

Video Transformers for Classification and Captioning Tasks

Team: LENSv7

Members:

Nam Nguyen The

Vojtěch Sýkora

Leon Trochermann

Swadesh Jana

Eric Nazareus

Content

1. Introduction
2. Related Works
3. Methods
4. Experiments
5. Results & Discussion
6. Demo
7. Conclusion



Introduction

Introduction

Problem statement & main contributions:

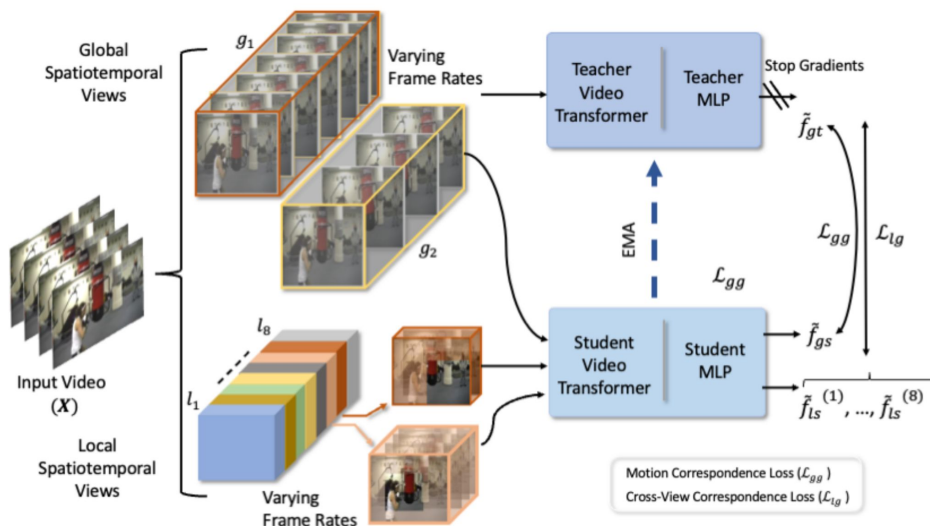
- **Adapt** video transformer-based models on **2 different downstream tasks**:
 - Video Classification
 - Video Captioning
- Video-encoder models used:
 - Self-supervised Video Transformer (SVT)
 - Video Mamba
- Extend the models to a difficult dataset: **Charades**
- Build an appropriate **data processing pipeline**
- Run a wide range of **experiments** to evaluate our approach



Related Work

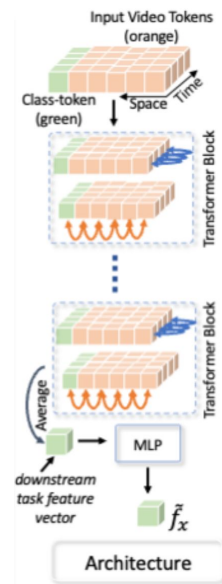
SVT - Training

- Self-supervised teacher-student training
- Generate multiple views at different spatial and temporal resolutions
- Extract features from each view
- Loss is based on the teacher-student output correspondence
- SVT learns features independent of spatial or temporal resolution




SVT - Inference

- Split video into frames and append a class-token (CLS)
- Then run it through the transformer
- The CLS token now represents a feature vector of the video
- Can be used for various downstream tasks
 - Originally an MLP for classification was used
- In our case, the transformer weights are frozen and only the MLP head is replaced for downstream tasks



