# Real-Time Video Captioning on CPU and GPU: A Comparative Study of Classical and Transformer Models

Othmane Sebban[1], Ahmed Azough[2], Mohamed Lamrini[3]
Faculty of Sciences Dhar El Mahraz, Sidi Mohamed Ben Abdellah University, Fez 30003, Morocco[1,3]
Paris La Défense, Léonard De Vinci Pôle Universitaire, Research Center, Paris, France[2]

***Abstract*—This paper proposes a scalable and hardware-adaptable approach to automatic video caption generation by comparing two architectures: a traditional encoder–decoder framework combining InceptionResNetV2 with GRU and a transformer-based model integrating TimeSformer with GPT-2. The system supports CPU and GPU deployment through a unified pipeline built on FFmpeg and ImageMagick for keyframe extraction and subtitle embedding. Experimental evaluations on the MSVD and VATEX datasets demonstrate that the TimeSformer–GPT-2 architecture significantly outperforms baseline models, particularly in GPU settings, achieving top results across BLEU, METEOR, ROUGE-L, and CIDEr metrics. This superiority is attributed to its capacity to model spatiotemporal dependencies and generate contextually rich language. Designed for real-time operation, the system is also suitable for low-resource devices, enabling impactful applications such as assistive tools for the visually impaired and intelligent video indexing. Despite high computational demands and sequence-length limitations, the system presents promising directions for future development, including multilingual captioning, multimodal audio–visual integration, and lightweight models like TinyGPT for enhanced portability.**

***Keywords*—Video captioning; transformer; timesformer; GPT-2; real-time inference; spatio-temporal attention; multimedia accessibility; CPU/GPU deployment**

## I. Introduction

Automatic descriptions of visual content play a crucial role in improving accessibility, information retrieval, and the generation of multimedia content [1]. While this task may seem natural to humans, it remains complex for artificial intelligence, which has to transform visual data into relevant linguistic representations [2]. The generation of natural language descriptions for videos, known as video subtitling, thus represents a major challenge in computer vision and multimedia processing [3]. Advances in deep learning have enabled significant progress, but conventional encoder-decoder architectures still suffer from redundancies and a lack of semantic consistency. Modern approaches, such as sequence-to-sequence models [4] combined with attention mechanisms, improve the alignment between vision and language, but their results sometimes remain insufficient. To overcome these limitations, large-scale language models (LLMs) [5] enhance the ability to generate accurate and consistent descriptions.

Video subtitling is the demanding task of transforming visual sequences into accurate, coherent textual descriptions. Among recent approaches, encoder-decoder architectures have demonstrated their effectiveness by combining networks such as InceptionResNetV2 for feature extraction and GRU for sequence generation. In parallel, large-scale language models (LLMs) have distinguished themselves by their ability to reason and generalize, opening up new opportunities for this application. In this context, GPT-2 [6], a lightweight, high-performance model compatible with non-GPU environments [7], was integrated.

With this in mind, a module optimized for video subtitling was designed, combining the TimeSformer-base-finetuned-k600 encoder with GPT-2 as decoder, in an architecture based on large-scale language models (LLM). This solution has been adapted to run efficiently on CPU-equipped systems, facilitating its deployment on modest hardware configurations. This study aims to evaluate the benefits of this LLM approach compared with conventional sequence-to-sequence architectures. The system, capable of processing videos in real time, relies on FFMPEG [8] for sequence fragmentation and ImageMagick [9]-[10] for automatic subtitle integration. A final evaluation confirms the robustness and relevance of the proposed solution in the field of video captioning.

This paper is structured into five main sections, each addressing a key aspect of the study. Section 2 reviews existing work on video captioning for the visually impaired, highlighting its limitations and presenting two approaches to improvement based on advanced video processing techniques. Section 3 describes the methodology used to develop the system's key modules, highlighting the technical choices made to enhance the user experience. Section 4 presents the experimental results, together with an analysis demonstrating the effectiveness of the proposed solutions. Finally, Section 5 summarizes the study's contributions and suggests directions for future research.

## II. Related Work

In recent years, assistive technologies for the visually impaired have advanced considerably, particularly in navigation, access to environmental information, and video caption generation. M. Chen et al. [11] propose the TVT model, based on a Transformer architecture combining visual information and motion to produce textual descriptions. Although it performs well on MSVD and MSR-VTT sets, its use in real time remains limited due to its complexity.

Kevin et al. [12] introduced SWINBERT, which can capture complex spatiotemporal representations using a sparse

attention mask, reducing image redundancy. However, processing long, dense sequences generates a high computational cost, restricting their effectiveness in real time. L. Zhou et al. [13] conceive a model generating dense descriptions for non-truncated videos, relying on differentiable masking and attention mechanisms. Despite good textual consistency, the computational load associated with the use of Transformers compromises its real-time deployment. For their part, M. Amaresh and S. Chitrakala [14] are exploring encoder-decoder architectures combining CNN and LSTM, integrating attention mechanisms and spatiotemporal analysis, but oriented towards offline processing. Finally, L. Gao et al. [15] present the aLSTMs model, combining attention and LSTM to generate coherent descriptions. Although performing well on MSVD and MSR-VTT bases, this model remains limited by its computational complexity.

## III. Proposed Method

This section describes the design and implementation of the two video subtitling architectures evaluated: a classic encoder-decoder model (InceptionResNetV2-GRU) and a transformer-based model (TimeSformer-GPT2). It details the visual and textual modules, the datasets used (MSVD and VATEX), and the technical optimizations (FFmpeg, ImageMagick), ensuring real-time processing on CPU and GPU.

### *A. Video Captioning Based on Encoder–Decoder Architectures*

An automatic video captioning system is proposed based on an encoder-decoder architecture. It uses the Inception-ResNetV2 pre-trained CNN model to extract visual features, combined with a GRU (Gated Recurrent Unit) decoder to generate natural language text descriptions. The pipeline includes the encoding of videos as feature vectors, followed by the sequential generation of captions by the GRU. Developed with TensorFlow and Keras, the system is trained on the MSVD (Microsoft Video Description Corpus) dataset. It improves on previous approaches based on VGG16 and LSTM, offering a richer visual representation and more efficient sequence production. This model is particularly well suited to applications such as accessibility for the visually impaired, indexing of video content, and metadata creation. How it works is illustrated in Figure 1.

