electronics

MDPI

Article

Multi-Task Video Captioning with a Stepwise Multimodal Encoder

Zihao Liu, Xiaoyu Wu * and Ying Yu

State Key Laboratory of Media Convergence and Communication, Communication University of China, Beijing 100024, China
* Correspondence: wuxiaoyu@cuc.edu.cn

Abstract: Video captioning aims to generate a grammatical and accurate sentence to describe a video. Recent methods have mainly tackled this problem by considering multiple modalities, yet they have neglected the difference in modalities and the importance of shrinking the gap between video and text. This paper proposes a multi-task video-captioning method with a Stepwise Multimodal Encoder. The encoder can flexibly digest multiple modalities by assigning a proper encoding depth for each modality. We also exploit both video-to-text (V2T) and text-to-video (T2V) flows by adding an auxiliary task of video–text semantic matching. We successfully achieve state-of-the-art performance on two widely known datasets: MSVD and MSR-VTT: (1) with the MSVD dataset, our method achieves an 18% improvement in CIDEr; (2) with the MSR-VTT dataset, our method achieves a 6% improvement in CIDEr.

Keywords: multimodal fusion; video captioning; multi-task learning; computer vision; transformer; artificial intelligence

Citation: Liu, Z.; Wu, X.; Yu, Y. Multi-Task Video Captioning with a Stepwise Multimodal Encoder. *Electronics* **2022**, *11*, 2639. https://doi.org/10.3390/electronics11172639

Academic Editor: Jungong Han

Received: 14 July 2022
Accepted: 19 August 2022
Published: 23 August 2022

1. Introduction

Video captioning aims to describe video content with natural language. It is a valuable and challenging direction of both natural-language processing and computer vision. The task studied in this paper refers to generating a caption for a short video, which is different from a dense video-captioning task, as it generates more sentences for a longer video and localizes events additionally [1–3]. The advancement of video captioning can assist visually impaired people to use a social networking service [4], watch a movie [5], or travel independently by implementing the algorithm on smart devices as in [6–8]. Moreover, the study of video captioning can also promote various sub-tasks of video understanding (e.g., video retrieval [9] and visual question answering [10]).

To generate flawless captions, the model first needs to embed high-level semantic feature of videos (e.g., objects, the relationships between the objects, the motions of the objects and the connections of different clips). Then, it turns the features of the video space into a language space. Finally, it outputs grammatical and accurate captions. Hence, intuitively, video captioning can be seen as two stages: first understanding video and then describing video.

Currently, most dominant methods adopt the encoder–decoder sequence-to-sequence (Seq2Seq) framework inspired by machine translation. This classical framework uses the 2D Convolutional Neural Network (2D-CNN), the 3D Convolutional Neural Network (3D-CNN) or the latest Vision Transformers (ViT) [11] as the encoder, and uses Recurrent Neural Networks (RNNs) or Transformers [12] as the decoder. Researchers usually use off-the-shelf CNNs that perform well on large datasets as encoders, such as ResNet [13] and IncepResNetV2 [14]. In addition, 3D-CNN [15–17], RNN with attention mechanism [16–18] and self-attention [1,18–20] are often used to tackle video's problem of temporal dynamics.

Despite these improvements, considering the multi-modality in video and the multimodal nature of human perception, how to exploit that rich information has become a

focus of much attention. Video is not only images with a temporal dimension, but also a combination of motion information, audio information, overlaid text, speech information, etc. [21]. Existing methods still suffer from sufficiently using these. In this paper, we propose a Stepwise Multimodal Encoder (SME), which can flexibly and stepwise fuse various modalities by sharing one encoder within all modalities and encoding each with a different depth.

Moreover, video captioning is a cross-modal task. Thus, we believe it is essential to shrink the gap between video and text, which is often neglected by existing methods. In this paper, we use a Vision–Language Pretrained (VLP) model during the feature-extraction stage instead of a traditional Image-Classification Pretrained (ICP) model. Meanwhile, as shown in Figure 1, we also devise an auxiliary task that performs semantic matching between video and text. We exploit both the traditional video-to-text (V2T) flow (red box in Figure 1) and the text-to-video (T2V) flow (blue box in Figure 1). By sharing the parameters of the video encoder and optimizing the video–text semantic match loss, it can enhance the model's generalization.