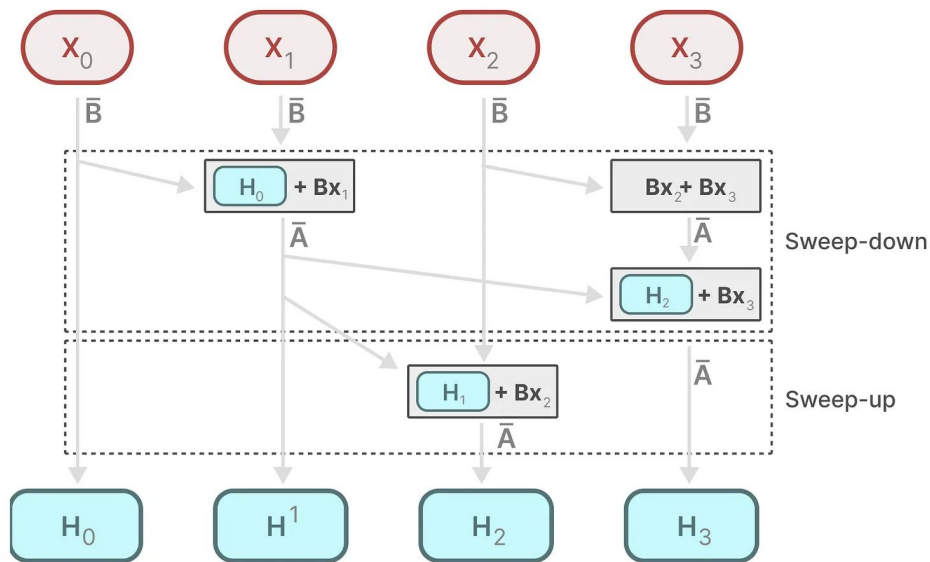
Video Mamba - Efficiency

	Training	Inference
Transformers	Fast! (parallelizable)	Slow... (scales quadratically with sequence length)
RNNs	Slow... (not parallelizable)	Fast! (scales linearly with sequence length)
 Mamba	Fast! (parallelizable)	Fast! (scales linearly with sequence length + unbounded context)

The Mamba architecture tries to **combine** the best of both worlds: **parallelization** in training and **linear scaling** with sequence length during inference.

Source: <https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state>

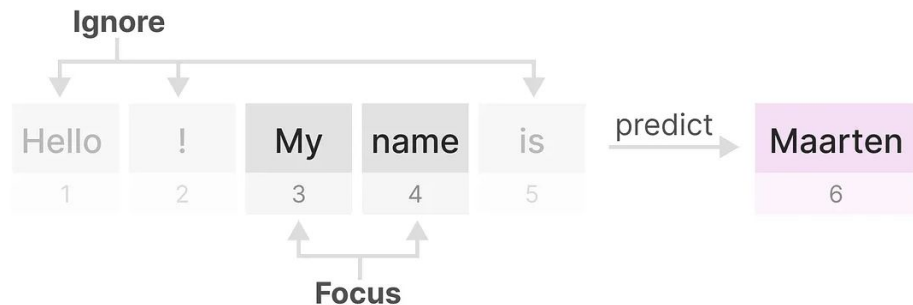
Video Mamba - Parallel Scan



Use the **parallel scan** algorithm to compute the hidden states efficiently:

$O(\log(n))$ steps

Video Mamba - Attention



Selective attention yields faster inference.

In fact, scales **linearly** with sequence length.

Source: <https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state>



Methods

Downstream Tasks

We use the previously mentioned backbones for:

- **Video Classification**

- The process of recognising occurrences of a set of actions in a video
- MLP used as classification head

- **Video Captioning**

- The process of generating descriptive text of the actions occurring in a video
- GPT-2 used as captioning head

