electronics

Article

Deep Learning-Based Context-Aware Video Content Analysis on IoT Devices

Gad Gad [1,*], Eyad Gad [2], Korhan Cengiz [3,4], Zubair Fadlullah [1,5] and Bassem Mokhtar [3,6]

[1] Department of Computer Science, Lakehead University, Thunder Bay, ON P7B 5E1, Canada; zubair.fadlullah@lakeheadu.ca
[2] School of Engineering and Applied Sciences, Nile University, Giza 12677, Egypt; e.gad@nu.edu.eg
[3] College of Information Technology, University of Fujairah, Fujairah 1207, United Arab Emirates; korhancengiz@uof.ac.ae (K.C.); bmokhtar@alexu.edu.eg (B.M.)
[4] Department of Electrical-Electronics Engineering, Trakya University, Edirne 22030, Turkey
[5] Thunder Bay Regional Health Research Institute (TBRHRI), Thunder Bay, ON P7B 7A5, Canada
[6] Department of Electrical Engineering, Faculty of Engineering, Alexandria University, Alexandria 21544, Egypt
* Correspondence: ggad@lakeheadu.ca

Abstract: Integrating machine learning with the Internet of Things (IoT) enables many useful applications. For IoT applications that incorporate video content analysis (VCA), deep learning models are usually used due to their capacity to encode the high-dimensional spatial and temporal representations of videos. However, limited energy and computation resources present a major challenge. Video captioning is one type of VCA that describes a video with a sentence or a set of sentences. This work proposes an IoT-based deep learning-based framework for video captioning that can (1) Mine large open-domain video-to-text datasets to extract video-caption pairs that belong to a particular domain. (2) Preprocess the selected video-caption pairs including reducing the complexity of the captions' language model to improve performance. (3) Propose two deep learning models: A transformer-based model and an LSTM-based model. Hyperparameter tuning is performed to select the best hyperparameters. Models are evaluated in terms of accuracy and inference time on different platforms. The presented framework generates captions in standard sentence templates to facilitate extracting information in later stages of the analysis. The two developed deep learning models offer a trade-off between accuracy and speed. While the transformer-based model yields a high accuracy of 97%, the LSTM-based model achieves near real-time inference.

Keywords: video content analysis; LSTM; transformer-based model; video captioning; Internet of Things (IoT)

check for updates

Citation: Gad, G.; Gad, E.; Cengiz, K.; Fadlullah, Z.; Mokhtar, B. Deep Learning-Based Context-Aware Video Content Analysis on IoT Devices. *Electronics* **2022**, *11*, 1785. https://doi.org/10.3390/electronics11111785

Academic Editors: Leonardo Galteri, Claudio Ferrari and Stefanos Kollias

Received: 3 May 2022
Accepted: 30 May 2022
Published: 4 June 2022

1. Introduction

Video content analysis (VCA) is the task of processing a sequence of frames to extract temporal and spatial information. Video captioning is a category of VCA where the goal is to generate the text description of an input video. Generating natural language-based captions is particularly useful if the output caption is received directly by a human. However, when the generated description is itself an input to another stage in an automated pipeline, restricting captions' sentence structure to simple templates reduces their complexity, making modeling them and applying postprocessing in later stages easier.

Video captioning tasks can be divided into two categories: single sentence captioning and multi sentence (dense captioning). While in single sentence captioning the goal is to summarize the input video in one abstract sentence, dense captioning systems are more flexible since they can describe multiple events simultaneously or generate a more detailed description of the input video. This work presents a dense captioning framework that generates multiple template-based captions describing multiple events of an input video.

Current video captioning approaches are based on an encoder–decoder architecture that combines convolutional neural networks (CNN) for spatial features extraction, and recurrent neural networks (RNN) for temporal encoding and decoding. Transformers [1] are also widely used due to their ability to learn patterns with long dependencies.

Due to the high dimensionality of videos and their captions, supervised video-to-text datasets are usually huge, open-domain datasets and are therefore not suitable for training lightweight models in a resource-limited environment. These datasets include the MPII Movie Description corpus (MPII-MD) [2]; the Microsoft Video Description dataset (MSVD) [3]; the Montreal Video Annotation Dataset (M-VAD) [4]; and the Microsoft Research Video to Text (MSR-VTT) [5]. Additionally, M-VAD and MSR-VTT datasets were annotated using the Amazon Mechanical Turk (AMT); therefore, annotations have inconsistent sentence structures. Generating natural language-based descriptions works best when the output is presented to a human user. However, standardizing captions' sentence structures (language model) in the intermediate stages of the analysis pipeline is important for the efficient postprocessing of the generated captions because every position in the sentence will have a fixed meaning.

The proposed framework can be used for classroom/lab monitoring, as shown in Figure 1, where a smart camera monitors the classroom and processes the collected frames in real-time to generate a description of the scene in a predefined sentence template. The generated caption is further processed to extract names, dates, and entities based on the application-specific metadata (context) that is given to the system. Finally, the information is stored in a database.

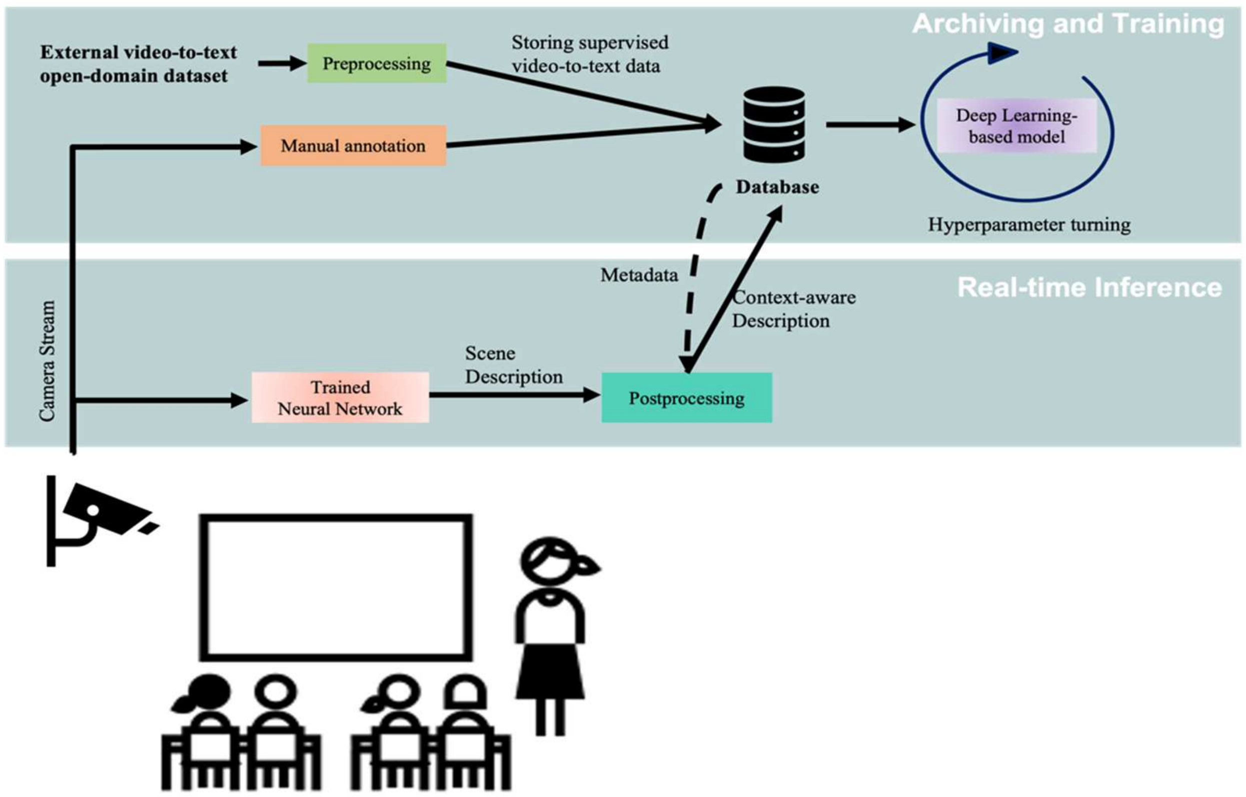


Figure 1. Overview of the proposed framework applied in a classroom monitoring application.

