Building Scalable Video Captioning Models

Sarthak Mohanty, Nitin Vegesna, and Aditya Chandaliya
Georgia Tech
smohanty@gatech.edu, nvegesna6@gatech.edu, achandaliya3@gatech.edu

Abstract

In this paper, we explore different approaches for video captioning and expand on one such approach. We leverage existing techniques from image captioning to create zero-shot and fine-tuned models utilizing large pre-trained models. We also apply the enhanced sequence modeling of self-attention transformers to process video data in a temporal form. We find that our model is able to generate decent captions, though some samples are fragmented and incoherent. Our final model had a BLEU score of 0.38. The results suggest that although the architecture of the model is good, we require more training data and computational power to get a well-performing model. In terms of future work, we look towards incorporating an audio analysis alongside the video analysis to improve clarity of captions.

1 Introduction

Captioning, or the process of generating high-quality image text pairs, has become increasingly useful with the advent of large-scale foundational generation models. A common practice used for image captioning by high-performing models is to leverage existing large web-crawled image-text pairs, such as CLIP (Radford et al., 2021) and ALIGN (Jia et al., 2021).

Observing that videos are simply a sequence of images, a theory is that we can leverage those same existing image-text connections to enable video captioning. This idea has been central to the development of video captioning models, which primarily analyze images separately and then operate on the results using some form of a sequence modeling component to reflect the temporal features. Such models have seen success and suggest that similar ideas can be expanded upon to generate more advanced and perceptive models. One such advancement is the use of transformers, a well-established sequence modeling component, instead of more traditional models such as LSTMs. In this paper, we utilize the approach of a state of the art model on a more complex task to test its ability to describe videos in short captions. In particular, we augment well-known image analysis models with transformers and evaluate their performance on video captioning tasks.

2 Related Work

There have recently been works that show efficient ways of adapting strong image-text models to video foundation models with minimal extra computation. One successful work is CLIP-Hitchhiker(Bain et al., 2022), which aggregates video features by taking the mean of image embeddings with text as similarity queries, not dissimilar to our approach.

One crucial breakthrough for multimodal understanding came from the integration of recurrent neural networks. One such paper is (Zhao et al., 2021), which applied LSTMs to perform audio-visual summarization. This model employed a two-stream LSTM to encode audio and visual data separately, then combined the data into a similar latent space via an audio-visual fusion LSTM. Lastly, the model uses an encoder to capture temporal relations between different parts of the data.

In particular, there is a general consensus that transformers are effective for video recognition. From the best of our knowledge, this perception stems from the adaptation of the ViT model (Dosovitskiy et al., 2021) architecture for videos by extending self-attention from 2D image space to 3D temporal volume. An example of such a model is ViViT (Arnab et al., 2021).

Finally, while the code has not been released publicly, VideoCoCa (Yan et al., 2023) seems to be the prominent approach for video captioning at scale. This approach is also suspected to be used in training OpenAI's SORA. We hope to replicate its success in this paper.

3 Dataset

For training our models, we utilize the **MSR-VTT** (Microsoft Research Video to Text) dataset (Xu et al., 2016). This is a large-scale dataset which consists of $10,000$ video clips from 20 categories. Each video clip is annotated with 20 English sentences by Amazon Mechanical Turks. For training, we sample one of the 20 sentences for each of the video clips.

To parse the dataset, consisting of the $10,000$ video clips, we converted the clips to a sequence of image frames and an audio .wav file. Due to the immense data storage issue in the Google Colaboratory RAM, we use every $40th$ image frame instead of every image frame to mitigate the some of the storage costs.

The training, validation, and test set is split into a $70-15-15$ split, where the data is shuffled (seed-based) to get a random set of data samples. For the captions, we utilize a pretrained Bert tokenizer and model to first tokenize the string captions, and then use the model to embed them, with a right-pad used in the batch to get a 3d tensor for Pytorch use.

