Constant-Memory Video Encoding for Language Grounding

Anthony Reyna[1], Rodolfo Corona[2], Dan Klein PhD[2], Trevor Darrell PhD[2]
[1]Ohlone College
[2]Department of Electrical Engineering and Computer Sciences, University of California, Berkeley

2022 Transfer-to-Excellence Research Experience for Undergraduates Program

Introduction

Given a video and a natural language query (NLQ) describing an event, a machine learning system capable of returning a small segment of video where the answer to the sentence can be found would be of immense utility for assistive technologies and beyond. Developing such a system requires a machine to be capable of understanding language as well as understanding video. In this work we expand on the development of such systems at this intersection of computer vision and natural language processing—what is referred to as *grounded language learning.* Additionally, due to the high computation costs that comes with the processing of videos of greater and greater length, this work has a focus on methods that will ultimately carry a lighter memory-footprint. Having the flexibility to handle videos of vastly varying length would indicate some level of resiliency towards the high variability of naturally occurring, real-world events.

Prior Work and its Shortcomings

TWO MAIN PROBLEMS:

1. Current methods must search the entire video at once to find the segment being described, making GPU memory costs untenable when dealing with videos of significant length [1]
2. It's computationally expensive and redundant to query the same video with multiple NLQ's, as the entire video must be reprocessed for each NLQ provided.

NLQ:
"Where'd I last see my orange utility knife?"

Fig 1: Sample of Ego4D data used for training [2]

Methods

- PerceiverIO architecture in order to reduce computational/memory costs
- This architecture makes extensive use of the latent space for its processing pipeline [3]
- The latent array's size does not depend upon the input, it is instead a fixed size decided by us
- The bulk of computations completed within this architecture occur in the latent space, making for a lightweight model from end-to-end
- This is an improvement over previous architectures because the amount of compute needed will only scale linearly with the size of the input video

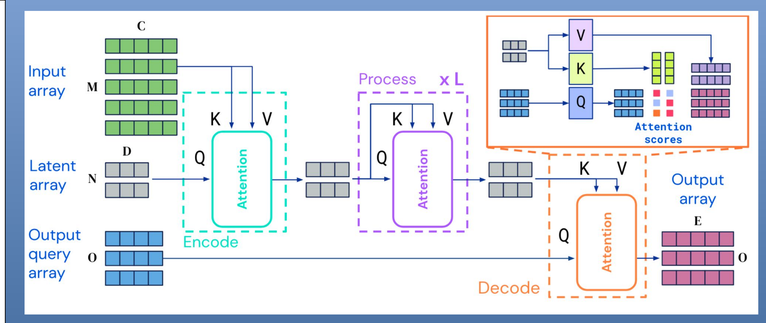

Fig 2: PerceiverIO architecture [3]

Results

Will have results ready before presentation. Will then get it printed and then i'll overlay it here

Where Do We Go From Here?

1. Improve current architecture's accuracy in retrieval task through fine tuning the model
2. Switch from processing entire video at once to processing in an "online" fashion—incrementally *as* the video comes in
3. Implement an evaluation/testing method to measure actual computation costs
4. Test boundaries of final model's manageable video lengths

Conclusion

As of now we've introduced a model that is computationally lightweight in comparison to the methods of previous work. While the retrieval rates can still be improved upon, what we've seen so far is promising. We now intend to begin development of our implementation of an architecture that keeps a memory cache of video as it comes in. We hope to show that this will keep the memory-footprint constant while upholding retrieval accuracy. This additionally flexibility of the model means that it would be more capable and more resilient to handling the variability inherent to real-world, natural phenomena.

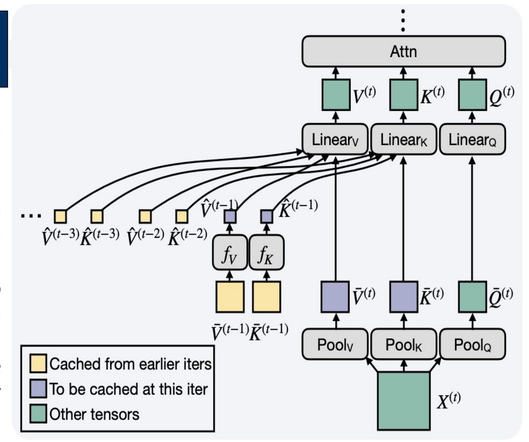

Fig : MeMViT implementation of iteratively caching memory

References

[1] Wu, Chao-Yuan, Yanghao Li, Karttikeya Mangalam, Haoqi Fan, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. "MeMViT: Memory-Augmented Multiscale Vision Transformer for Efficient Long-Term Video Recognition." arXiv, January 20, 2022. http://arxiv.org/abs/2201.08383.
[2] K. Grauman *et al.*, "Ego4D: Around the World in 3,000 Hours of Egocentric Video." arXiv, Mar. 11, 2022. Accessed: Jul. 09, 2022. [Online]. Available: http://arxiv.org/abs/2110.07058
[3] A. Jaegle *et al.*, "Perceiver IO: A General Architecture for Structured Inputs & Outputs." arXiv, Mar. 15, 2022. Accessed: Jul. 15, 2022. [Online]. Available: http://arxiv.org/abs/2107.14795

Acknowledgements

I'd like to first and foremost thank my mentor Rudy Corona for being the best guide one could ask for on this journey. I'd also like to extend a huge thank you to the TTE team for their endearing support. Lastly, a thank you to Professor Dan Klein and Professor Trevor Darrell for the opportunity to work in their labs throughout the duration of this program.

Contact Information

Anthony Reyna
AnthonyReynax@gmail.com
Github/Twitter: @NoStackEngineer

Funding

Many thanks to the Hopper-Dean Foundation for funding this research