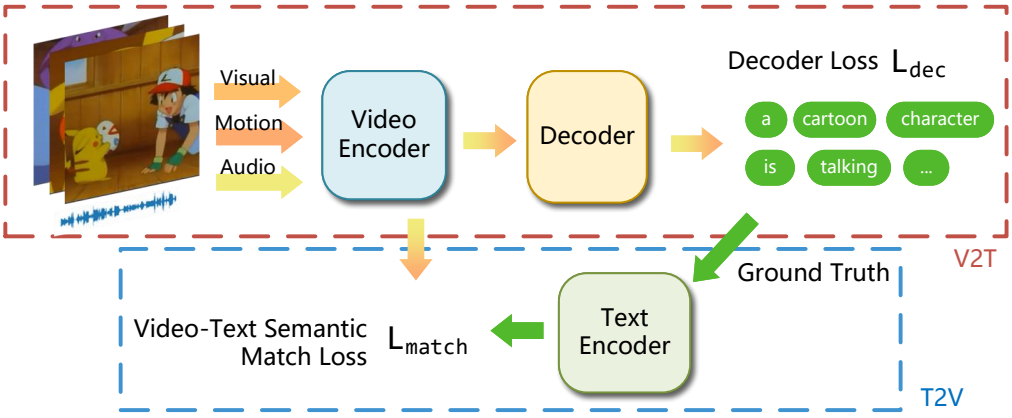

Figure 1. Use of two flows in our method. Traditional video-to-text (V2T) flow: the video encoder encodes the multimodal feature, and the decoder outputs the caption to obtain decoder loss; Text-to-video (T2V) flow: the text encoder encodes the ground-truth caption, and then the model calculates Video–Text Semantic Match Loss.

In summary, this work first extracts features of multiple modalities from the video using pretrained models, then performs feature fusion through a novel video encoder proposed in this paper, and finally decodes the fused feature to natural sentences. In addition, the encoder is also trained by an auxiliary task of semantic matching. The exploitation of multi-modality improves the accuracy of the output captions, and the multi-task learning enhances the generalization of the model. The main contributions of this work are summarized as follows:

- We propose a Stepwise Multimodal Encoder (SME), which can extend to multiple modalities and can adjust according to various modalities;
- We apply multi-task learning to further shrink the gap between the video modality and text modality;
- The experiment shows that our method outperforms several state-of-the-art methods on both widely used datasets: the MSVD dataset [22] and the MSR-VTT dataset [23].

2. Related Work

2.1. Video Captioning

With the rapid development in deep learning, models with encoder–decoder frameworks have become the mainstream in video-captioning tasks. Some works use an end-to-end approach [24], while the majority leverage pretrained models. Among them,

ResNet [13], InceptionResnetV2 [14], C3D [25] and I3D [26] are widely implemented as video feature extractors.

For the encoder, early work [27] applied mean pooling on feature vectors over a temporal dimension to avoid inconsistent length, while the temporal dynamic was ignored. Later, ref. [28] solved the problem using the Long Short-Term Memory (LSTM) network as encoder. As an improvement to this, ref. [29] leveraged the backward flow from the decoder's output to the encoder's input by using a reconstructor. In addition, ref. [30] leveraged an autoencoder to enhance the encoder. To embed richer temporal dynamics, ref. [31] applied hierarchical short Fourier transform and ref. [32] applied a hierarchical variant of LSTM. To learn more about the relationship between objects in the video, ref. [15] applied the graph convolutional network.

For the decoder, refs. [27,28] used LSTM as the decoder, and ref. [29,33] further improved it by applying an attention mechanism. To generate more accurate words, ref. [31] leveraged an object detector to build a vocabulary from which the decoder can select words. In another way, ref. [15] designed a teacher-recommended learning method that leveraged pretrained language models (e.g., BERT). The above methods generated captions autoregressively, which prompted the exposure-bias issue, while ref. [18] devised a bidirectional encoding structure to mitigate that. With the popularity of the Transformer, some works [1,2,19,20,24] used this to replace the traditional LSTM decoder.

2.2. Multimodal Learning of Video

Videos have multi-modality, and thus an advancing model should make full use of them, just as in human perception. Currently, many methods adopt multimodal learning in video-captioning tasks, where visual and motion features are most commonly used. Refs. [2,28,34] perform multimodal fusion at a later stage. Although they are flexible to accommodate multi-modality, each modality needs to be encoded and decoded separately, which brings more parameters and fewer opportunities for modal fusion. Refs. [1,20] perform multimodal fusion at an earlier stage, but lose the flexibility to accommodate more than two modalities. In addition, ref. [19] is a compromise in relation to the previous methods. For methods using object features [15,31,35], they prefer to design a separate module for the object feature.