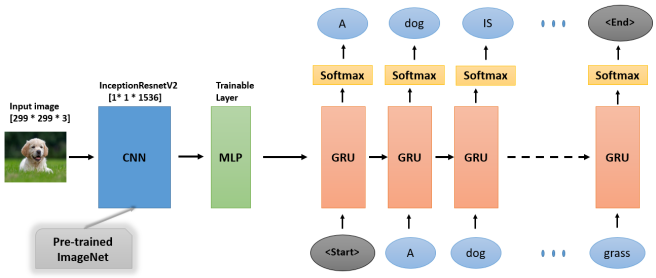


Fig. 1. General architecture of the encoder-decoder model for video captioning.

*1) Visual feature extraction with InceptionResNetV2:* Inception-ResNet-v2 is a neural network architecture that combines the advantages of Inception modules and residual connections [16], enabling efficient visual feature extraction while simplifying deep model training. It is based on three successive blocks - Inception-ResNet-A, B, and C - designed to process feature maps of decreasing dimensions. Block A operates on larger maps using convolutions of various sizes (1×1, 3×3) to capture information at different scales [16]-[17]. Block B continues this processing on intermediate maps, with controlled complexity, while Block C refines the representations on the most compact maps for final classification. Each of these blocks incorporates residual connections, which improve gradient propagation and stabilize learning, notably by alleviating the vanishing gradient problems encountered in very deep networks. The following figure illustrates the organization of these blocks. How it works is illustrated in Figure 2.

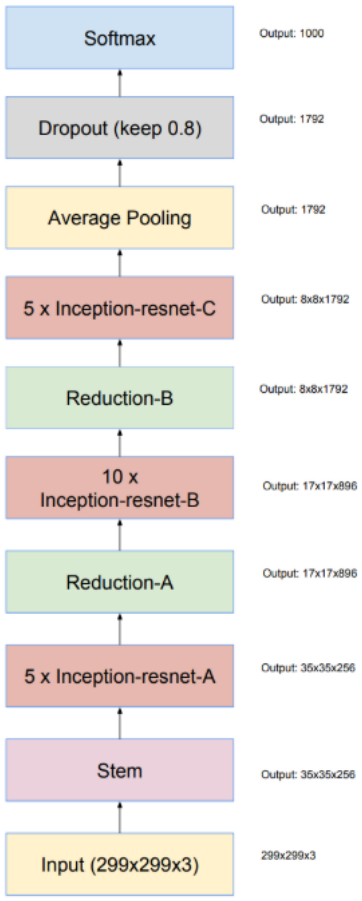


Fig. 2. Internal structure of the InceptionResNetV2 network.

*2) Sequential text generation using GRU decoder:* The decoder transforms visual representations into descriptive sentences, based on an RNN-type architecture. While conventional recurrent networks can model short-term dependencies, they run into difficulties with long sequences, notably due to the disappearance or explosion of the gradient. To overcome these limitations, we use gated recurrent units (GRUs) [18], which

control sequential learning using internal update, reset, and hidden memory mechanisms, as illustrated in figure 3. The decoder comprises three main components: an integration layer that encodes words in vector form, a multilayer GRU for temporal modeling, and a final linear layer that projects the hidden state into vocabulary space to predict the next word.

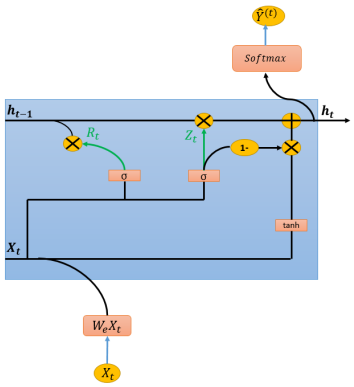


Fig. 3. Internal GRU mechanism (Gated Recurrent Unit).

In these equations (1, 2, 3, and 4) [19], $x_t$ represents the input, and $h_t$ is the hidden state at time $t$. The weights associated with the reset, update, and new information creation gates are denoted as $W_r$, $W_z$, and $W_u$, respectively. The hyperbolic tangent and sigmoid activation functions are symbolized by $\tanh$ and $\sigma$, respectively.

$$r_t = \sigma(W_{xr}x_t + U_{hr}h_{t-1}) \quad (1)$$
$$z_t = \sigma(W_{xz}x_t + U_{hz}h_{t-1}) \quad (2)$$
$$u_t = \tanh(W_{xu}x_t + U_{hu}(r_t \odot h_{t-1})) \quad (3)$$
$$h_t = (1 - z_t)h_{t-1} + z_t u_t \quad (4)$$

*3) Parameters for the encoder-decoder video captioning model:* Table I shows the main hyperparameters of a video caption generation model, combining InceptionResNetV2 for the extraction of visual representations and a GRU-based sequential decoder. The model processes 80 frames per video, from which it extracts vectors of dimension 1536. The text sequences produced are limited to 10 words, with a vocabulary of 1,500 tokens. Learning is performed over 3 epochs, with a learning rate of 0.0007 and a batch size of 8. 15% of the data is reserved for validation to control overlearning.

*4) Dataset used: Microsoft Video Description Corpus (MSVD):* As part of the experiment, the MSVD (Microsoft Video Description Corpus) dataset was selected for training and evaluation of the model. It consists of YouTube videos selected via Amazon Mechanical Turk and annotated with one-sentence descriptions. Only English captions are retained, after light pre-processing including lowercase casing, tokenization, and punctuation removal. The data follow the standard distribution proposed in [4]-[20], with 1,200 videos for learning, 100 for validation, and 670 for testing [21]. For each video, an image is extracted every ten frames, providing balanced temporal coverage. The set comprises 1,970 clips, each around 10 seconds long, and associated with around 40 descriptions, providing a linguistic diversity useful for training the generation system.

