Grenze International Journal of Engineering and Technology, June Issue

Video Captioning with Text and Audio Output

Sheela N[1], VaniAshok[2], Manimala S[3] and Chinmay G S[4]
[1-4]SJCE, JSS STU/CS, Mysore, India
Email: sheela_cse@sjce.ac.in, vanisj@sjce.ac.in, manimala@sjce.ac.in, chinmaypandit39@gmail.com

***Abstract*— Complex dynamics are inherent in real-world videos and the generation of open-domain video description should be capable of handling temporal structure and allow the input and output of the video description to be of arbitrary length. For this solution, we used an end-to-end sequence to sequence model for generating captions for videos. Here, we utilize Recurrent Neural Networks (RNN) especially Long Short-Term Memory (LSTM) as they are well suited for image caption generation and are among the best performing models today. In order to provide a summary of the incident in the video clip, our suggested LSTM model maps a sequence of video frames with a sequence of words using pairs of textual and video descriptions as input. By construction, our Language Model (LM) assumes both the temporal structure of the frame sequence and the sequence model of the generated sentences. We evaluate multiple iterations of the model using various video visual attributes on the Microsoft Research Video Description (MSVD) dataset in order to examine various aspects of the suggested model.**

***Index Terms*— CNN, RNN, LSTM, LM, MSVD.**

I. INTRODUCTION

The goal of the proposed work is to fill the accessibility gap in videos for visually and hearing-impaired users by evolving a deep learning model for automatic video captioning and audio generation. Despite the success of image captioning models, video processing frameworks still lack significant impact. This paper suggests using an enhanced Long Short-Term Memory network (LSTM) in a sequence-to-sequence structure to close this gap and solve the vanishing gradient issue common to Recurrent Neural Networks (RNNs). Using pre-trained Convolutional Neural Networks (CNNs), the model collects video information and produces accurate captions. These captions are then converted into audio descriptions, enhancing accessibility and allowing for better search functionalities based on combined text and audio.

The primary goal is to enhance accessibility for both visually and hearing-impaired users. The model, based on an LSTM encoder-decoder architecture, aims to generate accurate captions depicting video content for visually impaired viewers and convert these captions into audio descriptions to aid hearing-impaired viewers. This approach addresses the existing gap in video captioning solutions by ensuring both text and audio outputs, thus creating a more inclusive and searchable video experience. This method improves search results, recommendations, and accessibility. It enables better video content understanding for search algorithms and enhances user satisfaction through personalized recommendations. The system benefits various sectors like education, entertainment, and healthcare. For example, it aids learners in grasping course materials, allows viewers to fully participate in virtual worlds, and helps in the diagnosis of medical images through descriptive annotations.

Grenze ID: 01.GIJET.12.2.307_21

Current state-of-the-art methods for video captioning often employ Recurrent Neural Networks (RNN). However, RNNs face the vanishing gradient problem during training, which affects their ability to capture long-distance dependencies within the video sequences. This issue is more pronounced in video captioning than in text or audio tasks because of the need to understand the context across multiple frames. The vanishing gradient problem confines the performance of RNNs in producing accurate and fluent captions, making it a significant concern in the adaptation of RNNs for video captioning.

To address these challenges, the proposed video captioning system implements Long Short-Term Memory (LSTM) neural networks. LSTMs are designed to tackle the difficulties associated with traditional RNN models by using precisely structured gateways that selectively preserve relevant information over time while discarding irrelevant data. This approach prevents significant gradient decay, allowing LSTMs to retain crucial information across frames. The dynamic nature of LSTMs, which can manage different time links and process multiple frames simultaneously, making them highly effective for video captioning tasks. By leveraging LSTMs networks, this work aims to significantly enhance the accuracy and performance of video captioning compared to conventional RNNs methods.