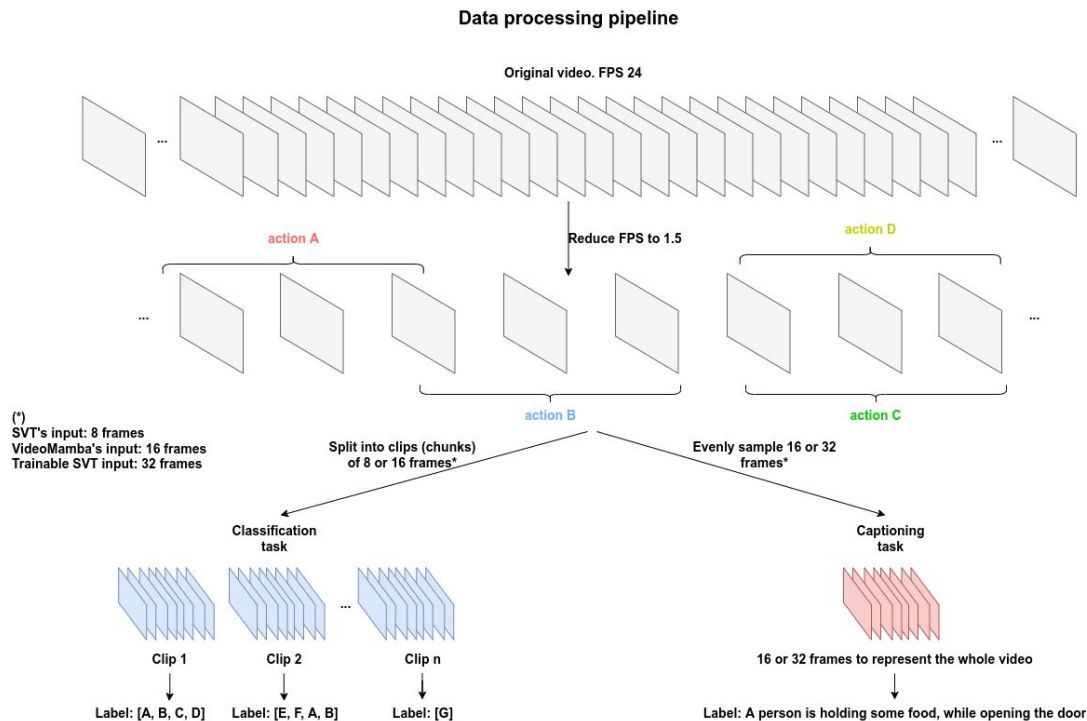
The Charades Dataset

- **Charades** is a dataset that was presented at the ECCV in 2016
- It is composed of 9,850 videos of daily indoor activities
- 267 actors were presented with activity prompts and acted them out, which was recorded
- Annotators generated class and caption annotations for these actions
 - Consensus was used with at least 4 workers
- In total, **Charades features**:
 - 66,500 classified actions (-> Video Classification)
 - 7,847 captioned videos (-> Video Captioning)

Challenges with Charades

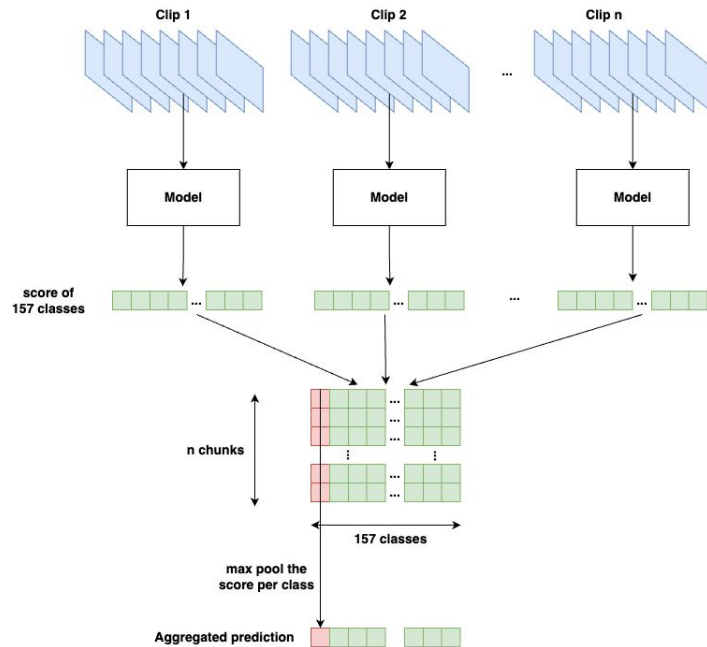
- **Video Classification on Charades** is not trivial
 - Videos can have multiple actions
 - Multi-label classification with 157 classes
 - The action annotations have a lot of overlap:
 - ~20M total action frames over ~6M unique action frames
 - Action classes are highly unbalanced:
 - Up to ~1:50 relative representation between some classes
 - Action timestamps are not precise:
 - E.g. they can exceed the end of the video
 - Neither naive multi-label nor single-label classification were viable, we needed a better solution