### B. Caption Generation with Large Language Models (LLMs)

*1) GPT-2 Integration for descriptive text generation:* This model is an adaptation of the Vision Transformer (ViT) [22], [23] to video data processing, enabling the simultaneous capture of spatial and temporal dynamics. Unlike conventional ViTs designed for still images, ViViT segments a video into a sequence of images, converted into tokens enriched by positional and identity embeddings. These representations are then processed successively by a spatial encoder and a temporal encoder to model the structural and evolutionary relationships between images.The architecture retains the structure of the ViT encoder, with adjustments to exploit dependencies between successive images. The overall operation of the model is shown in Figure 4.

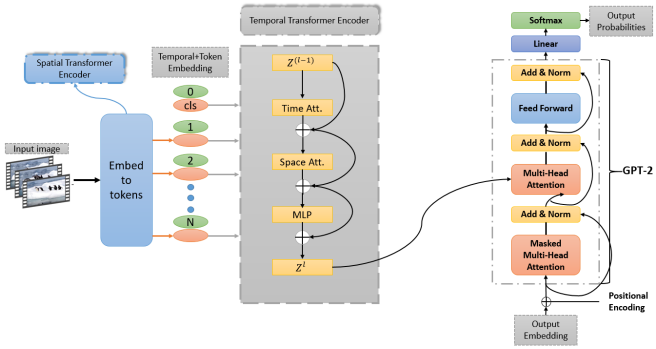


Fig. 4. TimeSformer + GPT-2 global architecture for video description generation.

*2) Spatio-temporal analysis with timesformer encoder:* The basic model is based on the "Base" structure of the Vision Transformer (ViT) and uses sequential spatio-temporal attention, applying attention first to the temporal axis, then to the spatial axis. This method, fitted to the Kinetics-600 dataset - comprising 392,000 videos for training and 30,000 for validation, spread over 600 human action categories - outperformed alternative approaches exploiting parallel or inverted attention. The TimeSformer model adopts this principle, illustrated in Figure 5, by dividing processing into two successive stages: temporal attention followed by spatial attention. This separation enables more efficient extraction of spatio-temporal features from videos.

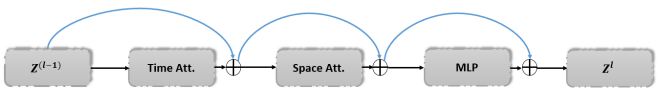


Fig. 5. Sequential spatial-temporal attention mechanism in timesformer.

At each layer, temporal attention is applied, followed by spatial attention and an MLP block, all integrated into a residual connection. This structure improves learning efficiency while reducing computational complexity, as shown by the $Z^{(l)}$ output described in equation 5.

TABLE I. CONFIGURATION PARAMETERS OF THE ENCODER-DECODER VIDEO CAPTIONING MODEL

| Parameter | Value | Description |
|---|---|---|
| video | None | Name of the video file to process |
| keep_temp | False | Keeps the extracted images if set to True |
| train_path | data/training_data | Folder containing the training videos |
| test_path | data/testing_data | Folder containing the testing videos |
| max_length | 10 | Maximum length of the text captions |
| batch_size | 8 | Number of samples per training batch |
| lr | 0.0007 | Learning rate for the optimizer |
| epochs | 100 | Number of training epochs |
| latent_dim | 512 | Latent dimension size of the GRU layer |
| validation_split | 0.15 | Proportion of data used for validation |
| num_encoder_tokens | 1536 | Dimension of the video feature vectors |
| num_decoder_tokens | 1500 | Number of words in the output vocabulary |
| time_steps_encoder | 80 | Number of frames extracted per video |

$$Z^{(l)} = Z^{(l-1)} + \text{TimeAtt}(Z^{(l-1)}) + \text{SpaceAtt}(Z') + \text{MLP}(Z'') \tag{5}$$

with :

$$Z' = Z^{(l-1)} + \text{TimeAtt}(Z^{(l-1)})$$
$$Z'' = Z' + \text{SpaceAtt}(Z')$$

The labels "Frame $t-\delta$", "Frame $t$", and "Frame $t+\delta$" denote the previous, current, and next frames in the video sequence. These serve as key inputs for spatio-temporal attention, enabling the model to capture temporal and spatial dependencies, as shown in Figure 6.

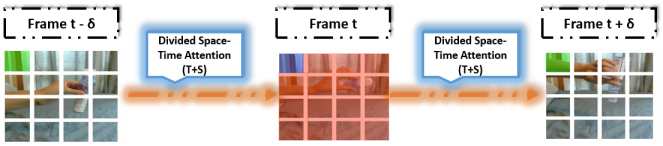


Fig. 6. Sequence of frames used for spatio-temporal attention.

*3) Unified decoder for contextual language generation:* It acts like a decoder, progressively generating a textual description. Each word is predicted by taking into account the words already produced, as well as the visual context extracted from the video. This mechanism ensures the harmonious integration of visual and linguistic information, resulting in an accurate and fluid description [23], [24].

Video subtitling begins by extracting visual features using pre-trained models such as ResNet, TimeSformer [23], [24] or I3D, which transform images into representative vectors. These vectors are then adapted to the GPT-2 input by adding special tokens (e.g., [CLS] for onset and [SEP] for separation). Trained on video-text corpora such as VaTeX [25], GPT-2 learns to generate relevant and coherent descriptions from visual content.

The unified video captioning framework based on GPT-2 models the conditional probability $P(C \mid V)$, where $C$ is the caption and $V$ the input video. It predicts each word in the output sequence based on the visual features extracted from $V$, as shown in Equation 6.

$$P(C \mid V) = \prod_{t=1}^{N} P(c_t \mid c_{<t}, \text{VideoEmbed}(V), \theta) \tag{6}$$

where $c_t$ represents the $t$-th token of the caption, VideoEmbed$(V)$ denotes the visual features mapped to GPT-2's embedding space, and $\theta$ refers to the model's parameters.

