Video Captioning using a Hybrid Transformer and RNN-based Encoder-Decoder

Alexandru-Cosmin Mihai
University Politehnica of Bucharest
alexcosmin.mihai@gmail.com

Mihai-Dan Masala
University Politehnica of Bucharest
mihaimasala@gmail.com

Dan-Teodor Poncu
University Politehnica of Bucharest
teodor.poncu@gmail.com

Traian Eugen Rebedea
University Politehnica of Bucharest
traian.rebedea@cs.pub.ro

ABSTRACT
Video captioning refers to generating a textual representation of the content presented visually in a video, a task that is necessary both for visually impaired users and for a better navigation through large video repositories. In this paper we describe a model that employs the heavily pretrained CLIP Vision Transformer (ViT) [9, 5] and the GPT2 language model [10] and discuss several approaches of combining the two models for video captioning. Our method relies on a novel hybrid model that combines the pretrained CLIP ViT with LSTM networks, outperforming CLIP + GPT2 models. We validate our method on the popular MSVD [4] and MSR-VTT [18] datasets, showing the potential of the proposed model for video captioning.

Author Keywords
Video captioning; Encoder-decoder models; Vision Transformer; Recurrent Neural Networks.

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous; I.4.9. Image Processing and Computer Vision: Applications.

DOI: 10.37789/rochi.2022.1.1.10

INTRODUCTION

Video captioning refers to the process of generating a textual representation in natural language of the relevant, semantic content presented in an input video. Similarly, image captioning aims to textually describe images, but video captioning is relatively more difficult than image captioning because it faces the additional challenges introduced by a new dimension: the time. Therefore, it is necessary to also put emphasis on activity recognition and understanding to some degree and exposing causality between events.

Both image and video captioning have important applications in Human-Computer Interaction. The most important ones are improving accessibility to visual content for visually impaired persons, indexing visual data in a database by its written description, and automatically converting information from visual to textual form so that it can be employed quicker by humans in specific scenarios (e.g. extracting the steps of a recipe from a cooking video).

RELATED WORK

Classical approaches

Modern video captioning techniques follow the pattern of employing an encoder-decoder architecture that uses neural networks as its components [1, 15, 16, 17, 19, 20]. The encoder part of the model is responsible for extracting visual features from the input video which basically encode the semantics of the visual content. These encodings are then passed to the decoder which generates the sequence of words whose semantic encodings match the visual features the best, such that the resulting sentence correctly describes the input video, or in other words, it describes the same entities and events as the video does.

For the encoder, usually a pretrained, 2D or 3D Convolutional Neural Network (CNN), a Recurrent Neural Network (RNN), a Vision Transformer (ViT) or a combination of them (e.g. CNN followed by an RNN or by a ViT [5]) is used. Options for the decoder include RNNs and Transformers [13].

Pretrained Transformers

In recent years, large Transformer models with tens and hundreds of billions of parameters trained on vast amounts of data have obtained outstanding results in a wide range and domains and tasks, especially when used as language models in natural language processing. One of the most popular such models is the Generative Pre-Trained Transformer (GPT) [10] family of models which was used in various text generation tasks such as generating stories starting from an initial prompt, creating text-based video games [22] or generating code from descriptive comments [23]. Interestingly, due to the large amount of diverse data that the models were trained on, the GPT models were shown to generalize to very different tasks, even without any fine-tuning, by simply giving them the appropriate input prompt (e.g. instructing the model to translate a following input text from one language to another [24]).

Another example of architecture that employs heavily pretrained Transformers is the one introduced by CLIP [9]. The CLIP architecture consists of an image encoder in the form of a Vision Transformer (ViT) and of a textual encoder in the form of a Transformer. These two are trained in a contrastive manner on a relatively simple task: determining if an input image and label match together. By

training the two image and text encoders using losses computed based on the cosine similarity of their generated embeddings, the objective is to teach both encoders to represent their inputs in a shared latent space. Keeping in mind our task of video captioning, the image encoder of the CLIP model is very promising because it can generate embeddings in a latent space that has been specifically designed to be shared with the embeddings of the textual information. Although we might use a different textual encoder, learning a mapping from the CLIP latent (textual) embedding space to our particular textual embedding space could be easier to do than to train an image or video encoder from scratch to generate "textually-aware" visual embeddings.