2.3. Multi-Task Learning

Multi-task learning is a widely used paradigm that enhances a model's generalization ability by sharing parameters across multiple related tasks. Specifically in the video-captioning task, ref. [36] applies entailment generation and unsupervised video prediction tasks, refs. [30,37] apply a joint embedding task, and [38] applies an attribute-prediction task. In addition, in [39], the video-captioning task is one of the five fine-tuned tasks.

Current methods still do not make full use of multi-modality. Moreover, we believe the model's generalization should be given more attention due to the limitations of the dataset and the specificity of the task. In this paper, we propose a method that maintains both multimodal flexibility and the full opportunity for modal fusion. Furthermore, as for the improvement in generalization, we apply the Transformer's decoder with SCE-loss [40] and additionally add an auxiliary task. The task we apply is similar to [30,37], in which we further leverage a pretrained model instead of training from scratch. The loss we apply theoretically works similar to [15] but without a huge external model.

3. Proposed Methods

Figure 2 shows the modules that make up our model. First, multiple features are extracted from an untrimmed video, and then these features are concatenated and decorated to form a video representation. Next, the video representation is sent to the SME to encode and acquire a discriminative multimodal feature. This feature is then exploited in two tasks: a Video-Captioning Decoder, and Video–Text Semantic Matching. In the former task, the caption generator takes the video feature and outputs a caption autoregressively. In the

latter task, a text encoder derives the feature representing the whole sentence, which will be used to calculate a semantic match loss with the feature representing the entire video.

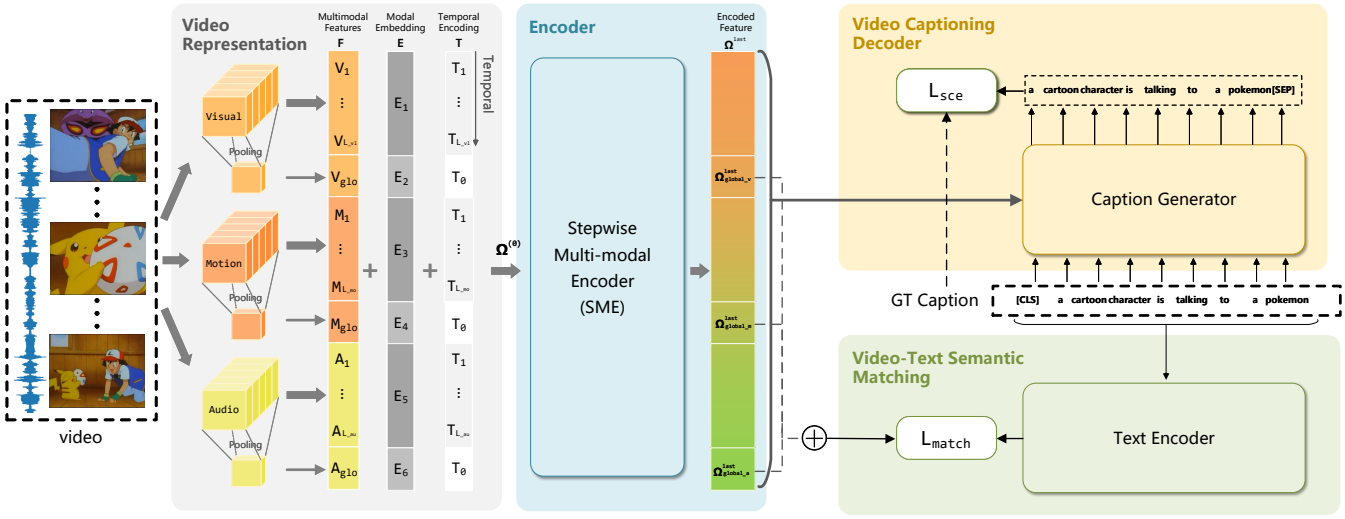

Figure 2. Overview of our multi-task multimodal method. Three modalities are used as an example in this figure. GT is short for Ground Truth. Our design contains four modules: Video Representation, Encoder, Video-Captioning Decoder and Video–Text Semantic Matching. Among them, the first two modules are shared by both tasks, which correspond to the last two modules each. ($\oplus$ refers to element-wise addition).

It is worth noting that our method is theoretically applicable to any N modalities, but we use three modalities as an example in the following description. We further elaborate on each module in the following sections.

3.1. Preliminary

Our method applies the Transformer network [12] as the backbone. Therefore, this section briefly introduces the original Transformer network before we describe our model. The Transformer is a Seq2Seq model consisting of an encoder and a decoder, among which self-attention (SA) plays a crucial role. The SA takes query, key and value as input, and to jointly attend more information, the Multi-Head Attention (MHA) is introduced, which performs multiple SAs with different projection matrices in parallel. In every encoder layer, MHA is used once, and in every decoder layer, MHA is used twice. A Feed-Forward Network is also applied to increase the non-linear fitting capability on both encoder and decoder. The decoder works autoregressively, which takes the previous results as inputs of the next step. Finally, the final outputs of the decoder are transformed to probabilities through a linear network and a SoftMax function.