4 Building a video captioner

4.1 Approach 1: Zero-shot VideoCoCa

An video captioner is very similar to a traditional language model that predicts text. We thus start by providing a brief description of language models. First, a tokenizer is used to break strings of text into discrete tokens. Once decomposed in this way, the text portion of our corpus can be represented as a sequence, $\mathbf{t} = [t_1, t_2, \ldots, t_n]$. We can then build a language model over the text by maximizing the following likelihood function:

$$L(\mathbf{t}) = \sum_j \log P(t_j | t_{j-k}, \ldots, t_{j-1}; \Theta)$$

where Θ is the parameters of the captioner that are to be optimized. To turn this language model into a captioner, you need only to condition on the video, composed as frames of images. The challenge here is that images are composed of many thousands of pixel values. Conditioning on all of this information is exceptionally inefficient with

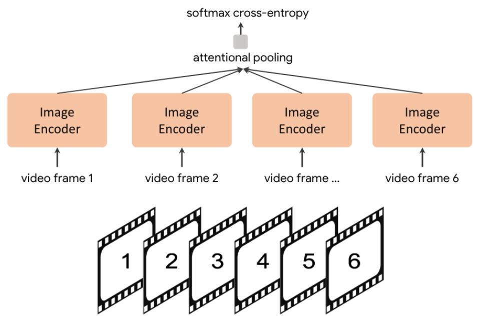


Figure 1: Augmenting CoCa for video captioning

our current neural networks, so we need a compressed representation space. Conveniently, CLIP provides just this.

Thus, given a pre-trained CLIP image embedding function F and some video $\mathbf{v} = [v_1, \ldots, v_n]$, we augment our language model objective as follows:

$$G(\mathbf{v}) = \frac{1}{n} \sum_{i=1}^{n} F(v_i)$$
$$L(\mathbf{t}, \mathbf{v}) = \sum_j \log P(t_j | t_{j-k}, \ldots, t_{j-1}; z_j; G(\mathbf{v}); \Theta)$$

Of course, training such a model from scratch would be extremely computationally expensive. Thus, we instead approximate this model objective by utilizing CoCa, a pre-existing image captioning model. (Yu et al., 2022)

CoCa adopts a cascaded decoder design, where the bottom half unimodal decoder encodes the text context with causally masked self-attention, and the top half multimodal decoder uses cross-attention to align image and text. The model is trained with joint contrastive loss and captioning loss.

To use CoCa, we pass each frame of the video through the pre-trained image encoder and average the embeddings. This gives us a zero-shot, **training-free** approach for video captioning.

4.2 Approach 2: Fine-tuned VideoCoCa

Note that the previous approach, while capturing spatial information, fails to encode temporal information. Thus, we we learn an additional pooler on top of the spatial and temporal feature tokens with a softmax cross-entropy loss.

4.3 Approach 3: ResNet + Transformer

With this approach we aim to implement the captioning process via a Residual Network (He et al., 2016) model to extract information from the sequence of images and transformer to capture sequential (temporal) dependencies and produce captions. First, we use the pretrained ResNet-18 model, with frozen convolutional blocks, as an encoder for each frame of the video. The component was frozen, since it will convert the images to a rich feature space, and it is already very good at extracting image features, as it was trained on a large and general dataset. The ResNet is projected onto a feature space with a size of 768 to match the embedding dim of the Bert model.

Then, the feature space tensors are passed through a Transformer. In the transformer, we use an encoder-decoder model, where relative positional bias and absolute positional embeddings are used and added to the feature embeddings. The transformer uses multi-head attention, with n layers and h heads in encoder and decoder. All other parameters were constant, but the norm-first argument, which was a hyperparameter as well. The norm-first argument determines whether the layer norm in the attention block is used prior or after the attention computations. In the encoder, no mask is used, since we are trying to capture long-term and sequential dependencies between all frames in the sequence. In the decoder, we use a lower triangular square mask during training to not attend to future tokens, since they are nonexistent in reality, but are available to allow for parallelism during training. After the transformer computation, to make this language generation, we project the tensor onto a space the size of the vocab size of Bert's tokenizer using a linear layer. This results in a tensor of size $sxbxd$, where b is the batch size, s is sequence length and d is the vocab size. The transformer implementation is directly based on the Attention is All you Need paper (Vaswani et al., 2017), with the addition of the norm-first hyperparameter.

For training, we utilize Cross Entropy Loss, since this can be thought of as a classification because for each token position, we must pick the correct token in the vocabulary. The stochastic optimizer used was Adam (Kingma and Ba, 2014), and the learning rate was a hyperparameter, ranging from $1e-3$ to $1e-6$, with a learning rate scheduler as a possible addition as well.

Due to the resources, the model was trained for only 15 epochs, since each training run was multiple hours long, and the hyperparameters were utilized in a grid search to find the best model parameters based on validation loss.

5 Experiments

Due to time constraints, we were only able to implement and analyze Approach 3. After conducting the grid search with provided hyperparameters, we gathered the optimal parameters, which were a learning rate of $1e{-}4$ + learning rate scheduler, 6 layers, 8 heads, and norm-first being $True$.

