COL775 Deep Learning Assignment 2

Harsh Pandey csy217545@iitd.ac.in
2021CSY7545

Sanket Gandhi tt1170896@hpc.iitd.ac.in
2017TT10896

Editor: "

About

This is the Assignment 2 of COL775 Deep Learning. The main task was to do Causal Reasoning from Videos. In this paper we implemented 2 baselines and also participated in the online challenge.

1 Baseline: CNN + BERT

In this section we will show the approaches we use and the design choices we made for CNN + BERT style model.

1.1 Pre processing and defining our loss

There are some pre-processing steps we do to formulate the question

1.1.1 Making question answer pair

- Note that we have different types of questions for videos. We convert them to a single format as explained below.
- For descriptive question it's already fine, so no processing needed. Here forming a question answer pair $< ques, ans >$ will be direct.
- For the rest of the 3 types of questions, we take the choice value and append it with the question. And now this becomes a binary task to predict whether the $ques + choice$ is correct or not. So a question answer pair becomes $< ques\_choice, ans >$

1.1.2 Forming Batches

- Since loading a bigger batch was impossible for the given memory of HPC so we segmented the batch into 2 steps.
- First we select B number of videos.
- Next for each video we select QPV number of questions.
- Since all the QPV questions will use the same Video, i.e. the same video embedding so we don't need to compute it multiple times.

- After computing the encoding of a video we expand the video embedding (which will be small as compared to a complete video) and then pass it jointly with the correct questions.
- Note that this will result in gradient flowing for all QPV questions so we scale down the learning rate so that gradient don't explode.

1.1.3 Masking using question type

- Since we will always have question type with us so we are using this to take prediction of model on only the relevant classes.
- We give a separate mask value which is added to the output (before softmax) of the classifier/decoder. Because of this the model will only consider the desired set of answers for a given question type and hence should make the learning more easier.
- But batching was not possible as the implementation only allowed batch size of 1.

1.2 Architecture 1

This is the first architecture which we used to train for VQA task.

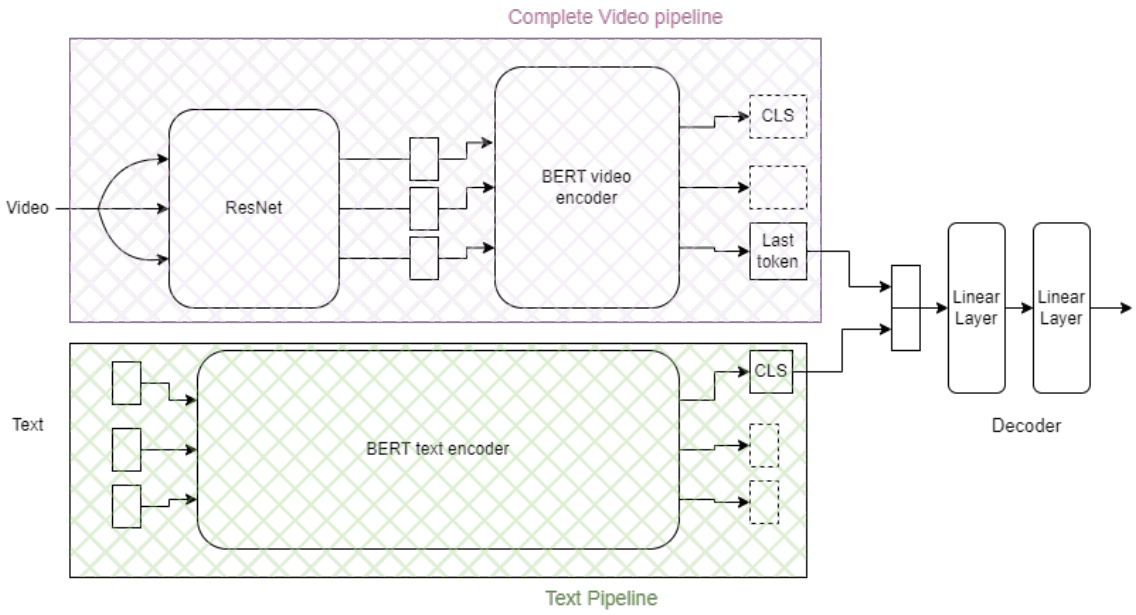

Figure 1: This is the Architecture 1

Note that in the figure 1.2 we can clearly see the architecture of the model.