II. Related Work

Initial works in video captioning involved video metadata tagging [1] and video and caption clustering to enable retrieval of videos and captions [6, 9, 17]. Subsequent approaches presented a two-step system which initially determined meaningful visual features and then converted these features to sentence level description using language models [4, 7, 14] that comprises of 'semantic parsing' which involves identifying the semantics of the target area such as the subject, verb and the object, then applying a template in order to generate the new sentence. This generally entailed training specific individual classifiers for candidate objects, actions and scenes. Probabilistic graphical frameworks are used to jointly reason over linguistic constraints and visual evidence, estimating the likely semantic constituents (subject, action, object, and scene) present in the video in order to facilitate sentence formation.

Although this helped in simplifying the problem by decoupling the content creation process from the structure realization step, it poses the problem of identifying a set of objects that are relevant and a set of actions to recognize. Furthermore, the concept of using templates to construct new sentences from characteristics does not fully explain all aspects of language usage in the particular context, such as which qualities to include and how to connect them to build an appropriate description. Instead of isolating content recognition from sentence creation, the proposed method directly aims to identify videos and generate human-provided sentences while learning a language model conditioned on visual indicators.

Some of the prototypes draw their motivation from image caption generation models that are discussed in [3, 16]. To do this they start with forming a fixed size vector or creating a small feature vector from an image using a CNN. The next step in learning involves the extraction of this vector to a function, which outputs a sentence describing the image from a sequence of words. Although, in principle, any RNN may be desirable for decoding the sequence, it inevitably results in long-term dependencies that may be detrimental to presentation. To address this problem, as LSTM models are more progressive in identifying long-term dependencies, they have been used as sequence decoders. Similarly, since we have variable size video as input, it uses LSTM where we have sequence to sequence transducers resulting the language model [13]. The authors of [15] pool the representations of the various frames after using LSTMs to capture the video description. Their approach involves feature extraction using CNN for each video frame, followed by the mean pooling across the frames to obtain a vector of features corresponding to the complete video. They then employ an LSTM which functions as a decoder for sequence and outputs a description given this vector. One significant limitation with such representation is that they can represent videos without respect to the time order of the frames and no temporal information can be harvested. In [3], the authors also produce video descriptions via an LSTM, but they use two-step approach, where a CRF algorithm is used to extract the semantic tuple which defines activity, object, tool, and location of the video, followed by LSTM which transforms this tuple into a sentence. Further, the model developed in [3] is limited to the domain of cookery videos only and hence is not as generic as ours which is focused on the generation of descriptions of videos in their natural environment. This was accomplished by another recent effort [12], which used LSTMs to predict the next frame sequence from an encoding of the previous frames. Their model resembles the language translation model in [13], where the authors use a single LSTM for text manipulation. Feeding the input text through a first LSTM to embed it into a fixed form and another LSTM to generate the translated text in another language. Rather than employing separate LSTMs for encoding and decoding, we use just one LSTM in the network, and its parameters are trained based on the data that is fed to

the network. This makes it possible for the LSTM to develop ways to create and share the weights for encoding and decoding processes. Another related work include [3] for activity classification, they use LSTMs that takes an image/flow frame representation and produces an activity class. However, our model provides captions at once after diagonalizing the concatenation of all the images of optical flow.

III. METHODOLOGY

For training video descriptions, we propose a sequence-to-sequence model that takes a series of frames. (x_1,...,x_n) as input. With the encoder being an input layer that permutes the sequence of characters through x_n, and the output layer that outputs the series of words (y_1,.., y_m) as well. Certainly the code for interpretation and output can be different each time. Due to this, we often have a lot of pictures with the tiny number of expressions. We compute the output sequences' (y_1,.., y_m) conditional probability in our model once the input sequence time series (x_1,...,x_n) is fed in as given by the equation (1)

$$p\ (y_1,...,y_m|x_1,...,x_n) \qquad (1)$$

This scenario closely resembles machine translation between two natural languages, wherein a word sequence from the source language is translated into a different word sequence specified in the destination language, word by word. As given in the Model [2, 13], there is a novel approach to meet the sequence to sequence problem confronting LSTM Recurrent Neural Network (RNN). We generalize this setting to inputs made up of video frame sequences, which helps to simplify and provide an easy-to-implement alternative to the RNN-based methods of video encoding. In the following section, we explain the details of our methodology (i.e. architecture, input and output representations for both video and sentences).