Our Pipeline: Overview



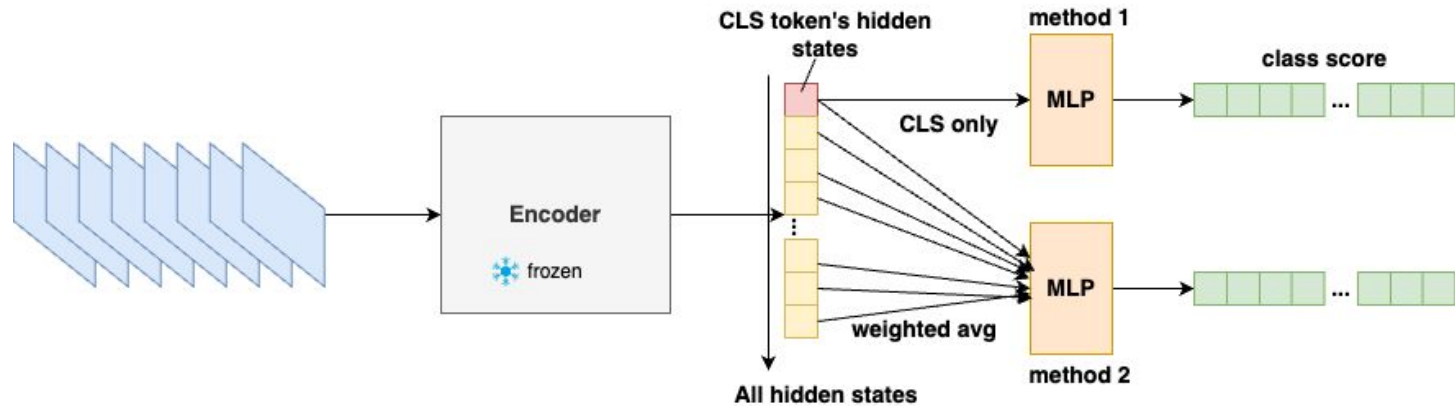
Our Pipeline: Video Classification

Sliding window prediction for classification task



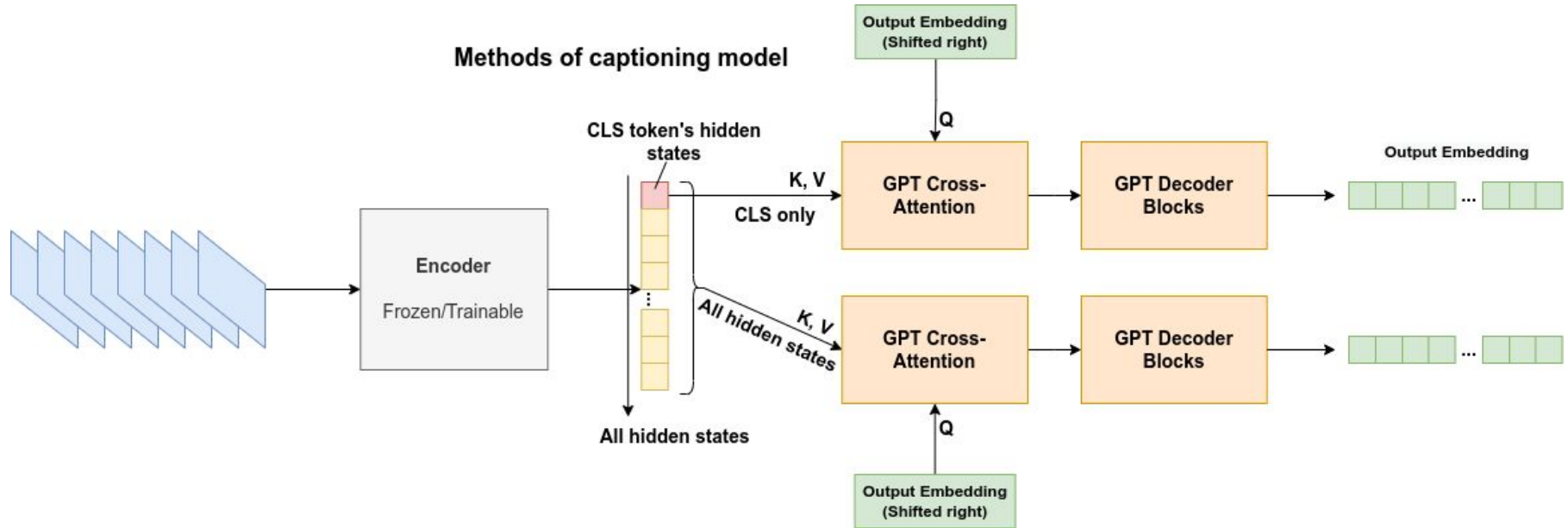
Video Classification in Detail

2 methods of classification model



Video Captioning in Detail

Methods of captioning model



Experiments

Video Classification

Best MLP head configuration

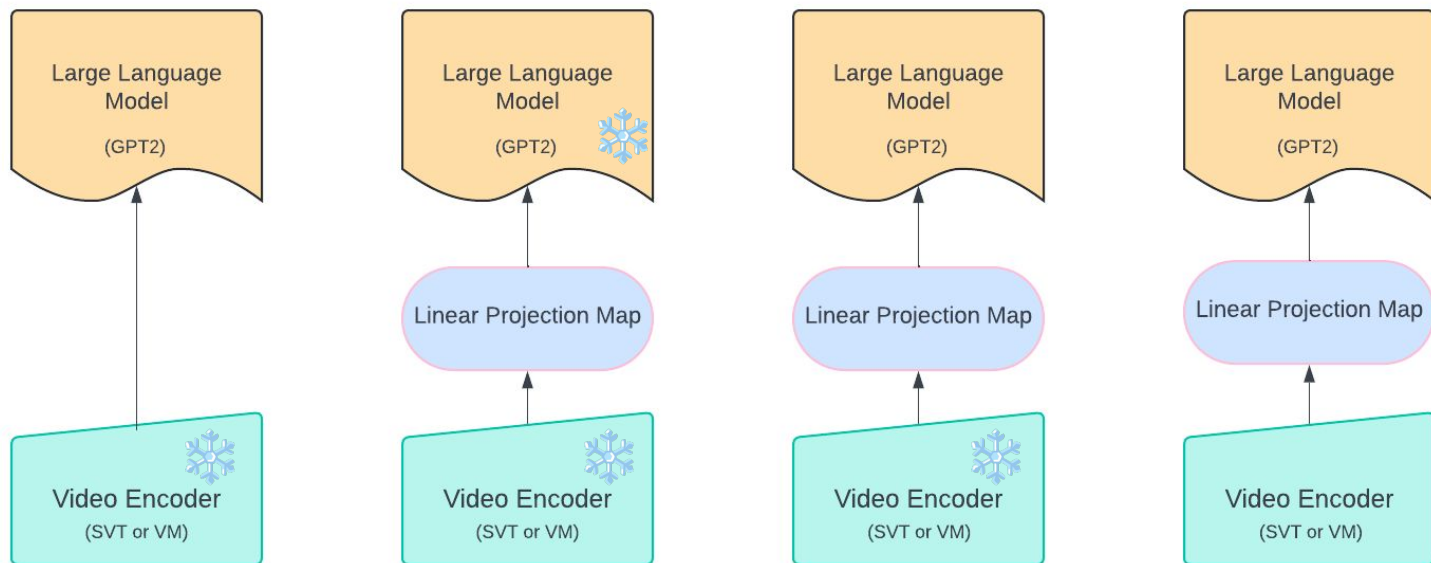
- Hidden states: CLS only or All token's hidden states
- Dropout: 0.7
- LayerNorm: False
- pos_weight: 2 (pos/neg imbalance fixing)
- Mean: [0.485, 0.456, 0.406]
- Std: [0.229, 0.224, 0.225]
- Optimizer: AdamW

Encoder	SVT	VideoMamba
Layers	[512, 512]	[1024, 1024]
Learning Rate	1e-4	1e-5

More challenges in captioning...

- For captioning, 8 frames in SVT is not enough - Doesn't capture enough information
- SVT has a time embedding layer, fixed for 8 frames in the pretrained weights -> Increase to 16 or 32
- But need to train the encoder as well -> Very computationally heavy but doable
- Two methods: Train the time embedding layer only or train the full encoder
- Training the encoder alongwith the decoder allows to have best captions (See Results)
 - Took 1 hour per epoch for the Charades captioning dataset on a 24GB L4 GPU
- See appendix for all experiments we tried

Video Captioning Methods



Video Captioning

Major hyperparameters for the fully-trainable model (in addition to those mentioned previously):

- Loss: Cross Entropy Loss
- Optimizer: AdamW
- Learning Rate: $2e-4$
- Epsilon: 0.000001
- Betas: [0.9, 0.999]
- Batch size: 4
- GPT2 max_tokens: 128
- Precision: 32



Results & Discussion

The classification task

- The classification model with both SVT and Mamba backbones are able to capture the actions in the videos. It achieves higher score than a few methods on the official benchmark.