The rest of this paper is structured as follows: In Section 2, recent related work is discussed. The methodology, including data processing and deep learning-model details are presented in Section 3. Experimental results are given in Section 4. Finally, our conclusion is discussed in Section 5.

2. Related Work

Following an era of combining traditional object/action detection methods to describe videos, deep learning (DL) models have stretched the limitations of spatiotemporal feature extraction, achieving state-of-the-art performance in applications such as machine translation [6], speech recognition [7], and image captioning [8]. Many of these approaches are based on an encoder–decoder architecture that starts with extracting features from input images using CNN and merging these features with a sequence learning model such as RNN or LSTM which predicts the output tokens sequence. An LSTM-based model was used [9] to generate natural language-based descriptions of cooking videos. One of the first sequence-to-sequence approaches was presented in [10], which relies on a stack of two LSTMs where the first LSTM layer encodes visual features, and the second layer decodes the output words sequentially. The model in [10] was trained on a YouTube video collection of MPII-MD [2] and M-VAD [11]. The authors of [12] uses a CNN feature extractor to calculate the embeddings for each input image before averaging them, and an LSTM decoder is then used to predict the output caption. Averaging the extracted features of video frames reduces complexity but results in losing the order of the frames.

Reinforcement earning (RL) is also used in video captioning. For example, [13] proposes a hierarchical reinforcement learning (HRL) video captioning framework that utilizes a two-level mechanism that views the textual and video context as an RL environment and trains two agents: a high-level agent (the manager) for goal setting at a low temporal resolution, and a low-level agent (the worker) for choosing actions according to the goals set by the manager. The authors in [14] used RL to address redundancy and inconsistency between sentences in dense video captioning by generating an event sequence, before feeding each proposed event to a video captioning network that was trained by RL with both event and episode rewards. Additionally, [15] presented an online RL algorithm for solving coupled algebraic Riccati equations of non-zero sum games using a policy iterative algorithm.

Self-attention [1] is a mechanism that relates elements at different positions in a sequence, resulting in a better representation of the same sequence. The self-attention mechanism recalculates the input sequence by giving more weights to relevant features which improves performance, especially for long temporal dependencies. Zhou et al. [16] train a transformer (attention-based network) end-to-end to produce video event proposals and captions simultaneously, allowing the direct influence of the language model to the video event proposal. Transformer networks were also applied to detect and recognize the actions of the person of interest [17] by, first, detecting people in video clips with a bounding box using a 3D CNN and a region proposal network (RPN). Secondly, a sequence of transformer blocks generates features for RPN proposals which are finally classified into action classes. Some methods [11,18] include audio or other elements of the video in video analysis. The authors of [18] present a weakly supervised multi-modal video captioning model. Lashin et al. [19] introduced a multi-modal dense video captioning module (MDVC), a multi-modal architecture to analyze audio, speech, and video to localize and describe events. Hessel et al. [11] used automatic speech recognition (ASR) to improve performance and a transformer-based model to encode video frames and speech tokens, generating captions for instructional videos. An industrial video analysis framework is proposed in [20] which is based Mask R-CNN image segmentation model [21] to detect objects along with a template-based sentence generation model.

The authors in [22,23] use generative adversarial networks (GAN) [24] and a YOLOv3 object detection model [25], respectively, to detect students' actions in a lab. In the first method, both the discriminator and the generator in the DCGAN are trained in parallel to classify behavior. The latter method proposes a framework based on the YOLOv3 object detector to recognize students' activities. These methods perform video classification and use large model sizes. Our proposed framework is unique in that it performs video captioning (which more informative compared with classification) while using lightweight models, enabling it to run on IoT devices. Additionally, this work is inspired by [26] but presents a new transformer-based model and achieves a high accuracy of 97%, compared to only 45% that was reported by [26]. Table 1 presents an overview of the existing literature with key differences.

Table 1. Summary of related work.

Category	Methods	Advantages	Disadvantages
Natural language-based video captioning	Uni-modal (visual features): [9,10]: LSTM-based seq-to-seq model. [16]: Transformer-based model to generate captions and event proposals. [13,14]: Reinforcement learning (RL)-based dense captioning methods Multi-modal (visual features + audio + other features): [11]: Automatic speech recognition ASR and transformer-based model. [18]: Weakly supervised trained model. [19]: Dense video captioning (Localize and caption one event or more).	1. Generating human-like sentences is more user-friendly compared with the other approaches. 2. Many huge natural language-based video-to-texts are available for training.	1. Computationally intensive and have a large memory footprint, therefore cannot be deployed on resource-limited devices. 2. Use large model sizes and need high computational power for training.
Template-based video captioning	[26]: Preliminary results of this work using custom model. [27]: Language processing to estimate activity–object correlation to improve activity recognition systems. [20]: Mask R-CNN [21] and template-based captioning. The proposed method trains a transformer model on SVO templates converted from natural language-based captions.	1. Use moderate computational power and memory footprint, making it a suitable option for resource-limited devices. 2. Require less data for training. 3. Restricting the language model of the output caption makes extracting information/adding context in postprocessing stage easier compared with natural language sentences.	1. Require custom dataset annotation since most video-to-text datasets use natural language-based captions. 2. Lack of proper language model conversion techniques to convert natural language-based captions of existing open-domain datasets to template-based captions.
Classification-based video description	[17]: 3DCNN for object detection and transformer-based for action classification. [22]: A generative adversarial network (GAN) [24] is used to recognize activities. [23]: Uses YOLOV3 [25] to classify students' behavior.	1. Use the least computational power compared to the other two approaches, therefore can be deployed on resource-limited devices. 2. Due to their relatively small model sizes, ensemble of specialized classifiers can be used to recognize a limited set of actions/objects.	1. Classification/Recognition of objects and actions is less informative than captioning.

Our proposed framework starts with collecting and preprocessing a dataset of video-caption pairs to train the neural network model on understanding the scene (video) and generate a description of it.