A. Sequence modelling using LSTMs

The suggested method for handling inputs and outputs of varying lengths is to encode a series of frames, one encoding step per frame, through the use of latent vector representation to represent the video content and then to decode those individual words into a sentence, again one at a time. Initially we start with the LSTM, which was originally given in [5] and build an LSTM layer based on what is presented in [18], Equation (2) illustrates how the LSTM computes a memory cell state ct, which is an encoding of all the cell's contents, and a hidden/control state ht for an input xt at time step t.

$$\begin{aligned} i_{t=}&\sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \\ f_t &= \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + b_f\right) \\ g_t &= \phi\left(W_{xg}x_t + W_{hg}h_{t-1} + b_g\right) \qquad (2) \\ c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\ h_t &= o_t \odot \phi(c_t) \end{aligned}$$

The trained parameters are denoted by weight matrices W_{ij} and biases b_j, hence in the encoding step, the input sequence X(x_1,...,x_n) is processed by the LSTM to obtain a number of hidden states (h_1,...,h_n) . In the process of decoding, it characterizes a distribution of the output sequence Y(y_1,.., y_m) as per the input sequence X as p(Y|X) as given in equation (3).

$$p(y_1,...,y_m \mid x_1,...,x_n) = \prod_{t=1}^{m} p(y_t|h_{n+t-1}, y_{t-1}) \quad (3)$$

In the case where the $p(y_t|h_{(n+t)})$ the distribution is provided as a softmax over the whole lexicon (Equation 5). It should be noted that the value of $h_{(n+t)}$ is derived by the recursion of $h_{(n+t-1)}$and $y_{(t-1)}$ as shown in Equation (2).

B. Video to Text using Sequence to Sequence approach

In contrast to [2,13], the S2VT method uses a single LSTM for both encoding and decoding, with the first LSTM encoding the input sequence into a fixed-length vector and another LSTM completing the mapping process from the vector to the sequence of outputs. The model will establish a link between the encoding and decoding variables in this way.The proposed model is a LSTM type which has only single layer of 1000 units. In our architecture, the hidden representation (ht) from LSTM layer serves as both the present input (xt) and the hidden state for subsequent time steps. It makes use of a single unified LSTM layer that handles both the corresponding video frame sequence and the output word sequence. Fig 1 depicts the S2VT model used in the proposed work.

During the training process and inference, the LSTM layer receives at the beginning a sequence of frames, which get encoded into hidden representations. The same is true for the LSTM layer, where the beginning-of-sentence tag <BOS> is presented, and the decoding task occurs to produce a series of words. During decoding stage, the model determines the log-likelihood of the forecast sentence output under the hidden state of the sequence of prior sentences and frames. Based on Equation 3 of a model with parameters θ and output sequence Y= (y_1,....,y_m),, it is written as follows in equation (4):

$$\theta^* = \sum_{t=1}^{m} \log p(y_t | h_{n+t-1}, y_{t-1}; \theta) \qquad (4)$$

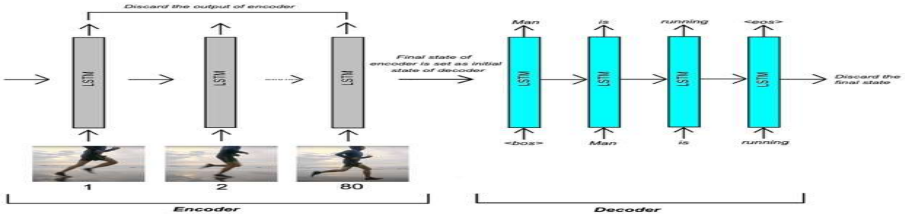

Figure 1. The S2VT model that we employed for the video description is a sequence to sequence model.

It utilizes a stacked LSTM that reads through the sequence of frames and then comes up with a sequence of words.