- First is the ResNet block, each frame of the video is passed through this block so get an encoding of each frame.

- After we got the encoding of each frame we add positional embedding to it and then pass it to a BERT encoder (BERT video encoder).
- Here we take the last encoded frame of the video.
- From the question part, we take the Embedding of the CLS token which will represent the entire question.
- Now in this architecture we append both the text and video embedding.
- Finally we pass this through 2 fully connected linear layers (the decoder part of the network).

1.3 Architecture 2

This is the second architecture we used for CNN + BERT style model.

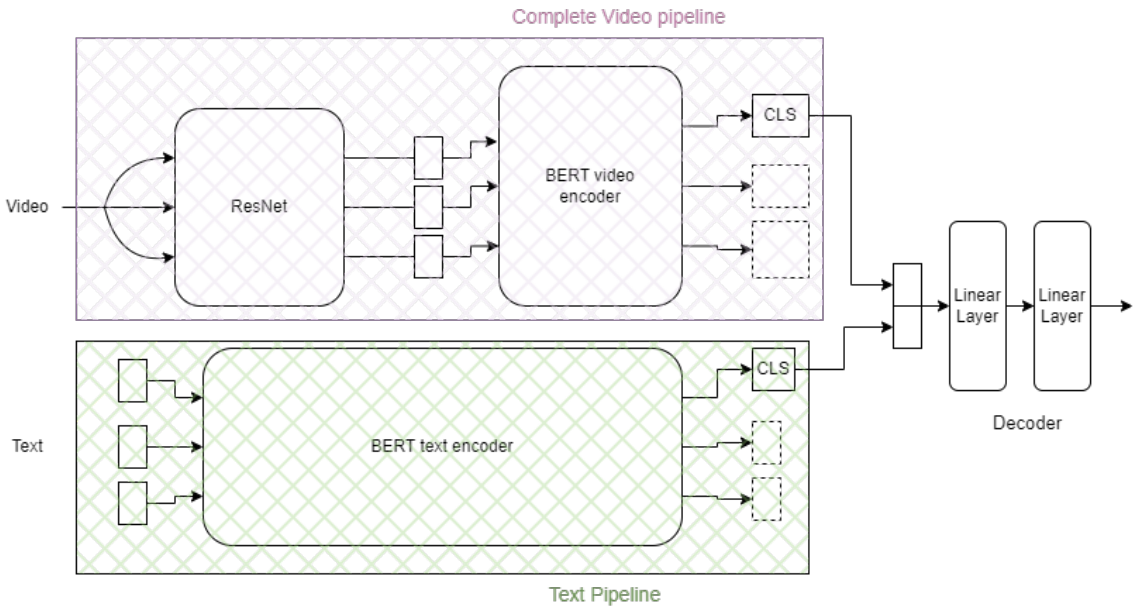

Figure 2: This is the Architecture 2

Note that in the figure 2.1 we can clearly see the architecture of the model.
This is very similar to the first Architecture.

- We noticed that even by freezing the video embeddings we were getting similar accuracy scores of about 40 percent.
- We noticed that this might be because of the fact that the last video embedding will only have the information about the last and related frames.

- So we added a CLS token at BERT video encoder step and took the output of CLS token and used it as an encoding of video will be better choice.

The rest of the part remains the same.

1.4 Architecture 3

This is the third architecture we used for CNN + BERT style model.