The preprocessing steps are described in Figure 2. The objective of this phase is to reduce vocabulary size and restrict captions to a predefined template. Our contribution includes how to create a subject-verb-object-based (SVO) caption from a natural language-based caption and replace several words with one representative word to reduce the vocabulary size. Finally, the developed video captioning models are lightweight and can be used on computationally limited IoT platforms. The performed analysis is optimized by:

1. Restricting the captions' language model to simple sentence templates to decrease the complexity and size of the model that is required to achieve a certain accuracy.
2. Developing two lightweight video captioning models, optimizing their parameters with hyperparameter tuning and evaluating their performance in terms of accuracy and inference time.
3. Predicting captions that have predefined sentence structures to make post processing of these captions easier relative to processing natural language-based captions.

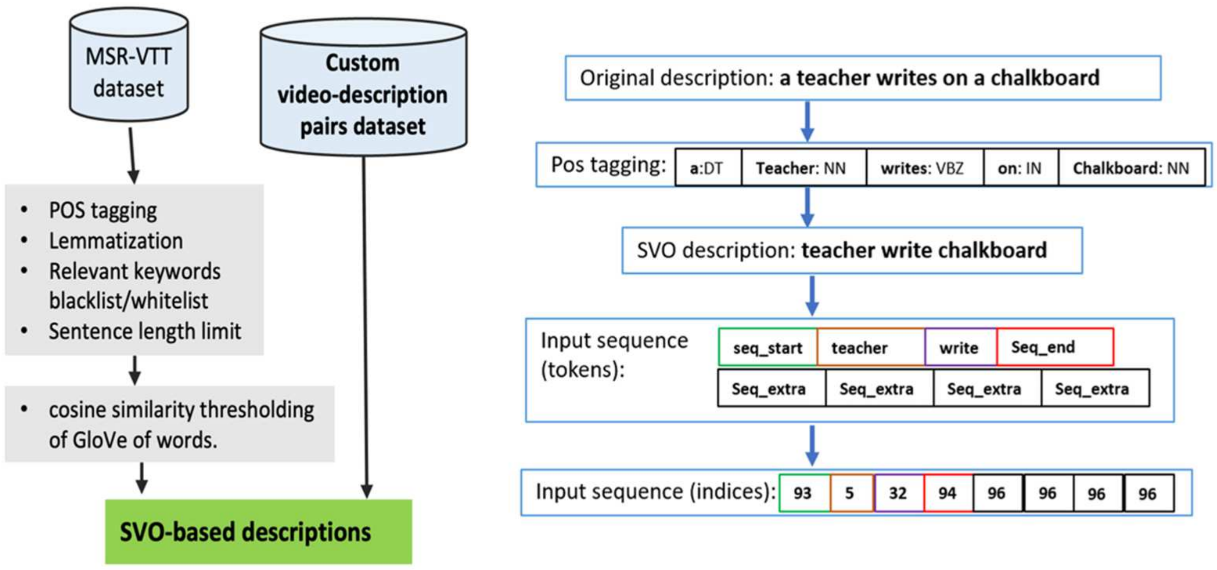

MSR-VTT dataset
Custom video-description pairs dataset
POS tagging
Lemmatization
Relevant keywords blacklist/whitelist
Sentence length limit
cosine similarity thresholding of GloVe of words.
SVO-based descriptions
Original description: a teacher writes on a chalkboard
Pos tagging: a:DT Teacher: NN writes: VBZ on: IN Chalkboard: NN
SVO description: teacher write chalkboard
Input sequence (tokens): seq_start teacher write Seq_end Seq_extra Seq_extra Seq_extra Seq_extra
Input sequence (indices): 93 5 32 94 96 96 96 96

Figure 2. **Left**: Preprocessing steps to perform on MSR-VTT and custom dataset for retrieving relevant data and converting captions to the subject-verb-object (SVO) template. **Right**: An example showing the result of applying these steps on a video caption.

3. Methodology

In this section, the proposed framework is explained in detail. The Dataset subsection describes how the MSR-VTT dataset is preprocessed to select video-caption pairs that belong to our use-case application (classroom monitoring) before the captions are converted to the SVO sentence format. Section 3.2 presents the proposed deep learning models including (1) transformer-based model; (2) LSTM-based model; (3) CNN feature extractor. Third, Section 3.3 describes caption generation.

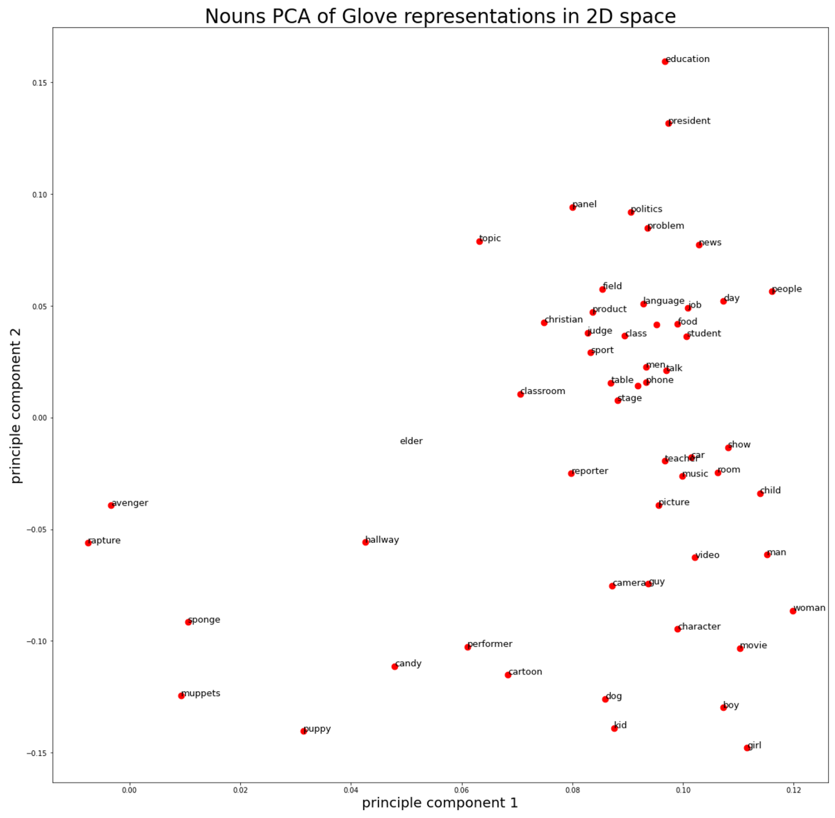
3.1. Dataset

The dataset used in this work is the MSR-VTT dataset, an open-domain video-to-text dataset that has 10,000 videos, 200,000 descriptions in total, and 20 captions per video. The first preprocessing step is to select a subset of video-caption pairs that belong to a particular domain/application. Since we use classroom monitoring as a use case, the dataset is filtered to include only the pairs of video descriptions that are relevant to that application. Although the videos are already tagged in MSR-VTT, none of these categories match our application. The preprocessing pipeline is illustrated in Figure 2 (left), as well as an example showing the result of applying these steps on a video caption (right). We first mined captions by searching for sentences that contained a whitelist of words, such as 'teacher', 'student', and 'class'. We used part-of-speech (POS) tags to exclude adjectives, adverbs, pronouns, etc., keeping only nouns and verbs. Lemmatization is then applied to return the basic form of each noun and verb so that later it is easy to calculate the frequency of words and compare the semantic distance between words more accurately. After applying the preprocessing steps shown in Figure 2, each of the 20 captions per video provided by the MSR-VTT dataset is compared to the SVO sentence structure. If the caption length and POS tags align with that of SVO, the video-caption pair is added as a standardized instance to a new dataset to train the proposed deep learning models. This simple approach is effective but has its limitations which are investigated in the results section.