The training objective consists of minimizing the negative log-likelihood of the predicted caption sequence conditioned on the video features, as shown in Equation 7.

$$\mathcal{L} = -\sum_{t=1}^{N} \log P(c_t \mid c_{<t}, \text{VideoEmbed}(V), \theta) \tag{7}$$

This fine-tuning process adapts GPT-2's language generation capabilities to the video-text domain, enabling the production of captions that are both semantically precise and aligned with visual content. As illustrated in Figure 7, each frame is divided into 2D patches—typically 7×7—then transformed into visual tokens by a Vision Transformer such as TimeSformer or CLIP. These tokens, enriched with spatial and temporal context, are structured using special markers like [CLS] and [SEP], then passed to a late fusion module that prepares them for decoding. GPT-2, operating autoregressively, generates one word at a time by leveraging both prior textual outputs and the embedded video information. It produces a sequence of logits that guide the selection of the most probable next token at each step. This integrated architecture ensures a smooth fusion of visual cues and language modeling, yielding accurate and naturally flowing captions.

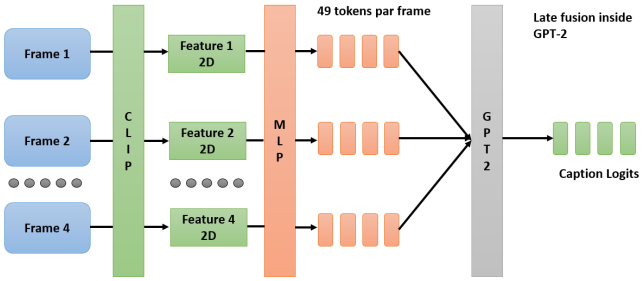


Fig. 7. Video subtitle generation pipeline via TimeSformer and GPT-2.

*4) Performance optimization on CPU vs. GPU:* A [26]-compatible GPU is an ideal solution for training video subtitling models such as TimeSformer and GPT-2, thanks to its ability to handle high resolutions, large batches of data, intensive spatio-temporal processing, as well as the use of mixed precision. Conversely, a CPU [26], although more affordable and accessible, is more suited to tasks of lower complexity due to its limited performance. The following table II compares learning parameters across CPU and GPU environments, detailing batch sizes, image resolutions, number of threads, and training schemes, to optimize performance according to available hardware resources.

During inference, a time clip is extracted from the center of the video. On a processor (CPU), the image is resized to 112 pixels on its shortest side, followed by a 112 × 112 center crop, to limit computation. A single softmax score is then used to generate the prediction. On a GPU, the TimeSformer-HR model exploits a higher resolution: resizing is performed at 224 pixels, with three 224 × 224 cuts analyzed to enrich the spatial information. The final prediction, obtained by averaging the softmax scores, thus benefits from the GPU's ability to process high-resolution data, enhancing the model's accuracy and robustness. Figure 8 compares two architectures designed for real-time caption generation. Sub-figure 8(a) illustrates the GPT-2 architecture, responsible for encoding the data and producing the textual descriptions. Sub-figure 8(b) shows the optimized version of TimeSformer, trained on Kinetics-600, capable of efficiently capturing spatio-temporal relationships in video sequences.

*5) Dataset used (VATEX):* VATEX is a large-scale dataset comprising around 41,250 10-second video clips, each accompanied by 10 manually annotated English captions. Learning is based on the official set, while performance is measured using the public test set [21], [25].Three Vision Transformer encoders are used for visual feature extraction, including an I-frame encoder initialized with the CLIP model [27], trained on the LAION-400M image-text corpus [28]. In comparison, SwinBERT [12] relies on the VidSwin architecture, trained on the video-oriented Kinetics-600 [29] dataset. It should be noted that LAION-400M is an image-text corpus, Kinetics-600 targets videos, and VATEX serves here as a reference for experimentation. SwinBERT outperforms the proposed method, thanks in particular to the effectiveness of its pre-training phase on Kinetics-600.

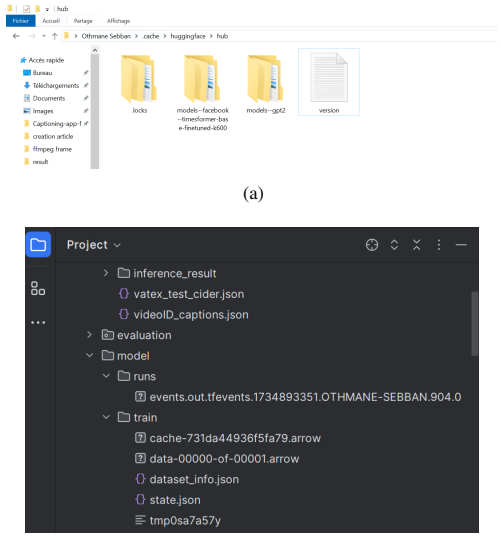


Fig. 8. Illustration of video captioning components: (a) GPT-2 used for text decoding, and (b) TimeSformer for encoding video frames into visual embeddings.

## C. System Optimization Using FFmpeg and ImageMagick

To train the models, FFmpeg [8] was used, integrated via MoviePy [30], to facilitate the manipulation of complex video content and to optimize the use of large language models (LLMs). This free software supports a wide variety of multimedia formats, including video and audio, and includes a library dedicated to keyframe extraction. These keyframes generally represent significant visual changes. However, FFmpeg does not detect more subtle variations, often referred to as interesting images, which may nonetheless contain information relevant to analysis. MoviePy [31] simplifies the addition of text to videos by automating its generation and positioning using the TextClip class, then embedding it in the video via CompositeVideoClip, with FFmpeg support in the background. Once the text has been merged, it is encoded using standard codecs such as libx264 for video and AAC for audio, ensuring optimum compatibility with the majority of media players. In addition, key frame extraction aims to identify representative frames in a video sequence. This process can be carried out using a variety of tools, including FFmpeg, OpenCV's absdiff() function or the DMD algorithm [32]. Figure 9 illustrates the images extracted from a video file in MP4 format using the following FFmpeg command: `ffmpeg -i file.mp4 -vf "select='not(mod(n,4))',scale=320:240" -vsync vfr -frames:v 4 frame_%03d.png`.