An approach that employs the pretrained CLIP ViT and GPT2 for the relatively easier problem of image captioning was introduced by Mokady, Hertz, and Bermano [7]. Our work builds on top of this and tries to adapt CLIP and GPT2 to the problem of video captioning, while also introducing a better performing hybrid model in the end that uses CLIP ViT as a visual encoder and LSTMs as a textual decoder.

METHODS

Baselines

As a comparison baseline, we used an encoder-decoder architecture that follows the more classical pattern of employing CNNs and RNNs. As such, we used the Inception-ResNet-V2 [12] pretrained on ImageNet [11] as the visual encoder in combination with an LSTM encoder and an LSTM decoder with attention. The 2D CNN extracts visual features from several equally distanced frames from the input videos. The number of selected frames is the median length of all videos in the respective dataset (in our case, 240 for MSVD and 360 for MSR-VTT). These features are then processed as a sequence by the encoding LSTM whose implicit objective is to capture temporal or order information in its hidden states at each timestep. The encoding LSTM hidden states at each timestep (i.e. at each frame) are kept and used as entries in the attention memory of the LSTM decoder. The initial hidden and cell states of the LSTM decoder are computed through two learned linear projections of the mean of all the encoder's hidden states.

Initial CLIP & GPT2 approaches

The main challenge in using CLIP and GPT2 for the problem of video captioning consists in projecting the textually aware CLIP visual embeddings to the input embedding space of GPT2 such that the visual information can be fed to GPT2 as a prefix in its input prompt.

Initially, we experiment with straightforward approaches to handle this projection. One such approach was to use a single CLIP embedding to represent the entire video: either the embedding of the middle frame in each video or the mean of the CLIP embeddings of 64 equally distanced frames from each video. When using a single CLIP embedding to represent the entire video, we experimented with the techniques introduced by Mokady, Hertz, and Bermano [7] to adapt the CLIP embeddings to GPT2 embeddings. The first technique projects the single CLIP embedding into 10 GPT2 input embeddings using a Multi-layer Perceptron (MLP). The other adaptation technique proposed by the authors that we used was to first project the CLIP embedding into 10 vectors having the dimension of the GPT2 embeddings, then concatenate these with an equal number of learned constants which also have the dimension of the GPT2 embeddings. Finally, the resulting sequence is passed through a Transformer encoder and the resulting embeddings associated with the learned constants are used in the input prompt of the GPT2.

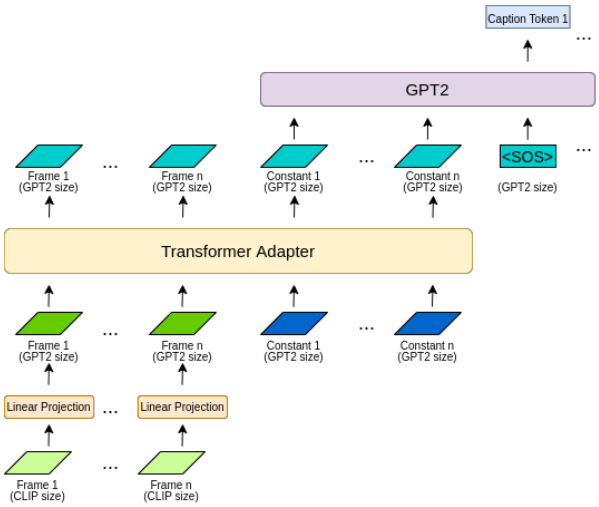

Figure 1. The frame level approach for projecting the CLIP embeddings into the GPT2 input embedding space.

Another approach that we used was to project all the CLIP embeddings of all the selected frames from a video through an MLP and such obtain a GPT2 input embedding for each frame.