To further reduce vocabulary size, words' embeddings are clustered and one representative word (the most frequent) replaces all the words in that cluster in their respective sentences. This is achieved by converting words to their Global Vectors for Word Representation (GloVe) [28], a vector representation of words where linear distances reflect the semantic relations between words. This is particularly important in our application since we use the distance between words' embeddings and apply a manually tuned threshold to cluster close embeddings (which represent semantically related words) and replace them with the embeddings vector that corresponds to the most frequent word. This process is applied to nouns and verbs separately. Figure 3 shows the 2D plot of the PCA of the GloVe representation nouns. To cluster word embeddings, we use Euclidean distance (Equation (1)), which was used in [29] to estimate the semantic similarity between words.

$$euclidean\ distance = (\sum(w_i|(u_i - v_i)|^2)^{1/2} \tag{1}$$

where u and v are the two input 1D arrays, $\overline{u}$ and $\overline{v}$ are the means of the elements of u and v, respectively, and w is a 1D weights array. After comparing the generated clusters from each distance algorithm to what is expected, Euclidean distance is used because it was found to yield relatively better results. Words in each cluster are replaced by the most frequent word in that cluster to reduce the vocabulary size and simplify the language model.



Verb	Neighbor verbs
Speak	listen, tell, learn, hear
Ask	call, answer, tell, need, want
Sit	stand, go, wait, let
Drink	feed, eat
Be	have, has, get

Noun	Neighbor nouns
Elder	husband, father, son, brother, daughter
dress	Shirt, coat, sweater, wear, vest, skirt
House	office, room, official
Animal	Animal, cat, dog, pet
infant	Boy, girl, mother, daughter, infant, child

Figure 3. Left: 2D plot of the first two principal components of a subset of nouns' GloVe representations of the MSR-VTT dataset. **Right Top**: samples of verbs and their respective neighbor verbs measured by the Euclidean distance between their GloVe vectors. **Right Bottom**: samples of nouns and their respective neighbor nouns measured by the Euclidean distance between their GloVe vectors.

3.2. Deep Learning-Based Models

Figure 4 demonstrates a diagram of the proposed framework. The diagram shows the model, which will be explained later, at the core as well as two input branches. A sequence of the frames' embeddings, generated from a separate CNN feature extractor, is fed to the first branch, and a sequence of caption tokens (words that were already predicted or placeholder tokens) is fed to the other branch, simultaneously, to generate the next caption token. The input caption tokens are stacked between two special tokens that indicate the beginning and end of a caption: "seq_start" and "seq_end". Additionally, the "seq_extra" special token works as a placeholder for the tokens to be predicted in the following iterations. After that, the tokens are replaced by their respective indices that are fed to the network. The following two subsections describe the details of the two proposed models.

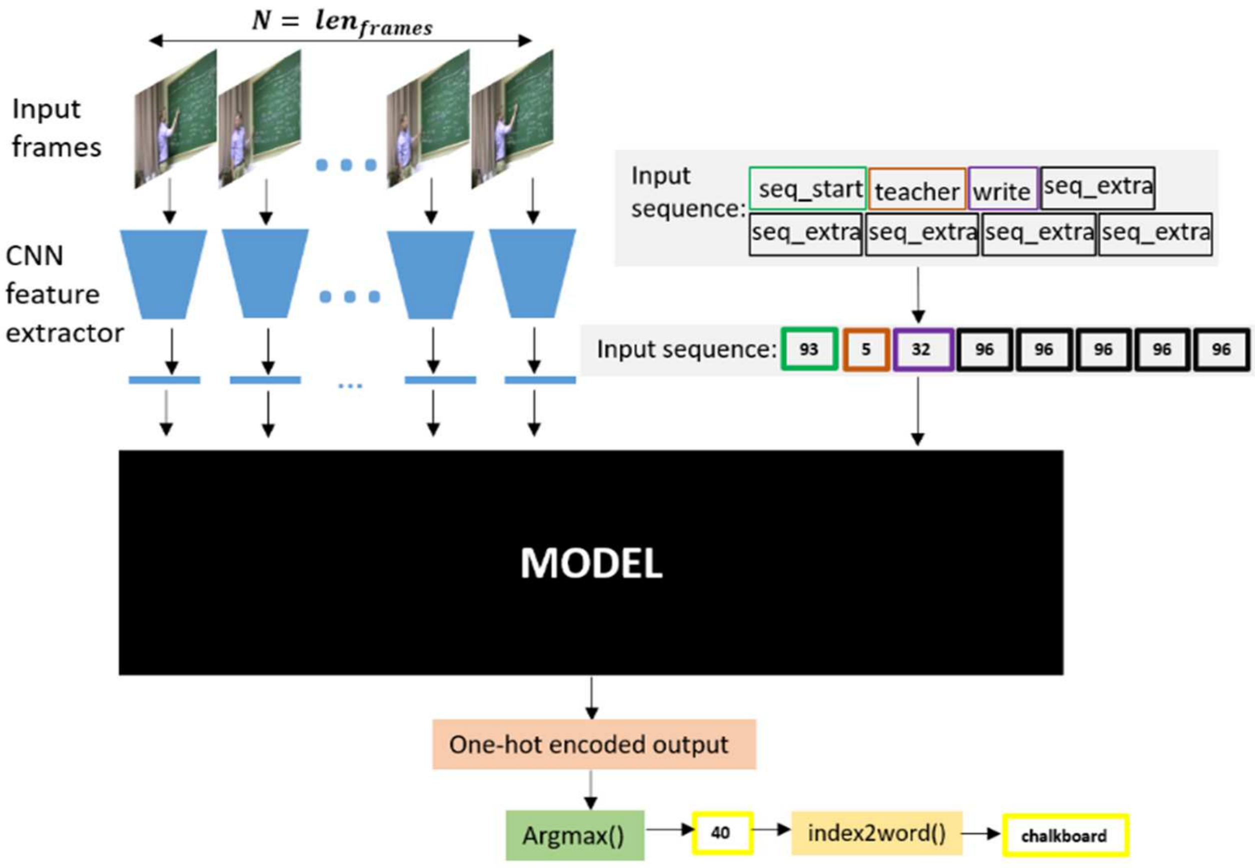


Figure 4. Overview of the proposed system. The block "Model" refers to any of the two proposed models.

3.2.1. Transformer-Based Model

A transformer-based network (also called attention-based) is based on the transformer block (block diagrams of both architectures are shown in Figure 5). The network has two branches; the first branch receives the sequence of the frames' embedding vectors (generated by a pre-trained CNN feature extractor), and the second branch receives a sequence of token indices.

The tokens are converted to a sequence of meaningful embeddings with an embedding layer. An embedding layer can either be trained as part of the model to learn representative embeddings for each token, or import embeddings from a pre-trained global word representation, such as GloVe [28]. After that, the weights of the imported embeddings layer are made untrainable. In our case, we trained the embedding layer since the target language model (sentence structure) is SVO-based which is more constrained than the natural language-based corpuses that global word embeddings are trained on. Then, the frames' embeddings and tokens pass through the transformer layers. Finally, the frames' embeddings are given to an LSTM layer and the output is used as an initial state to another LSTM layer that is applied on caption embeddings (as shown in Figure 5). Finally, a linear layer reduces the output size to the vocabulary size and a softmax activation is applied to generate a one-hot encoded token. A distinctive feature of transformer blocks is that the sizes of the input and the output are the same, but individual element in the output sequence is more "context-aware", relative to the input sequence.