ImageMagick is currently used to integrate full descriptions into videos as part of the subtitling process. An integrated tool [32], combined with a command-line routine, automates the cropping of images by removing 8 pixels at the left and 40 pixels at the top, reducing their size by around 20%. In addition, partial conversion of PPM files to JPEG is performed by a batch script. Interstitial spaces between mosaic segments were also corrected for archiving purposes. In all, the 67 selected mosaics and still images occupy 764 Kb.

TABLE II. HARDWARE CONFIGURATION COMPARISON FOR TRAINING CAPTIONING MODELS ON CPU VS. GPU

| **Parameter** | **CPU** | **GPU** | **Description** |
|---|---|---|---|
| Device | ”cuda” if GPU available, else ”cpu” | ”cuda” if GPU available, else ”cpu” | Automatic device detection |
| Encoder Model | facebook/timesformer-base-finetuned-k600 | facebook/timesformer-base-finetuned-k600 | Model used for encoding |
| Decoder Model | Gpt-2 | Gpt-2 | Model used for decoding |
| Video Frames | 4 | 16 | Number of frames used per video |
| Image Resolution | 112×112 | 224×224 | Size of the images used |
| Batch Size | 2 | 6 | Batch size for training |
| Batch Learning | Disabled | Enabled | Use of `GradScaler` for AMP |
| Learning Rate | $1e^{-5}$ | $5e^{-7}$ | Learning rate for the optimizer |
| Epochs | 100 | 100 | Number of epochs for training |
| Scheduler | linear | linear | Learning rate scheduler |
| Num Workers | 1 | 8 | Number of workers for `DataLoader` |
| Collate Function | custom.collate.fn | default.data.collator | Data collation method |
| Pin Memory | False | True | Memory management in `DataLoader` |

Fig. 9. Extracting key frames from video using FFmpeg.

## IV. EXPERIMENTAL RESULTS

The Experimental Results section provides an in-depth analysis of the system's performance, examining each of its key modules. Subsection A outlines the evaluation criteria and the fundamental components of the system. Subsection B presents the results obtained by the video subtitle generation models developed according to the two approaches explored.

### A. The Evaluation Metrics

To assess the quality of the trained model, four metrics are used. Among these, the BLEU (Bilingual Evaluation Understudy) metric quantifies linguistic similarity by comparing the n-grams shared between the generated sentences and the reference sentences. The calculation of the BLEU score is presented in equation 8.

$$\text{BLEU} - N(ci, Si) = b(ci, Si) \exp\left(\sum_{n=1}^{N} \omega_n \log P_n(ci, Si)\right) \tag{8}$$

where

$$b(ci, Si) = \begin{cases} 1 & \text{if } lc > ls \\ e^{1-\frac{ls}{lc}} & \text{if } lc \leq ls \end{cases}$$ is a brief penalty; lc is

The total length of the sentences generated (candidates) is noted as $l_S$, while $l_c$ designates the optimal reference length of the corpus. When a candidate sentence is associated with several references, the one whose length is closest to that of the candidate is selected. In addition, the weights $\omega_n$, assigned to the $n$-grams, are generally set to a constant value, as indicated in [18]-[33], as illustrated in the equation 9.

$$P_n(ci, Si) = \frac{\sum_k \min(h_k(ci), \max(h_k(S_{ij})))_{j \in \mathcal{M}}}{\sum_k h_k(ci)} \tag{9}$$

B-N, short for BLEU-N, is a precision measure that evaluates the quality of generated sentences by focusing on short n-grams, generally up to 4 words. It calculates the proportion of linguistic units (n-grams) shared between a candidate sentence and one or more reference sentences.

METEOR is an evaluation metric that measures both precision and recall at the unigram level. It takes into account not only exact matches, but also synonyms from WordNet and partial matches based on word fragments (truncated tokens). The METEOR score is given by the equation 10 [18]-[33].

$$\text{METEOR} = (1 - \text{Pen}) \times \text{Fmean} \tag{10}$$

The term $\text{Pen} = \gamma \left(\frac{\text{ch}}{m}\right)^m$ represents a penalty factor. In this expression, $m$ corresponds to the total number of alignments between the candidate sentence and the reference sentence, while ch designates the number of contiguous segments correctly aligned (words identical and in the same order), as specified in equations 11, 12, and 13.

$$P_m = \frac{|m|}{\sum_k h_k(ci)} \tag{11}$$

$$R_m = \frac{|m|}{\sum_k h_k(S_{ij})} \tag{12}$$

$$F_{\text{mean}} = \frac{P_m \cdot R_m}{\alpha P_m + (1 - \alpha) R_m} \tag{13}$$

METEOR is a measure based on the harmonic mean between precision and recall, calculated between a reference sentence and the best-aligned candidate.

ROUGE-L is based on the longest common subsequence (LCS) identified between two sentences. The length $l(ci, sij)$ of this LCS forms the basis of its evaluation, as shown in equation 14.

$$\text{ROUGE-L}(ci, Si) = \frac{(1+\beta^2) \cdot Rl \cdot Pl}{Rl + \beta^2 \cdot Pl} \tag{14}$$

where:

$$Rl = \max_j \left( \frac{l(ci, S_{ij})}{|S_{ij}|} \right) \quad \text{(recall based on LCS)}$$

$$Pl = \max_j \left( \frac{l(ci, S_{ij})}{|ci|} \right) \quad \text{(precision based on LCS)}$$

The **ROUGE-L** metric evaluates precision based on the longest common subsequence (LCS), while the parameter$E$ is a constant to reinforce the importance of recall in the calculation. Other variants of the RED metric, such as **ROUGE-N** and **ROUGE-S**, are also used for finer comparisons [18]. For its part, **CIDEr-D** measures the average cosine similarity between the n-grams of a candidate sentence and those of references, taking into account both precision and recall. Its mathematical formulation is given by equation 15 [18]-[33].

$$CIDEr - D_n(ci, Si) = \sum_{n=1}^{N} \omega_n \text{CIDEr-D}(ci, Si) \tag{15}$$

This formula aggregates the *CIDEr-D* scores calculated for different levels of *n-grams* (from 1-gram to $N$-gram), assigning them a specific weight $\omega_n$. This enables us to assess the similarity between the generated legend $c_i$ and reference legends $S_i$ at several levels of linguistic granularity, while incorporating a weighting mechanism that mitigates the impact of excessive repetition and too-frequent *n-grams*.