CLIP & Frame level Transformer Adapter & GPT2

Previously, we had only used an MLP as the adapter network when projecting each frame individually from the CLIP embedding space to the GPT2 input embedding space. The next step we took was to implement a Transformer encoder adapter that worked with individual frames' CLIP embeddings as inputs. We experimented with 2 ways of projecting the input frame level CLIP embeddings into the GPT2 input space with the help of this new transformer adapter, both considering 10 equally distanced frames from each input video.

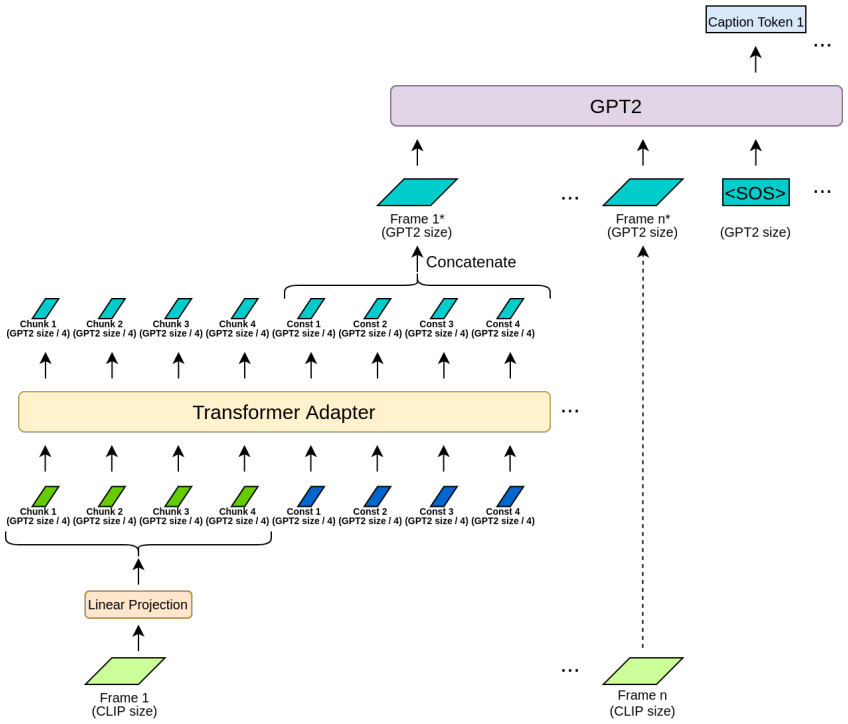

Figure 2. The frame chunk level approach for projecting the CLIP embeddings into the GPT2 input embedding space.

The first approach was to pass the entire sequence of frame level features through the adapter. Before doing so, we linearly projected each CLIP frame feature to the dimensionality of the GPT2 input space through a learned, fully connected layer, going from an embedding size of 512 to one of 768. Apart from the frame features, a sequence of learned constants – having the role of special visual "tokens"– that is equal in length and overall shape to the sequence of frame features is passed through the adapter transformer after being concatenated to the sequence of frame features, similarly to image captioning approaches [7]. These learned constants, which are used for all the input sequences of frames, are meant to help with the task adaptation of the CLIP and GPT2 networks to working together by being optimized, together with the adapter transformer, to "absorb" information from the input CLIP embeddings through self-attention in a way that generates the best translation into the GPT2 input space. In the end, the resulting projected embeddings associated with the input learned constants are used as part of the GPT2 input prompt. A visual representation of this approach can be seen in Figure 1.

CLIP + Frame-chunk level Transformer Adapter + GPT2

For the second approach of employing a transformer adapter, we first divided each frame's CLIP embedding into four chunks of equal size in the initial linear projection to the GPT2 input space, before passing them through the adapter.

This time, instead of passing a sequence of frame-level features through the adapter transformer, we pass individually each sequence of chunks associated with a frame, together with the four constants of the same shape that are trainable by the model. After each frame's chunks and constants are embedded by the transformer, for each frame we keep only the four constants, similarly to the previous approach, and concatenate them together to form the frame-level GPT2 input space embedding which is later used in the prompt of GPT2.