The main component of the transformer block is the multi-head self-attention layer. Hyperparameter tuning is applied to select the number of heads. Figure 5 shows the components of the transformer block which includes a dropout activation function, linear transformations, and layer normalization.

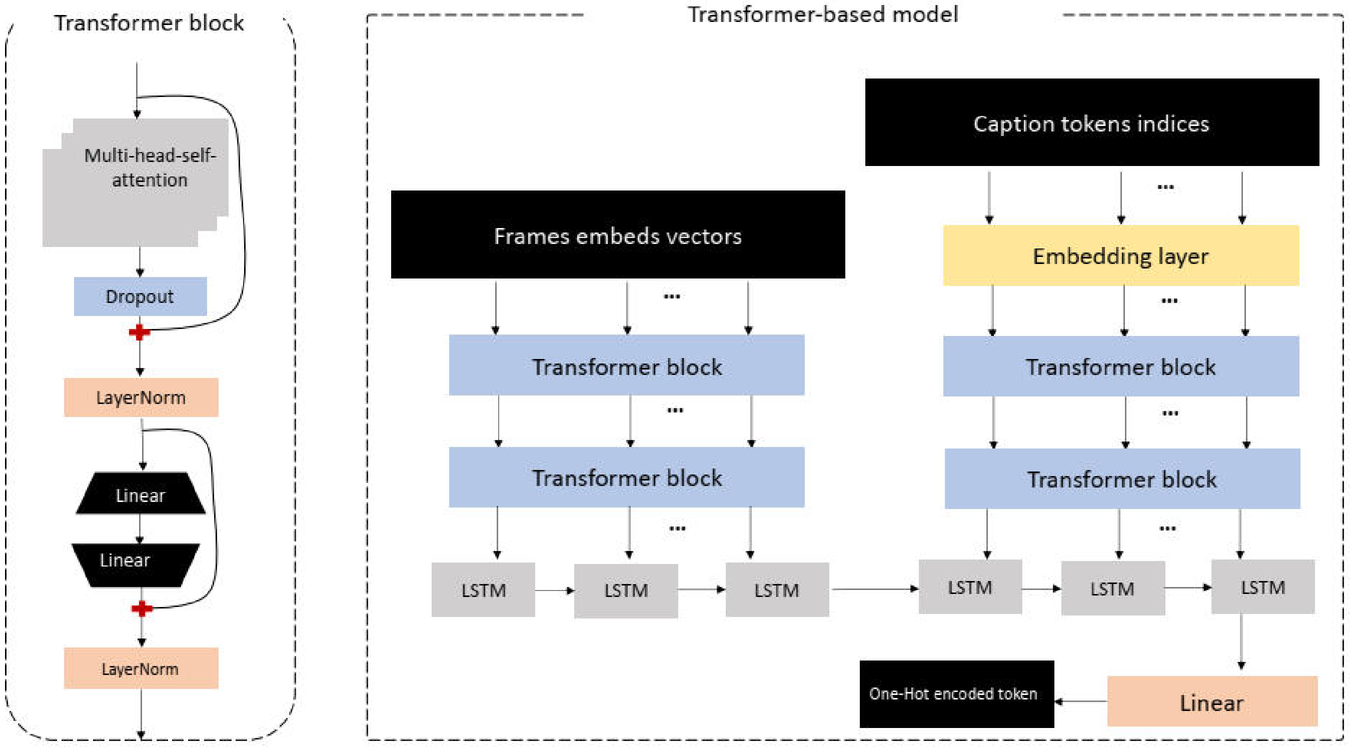


Figure 5. Left: the transformer block. Right: the proposed transformer-based model architecture.

3.2.2. LSTM-Based Model

The second model, shown in Figure 6, is an LSTM-based model also consisting of two branches, one for captions and the other for frames. However, this model relies on LSTM layers to encode the temporal data in both modalities. The output embeddings from each branch are then concatenated and fed to the LSTM layers. Finally, a linear layer reduces the output size to the vocabulary size and a softmax activation is applied to generate a one-hot encoded token.

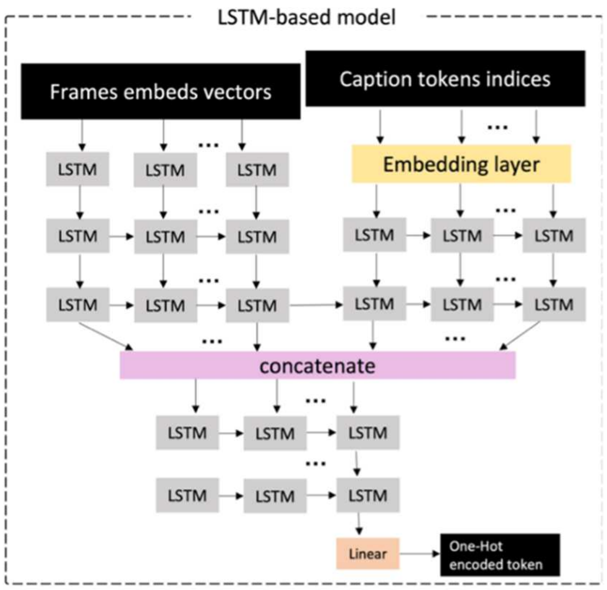


Figure 6. The LSTM-based model architecture.

Despite that every activity is represented by only one caption and each caption in strictly three tokens (subject, verb, and object), the length of any caption is set to eight. This is due to two reasons (1) The model can detect up to two activities, each with three tokens. (2) The special tokens "seq_start" and "seq_end" must be included at the beginning and end of the caption; for example, the description "a teacher writes on a chalkboard" is processed as in Figure 2. Therefore, the total number of tokens to be generated is 3 + 3 + 2 = 8 tokens.

3.2.3. CNN Feature Extractor

Three CNN backbone architectures that were pre-trained on ImageNet [30] were evaluated and compared in terms of their inference time: GoogLeNet [31], ResNet-18 [32], and ResNet-50. Since the CNN model is used to extract embeddings/features of the input frames, it is separated from the rest of the model to balance the time cost of the CNN inference and the informativeness of the generated vector.

3.3. Generating Captions

For an input video, caption generation starts with the CNN feature extractor generating the embeddings vector of each frame. Captions are tokenized into words, and special tokens such as "seq_start", "seq_end", "seq_extra", and "seq_unkown" are added where appropriate. Algorithm 1 describes how captions are generated from an input list of frames. Algorithm 1 uses *ix2word*, a dictionary that maps indices to words, to return a list that starts with "seq_start", ends with "seq_end", and has one or two SVO-based captions in between.