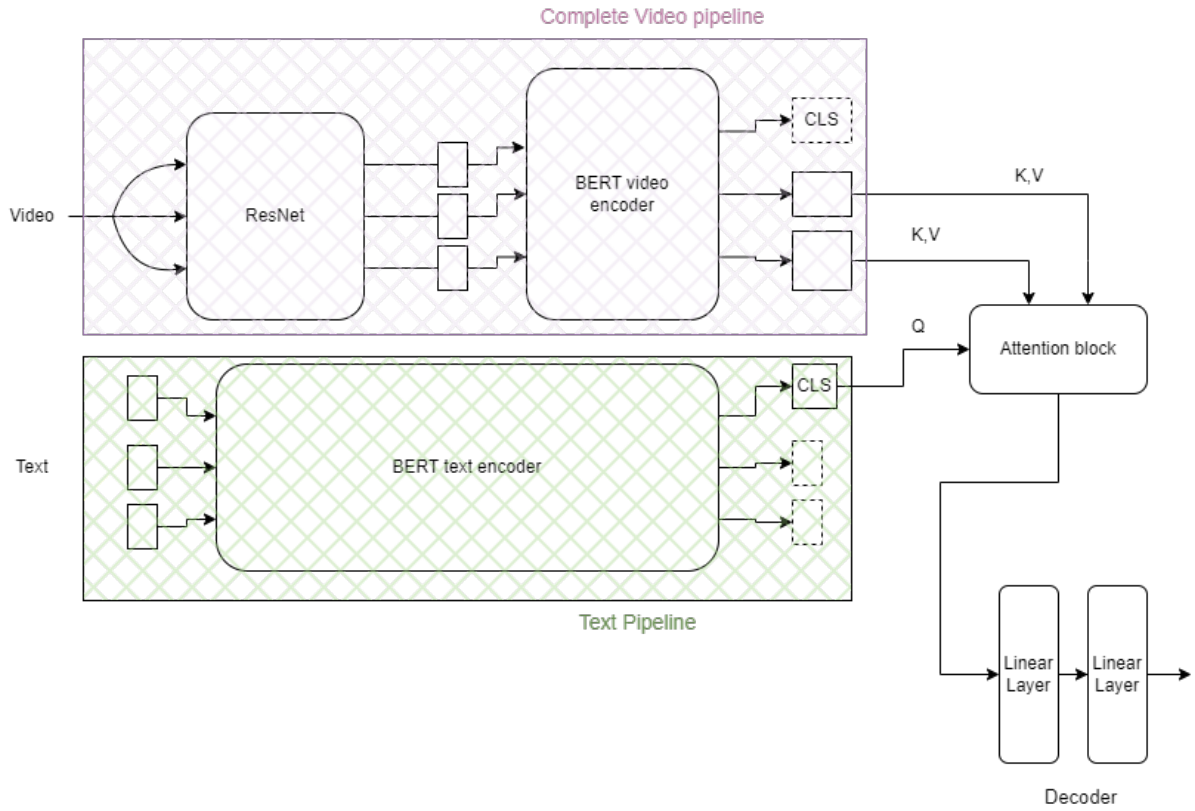

Figure 3: This is the Architecture 3, attention of VIdeo embeddings with the question embedding (the CLS token one)

Note that in the figure 2.1 shows the architecture of the model.
This is a bit different from the previous Architectures.

- Even taking CLS token didn't result in significant gain.

- So rather than just taking 1 encoding of video and discarding all the other, we calculated attention of the embedding of the video frames with the embedding of the questions (just 1 i.e. the CLS token).

1.5 Results

This section contains the loss curves and accuracies of the models achieved.

Model	F1 macro	F1 micro	Accuracy
TransformerClsToken	0.141	0.428	0.428
TransformerEndToken	0.258	0.462	0.462
TransformerAttn	0.311	0.542	0.542
Resnet3D	0.211	0.430	0.430

Table 1: Execution accuary

We show the plot of loss for the main model here.

