT-QA [12] is a VideoQA benchmark derived from MSR-VTT, and contains 243K QA pairs. The dataset follows the standard splits released in MSR-VTT [13].

ActivityNet-QA [15] contains 58K QA pairs for VideoQA where the train, val and test sets have 32K, 18K and 8K pairs, respectively.

Kinetics [5] is the largest action classification benchmark.

We evaluate on both Kinetics 400 and 600, containing approximately 267K clips from 400 classes and 446K clips from 600 classes, respectively.

C. Implementation Details

C.1. Pretraining

As described in the main manuscript, we pretrain our model by the proposed bidirectional loss, which consists of the forward and backward generation losses. As described in the main manuscript, our framework pretrains a model consisting of a visual encoder (VE), a text encoder (TE), a multimodal encoder (MME) and a decoder (Figure Ba). After pretraining, a different subset of these components depending on the downstream task is transferred and finetuned, which is described in the following sections.

For pretraining, we initialize the text encoder and the decoder with the standard BERT and GPT-2 weights respectively pretrained on large-scale unlabelled corpora [2, 9]. Similarly, we initialize our visual encoder using the pretrained weights on Kinetics 400 in [1]. Our entire model is pretrained end-to-end using the Adam optimizer [6] for 1.5M iterations with the batch size of 2048. We adopt a weight decaying factor of 0.01, and use the cosine learning rate decay with a linear warm-up of 500 iterations.

C.2. Multimodal Video Captioning

Given an MV-GPT model pretrained on HowTo100M (Figure Ba), the entire pretrained MV-GPT is transferred for multimodal video captioning as our main target task as depicted in Figure Bb. The differences are the input and output configurations as described in the main manuscript; during pretraining, we feed present utterances (PU) as inputs predicting future utterances (FU) in forward generation and vice versa in backward generation whereas our model predicts captions given present utterances for captioning. Note also that we feed a special BOS token to initiate the sentence generation from the decoder.

We finetune the entire model end-to-end for 1K iterations with an initial learning rate of 0.0001 and a batch size of 512, and use the best validation checkpoint selected based on the Meteor score. For testing, we perform beam search with a beam size of 5 as in [7]. Note that we initialize the decoder using the weights of GPT-2 [9] when we test models trained by encoder-only pretraining methods.

C.3. Generative VideoQA

Generative VideoQA requires generating an open-ended answer given multimodal video and a question. While a question is given as an additional text input, we simply concatenate it to the present utterance; this allows us to use the original MV-GPT model for this task without any change as depicted in Figure Bc.

C.4. VideoQA

Following previous work [11], we formulate this task as a classification problem of predicting a predefined answer class. Note that we simply concatenate the input question to the utterances from the clip and feed the concatenation as a single textual input. Although we do not decode any textual outputs in this task, we still make use of the decoder as an additional multimodal encoder since our decoder is also trained to contextualize the input embeddings by applying the masked language modeling on the decoder outputs (see Section 3.1.2 in the main manuscript). Instead of feeding the BOS token and predicting next tokens, we first obtain the embeddings of the inputs from the decoder, average-pool these embeddings, and feed the pooled embedding to a two-layered MLP classifier to predict the answer (Figure Bd). Note that we use the entire pretrained model but append a randomly-initialized classifier.

For every experiment, we finetune the entire model end-to-end on the downstream benchmark for 20K iterations with a batch size of 512 and report the results using the checkpoint with the best answer accuracy.

C.5. Action Classification

Our goal with the experiments in action classification is to show the effectiveness of the pretrained visual encoder in MV-GPT, and therefore we simply discard all the other components and append a randomly initialized classification layer to the visual encoder as illustrated in Figure Be. For finetuning, we follow all the exact evaluation protocols used in [1].

C.6. Video Retrieval

The common practice for retrieval is to train a video-text joint embedding using *discriminative* losses only, typically in the form of a standard NCE loss [4], where each video clip has a single corresponding textual caption. In the retrieval experiments, we investigate whether our generative pretraining loss can provide a boost to performance. Since each example forms two inputs-target triplets, *i.e.* (F, U, W) and (F, W, U) , in our bidirectional frameworks, we apply NCE losses on both (Bi-NCE; Figure Ca). Note that we use an additional textual encoder to compute embeddings of the target texts. We then add our generative pretraining loss to this framework (Figure Cb). Finally for finetuning, we transfer the visual/textual/multimodal encoders and the additional text encoder of the pretrained model, and train the network using an NCE loss with the text query provided in the downstream benchmark.

For pretraining, we down-weight the bidirectional NCE losses with a factor of 0.001, and follow the same hyperparameters used in the regular MV-GPT pretraining. For finetuning, we train the entire network end-to-end for 1K