Algorithm 1. Sequential Caption Generation

Input: frames (list of frames),
max_{len} (Maximum length of the generated description sentence),
word2ix (a dictionary mapping each token to index),
ix2word (a dictionary mapping each generated index to token),
model (trained model)
Output: caption (generated sentence)

```
1.     caption ← ["seq_start"]
2.     vectors ← [ ]
3.    for (i = 1 to length(frames)) do
4.    vectors ← add CNN (frames[i])
5.    for (i = 1 to max_len) do
6.    ix_tokens = Algorithm2 (caption, word2ix, max_len)
7.    props ← model.predict (vectors, ix_token)
8.    index ← argmax (props)
9.    token ← ix2word (index)
10.   caption ← add token
11.   if (token = "seq_end") then
12.   break
13.   return caption
```

In each iteration (forward pass), a single token is generated. However, the input size is fixed because the structure of the network is static. Therefore, Algorithm 2 pads the input sequence with a special token called the padding_element to reach the defined input size, max_{len}. In the first iteration, the only known token is the "seq_start" token. Algorithm 2 is used to add "seq_extra" tokens to reach a fixed input length. It then converts input tokens to indices. When the model predicts the next token in the input sequence it is compared to the stopping token, "seq_end". If this is the predicted token, it means that the network has predicted the whole caption; if not, then the predicted token is appended to the input sequence, and a new iteration starts until the "seq_end" token or a certain number of iterations is reached.

Algorithm 2. Padding Captions

Input: word_tokens (list of tokens to pad),
word2ix (a dictionary mapping each token to index),
max_{len} (maximum length of the caption),
Output: ix_tokens (list of padded indices tokens)

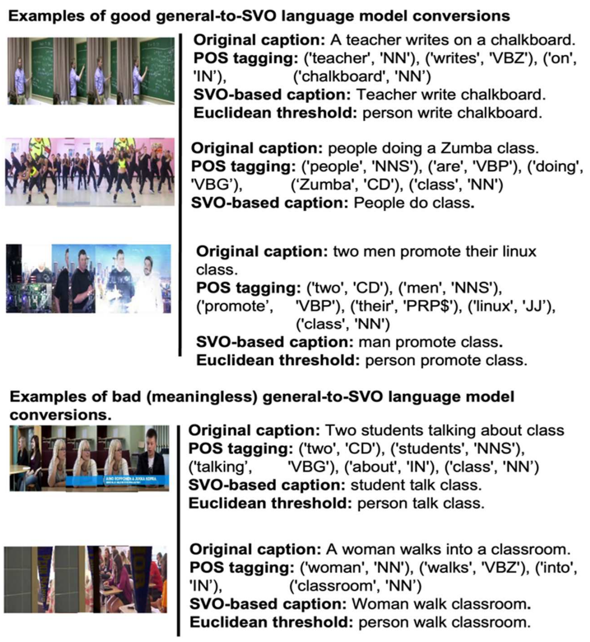
```
1. $padding\_element \leftarrow$ "seq_extra"
2. $padding_{len} \leftarrow max_{len} - length(word\_tokens)$
3. for ($i = 1$ to $padding_{len}$) do
4.     word_tokens ← add padding_element
5. for ($i = 1$ to length(word_tokens)) do
6.     ix_tokens ← add word2ix (word_tokens[i])
7. return ix_tokens
```

4. Results

In this section, the results of caption generation, conversion of natural language-based captions to SVO-based sentences, hyperparameter tuning, and models' inference-time comparison results are presented. Generated SVO captions are evaluated using accuracy for two reasons:

1. The length of SVO-captions is relatively small and fixed, therefore, comparing the corresponding words of the predicted and the ground truth captions makes more sense than machine translation (MT) quality metrics like BLEU [33] which is more suitable for assessing natural language-based captions.
2. Accuracy is used by other template-based methods [20,26] and is therefore needed for comparison.

Figure 7 shows examples of correct and incorrect SVO-based sentences that were generated from the natural language-based sentences of the original dataset (MSR-VTT) [5]. Correctness here refers to whether the generated caption is meaningful to be further processed and stored in the database. For each example, samples of the video frames, original caption, POS tagging of each token in the original caption, and the generated SVO-based caption are shown. The generated captions are tokenized, and the number of unique tokens is reduced by applying the Euclidean distance thresholding approach described earlier. For example, the SVO captions' tokens of the first, third, fourth, and fifth samples in Figure 7 were reduced by replacing "teacher" and "student" with "person".


Examples of good general-to-SVO language model conversions

Original caption: A teacher writes on a chalkboard.
POS tagging: ('teacher', 'NN'), ('writes', 'VBZ'), ('on', 'IN'), ('chalkboard', 'NN')
SVO-based caption: Teacher write chalkboard.
Euclidean threshold: person write chalkboard.

Original caption: people doing a Zumba class.
POS tagging: ('people', 'NNS'), ('are', 'VBP'), ('doing', 'VBG'), ('Zumba', 'CD'), ('class', 'NN')
SVO-based caption: People do class.

Original caption: two men promote their linux class.
POS tagging: ('two', 'CD'), ('men', 'NNS'), ('promote', 'VBP'), ('their', 'PRP$'), ('linux', 'JJ'), ('class', 'NN')
SVO-based caption: man promote class.
Euclidean threshold: person promote class.

Examples of bad (meaningless) general-to-SVO language model conversions.

Original caption: Two students talking about class
POS tagging: ('two', 'CD'), ('students', 'NNS'), ('talking', 'VBG'), ('about', 'IN'), ('class', 'NN')
SVO-based caption: student talk class.
Euclidean threshold: person talk class.

Original caption: A woman walks into a classroom.
POS tagging: ('woman', 'NN'), ('walks', 'VBZ'), ('into', 'IN'), ('classroom', 'NN')
SVO-based caption: Woman walk classroom.
Euclidean threshold: person walk classroom.

Figure 7. Examples of good and bad conversions of captions from the natural language descriptions to the SVO-based captions used in our framework.

Hyperparameter tuning includes the following parameters: number of heads of the transformer block (shown in Figure 5), learning rate, word embeddings vector size, etc. A sample of validation accuracies obtained in the hyperparameter tuning experiments are plotted in Figure 8. Table 2 shows the exact parameter values and the obtained accuracy for each experiment. While Table 2 and Figure 8 compare the transformer-based model and the LSTM-based model in terms of accuracy, Table 3 and Figure 9 compare both models as well as CNN feature extractors in terms of their inference time on different platforms. The LSTM-based model is found to be 13× faster than the transformer-based model, generating a single token in 0.2 s as measured on the MAXN power mode of the Nvidia Jetson nano board.