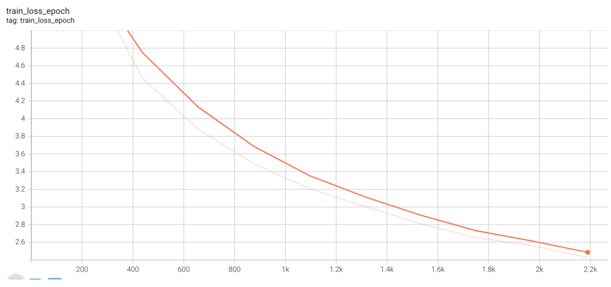


Figure 2: Train loss of optimal parameters

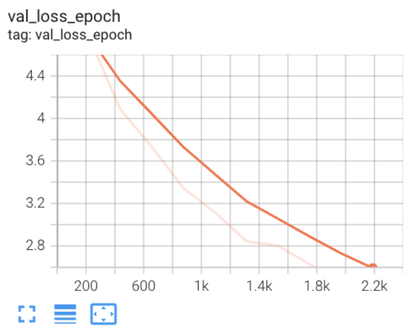
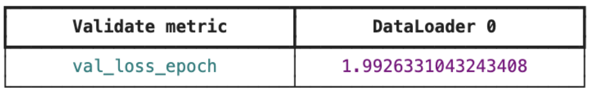


Figure 3: Validation loss graph of optimal parameters



Validate metric	DataLoader 0
val_loss_epoch	1.9926331043243408

Figure 4: Minimum validation loss of optimal parameters

The results show stable training, with likely better results with more training, as convergence hasn't occurred, and the training loss was steadily decreasing.

6 Analysis and Conclusion

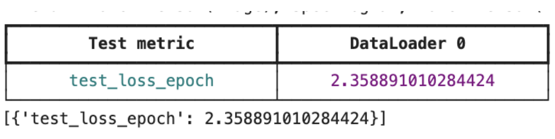
With minimal training, we were able to get pretty good captions, but also incoherent captions. Rows

Predicted Caption	**Ground Truth Caption**
A man in a purple shirt and red hat is playing guitar while playing on a microphone	A man in a blue shirt and blue shirt is playing guitar and singing into a microphone
A person is trying up through from the	A person is carrying someone away from danger
A group of the sports car are very at at a busy race show	A series of convertible sports cars are lined up at a large car show

Table 1: Comparison of Predicted and Ground Truth Captions

	Evaluation Metrics		
	Recall	**Precision**	**F1**
ROUGE-1	0.6860699710113917	0.6493109267925568	0.6618847461242713
ROUGE-2	0.45838117034278275	0.2990833003742873	0.35434326640328534
ROUGE-L	0.6722172025069564	0.6365854570425724	0.6487584559993945
BLEU		0.3899576889070564	

Table 2: Combined Evaluation Metrics (ROUGE & BLEU Scores)



Test metric	DataLoader 0
test_loss_epoch	2.358891010284424

```
[{'test_loss_epoch': 2.358891010284424}]
```

Figure 5: Test loss with optimal parameters

1 and 3 of Table 1 provide very good and similar captions, however row 2 does not finish its sentence and does not make sense. This is very promising, since we only used 7000 images for training, however, with a Bleu score of 0.38 and moderately good ROUGE score, we can say that we are getting slightly good captions, but not near human-level.

While the general architecture of our models are sound, more training and high-quality data is required to achieve good results, especially when considering the size of video data.

Another chance to greatly improve upon our models is to incorporate audio data. Although we parsed audio data from the MSR-VTT dataset, the data was too large to appropriately utilize, given our resources on Google Colaboratory. Further work could thus greatly improve the quality of our captions by incorporating the audio features, perhaps with an Audio Spectrogram Transformer.

References

Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. 2021. Vivit: A video vision transformer.

Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. 2022. A clip-hitchhiker's guide to long video retrieval.

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. 2021. An image is worth 16x16 words: Transformers for image recognition at scale.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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. 2021. Learning transferable visual models from natural language supervision.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. 2016. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5288–5296.

Shen Yan, Tao Zhu, Zirui Wang, Yuan Cao, Mi Zhang, Soham Ghosh, Yonghui Wu, and Jiahui Yu. 2023. Videococa: Video-text modeling with zero-shot transfer from contrastive captioners.

Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. 2022. Coca: Contrastive captioners are image-text foundation models.

Bin Zhao, Maoguo Gong, and Xuelong Li. 2021. Audiovisual video summarization.