The optimization of the log-likelihood goes over the whole dataset by means of stochastic gradient descent. The computation of the loss is only carried out during the time when LSTM is engaged in the process of decoding. The fact that this loss is multiplied as time proceeds enables the lamination of a correct hidden state representation (hn) based on the input sequence. At the next LSTM layer, the output (z_t) of that layer is employed to generate the emitted word (y). We use a softmax to attain the probability distribution of the words y' in the vocabulary V as indicated by the equation (5).

$$p(y|z_t) = \frac{\exp\,(W_y z_t)}{\sum_{y^1 \in v} \exp\,(W_{y^1} z_t)} \qquad (5)$$

We observe that, in the decoding step, the visual frame code of the first LSTM layer is nothing but a lengthy sequence of zero vectors which serve as padding input. We also need an explicit end-of-sentence tag (<EOS>), so that each sentence can be ended, as only in this way the model can determine a distribution on sequences of different length. At test time, in each de-coding step we select the word y_t whose probability after the softmax (in Equation 5) is maximum until it produces the <EOS> token.

C. RGB frames

Similar to earlier LSTM-based image captioning systems [3, 16] and video-to-text systems, we use a convolutional neural network (CNN) to collect input images and feed the output of the most recent layer as input to the LSTM unit. The 16-layer VGG model is used in this work to report the results [11]. The subset of the ImageNet 1.2M image ILSVRC-2012 object classification dataset is used to test our CNNs [10]. After that, the CNN receives this. From the input (x_t) to the LSTM layer, the input is made up of lower dimension features. We train a new linear embedding of the features after removing the previous final fully-connected classification layer. The embedding weights are learned concurrently with the LSTM layers during training.

D. Text Input

The intended word sequence is denoted in terms of one-hot vector encoding (1-of-N encoding, where N is the size of available vocabulary). Our study context is to follow the one we have with frame features embedded. During the back propagation, we learned the parameters of the linear transformation we applied to the data inputs. The tensor in the vector with the encapsulated word is connected to the output of the LSTM layer, and the latter (ht) is input to the next LSTM layer. A softmax is applied on the entire LSTM's output during which point of time an exhaustive map of the vocabulary is used (which is in Equation 5).

IV. Experimental Setup

The experimental set up of the suggested work is as shown in Fig 2.

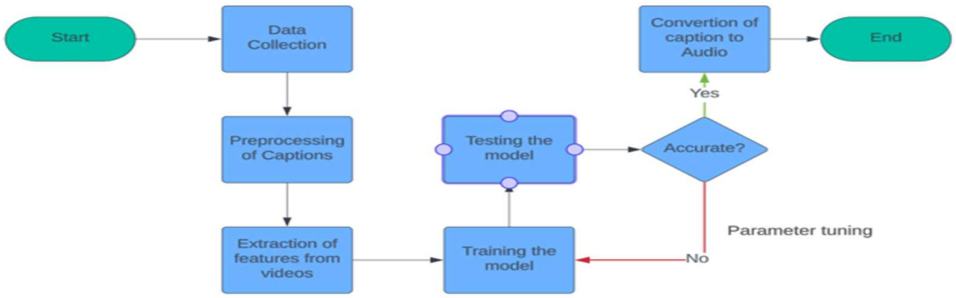

Figure 2. Experimental setup

A. Data Acquisition

The present study is based on MSVD dataset that was chosen from Microsoft collection of videos. The questionnaire dataset encompasses 1450 thirty-second YouTube clips labelled for training with a number of 100 held-out videos allocated for test. An ID is assigned to every video, using which the video could be identified with approximately 15 to 20 descriptive captions associated with a particular ID. Upon downloading the dataset, it's organized into two main folders: training data and testing data. Within every top-level folder is hidden the sub-folders that are identified by the names of the videos, which consist of clips assigned for training or testing. In these video subfolders we also encounter the feat subfolders which, in short, stand for features. It is inside these subfolders artists present their music videos. Furthermore, there are files called training label and testing label for storing the captions for all the frames corresponding to each video ID. This parsing can be done and read with regard to the JSON files.