### B. Quantitative Results

The proposed method was evaluated using the BLEU, ROUGE-L, METEOR, and CIDEr metrics to measure its efficiency. Tests were carried out on two Seq2Seq models based on LSTM and GRU, respectively, as well as on the Timesformer-base fine-tuned on Kinetics-600 with GPT-2, evaluated on both CPU and GPU.

The Neleac/timesformer-gpt2-video-captioning pre-trained model [22], [23] is based on a modular architecture built around three components. A visual preprocessor (MCG-NJU/videomae-base) adapts images extracted from videos, a tokenizer (GPT-2) encodes text sequences, and an encoder-decoder model combines a Vision Transformer with a GPT-2 decoder to automatically produce subtitles. This configuration, illustrated in figure 10, enables efficient, context-sensitive generation of descriptions from video content.

The results, presented in Table III, show the best performances in bold. Three modules were evaluated, including the pre-trained Timesformer-GPT2 subtitling model, as well as two encoder-decoder architectures designed for video description generation. Compared with previous methods, the proposed approach outperformed across all considered metrics, including BLEU-1 to BLEU-4, METEOR, CIDEr, and ROUGE-L.

```python
import av
import numpy as np
import torch
from transformers import AutoImageProcessor, AutoTokenizer, VisionEncoderDecoderModel

device = "cuda" if torch.cuda.is_available() else "cpu"

# load pretrained processor, tokenizer, and model
image_processor = AutoImageProcessor.from_pretrained("MCG-NJU/videomae-base")
tokenizer = AutoTokenizer.from_pretrained("gpt2")
model = VisionEncoderDecoderModel.from_pretrained("Neleac/timesformer-gpt2-video-captioning").to(device)

# load video
video_path = "never_gonna_give_you_up.mp4"
container = av.open(video_path)

# extract evenly spaced frames from video
seg_len = container.streams.video[0].frames
clip_len = model.config.encoder.num_frames
indices = set(np.linspace(0, seg_len, num=clip_len, endpoint=False).astype(np.int64))
frames = []
container.seek(0)
for i, frame in enumerate(container.decode(video=0)):
    if i in indices:
        frames.append(frame.to_ndarray(format="rgb24"))

# generate caption
gen_kwargs = {
    "min_length": 10,
    "max_length": 20,
    "num_beams": 8,
}
pixel_values = image_processor(frames, return_tensors="pt").pixel_values.to(device)
tokens = model.generate(pixel_values, **gen_kwargs)
caption = tokenizer.batch_decode(tokens, skip_special_tokens=True)[0]
print(caption) # A man and a woman are dancing on a stage in front of a mirror.
```

Fig. 10. Example of generated video captioning from the TimeSformer-Gpt2 pre-trained model.

The models compared in the table fall into two categories: on the one hand, classic encoder-decoder architectures based on InceptionResNetV2 coupled with GRU; on the other, those based on the combination of TimeSformer and GPT-2. The latter captures spatio-temporal relationships more effectively and produces more accurate descriptions. Although they perform well on the CPU, they reach their full potential on the GPU, particularly in their pre-trained version, which considerably accelerates inference. Overall, this architecture outperforms encoder-decoder approaches, but at the cost of higher hardware consumption.

## V. Comparative Analysis and Interpretation

This study compared two architectures for video captioning: a traditional solution based on an encoder-decoder framework combining InceptionResNetV2 and GRU, and a more advanced approach integrating a pre-trained TimeSformer encoder with a GPT-2 decoder. Experimental results demonstrate that the TimeSformer-GPT2 configuration, when executed on a GPU, achieves superior performance, notably with a BLEU-4 score of 0.276, compared to only 0.058 for the baseline. Improvements were also recorded in METEOR (0.821) and CIDEr (0.889), highlighting the effectiveness of combining a spatio-temporal attention mechanism with a generative language model for producing coherent and accurate outputs.

In particular, while the pre-trained Neleac/timesformer-gpt2-video-captioning model proved highly effective, the optimized Timesformer-base fine-tuned (GPU) version demonstrated superior performance on the majority of evaluation metrics, validating the effectiveness of the implemented adaptation process.

These findings emphasize the strength of TimeSformer in capturing complex temporal and spatial dependencies within video sequences. Coupled with GPT-2's capacity to generate contextually appropriate language, this synergy allows for a

TABLE III. QUANTITATIVE EVALUATION OF VIDEO CAPTIONING MODELS USING BLEU, METEOR, ROUGE-L, AND CIDER METRICS

| Model | B-1 | B-2 | B-3 | B-4 | ROUGE-L | CIDEr | METEOR |
|---|---|---|---|---|---|---|---|
| GRU-based Encoder–Decoder (our model) | 0.685 | 0.331 | 0.250 | 0.058 | 0.241 | 0.372 | N/A |
| Timesformer-base finetuned (CPU) (our model) | 0.803 | 0.492 | 0.370 | 0.182 | 0.567 | 0.765 | 0.714 |
| Timesformer-base finetuned (GPU) (our model) | **0.887** | **0.571** | **0.498** | **0.276** | **0.687** | **0.889** | **0.821** |
| Timesformer-GPT2 video captioning (GPU) [22], [23] | 0.872 | 0.562 | 0.475 | 0.259 | 0.672 | 0.884 | 0.812 |

deeper understanding and expression of visual content. The architecture relies on a divided attention mechanism—applying temporal and then spatial attention—to enhance the quality of feature extraction. GPT-2, in turn, adapts its linguistic output based on these extracted patterns, resulting in semantically rich and logically structured narratives.