The thought process behind this approach is that we try to break the input CLIP embedding in multiple sub-components, then project these individual sub-components to the GPT2 input space with the hope that it is a simpler task and, finally, we put together the resulting projected sub-components into a single final embedding for the frame, with the added caveat of also employing the learned constants for helping with the task adaptation.

This idea is conceptually similar to breaking a vector into a set of base vectors and then applying a function on the base vectors instead of the "assembled" vector. A schematic of how this approach works is presented in Figure 2.

	MSVD	MSR-VTT
#Total video clips	1,970	10,000
#Training clips	1,200	6,513
#Validation clips	100	497
#Testing clips	670	2,990
Mean #frames	275.0	407.9
Median #frames	240.0	360.0
#Total captions	85,529	200,000
Mean #words	7.10	9.28
Median #words	6.0	8.0

Table 1. Statistics of the MSVD and MSR-VTT video captioning datasets.

A hybrid approach: CLIP & LSTMs

Finally, after obtaining improved results with the CLIP & transformer adapter & GPT2 model, we decided to take a step back and try to combine the two base approaches at the end of the spectrums: on one side, using simpler models with an LSTM encoder and an LSTM decoder with attention, and, on the other side, using extensively pretrained large transformer models. Thus, we introduce a hybrid model consisting of the pretrained CLIP Vision Transformer, an LSTM visual encoder, and an LSTM textual decoder.

DATASETS

We report the results of our experiments on the MSVD and MSR-VTT video captioning datasets. Both are generalist datasets that capture a wide variety of actors, objects, actions and environments. A quantitative description of the two datasets is presented in Table 1. The MSR-VTT has more samples and, on average, its videos and captions are longer than the MSVD dataset.

RESULTS

We evaluate our proposed methods using popular natural language metrics in the form of BLEU4 [8], METEOR [3], ROUGE-L [6] and CIDEr [14].

Initial CLIP & GPT2 approaches

In Table 2 we present the results obtained on the MSVD validation set by the initial, more simplistic approaches to adapting the CLIP and GPT2 models to the problem of video captioning.

The best performing model among these initial pretrained transformers models is the one using a transformer as the adapter that projects the mean of the CLIP embedding of 64 equally distanced samples and which also fine-tunes the GPT2 model during training. Considering for comparison the model which uses the middle frame of a video as the single representative of a video, this best performing model brings an improvement of 16.68% in BLEU4, 9.14% in METEOR, 5,21% in ROUGE-L, and 41.05% in CIDEr.

Adapter	Frames	Trained	BLEU4	METEOR	ROUGE-L	CIDEr
MLP	Uniform 64 mean	adapter	0.2368	0.2700	0.5337	0.2260
MLP	Uniform 10	adapter + GPT2	0.2618	0.2873	0.5506	0.2606
Transf.	Uniform 64 mean	adapter + GPT2	**0.2818**	**0.2899**	**0.5625**	**0.2903**
Transf.	Middle frame	adapter + GPT2	0.2415	0.2656	0.5346	0.2058

Table 2. MSVD validation set metrics for the initial models using CLIP & GPT2 architectures.

The models which use the mean of different frame features perform better than the middle frame baseline, suggesting that even this relatively crude method of aggregating information from the video is beneficial, although temporal information is lost in the process.

It is also worth noting that the second best performing model is the one using the distinct CLIP features of 10 equally distanced frames from the input video, without computing their mean. Given that this model uses the simpler MLP as the adapter network (although with a simpler task of projecting a single embedding to a single embedding, not one to many), this is a promising result that suggests that this approach makes better use of temporal information in a beneficial way for video captioning.

MSVD validation set

In Table 3 we present the results of the most relevant models we experimented with on the MSVD validation set.