	SVT	Video Mamba
Number of Params	121 M	73.6 M
Input Frames	8	16
mAP	24.87	29.38

40	MoViNet-A2	32.5	×	MoViNets: Mobile Video Networks for Efficient Video Recognition	🔗	📄	2021	
41	Timeception (R2D)	31.6	×	Timeception for Complex Action Recognition	🔗	📄	2018	
42	MultiScale TRN	25.2	✓	Temporal Relational Reasoning in Videos	🔗	📄	2017	
43	Co Slow_64	25.2	6.9x1	×	Continual 3D Convolutional Neural Networks for Real-time Processing of Videos	🔗	📄	2021
44	Slow-8×8	24.1	54.9x1	×	Continual 3D Convolutional Neural Networks for Real-time Processing of Videos	🔗	📄	2021
45	Asyn-TF	22.4	✓	Asynchronous Temporal Fields for Action Recognition	🔗	📄	2016	

Experiments with using all the hidden states vs the CLS token's

- We use **learnable** weighted average to aggregate the information from all the hidden states
- Model: Video Mamba

	CLS only	All hidden states
Num tokens	1	3137
mAP	29.38	24.57

Experiments with stride in sliding window inference

	SVT	VideoMamba
Without overlap (stride=clip length)	24.87	29.38
With overlap	25.01 (stride=3s)	29.82 (stride=3s)
		29.60 (stride=6s)

The table shows best evaluation mAP scores

The classification task - Analysis

- Video Mamba shows significantly better result compared to SVT with the same training configuration though having fewer parameters.
- Our models are still below a lot of methods benchmarked on the Charades datasets. This is because they have the whole network fully trainable, use many more input frames and a bigger architecture.
- Using CLS token only yields a better result. We hypothesize that this is due to the way the self-supervised pretraining is done, map only CLS hidden state of the teacher and student model.
- Sliding window inference with overlap show a small improvement.

Captioning Results

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-1 F-measure	ROUGE-2 F-measure
SVT + GPT2*	0.2248	0.1056	0.0487	0.0250	0.3101	0.0794
VM + GPT2*	0.2333	0.1043	0.0465	0.0233	0.2884	0.0651
Reference**	0.509	0.308	0.196	0.134	-	-

*Full trainable models.

** Zhao et al., 2018

The captioning task - Analysis

- SVT with more frames can produce slightly better results.
- Currently there is no official benchmark on Charades for captioning on PapersWithCode. But some paper claims to have much higher score. This might be due to the reason that we are using fewer frames in the encoder and smaller architectures.
- Using all the hidden states works much better, since the context from each frame is passed to the GPT2 decoder that allows for better output.

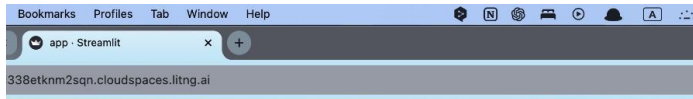


Time for a Demo!

Conclusion

Conclusion

- In this work, we fine-tuned SVT and Video Mamba on 2 downstream tasks: video classification and video captioning on the Charades dataset.
- Charades is a very challenging dataset due to long clip duration, multi-label and a large number of class with highly imbalanced distribution
- Our models are able to capture the content of the videos reasonably.
- There is still room for improvement by using more frames, allowing fully trainable encoders (classification) and using a bigger generative head (captioning).



Video Upload and Prediction

Choose a model

Self-supervised Video Transformer

Upload video



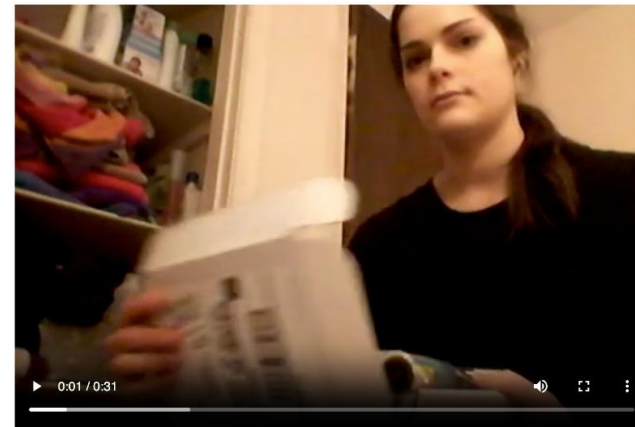
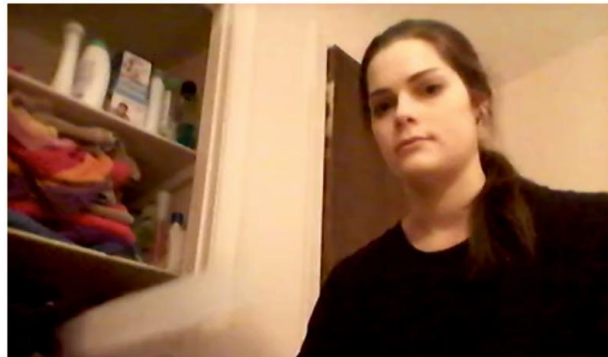
Drag and drop file here