3.2. Video Representation

This module aims to obtain a video representation that incorporates multi-modality and preserves temporal dynamics. The final video representation ($\mathbf{\Omega}^{(0)}$) is the sum of three vectors ($\mathcal{F}$, $\mathcal{E}$ and $\mathcal{T}$) as shown in Equation (1), where $\mathcal{F}$ is a multimodal feature vector, $\mathcal{E}$ is a modal embedding vector, and $\mathcal{T}$ is a temporal encoding vector. Among them, the multimodal feature vector is derived from the video features extracted by the pretrained models and contains most information, while the modal embedding vector and temporal encoding vector serve as supplementary information. By adding $\mathcal{E}$ and $\mathcal{T}$, the model can distinguish between different modalities concatenated together and learn the temporal order of the features.

$$\mathbf{\Omega}^{(0)} = \mathcal{F} + \mathcal{E} + \mathcal{T}. \tag{1}$$

3.2.1. Multimodal Feature Vector

The Multimodal feature vector is the most dominant vector. It contains most of the information in a video and is obtained by concatenating features of different modalities after projecting to an identical dimension. Take three modalities (visual, motion and audio) as an example, the original visual ($\boldsymbol{v} \in \mathbb{R}^{L_{vi} \times d_{vi}}$), motion ($\boldsymbol{\mu} \in \mathbb{R}^{L_{mo} \times d_{mo}}$), and audio ($\boldsymbol{a} \in \mathbb{R}^{L_{au} \times d_{au}}$) features are extracted from an untrimmed video by pretrained models, where L is the variable length of temporal dimension, d is the size of the feature vector, and vi, mo, au are short for *visual*, *motion*, and *audio*, respectively. Considering that the size of these features varies, we embed them into a space of the same shape:

$$\begin{cases} \boldsymbol{V} = \boldsymbol{v}\boldsymbol{W}^{e}_{vi} + \boldsymbol{b}^{e}_{vi}; \\ \boldsymbol{M} = \boldsymbol{\mu}\boldsymbol{W}^{e}_{mo} + \boldsymbol{b}^{e}_{mo}; \\ \boldsymbol{A} = \boldsymbol{a}\boldsymbol{W}^{e}_{au} + \boldsymbol{b}^{e}_{au}, \end{cases} \tag{2}$$

where $\boldsymbol{W}^{e}_{vi}, \boldsymbol{W}^{e}_{mo}, \boldsymbol{W}^{e}_{au}, \boldsymbol{b}^{e}_{vi}, \boldsymbol{b}^{e}_{mo}, \boldsymbol{b}^{e}_{au}$ are learnable parameters. We additionally extracted the global video-level features ($\boldsymbol{V}_{global}$, $\boldsymbol{M}_{global}$, $\boldsymbol{A}_{global}$) of each modality by taking the average over the temporal dimension on $\boldsymbol{V}, \boldsymbol{M}, \boldsymbol{A}$:

$$\begin{cases} \boldsymbol{V}_{global} = Average(\boldsymbol{V}); \\ \boldsymbol{M}_{global} = Average(\boldsymbol{M}); \\ \boldsymbol{A}_{global} = Average(\boldsymbol{A}), \end{cases} \tag{3}$$

Then we concatenate these features over the temporal dimension to form a concatenated multimodal feature $\mathcal{F}$:

$$\mathcal{F} = [\boldsymbol{V}, \boldsymbol{V}_{global}, \boldsymbol{M}, \boldsymbol{M}_{global}, \boldsymbol{A}, \boldsymbol{A}_{global}], \tag{4}$$

$$\mathcal{F} \in \mathbb{R}^{(L_{vi}+L_{mo}+L_{au}+3) \times d_{model}}, \tag{5}$$

where $(L_{vi} + L_{mo} + L_{au} + 3)$ is the total length (over temporal dimension) of the multimodal feature and d_{model} is the unified model dimension.

In particular, a VLP model called CLIP [41] is leveraged for visual feature extraction instead of an ICP model. The VLP is to train the model by minimizing the distance between the encoded visual features and the encoded language features. Thus, we believe the VLP models could shrink the gap between video and text and improve our task. Recent works [42–45] have shown that the visual encoder under VLP outperforms others in a variety of downstream tasks, and among them, the CLIP excels the most. Therefore, we take advantage of it and explore the approach to fusing it with other modalities.