The initial model that uses both pretrained Transformers and adapts the mean of all the CLIP embeddings to the GPT2 input embedding size using a Transformer adapter performs poorly, even in comparison with the classical baseline that employs Inception-ResNet-V2 in combination with an encoding video LSTM and a decoding textual LSTM and therefore we dropped it from our next experiments on the MSR-VTT dataset. It is worth noting though that this simplistic model manages to obtain the best METEOR score across all our models. As such, in comparison with the classical baseline, this initial adaptation of CLIP and GPT2 obtains the following relative results: -40.76% BLEU4, +16.94% METEOR, -18.07% ROUGE-L, -61.26% CIDEr.

Comparing the frame-level and chunk-level improved models, we see that the chunk-level adapter model performs better with respect to BLEU4, with a favorable difference of 4.47%, and with respect to METEOR, with an increase of 3.33%. The frame-level model performs better on ROUGE-L with a slight positive difference of 1.04% and with a larger positive difference of 14.33% on CIDEr.

Model	BLEU4	METEOR	ROUGE-L	CIDEr
IRN-V2 + LSTM enc + LSTM dec	0.4757	0.2479	0.6866	0.7493
CLIP + Mean Transf. Adapt. + GPT2	0.2818	**0.2899**	0.5625	0.2903
CLIP + Frame-level Transf. Adapt. + GPT2	0.4628	0.2766	0.7296	0.9485
CLIP + Chunk-level Transf. Adapt. + GPT2	0.4835	0.2858	0.7221	0.8296
Hybrid CLIP + LSTM enc + LSTM dec	**0.5685**	0.2756	**0.7391**	**0.9662**

Table 3. MSVD validation set results for our most relevant models.

The relatively small differences between performances of the improved frame-level and chunk-level models, apart from not showing a clear superior approach, also makes them both good candidates for evaluation on the test set of MSVD.

When compared to the previously introduced CLIP + frame/chunk-level adapter + GPT2 models, the hybrid approach performs better across all metrics on the evaluation set, except for the METEOR metric, where the chunk level CLIP + GPT2 model still performs better. As such, we see for the hybrid model, in comparison with the best metrics obtained by any CLIP + GPT2 model: 17.58% increase in BLEU4, 3.57% decrease in METEOR, 1.30% increase in ROUGE-L and 1.87% increase in CIDEr.

It is also worth mentioning that training the best performing hybrid model took significantly less time when compared to the CLIP + GPT2 alternatives. On average, training the hybrid model for an epoch on the MSVD training set using a single RTX 3080 mobile GPU took 187 seconds, which, when compared to the average of 860 seconds per epoch of the frame level CLIP + GPT2 model, amounts to a reduction of 78.26% in training time.

MSVD test set

In Table 4 are presented the results of our most relevant models and of some of the top models from the literature on the MSVD test set.

Compared to the classical baseline, the best performing model overall that employs in any way pretrained transformers, namely the hybrid, CLIP + LSTM model, performs better across all evaluation metrics on the test set, except for METEOR. As such, compared to the best non-transformer model, we see: 15.09% increase in BLEU4, 12.45% decrease in METEOR, 4.24% increase in ROUGE-L and 11.73% increase in CIDEr.

It is also worth noting the improvement brought by the refined CLIP + GPT2 models, when compared to the best model from the first iteration of experiments on this architecture, namely the CLIP + GPT2 model that

Model	BLEU4	METEOR	ROUGE-L	CIDEr
Yao et al., 2015 [19]	0.4190	0.2960	-	0.5160
Yu et al., 2016 [20]	0.4990	0.3260	-	0.6580
Ballas et al., 2015 [2]	0.4330	0.3160	-	0.6800
Zhang et al., 2020 [21] (SotA)	**0.5430**	**0.3640**	**0.7390**	**0.9520**
IRN-V2 + LSTM enc + LSTM dec	0.4519	0.2444	0.6592	0.6561
CLIP + Mean Transf. Adapt. + GPT2	0.2450	0.2650	0.5312	0.2042
CLIP + Frame-level Transf. Adapt. + GPT2	0.4431	0.2656	0.6925	0.7409
CLIP + Chunk-level Transf. Adapt. + GPT2	0.4384	0.2756	0.7024	0.7229
Hybrid CLIP + LSTM enc + LSTM dec	0.5201	0.2693	0.7074	0.7331