Compared to models such as SWINBERT and TVT, the proposed method demonstrates a notable reduction in computational overhead, especially during inference on CPU-based systems. While many transformer-based solutions are primarily designed for offline processing, the presented system remains compatible with lower-resource environments without compromising output quality. These results align with the contributions of Bertasius et al. and Radford et al. [22]-[23], while offering a more scalable alternative for practical deployment.

From a conceptual standpoint, this work supports the hypothesis that coupling a high-performance visual encoder with a Large Language Model (LLM) improves the generation of multimodal language. Practically, it illustrates the feasibility of integrating sophisticated captioning mechanisms into mobile or embedded systems, particularly for assistive technologies aimed at the visually impaired or for intelligent indexing of audiovisual content.

Despite encouraging results, the proposed approach presents some limitations. The training phase, especially when involving the TimeSformer component, requires considerable computational resources. Furthermore, the fixed-length format (limited to 10 words) imposes constraints on expressive capacity, particularly in visually complex scenes. Additionally, the evaluation was limited to two datasets (MSVD and VATEX); further validation on diverse, especially multilingual, corpora would be beneficial to assess the generalizability of the system.

Analysis of the videos in the MSVD and VATEX datasets reveals limitations specific to each model. The GRU model favors static elements, sometimes neglecting the main action, while the Timesformer-base finetuned (GPU) model, while effective at capturing movement, can confuse context or over-interpret complex scenes. These errors highlight the difficulty of accurately understanding a scene and underscore the value of multimodal approaches that combine visual, sound, and temporal signals.

Potential enhancements include the incorporation of compact architectures such as TinyGPT and the use of knowledge distillation techniques to reduce training complexity. Another promising direction involves exploring multimodal pipelines that combine audio and visual data to further enrich the generated outputs. Moreover, extending the system to handle real-time processing of long-form video content or deployment in unstructured environments remains a key challenge for improving portability and robustness. Figure 11 shows the final results of the proposed system, illustrating optimal performance in terms of video captioning. It highlights the model's ability to produce accurate and consistent descriptions, demonstrating its effectiveness.

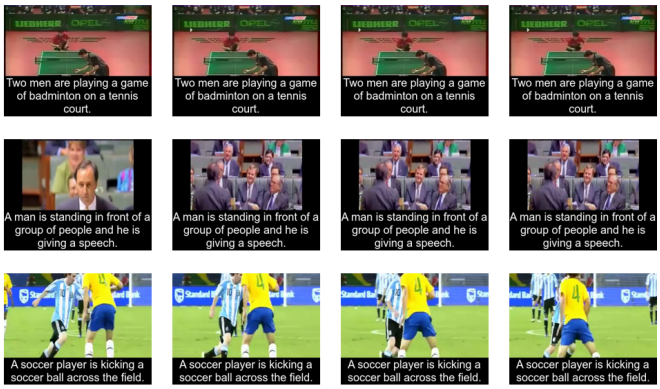


Fig. 11. Video captioning: key frame extraction with FFmpeg and caption insertion using ImageMagick.

## VI. CONCLUSION AND FUTURE WORK

This paper has proposed a comparative study between classical video captioning architectures, such as InceptionResNetV2-GRU, and advanced approaches combining TimeSformer and GPT-2. Experiments on MSVD and VATEX datasets demonstrated the clear superiority of the TimeSformer-GPT2 model, particularly on GPUs, with high scores according to the BLEU, METEOR, ROUGE-L, and CIDEr metrics. This performance is explained by the model's ability to capture spatio-temporal dependencies while generating linguistically consistent descriptions efficiently. The proposed pipeline, integrating FFmpeg and ImageMagick tools, demonstrates the feasibility of a robust and adaptable automated subtitling system, compatible with different hardware environments. This flexibility opens up concrete prospects, notably in assistive technologies for the visually impaired and intelligent indexing of audiovisual content. Despite the promising results, there are several avenues for improvement worth exploring. Adopting lighter models such as TinyGPT or DistilGPT would reduce the computational load and facilitate deployment on mobile devices. The addition of audio modality could enrich the descriptions generated, particularly for complex scenes.

Furthermore, adaptation to specific multilingual corpora (educational, medical) would reinforce the system's generalizability. Finally, the processing of long continuous videos with dynamic generation of summaries or subtitles would

pave the way for more advanced applications, notably in visual assistance or intelligent streaming. This work paves the way for a new generation of intelligent video subtitling systems, capable of reconciling performance, accessibility, and lightness, and designed for concrete applications on mobile and embedded platforms.

## ACKNOWLEDGMENT

Dr. Othmane Sebban thanks Dr. Ahmed Azough and Dr. Mohamed Lamrini, as well as the research team, for their essential support in this study.

## REFERENCES

[1] L. Guo et al., "Non-autoregressive image captioning with counterfactuals-critical multi-agent learning," Proceedings of the Twenty Ninth International Joint Conference on Artificial Intelligence, pp. 767–773, Jul. 2020. doi:10.24963/ijcai.2020/107.

[2] Abdar, Moloud et al., A Review of Deep Learning for Video Captioning. 2023.

[3] Y. Zheng, Y. Zhang, R. Feng, T. Zhang, and W. Fan, "Stacked multimodal attention network for context-aware video captioning," IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 1, pp. 31–42, Jan. 2022. doi:10.1109/tcsvt.2021.3058626.

[4] H. Wang, C. Gao, and Y. Han, "Sequence in sequence for video captioning," Pattern Recognition Letters, vol. 130, pp. 327–334, Feb. 2020. doi:10.1016/j.patrec.2018.07.024.

[5] C. Zhang et al., "A simple LLM framework for long-range video question-answering," Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pp. 21715–21737, 2024. doi:10.18653/v1/2024.emnlp-main.1209.

[6] Y. Lu et al., "Set prediction guided by Semantic Concepts for diverse video captioning," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, no. 4, pp. 3909–3917, Mar. 2024. doi:10.1609/aaai.v38i4.28183.

[7] A. Tariq, M. Elhadef, and M. Ghani Khan, Vidcap-LLM: Vision-Transformer and large language model for video captioning with linguistic semantics integration, 2024. doi:10.2139/ssrn.4812289.