Limit 200MB per file • MP4, MOV, AVI, MKV, MPEG4

Browse files



2MAZY.mp4 2.7MB



Predict action class

Action classification:

Top 5 classes:

Predicted action: Holding a box. Probability: 0.56

Predicted action: Holding a book. Probability: 0.33

Predicted action: Opening a box. Probability: 0.31

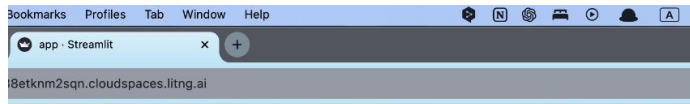
Predicted action: Watching/Reading/Looking at a book. Probability: 0.23

Predicted action: Taking food from somewhere. Probability: 0.18

Predict caption

Caption: a person is sitting at a table and opening a box. the person begins laughi

Demo Video 1



Demo Video 2

Predict action class

Action classification:

Top 5 classes:
Predicted action: Holding some food. Probability: 0.37
Predicted action: Someone is eating something. Probability: 0.34
Predicted action: Sitting in a chair. Probability: 0.29
Predicted action: Holding a dish. Probability: 0.27
Predicted action: Holding a phone/camera. Probability: 0.25

Predict caption

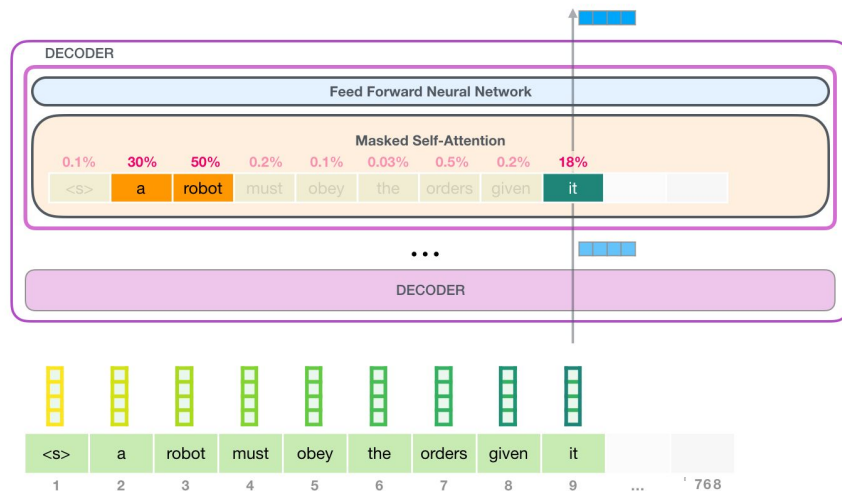
Caption: person is sitting at a desk. Another person is working on a sandwich.



Thank You!