Table 4. MSVD test set results for our best performing models and for some of the top models from the literature (best values are in bold, best values among our models are underlined).

computed the mean of all the CLIP embeddings associated with the different input frames, then used a Transformer adapter to project this single mean embedding into a prefix for the GPT2 prompt. This mean computing approach obtained better results than the per individual frame projection one using a multilayer perceptron instead of the transformer for the adapter network. Compared to this initial CLIP + GPT2 best model, our best metrics for any CLIP + GPT2 models were improved as such: 80.86% increase in BLEU4, 4.0% increase in METEOR, 32.23% increase in ROUGE-L and 363% increase in CIDEr for the MSVD test set.

With respect to the other models in the literature, we managed to obtain better results across multiple metrics than the earlier state of the art (SotA) models, but when compared to the most up to date SotA on the MSVD dataset we could find, namely the method proposed by Zhang et al. [21], our best overall model – the hybrid CLIP + LSTM one, is still behind it. As such, our best model has: a 4.22% disadvantage on BLEU4, a 26.02% disadvantage on METEOR, a 4.28% disadvantage on ROUGE-L and a 22.30% disadvantage on CIDEr.

It is worth noting that the aforementioned SotA model uses a complex approach regarding the embedding of visual features by combining the features extracted by multiple 2D and 3D CNNs, as well as the information extracted about objects detected in the input by a pretrained object detector. These aspects make the visual embedding process resemble a pseudo-manual feature engineering approach which would make, for example, the execution of such a model in near real-time very difficult, as the execution path seems to be longer and going through multiple submodules. This

Model	BLEU4	METEOR	ROUGE-L	CIDEr
IRN-V2 + LSTM enc + LSTM dec	0.3815	0.2076	0.5738	0.3509
CLIP + Frame-level Transf. Adapt. + GPT2	0.3616	0.2312	0.5892	0.3892
CLIP + Chunk-level Transf. Adapt. + GPT2	0.3480	0.2304	0.5845	0.4015
Hybrid CLIP + LSTM enc + LSTM dec	**0.4250**	**0.2356**	**0.6154**	**0.4815**

Table 5. MSR-VTT validation set results for our most relevant models.

approach also implies the need for a potentially larger memory capacity in order to store the weights of the different networks as well as their gradients throughout the training process, making it less feasible to train and use with limited resources.

MSR-VTT validation set

In Table 5 we show the results of our models on the MSR-VTT validation set.

We can observe that across all metrics there is a general decrease in values. This observation is also true for the SotA models from the literature presented in the following section. This suggests that not only MSR-VTT is a larger dataset than MSVD, but that it is overall a more difficult dataset for the video captioning task.

The hybrid model is, once again, the best performing model overall from our proposed models. Compared to the best metrics obtained by any of the model that employs CLIP and GPT2, the hybrid model obtains: 17.53% increase in BLEU4, 1.90% increase in METEOR, 4.45% increase in ROUGE-L and 19.93% increase in CIDEr. Compared to the baseline classic architecture model, namely the one based on the Inception-ResNet-V2 encoder, the advantages are also significant for the hybrid model: 11.40% increase in BLEU4, 13.49% increase in METEOR, 7.25% increase in ROUGE-L and 37.22% increase in CIDEr.

MSR-VTT test set

In Table 6 are presented the results of our most relevant models and of some of the best models from the literature on the MSR-VTT test set.

As was the case when presenting the results on the MSVD test set, on the MSR-VTT test dataset, our hybrid model, employing the CLIP visual embeddings and an LSTM video encoder and an LSTM textual decoder, does not manage to surpass the existing SotA model. The differences compared to the SotA [21] are: a 4.54% disadvantage on BLEU4, a 21.01% disadvantage on METEOR, a 2.54% disadvantage on ROUGE-L and a 10.29% disadvantage on CIDEr.