Inputs (video frames and transcript)			Generated captions	
 <p>Transcript: So then what I do is I add some parsley now, you don't have to put parsley do this recipe because it really doesn't call for parsing but you know something I like the atom countries and vitamins that come to the purse like unless he gives it a nice look so I just put that it but like I said, you don't have your so this about half a cup fresh firstly I use around or that's my favorite and now we just turn off the heat and we put this beside and going to be stitch.</p>			GT:	add parsley to the pot
			No PT:	add chopped tomatoes and ground beef and mix well
			MV-GPT:	add some chopped parsley
 <p>Transcript: Well repeat the process and proceed to make the rest of the rolls. We will now preheat the oil for deep frying once the oil is heated add in a roll at a time and deep fry on medium heat until golden brown in color this process of deep frying takes a good 4 to 5 minutes first spring roll once browned evenly drain the excess oil and place them on a serving platter serve these delicious spring rolls with a hot and spicy Szechuan sauce these crunchy and delicious spring rolls make perfect appetizers.</p>			GT:	fry the rolls in oil
			No PT:	fry fish in oil
			MV-GPT:	fry the spring rolls in oil
 <p>Transcript: What we're going to do is add some oil to a preheated pan add the shallots in followed by the chopped garlic and I'll stir and saute this just for a minute or two until they're fragrant. Then we'll place the shrimp in followed by the fried tofu and give it a stir until the shrimp become pinkish.</p>			GT:	add some oil chopped shallots garlic and salt to pan
			No PT:	add some oil to a pan and saute
			MV-GPT:	add oil and shallots to a preheated pan
 <p>Transcript: I'm just using often onion just a regular brown on him. I'm going to put that now our sausages are pretty much cook.</p>			GT:	add a sliced onion to the sausage pan
			No PT:	place the sausages on the grill
			MV-GPT:	add onion to the sausages
 <p>Transcript: So now we're ready for the next step and I'm gonna add in the chicken broth and then you're also gonna add in your sauce at this point and the beans now just give that a star and then we're gonna raise the heat up to like medium-high and you're gonna bring this up to a boil and believe it or not.</p>			GT:	add in the chicken broth sauce and the beans
			No PT:	add crushed tomatoes to the pan and stir
			MV-GPT:	add chicken broth and beans to the pot
 <p>Transcript: N/A</p>			GT:	spread mustard on the bread
			No PT:	flip the sandwiches
			MV-GPT:	spread the sauce on the bread

Figure D. Qualitative examples on YouCook2. GT. Ground-truth caption. No PT. No multimodal pretraining on HowTo100M. MV-GPT. Our pretrained model.






Inputs (video frames and transcript)			Generated captions	
 <p>Transcript: And as you can tell iris is really really excited about this at all. I'm gonna use as a shallow baking sheet. It's about half an inch thick it's about a quarter-inch full of water and the sticks will absorb a little bit of water but not much you just need to make sure you have a pan that's long enough to hold the skewers. So our skewers have been soaking for about 45 minutes the meat the vegetables and the teriyaki sauce have been also marinating for about 45 minutes.</p>			GT:	wash some skewers by soaking in water
			No PT:	soak the seaweed in water
			MV-GPT:	soak the skewers in water
 <p>Transcript: And there's the basil. There we go.</p>			GT:	add some basil to the pot
			No PT:	add paprika powder and tomato puree to the pan
			MV-GPT:	add basil to the pot
 <p>Transcript: So now we're ready for the next step and I'm gonna add in the chicken broth and then you're also gonna add in your sauce at this point and the beans now just give that a star and then we're gonna raise the heat up to like medium-high and you're gonna bring this up to a boil and believe it or not.</p>			GT:	add in the chicken broth sauce and the beans
			No PT:	add crushed tomatoes to the pan and stir
			MV-GPT:	add chicken broth and beans to the pot
 <p>Transcript: They really do taste the flavor. It's dramatically different. Really.</p>			GT:	continue chopping the cabbage
			No PT:	add salt to the pan
			MV-GPT:	chop the cabbage
 <p>Transcript: N/A</p>			GT:	remove from the oven and slice
			No PT:	bake the pizza in the oven
			MV-GPT:	remove the pizza from the oven and serve

Figure E. More qualitative examples on YouCook2. GT. Ground-truth caption. No PT. No multimodal pretraining on HowTo100M. MV-GPT. Our pretrained model.

iterations with a batch size of 512 and we report the scores from the best checkpoints on the validation set.

D. More Qualitative Examples

We show more qualitative examples on YouCook2 in Figure D. These qualitative examples demonstrate that our MV-GPT model can capture both textual cues (*e.g.*, the word ‘parsely’ in the first example) and visual cues (*e.g.*, the action of ‘spreading sauce’ in the last example) whereas the model without pretraining is often unable to capture these. Additionally, we include some video clips with the qualitative results in the supplementary material package.

E. Limitations and Broader Impact

Limitations: Our approach is not always successful, in particular in the presence of a significant domain shift between the pretraining data and the downstream application. Future work will address this limitation, for example by collecting curated pretraining data.

Broader Impact: Large, uncurated datasets scraped from the web may contain unintended biases, and models pretrained on such datasets may inadvertently amplify these biases. Therefore, applications of our work beyond the academic setting presented here should first carefully examine and filter the pretraining dataset for potentially

harmful biases in the data.

References

- [1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. ViViT: A video vision transformer. In *ICCV*, 2021. 3
- [2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019. 3
- [3] Zhang *et al.* Open-ended long-form video question answering via hierarchical convolutional self-attention networks. In *IJCAI*, 2019. 1
- [4] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, 2010. 3
- [5] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 2
- [6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3
- [7] 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 multi-modal understanding and generation. *arXiv e-prints*, 2020. 3
- [8] 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 *ICCV*, 2019. 2
- [9] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *Technical Report*, 2019. 3
- [10] Paul Hongsuck Seo, Arsha Nagrani, and Cordelia Schmid. Look before you speak: Visually contextualized utterances. In *CVPR*, 2021. 2
- [11] Zineng Tang, Jie Lei, and Mohit Bansal. DECEMBERT: Learning from noisy instructional videos via dense captions and entropy minimization. In *NAACL*, 2021. 3
- [12] Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwang Zhang, Xiangnan He, and Yueting Zhuang. Video question answering via gradually refined attention over appearance and motion. In *ACM MM*, 2017. 2
- [13] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *CVPR*, 2016. 2
- [14] Youngjae Yu, Jongseok Kim, and Gunhee Kim. A joint sequence fusion model for video question answering and retrieval. In *ECCV*, 2018. 2
- [15] Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. Activitynet-qa: A dataset for understanding complex web videos via question answering. In *AAAI*, 2019. 2