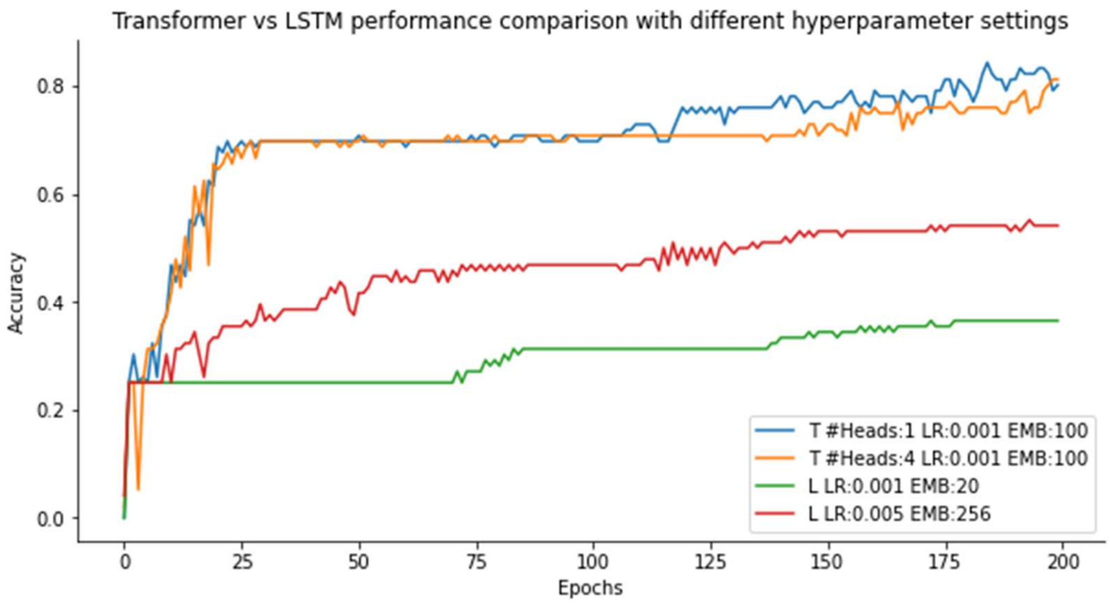


Figure 8. Transformer-based VS LSTM-based models performance comparison with different hyper-parameters settings. Accuracies of transformer-based models are significantly better than accuracies of LSTM-based models.

Table 2. Selected hyperparameters for tuning while training the two proposed models.

Model	**Selected Hyperparameters**			**Accuracy (%)**
	# of Attention Heads	**Learning Rate**	**Word Embeddings**	
Transformer	1	0.001	100	69.8
	1	0.005	100	80.2
	1	0.0005	100	95.6
	2	0.001	100	83.3
	4	0.001	100	81.2
	4	0.0005	100	**97**
LSTM-based model	N/A	0.005	20	52.1
	N/A	0.005	100	52.1
	N/A	0.001	80	54.2
	N/A	0.001	256	**65.6**
	N/A	0.0005	20	25
	N/A	0.0005	256	59.4

Table 3. Time cost (in seconds) for a single inference by the CNN and for generating one token by the video captioning models on both power modes of the Jetson nano board.

Category	**Model**	**Power Mode**	
		MAXN	**5 W**
CNN feature extractor	GoogLeNet	0.33 ± 0.003 s	0.49 ± 0.009 s
	RESNET-18	0.65 ± 0.006 s	1.09 ± 0.007 s
	RESNET-50	1.166 ± 0.304 s	1.536 ± 0.056 s
Token generation	LSTM-based	0.21 s	0.3 s
	Transformer	2.8 s	3.2 s

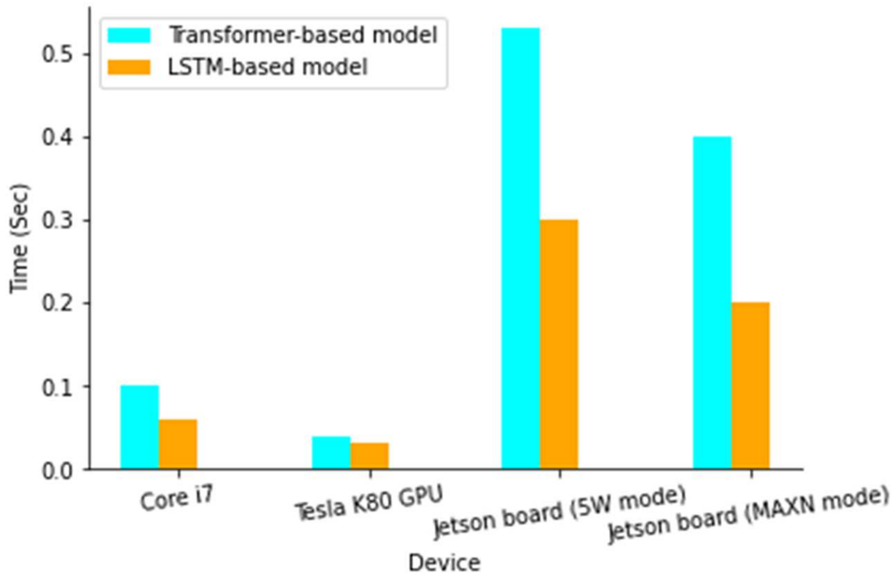


Figure 9. Model inference time comparison on different platforms.

In addition to the parameters' values used in Table 2, other parameters were also included in hyperparameter tuning, such as batch size, number of epochs, and layers' configurations like the keep probability of the dropout layer and the choice of activation layers. However, the subset of parameters presented in Table 2 were experimentally found to have the most influence on performance. As can be seen from Figure 9, The transformer-based model obtained higher accuracy than the LSTM-based model. Moreover, compared with the model presented in [26] on the same task, the transformer-based model also achieved an accuracy of 97%, compared to 45% by the model in [26].

To evaluate the computing time of the different components of the proposed framework, the developed caption generation models and the off-the-shelf CNN feature extractors that were used were tested on a Nvidia Jetson Nano board, a 128-core Maxwell GPU and a Quad-core ARM A57 @1.43 GHZ CPU. The board has two power modes depending on the capability of the power source: 5 W and MAXN modes. We tested the performance of our framework in both modes. Equation (2) calculates the total time (TT) in seconds to generate a description sentence composed of len_{tokens} words from a sequence of frames of length len_{frames}.

$$TT = (len_{tokens} \times 0.21) + 0.33 \tag{2}$$

The previous equation assumes using GoogLeNet as the CNN feature extractor, as well as the LSTM-based model, and the Jetson MAXN mode. This equation is not dependent on len_{frames} because at any time t, the system uses the $len_{frames} - 1$ embedding vectors that were processed in previous steps and needs to apply CNN only on the current frame at time t.

Table 3 shows the time comparison of a single inference among three CNN architectures: GoogLeNet [31], ResNet-18 [32], and ResNet-50 on both power modes, as well as the time cost for a single token generation. GoogleNet has the lowest time cost which is expected due to its smaller size relative to the other two CNN models. Since in the proposed system the CNN model is used for feature extraction (to generate vector embeddings of frames), GoogleNet is selected for its low inference time. Figure 9 compares the inference time of the transformer model vs. the LSTM-based model on different platforms.

5. Conclusions and Discussion

We proposed a deep learning-based light-weight video content analysis framework to be deployed on IoT platforms. Firstly, the presented framework performs data mining on a massive open-domain video-to-text dataset: MSR-VTT to extract video-caption pairs that are relevant to a particular domain (classroom in this case). Second, it generates SVO sentence templates from natural language-based captions to simplify the language model of captions and consequently reduce the size of the deep learning model to be trained on the dataset. Third, reduces the vocabulary size by clustering similar words based on the distance between their GloVe [28] representations to replace the words of each cluster with its most frequent word. finally, we presented two video captioning deep learning-based models: a transformer-based model and an LSTM-based model of which the accuracy is optimized through hyperparameter tuning and the inference time is measured on different platforms, including IoT and GPU-enabled devices. The transformer-based model achieved high accuracy while the LSTM-based model has low inference time making it suitable for IoT devices.