Model	BLEU4	METEOR	ROUGE-L	CIDEr
Wang et al., 2018 [17]	0.3813	0.2658	-	-
Wang et al., 2019 [16]	0.4130	0.2870	0.6210	**0.5340**
Zhang et al., 2020 [21] (SotA)	**0.4360**	**0.2880**	**0.6210**	0.5090
IRN-V2 + LSTM enc + LSTM dec	0.3562	0.2010	0.5658	0.3540
CLIP + Frame-level Transf. Adapt. + GPT2	0.3570	0.2265	0.5856	0.3983
CLIP + Chunk-level Transf. Adapt. + GPT2	0.3437	0.2268	0.5811	0.3953
Hybrid CLIP + LSTM enc + LSTM dec	0.4162	0.2275	0.6052	0.4566

Table 6. MSR-VTT test set results for our best models and for top models from the literature (best values are in bold, best values among our models are underlined).

MSVD qualitative analysis

In Figure 3, relevant frames from a sample video in the MSVD validation set are presented and, in Table 7, the captions generated by our top performing models, together with some human, ground truth annotations are presented.

We can observe by looking at these predictions that none of the models manage to generate captions that correctly describe the complete semantics of the video, but all of them generate captions that are plausible from a grammatical point of view.

The baseline classical model based on the Inception-ResNet-V2 encoder is the only one which manages to correctly identify the age and gender of one of the actors in the scene, namely the boy. It predicts that the boy is eating, which, although incorrect, suggests that the model is aware

Figure 3. Frames extracted from a sample video in the MSVD validation set presenting a boy and a woman playing catch with a small pumpkin [4].

Model	Caption
IRN-V2 + LSTM enc + LSTM dec	“a boy is eating”
CLIP + Frame-level Transf. Adapt. + GPT2	“a girl is doing a dance”
CLIP + Chunk-level Transf. Adapt. + GPT2	“a girl is talking”
Hybrid CLIP + LSTM enc + LSTM dec	“a girl is dancing”
Human 1	“a boy and woman are tossing a pumpkin”
Human 2	“a woman and child is tossing a pumpkin to each other”
Human 3	“people are throwing a pumpkin”

Table 7. Predictions of our most relevant models and 3 of the human annotated captions for the video presented in Figure 3.

of the pumpkin in the video, or at least that some eatable object is present.

All the CLIP + GPT2 models, both the frame level and the chunk level one, and the hybrid one employing CLIP and LSTMs, fail to recognize all the actors in the scene, the correct action, and the main object of interest (the pumpkin). They all mention that there is a girl in the video. This result might be explained by the models confusing the little boy with a girl, which might be caused by an implicit bias in the dataset used to pretrain the CLIP visual encoder, namely that children with longer haircuts, such as the boy in this video, are more probable to be females. If this is the case, it is worth noting that the models correctly recognized the age of the child and therefore used the word "girl" instead of "woman".

All the models recognized semantically reasonable actions, such as talking or dancing, but none of them managed to capture the true meaning of the motions presented in the input video. The frame level CLIP + GPT2 model and the hybrid, CLIP + LSTMs model both recognized the same action of dancing, but the GPT2 decoder generated a longer sentence describing the same thing, which might be an indication of the more expressive or "talkative" nature of the GPT2 model, compared to the relatively simpler LSTM decoder.

MSR-VTT qualitative analysis

Figure 4 presents frames from one of the validation videos in the MSR-VTT validation dataset and Table 8 showcases the captions generated by our most relevant models given this video as input.

Only the model employing the CLIP visual encoder, the frame level adapter and the GPT2 decoder, as well as the hybrid model manage to generate a correct caption, namely by identifying the correct action taking place. The other two

Figure 4. Frames extracted from a sample video in the MSR-VTT validation set presenting multiple men running on a track [18].

models recognize that there is a sport event presented in the video, but fail to correctly identify its kind, misclassifying it as a soccer game, which can be explained by the background consisting of a grass field and spectator seats which resemble a soccer field.