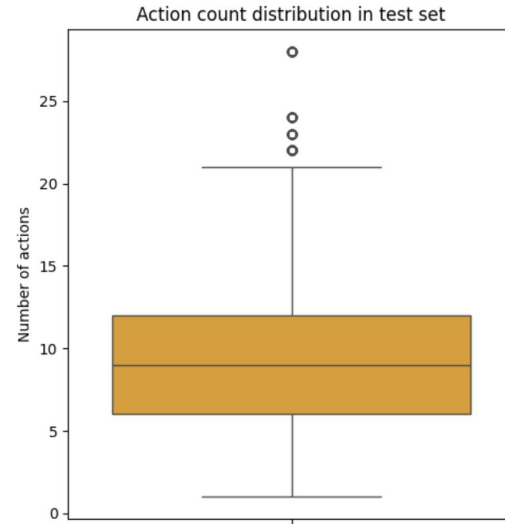
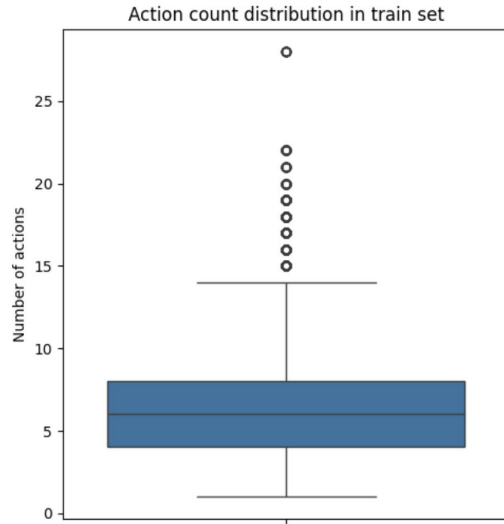
Appendix - GPT-2 for captioning

- Predicts next word based on current input sentence and hidden state input (optional)
- Blocks of decoders stacked together
- Each decoder consists of:
 - Self-attention on input features
 - Feed-forward Neural Network
- Hidden state input - Video encoder features
 - Can be only CLS token or all tokens
- GPT-2 small
 - Dimension: 768
 - Decoder blocks: 12
 - Parameters: 117M

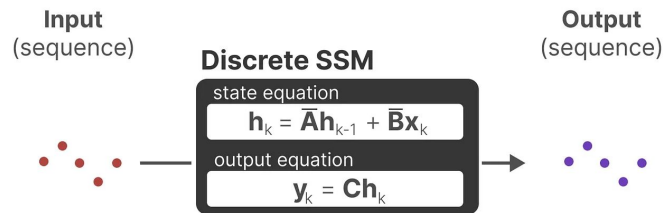
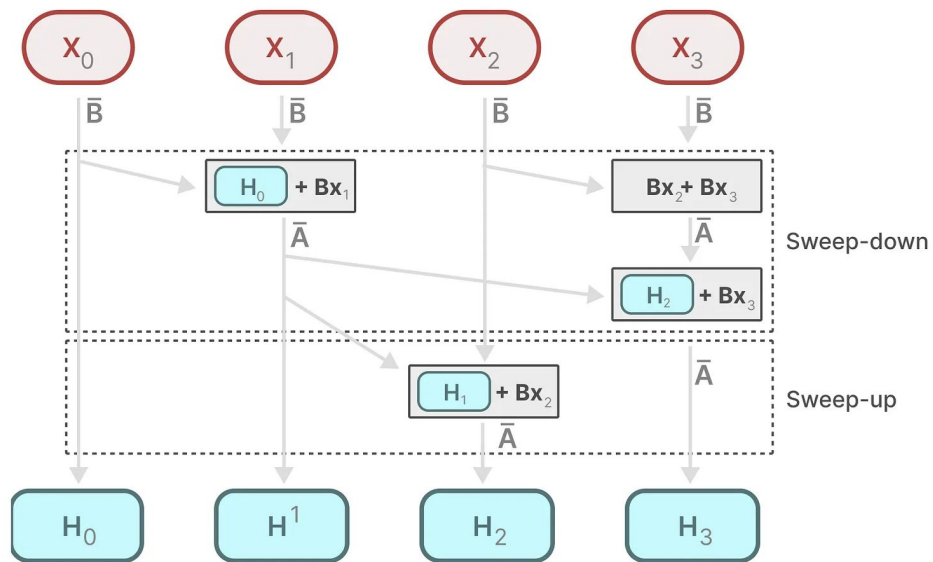


Source: <https://jalammar.github.io/illustrated-gpt2/>

Appendix - Charades Actions Per Video



Appendix - Video Mamba Equations



Source: <https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state>

Appendix - Captioning experiments we tried

1. Encoder frozen -> take only cls token state -> Decoder trainable
2. Encoder frozen -> take all hidden states -> Decoder trainable
3. Encoder frozen -> take all hidden states -> linear projection mapper -> Decoder trainable
4. Encoder frozen -> multiple clips of a single video -> get cls token for each clip -> concat as hidden state to trainable decoder
5. Encoder trainable -> take only cls token state -> Decoder trainable
6. Encoder trainable -> take all hidden states -> Decoder trainable