While template-based video captioning methods have many benefits such as reducing the complexity of the language model and making information extraction from the generated caption easier, the scarcity of video-to-text datasets with template-based captions poses a challenge to adopting these methods. Points 1 and 2 in the conclusion present a suggested technique to address this problem. However, the proposed language model conversion technique also has its limitations that should be addressed in future work, including the fact that some natural language captions cannot be converted into SVO captions and others produce meaningless SVO captions. Therefore, manually annotated, template-based datasets are required to train high quality template-based video captioning models.

Author Contributions: Conceptualization, G.G.; methodology, G.G.; Formal analysis, G.G.; validation, E.G. and Z.F.; supervision, Z.F. and B.M.; funding acquisition, G.G., K.C., Z.F. and B.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research is partially funded by the Vector institute through the VI scholarship in AI.

Data Availability Statement: The code that was implemented in this project for creating both neural networks, training, hyperparameter tuning, and evaluation is available at: https://github.com/gadm21/videoToSeq (accessed on 1 May 2022). The video-description pairs that were mined from the MSR-VTT dataset are available at the same GitHub repository.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*. June 2017, Volume 2017-December, pp. 5999–6009. Available online: https://arxiv.org/abs/1706.03762v5 (accessed on 17 March 2021).
2. Rohrbach, A.; Rohrbach, M.; Tandon, N.; Schiele, B. A dataset for Movie Description. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015. [CrossRef]
3. Chen, D.L.; Dolan, W.B. Collecting highly parallel data for paraphrase evaluation. In *ACL-HLT 2011, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, Oregon, 19–24 June 2011*; IEEE: Manhattan, NY, USA, 2011; Volume 1.
4. Torabi, A.; Pal, C.; Larochelle, H.; Courville, A. Using Descriptive Video Services to Create a Large Data Source for Video Annotation Research. March 2015. Available online: http://arxiv.org/abs/1503.01070 (accessed on 13 March 2021).

5. Xu, J.; Mei, T.; Yao, T.; Rui, Y. MSR-VTT: A large video description dataset for bridging video and language. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; Volume 2016-December. [CrossRef]
6. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078. [CrossRef]
7. Bahdanau, D.; Chorowski, J.; Serdyuk, D.; Brakel, P.; Bengio, Y. End-to-end attention-based large vocabulary speech recognition. In Proceedings of the ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing, Shanghai, China, 20–25 March 2016; Volume 2016-May. [CrossRef]
8. Vinyals, O.; Toshev, A.; Bengio, S.; Erhan, D. Show and tell: A neural image caption generator. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015. [CrossRef]
9. Donahue, J.; Saenko, K.; Darrell, T.; Austin, U.T.; Lowell, U.; Berkeley, U.C. Long-term Recurrent Convolution Networks for Visual Recognition and Description. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2625–2634. [CrossRef] [PubMed]
10. Venugopalan, S.; Rohrbach, M.; Donahue, J.; Mooney, R.; Darrell, T.; Saenko, K. Sequence to sequence—Video to text. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; Volume 2015, pp. 4534–4542. [CrossRef]
11. Hessel, J.; Pang, B.; Zhu, Z.; Soricut, R. A case study on combining ASR and visual features for generating instructional video captions. *arXiv* **2019**, arXiv:1910.02930. [CrossRef]
12. Venugopalan, S.; Xu, H.; Donahue, J.; Rohrbach, M.; Mooney, R.; Saenko, K. Translating videos to natural language using deep recurrent neural networks. *arXiv* **2015**, arXiv:1412.4729. [CrossRef]
13. Wang, X.; Chen, W.; Wu, J.; Wang, Y.F.; Wang, W.Y. Video captioning via hierarchical reinforcement learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4213–4222.
14. Mun, J.; Yang, L.; Ren, Z.; Xu, N.; Han, B. Streamlined dense video captioning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 6588–6597.
15. Xin, X.; Tu, Y.; Stojanovic, V.; Wang, H.; Shi, K.; He, S.; Pan, T. Online reinforcement learning multiplayer non-zero sum games of continuous-time Markov jump linear systems. *Appl. Math. Comput.* **2022**, *412*, 126537. [CrossRef]
16. Zhou, L.; Zhou, Y.; Corso, J.J.; Socher, R.; Xiong, C. End-to-End Dense Video Captioning with Masked Transformer. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018. [CrossRef]
17. Girdhar, R.; Carreira, J.J.; Doersch, C.; Zisserman, A. Video action transformer network. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; Volume 2019-June. [CrossRef]
18. Rahman, T.; Xu, B.; Sigal, L. Watch, listen and tell: Multi-modal weakly supervised dense event captioning. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; Volume 2019-October. [CrossRef]
19. Iashin, V.; Rahtu, E. Multi-modal dense video captioning. *arXiv* **2020**, arXiv:2003.07758.
20. Namjoshi, M.; Khurana, K. A Mask-RCNN based object detection and captioning framework for industrial videos. *Int. J. Adv. Technol. Eng. Explor.* **2021**, *8*, 1466. [CrossRef]
21. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
22. Cheng, Y.; Dai, Z.; Ji, Y.; Li, S.; Jia, Z.; Hirota, K.; Dai, Y. Student Action Recognition Based on Deep Convolutional Generative Adversarial Network. In Proceedings of the 32nd Chinese Control and Decision Conference, CCDC 2020, Hefei, China, 22–24 August 2020; pp. 128–133. [CrossRef]
23. Rashmi, M.; Ashwin, T.S.; Guddeti, R.M.R. Surveillance video analysis for student action recognition and localization inside computer laboratories of a smart campus. *Multimed. Tools Appl.* **2021**, *80*, 2907–2929. [CrossRef]
24. Creswell, A.; White, T.; Dumoulin, V.; Arulkumaran, K.; Sengupta, B.; Bharath, A.A. Generative adversarial networks: An overview. *IEEE Signal Process. Mag.* **2018**, *35*, 53–65. [CrossRef]
25. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. April 2018. Available online: http://arxiv.org/abs/1804.02767 (accessed on 2 July 2021).
26. Gad, G.; Gad, E.; Mokhtar, B. Towards Optimized IoT-based Context-aware Video Content Analysis Framework. In Proceedings of the 2021 IEEE 7th World Forum on Internet of Things (WF-IoT), New Orleans, LA, USA, 14 June–31 July 2021; pp. 46–50. [CrossRef]
27. Motwani, T.S.; Mooney, R.J. Improving video activity recognition using object recognition and text mining. In *ECAI 2012*; IOS Press: Amsterdam, The Netherlands, 2012; pp. 600–605.
28. Pennington, J.; Socher, R.; Manning, C.D. GloVe: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014. [CrossRef]
29. Ayeldeen, H.; Hassanien, A.E.; Fahmy, A.A. Lexical similarity using fuzzy Euclidean distance. In Proceedings of the 2014 International Conference on Engineering and Technology (ICET), Cairo, Egypt, 19–20 April 2014; pp. 1–6.
30. Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009. [CrossRef]

31. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015. [CrossRef]
32. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; Volume 2016-December. [CrossRef]
33. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. Bleu: A method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting of the Association for Computational Linguistics, Philadelphia, PA, USA, 7–12 July 2002; pp. 311–318.