Model	Caption
IRN-V2 + LSTM enc + LSTM dec	“a soccer game is played”
CLIP + Frame-level Transf. Adapt. + GPT2	“**a group of runners are running on a track**”
CLIP + Chunk-level Transf. Adapt. + GPT2	“a soccer team is playing soccer”
Hybrid CLIP + LSTM enc + LSTM dec	“**a group of people are running on a track**”
Human caption 1	“runners in an event are running around the track”
Human caption 2	“a group of men are racing around a track”
Human caption 3	“people running in race”

Table 7. Predictions of our most relevant models and 3 of the human annotated captions for the video presented in Figure 4.

CONCLUSIONS

In this paper we introduced two ways to adapt the heavily pretrained CLIP and GPT2 Transformer-based models to the problem of video captioning, then we presented a hybrid CLIP + LSTMs model that outperformed our models based exclusively on the CLIP + GPT2 architecture or on the classical CNN + LSTMs architecture.

Although the newly introduced hybrid model does not manage to surpass the performance of the current state of the art in video captioning, it comes relatively close to it while proposing a new approach that requires no manual or

semi-manual feature engineering. Since this approach does not rely on pretrained object or action detectors, as the SotA and other well performing models in the literature do, it brings the potential to generalize better to different datasets and environments in general.

REFERENCES

1. Aafaq, N., Mian, A., Liu, W., Gilani, S. Z., and Shah, M. (2019). Video description: A survey of methods, datasets, and evaluation metrics. ACM Computing Surveys (CSUR), 52(6):1–37.
2. Ballas, N., Yao, L., Pal, C., & Courville, A. (2015). Delving deeper into convolutional networks for learning video representations. arXiv preprint arXiv:1511.06432.
3. Banerjee, S., & Lavie, A. (2005, June). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization* (pp. 65-72).
4. Chen, D. and Dolan, W. B. (2011). Collecting highly parallel data for paraphrase evaluation. In Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies, pages 190–200.
5. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.
6. Lin, C. Y. (2004, July). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out* (pp. 74-81).
7. Mokady, R., Hertz, A., and Bermano, A. H. (2021). Clipcap: Clip prefix for image captioning. arXiv preprint arXiv:2111.09734.
8. Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002, July). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics* (pp. 311-318)
9. Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In International Conference on Machine Learning, pages 8748–8763.
10. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. OpenAI blog.
11. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. International journal of computer vision, 115(3):211–252.
12. Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In Thirtyfirst AAAI conference on artificial intelligence.
13. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.
14. Vedantam, R., Lawrence Zitnick, C., & Parikh, D. (2015). Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4566-4575).
15. Venugopalan, S., Xu, H., Donahue, J., Rohrbach, M., Mooney, R., and Saenko, K. (2014). Translating videos to natural language using deep recurrent neural networks. arXiv preprint arXiv:1412.4729.
16. Wang, B., Ma, L., Zhang, W., Jiang, W., Wang, J., and Liu, W. (2019). Controllable video captioning with pos sequence guidance based on gated fusion network. In Proceedings of the IEEE/CVF international conference on computer vision, pages 2641–2650.
17. Wang, J., Wang, W., Huang, Y., Wang, L., and Tan, T. (2018). M3: Multimodal memory modelling for video captioning. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7512–7520.
18. Xu, J., Mei, T., Yao, T., and Rui, Y. (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.
19. Yao, L., Torabi, A., Cho, K., Ballas, N., Pal, C., Larochelle, H., and Courville, A. (2015). Describing videos by exploiting temporal structure.
20. Yu, H., Wang, J., Huang, Z., Yang, Y., & Xu, W. (2016). Video paragraph captioning using hierarchical recurrent neural networks. Proceedings of the IEEE conference on computer vision and pattern recognition, 4584-4593.
21. Zhang, Z., Shi, Y., Yuan, C., Li, B., Wang, P., Hu, W., & Zha, Z.-J. (2020). Object relational graph with teacher-recommended learning for video captioning. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 13278-13288.
22. GPT-3 Powers the Next Generation of Apps. https://openai.com/blog/gpt-3-apps/
23. OpenAI Codex. https://openai.com/blog/openai-codex/
24. Better Language Models and Their Implications. https://openai.com/blog/better-language-models/