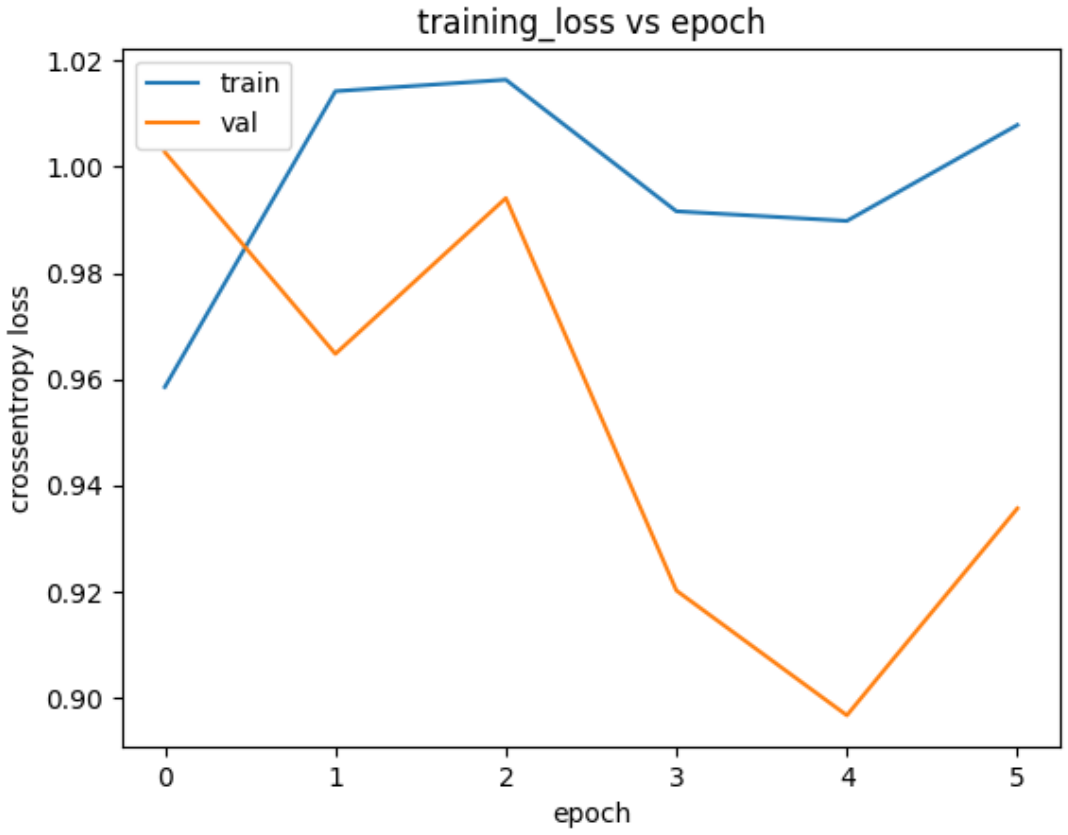

Figure 4: This is the Loss of Architecture 3

2 Baseline: VideoCLIP

2.1 Details:

In this section will give details of how we fine-tuned VideoCLIP on CLEVERER.

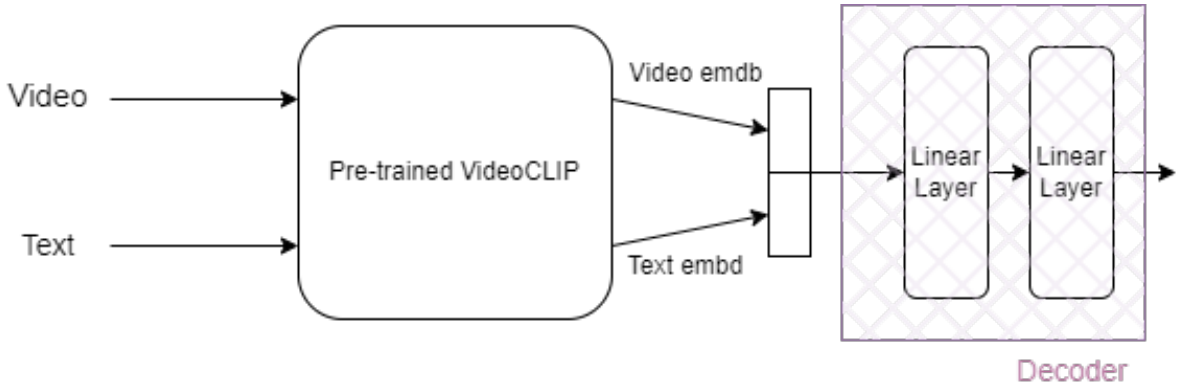

Figure 5: This is the architecture used to finetune VideoCLIP for our task

We used the version of VideoCLIP provided in github repository `https://github.com/facebookresearch/fairseq/tree/main/examples/MMPT`. Note that this already had the pretrained model for VideoCLIP so our task was to finetune the model.