[8] R. Radarapu, A. S. Gopal, M. NH, and A. K. M., "Video summarization and captioning using dynamic mode decomposition for surveillance," International Journal of Information Technology, vol. 13, no. 5, pp. 1927–1936, May 2021. doi:10.1007/s41870-021 00668-0.

[9] S. Sarwar et al., "Advanced Audio Aid for Blind people," 2022 International Conference on Emerging Technologies in Electronics, Computing and Communication (ICETECC), pp. 1–6, Dec. 2022. doi:10.1109/icetecc56662.2022.10069052.

[10] C. Soto and S. Yoo, "Visual detection with context for document layout analysis," Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP IJCNLP), 2019. doi:10.18653/v1/d19-1348.

[11] M. Chen, Y. Li, Z. Zhang, et S. Huang, "TVT: Two-View Transformer Network for Video Captioning," Proceedings of Machine Learning Research, vol. 95, pp. 847–862, 2018.

[12] Lin, Kevin et al., Swinbert: End-to-End Transformers with Sparse Attention for Video Captioning. 2021.

[13] L. Zhou, Y. Zhou, J. J. Corso, R. Socher, and C. Xiong, "End-to-end dense video captioning with masked transformer," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Jun. 2018. doi:10.1109/cvpr.2018.00911.

[14] M. Amaresh and S. Chitrakala, "Video captioning using Deep Learning: An overview of methods, datasets and metrics," 2019 International Conference on Communication and Signal Processing (ICCSP), pp. 0656–0661, Apr. 2019. doi:10.1109/iccsp.2019.8698097.

[15] L. Gao, Z. Guo, H. Zhang, X. Xu, and H. T. Shen, "Video captioning with attention-based LSTM and semantic consistency," IEEE-Transactions on Multimedia, vol. 19, no. 9, pp. 2045–2055, Sep. 2017. doi:10.1109/tmm.2017.2729019.

[16] M. W. Kesiman, K. T. Dermawan, and I. G. Darmawiguna, "Balinese carving ornaments classification using INCEPTIONRESNETV2 architecture," 2022 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM), pp. 1–5, Nov. 2022. doi:10.1109/cenim56801.2022.10037265.

[17] R. R. Rajalaxmi et al., "Deepfake detection using inception-ResNet-V2 Network," 2023 7th International Conference on Computing Methodologies and Communication (ICCMC), pp. 580–586, Feb. 2023. doi:10.1109/iccmc56507.2023.10083584.

[18] V. KILIC¸, "Deep gated recurrent unit for Smartphone-based image captioning," Sakarya University Journal of Computer and Infor mation Sciences, vol. 4, no. 2, pp. 181–191, Aug. 2021. doi:10.35377/saucis.04.02.866409.

[19] C.Zhu,Q.Jia, W.Chen, Y. Guo, and Y. Liu, "Deep learning for video-text retrieval: A Review," International Journal of Multimedia Information Retrieval, vol. 12, no. 1, Feb. 2023. doi:10.1007/s13735-023-00267-8.

[20] S. Venugopalan et al., "Translating videos to natural language using deep recurrent neural networks," Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2015. doi:10.3115/v1/n15-1173.

[21] Y. Shen et al., "Accurate and fast compressed video captioning," 2023 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 15512–15521, Oct. 2023. doi:10.1109/iccv51070.2023.01426.

[22] G. Bertasius, H. Wang, and L. Torresani, "Is Space-Time Attention All You Need for Video Understanding?", arXiv preprint arXiv:2102.05095, 2021.

[23] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," OpenAI, 2019.

[24] D. Xu, W. Zhao, Y. Cai, and Q. Huang, "Zero-textcap: Zero-shot framework for text-based image captioning," Proceedings of the 31st ACM International Conference on Multimedia, pp. 4949–4957, Oct. 2023. doi:10.1145/3581783.3612571.

[25] X. Wanget al., "Vatex: A large-scale, high-quality multilingual dataset for video-and-language research," 2019 IEEE/CVF Interna tional Conference on Computer Vision (ICCV), pp. 4580–4590, Oct. 2019. doi:10.1109/iccv.2019.00468.

[26] H.-R. Huang et al., "Accelerating video captioning on Heterogeneous System Architectures," ACM Transactions on Architecture and Code Optimization, vol. 19, no. 3, pp. 1–25, May 2022. doi:10.1145/3527609.

[27] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., "Learning Transferable Visual Models from Natural Language Supervision," arXiv preprint arXiv:2103.00020, 2021.

[28] C. Schuhmann, R. Beaumont, R. Vencu, et al., "LAION-400M: Open Dataset of CLIP-Filtered 400 Million Image-Text Pairs", arXiv preprint arXiv:2111.02114, 2021.

[29] A. Li, H. Wang, L. Torresani, and G. Mori, "The AVA-Kinetics Localized Human Actions Video Dataset", arXiv preprint arXiv:2005.00214, 2020.

[30] A. A. Zhuravlev and K. A. Aksyonov, "Dependence comparison of the effectivness of software tools for splitting video into frames on format and resolution using a road survey as an example," 2024 International Russian Automation Conference (RusAutoCon), pp. 468–472, Sep. 2024. doi:10.1109/rusautocon61949.2024.10693956.

[31] S. R. Solanki and D. K. Khublani, "From script to screen: Unveiling text-to-video generation," Generative Artificial Intelligence, pp. 81–112, 2024. doi:10.1007/979-8-8688-0403-8 3.

[32] B. Taraghi, H. Amirpour, and C. Timmerer, "Multi-codec Ultra High Definition 8K MPEG-Dash Dataset," Proceedings of the 13th ACMMultimedia Systems Conference, Jun. 2022. doi:10.1145/3524273.3532889.

[33] M. S. Wajid, H. Terashima-Marin, P. Najafirad, and M. A. Wajid, "Deep learning and knowledge graph for image/video captioning: a review of datasets, evaluation metrics, and methods," Engineering Reports, vol. 6, no. 1, 2024, doi: 10.1002/eng2.12785.