B. Data Pre-processing

First and foremost, the captions are prepared that will be displayed in sync with the movie trailers. This will be the root element that will come out to describe the clip by means of the information extracted from the video. We shall extend that by placing bof (beginning of file) and eof (end of file) to attach at the start and end of video captions. The sentences are also surface-padded to have equal words and sentence lengths (padding). The captions are padded between 6-10 words and also cleaning is facilitated by removing punctuations, stop words, or any other irrelevant elements from the captions.

C. Data Augmentation

Considering the scarceness of video data, a data augmentation strategy is applied. The same video with varying captions is utilized to expand the dataset. The strategy intends to increase how diverse and resilient the training data is by having various textual descriptions of the identical visual material. Using various captions for each particular video, our model is capable of generalizing better, and it gets the hang of language and the same visual context. Consequently, the number of data increases, and the model will be able to obtain richer features and make captioning of video tasks better by data augmentation. Besides, it is also advantageous in the context of minimizing the probability of overfitting by giving the model a variety of linguistic expressions and keeping the overall visual content constant.

D. Extraction of Features from Videos

An information extraction component should be included which recognizes the video features that are the most critical and connect them with a concise summary of the visual features of that video is necessary. We are using CNNs to tackle this problem. The key process concerning output of features of video consists of a huge amount of pictures or frames, that stodges together. To convey the crucial moments of each video, every frame is of each of the dataset videos. This heterogeneity being the case, a homogenous approach is picked where only from 80th frame of each video is selected to ease the process which is uniform. Here, these frames are passed to a pre-trained VGG16 network that results in 4096 features for each frame. The next step would be concatenating all the frames dimensional output from all 80 frames. Using the resultant frame of shape (80, 4096). This shows that there are 80 frames and 4096 extracted features in each frame. In this method, every frame of each video is done with a Vectorizer and NumPy arrays. Since pre-extracted features are efficiently provided by the dataset, it logically indicts the follow-up phase of the proposed work

E. Training the Model

At this stage, a training process of captioning model involves the use of prepared captions accompanied by extracted features from the video. An encoder-decoder model, a component of a neural network system is utilized to study the relationship between video and their captions. The decoder part of the architecture extracts features that are informative from the input video data for the encoder, as a result, captions are created based on these features. The model applies the iterative optimization technique, for example: backpropagation and gradient descent to bring the difference between the predicted captions and the ground truth captions (the actual transcription of the video) to minimum and hence improving the model's ability to accurately describe video content. The encoder-decoder model's training architecture is as shown in Fig 3.

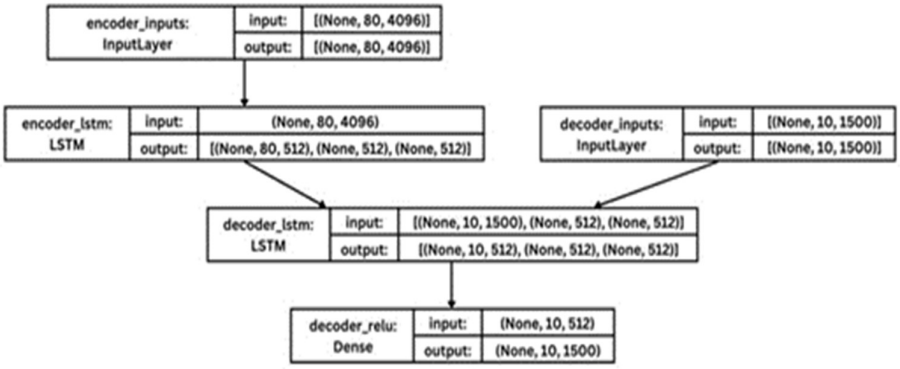

Figure 3. Encoder - Decoder Model Architecture for Training the model.

Testing the Model: Following model training, the model's accuracy, generalization ability and performances are examined for the dataset which has separated it from the testing dataset. By consequently comparing the model predictions with the captions of the ground truth during the test pattern, directions are well-defined as for which the model lacks in robustness accuracy. It builds a dataset needed for adjustments which ensures the avoidance of cases where the model is not able to generalize unseen data.