- Since the model is already pretrained enough to generate better encoding of video and text.
- We added a classifier or decoder layer at the end of the vclip and now we are ready to finetune this model.
- We will take the output of complete model and use the standart cross entropy loss to optimize this.
- Note that here as well we used all the above mentioned tricks like masking outputs,

2.2 Results:

The result can be found in figure 2.2. But as visible the VideoCLIP is not training very well. The best accuracy achieved was 44.14.

3 Competitive part

We used slot former as it is for the competitive part. We only did two experiments and that too for limited dataset by removing the initial logic where only those frames were considered where the object does not appear. Due to very heavy computational requirements we only used 70 percent of the training data. We only half-trained the slot attention for 8 epochs which itself took 2.5 days. Then slotformer was trained for 41 epochs and the aloe model for 60 epochs.

The results are there in table 2

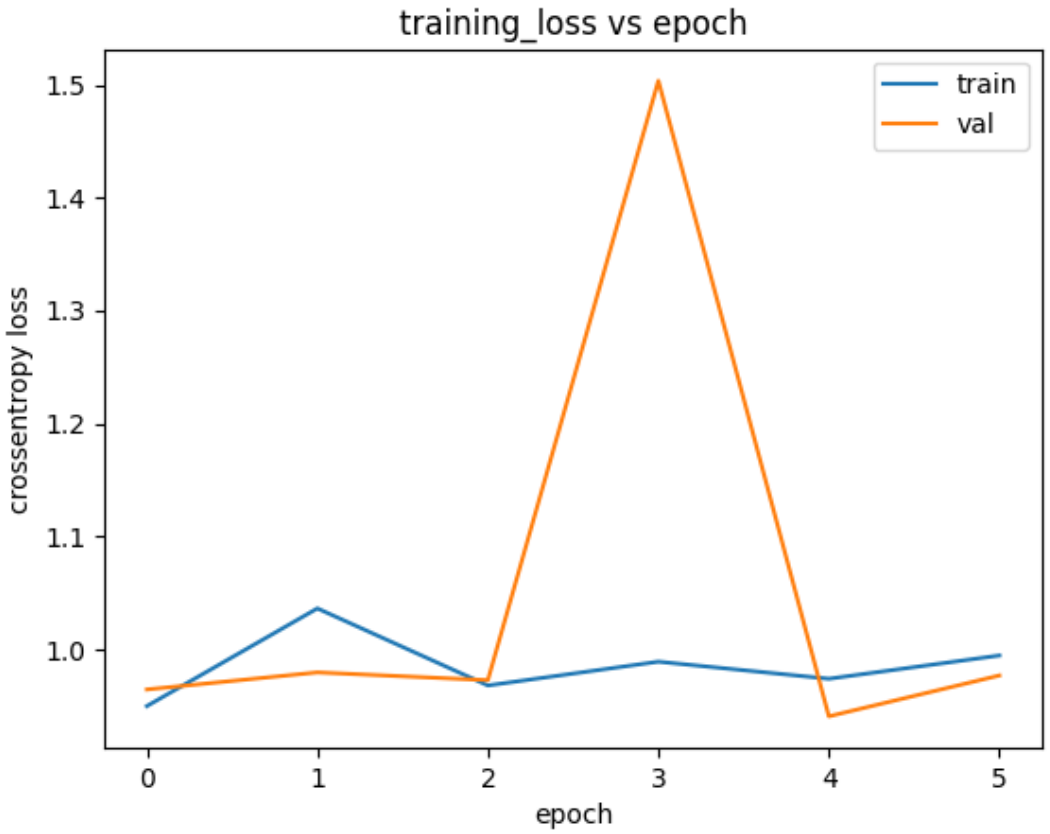

Figure 6: This is the loss curve for finetuning VideoCLIP

category	accuracy
Descriptive	76.98268335273573
Explanatory-per opt.	88.453834
Explanatory-per ques.	65.47397
Predictive-per opt.	86.55249
Predictive-per ques.	73.7507018
Counterfactual-per opt.	82.449675
Counterfactual-per ques.	52.112524
Average-per ques.	67.07996

Table 2: Execution accuary