Parameter Tuning: Parameter tuning is the process of continuous fine-tuning of those settings so as to give the model more flexibility in improving its performance. Over and over again the hyper parameters including the learning rate, optimizer, and network architecture are tuned, improved, and the model is re-trained, leading to the finding of the configuration that gets the highest accuracy of predictions, best generalization while still holding minimum possible training time and computational resources. This kind of iterative adjustment to parameters will guarantee that the model that it produced will give the best output when working on the captioning task mentioned.

V. Result Analysis

Video captioning that involves writing captions for videos has been usually assessed through the performance by automatic systems, which may determine the written captions' relevancy and correctness. The existing measures that have been suggested for assessing the video captioning models' performance are as follows. Each of the metrics is different and offers smart impression of numerous aspects of captioning. Key metrics of video captioning research that are currently available are discussed, highlighting the parameters of some of them based on their advantages and disadvantages, as well as the possibility of using them.

Evaluation metrics: There are different metrics available for evaluation of video captioning model. Each metrics are different and offers smart impression of numerous aspects of video captioning. We are using BLEU (Bilingual Evaluation Understudy) as a metric of evaluation in our project as it is simple, efficient, and robust to variations, standardization and benchmarking.

BLEU Metric details: the result is multiplied by an exponential brevity penalty factor after computing the geometric mean of the test corpus adjusted precision scores. At the moment, the only text normalization performed prior to the precision calculation is case folding. Using n-grams of length N and weights w_n of length N that add up to one, First, the updated n-gram precisions' p_ngeometric mean is computed. Next, we designate r as the effective reference corpus length and c as the candidate translation length. . Equations (6) and (7) are used to calculate the Brevity Penalty BP.

$$BP = \begin{cases} 1 & if\ \ c > r \\ e^{\frac{1-r}{c}} & if\ c \leq r \end{cases} \qquad (6)$$

Then, $$BLEU = BP.\exp(\sum_{n=1}^{N} w_n log p_n) \qquad (7)$$

N = 4 and uniform weights wn = 1/N are our default parameters in the baseline and the number of perfect translations of a given source sentence is usually big. Such translations can differ in terms of the words used or the order of the words used even when they translate the same words but still humans are able to distinguish a good translation and a bad one easily. The paper [8] embraces the ensemble strategy and that it is based on the integration of different data such as MSVD, MPII-MD and M-VAD, finally leading to obtaining an average of 66.125 accuracy. In our proposed work, accuracy of 67.38 is obtained providing only MSVD dataset, as other two datasets were not available to get the study done. Regardless of the fact that no additional datasets used, the approach introduced here indicates competitive performance turns out the issues of data availability and approximation are still of paramount importance in video captioning research. Table 1 shows the accuracy obtained for different models. Fig 4 and Fig5 shows the Graph of Epochs vs Accuracy and Graph of Epochs vs Loss respectively and Fig 6, Fig 7 and Fig 8 shows the snapshot of the output of video captioning with User Interface.

TABLE I: PERFORMANCE OF DIFFERENT MODELS

Model	B@1	B@2	B@3	B@4	Average
LSTM	82	70.5	60.42	56.6	67.38

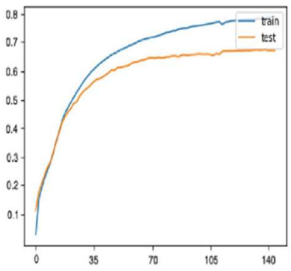

Fig4.Graph of Epochs vs Accuracy

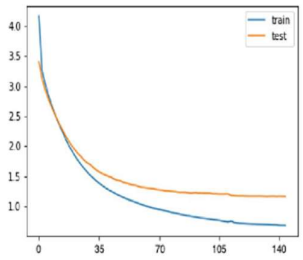

Fig 5. Graph of Epochs vs Loss

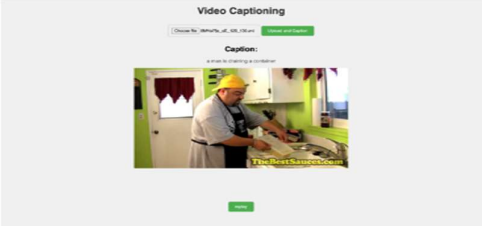

Fig 7. Video Captioning with UI

Figure 6. Results obtained

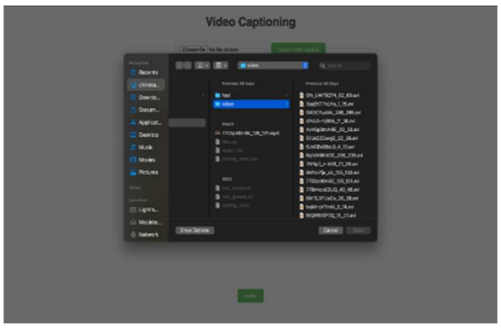

Fig 8. Video Captioning with UI

VI. CONCLUSIONS

This paper aimed at developing a new framework for video description. Compared to the related work we build descriptions based on the sequence-to-sequence architecture which contains a sequence of frames to read, through them a sequence of words is generated. It helps us in handling variable lengths of inputs and outputs at the same time and enjoy the feature of having some sort of time dependency built within the model. In contrast to prior approaches, we set spatial and temporal correlation as our primary focus. Regardless of the fact that no additional datasets used, the approach introduced here indicates competitive performance turns out the issues of data availability and approximation are still of paramount importance in video captioning research. However, our Future work will focus on gathering different datasets to further evaluate the model. This sets the stage for the next phase of technical validation and model comparison.

REFERENCES

[1] Yang, Y., Zhou, J., Ai, J., et al. "Video captioning by adversarial lstm." IEEE Transactions on Image Processing 27.5 (2018): 5600-5611.

[2] Cheng, X., Lu, J., Feng, J., Yuan, B., & Zhou, J. "Scene recognition with objectness."Pattern Recognition 74 (2018): 474-487.

[3] Lee, S., & Kim, I. "Multimodal feature learning for video captioning." Mathematical Problems in Engineering 2018 (2018).

[4] Yan, C., Tu, Y., Wang, X., et al. "Stat: spatial-temporal attention mechanism for video captioning." IEEE Transactions on Multimedia 22.1 (2019): 229-241.

[5] Zhang, Z., Shi, Y., Yuan, C., et al. "Object-relational graph with teacher-recommended learning for video captioning." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 13278-13288.

[6] Gao, L., Lei, Y., Zeng, P., Song, J., Wang, M., & Shen, H. T. "Hierarchical representation network with auxiliary tasks for video captioning and video question answering." IEEE Transactions on Image Processing 31.1 (2021): 202-215.

[7] Yang, B., & Zou, Y. "Clip meets video cautioner's: attribute-aware representation learning promotes accurate captioning." arXiv preprint arXiv: 2111.15162 (2021).

[8] Dosovitskiy, A., Beyer, L.Kolesnikov, A. Weissenborn, D.Zhai, (2022). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

[9] Carion, N., Touvron, H., Cord, M., & Mishkin, D. (2023). End-to-end object detection with transformers. ArXiv preprint arXiv:2003.11986.

[10] K. Cho, B. van Merrie¨nboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. arXiv:1409.1259, 2020.

[11] I.Sutskever,O.Vinyals,andQ.V.Le.Sequence to sequence learning with neural networks. In NIPS, 2024.

[12] S.Hochreiter and J. Schmidhuber. Long short-term memory.Neural computation, 9(8), 1997

[13] W.Zaremba and I. Sutskever. Learning to execute.arXiv:1410.4615, 2024.

[14] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recur- rent convolutional networks for visual recognition and de- scription. In CVPR, 2025.

[15] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. CVPR, 2025.

[16] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko. Translating videos to natural language using deep recurrent neural networks. In NAACL, 2020.

[17] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville. Describing videos by exploiting temporal structure. arXiv:1502.08029v4, 2023.

[18] N. Krishnamoorthy, G. Malkarnenkar, R. J. Mooney, K. Saenko, and S. Guadarrama. Generating natural-language video descriptions using text-mined knowledge. In AAAI, July 2023.