3.2.2. Modal Embedding Vector

The modal embedding vector allows the model to distinguish between different modalities in $\mathcal{F}$. We distinguish different modalities by adding different learnable embedding vectors to different positions in $\mathcal{F}$. In addition, we treat global features specifically by assigning them to different embedding vectors.

Suppose there are N modalities. We initialize a one-hot vector $\boldsymbol{\tau}_j$ of length $2N$ for the features at each position (j) in $\mathcal{F}$ (The shape of $\boldsymbol{\tau}$ is $(L_{vi} + L_{mo} + L_{au} + 3) \times 2N$). The features of the *i-th* modality correspond to the $(2i-1)$*-th* dimension of this one-hot vector ($\tau_{j,2i-1}$), and the global features of the *i-th* modality correspond to the 2*i-th* dimension ($\tau_{j,2i}$). The value of the corresponding dimension is set to 1 and the others to 0. In this paper, we have three modalities, then the shape of $\boldsymbol{\tau}$ is $(L_{vi} + L_{mo} + L_{au} + 3) \times 6$. Therefore, the $\boldsymbol{\tau}_j$ of the global feature of the visual modality is $[0, 1, 0, 0, 0, 0]$, and the $\boldsymbol{\tau}_j$ of the non-global features of the audio modality is $[0, 0, 0, 0, 1, 0]$. After obtaining $\boldsymbol{\tau}$, we calculate the product of $\boldsymbol{\tau}$ and a learnable embedding matrix $\boldsymbol{W}_{emb}$ to obtain the modal embedding vector $\mathcal{E}$:

$$\mathcal{E} = \boldsymbol{\tau} \times \boldsymbol{W}_{emb} = [\underbrace{\boldsymbol{E}_1, \ldots, \boldsymbol{E}_1}_{L_{vi}}, \boldsymbol{E}_2, \underbrace{\boldsymbol{E}_3, \ldots, \boldsymbol{E}_3}_{L_{mo}}, \boldsymbol{E}_4, \underbrace{\boldsymbol{E}_5, \ldots, \boldsymbol{E}_5}_{L_{au}}, \boldsymbol{E}_6], \tag{6}$$

$$\mathcal{E} \in \mathbb{R}^{(L_{vi}+L_{mo}+L_{au}+3)\times d_{model}}, \tag{7}$$

where $\boldsymbol{W}_{emb} \in \mathbb{R}^{6\times d_{model}}$, L_{vi}, L_{mo}, L_{au} represent the length of visual, motion, and audio features, respectively (excluding global features), and the shape of $\mathcal{E}$ is same as $\mathcal{F}$. As shown in Equation (6), $\mathcal{E}$ can be divided into several parts, where $\boldsymbol{E}_k$ denotes the *k-th* embedding.

3.2.3. Temporal Encoding Vector

The temporal encoding vector injects temporal information. The purpose of this is to allow the model to know which period of the video the input features correspond to, respectively. Since the vanilla SA is not concerned with the order of the input sequence, we leverage the same temporal encoding as the Transformer [12].

We determine the temporal feature vector in terms of how the visual features are sampled for extraction. In this paper, we extract two frames per second as keyframes for visual features, which means that there will be one visual feature every 0.5 s. Therefore, the features extracted within $[0, 0.5\text{s})$ will be assigned to position 1 ($pos = 1$), the features within $[0.5\text{s}, 1.0\text{s})$ will be assigned to position 2 ($pos = 2$), and so on. Specifically, we assign $pos = 0$ to global features. This method can be adapted to different sampling frequencies.

Given position pos and dimension i, we use Equation (8) from [12] to calculate the value of the temporal encoding vector:

$$\begin{cases} T_{pos,2i} = sin(pos/10,000^{2i/d_{model}}); \\ T_{pos,2i+1} = cos(pos/10,000^{2i/d_{model}}). \end{cases} \tag{8}$$

where the temporal encodings have the same dimension d_{model} as $\mathcal{F}$ and $\mathcal{E}$. Therefore, the final sequence of temporal encoding takes the form:

$$\mathcal{T} = [\boldsymbol{T}_1, \ldots, \boldsymbol{T}_{L_{vi}}, \boldsymbol{T}_0, \boldsymbol{T}_1, \ldots, \boldsymbol{T}_{L_{mo}}, \boldsymbol{T}_0, \boldsymbol{T}_1, \ldots, \boldsymbol{T}_{L_{au}}, \boldsymbol{T}_0], \tag{9}$$

$$\mathcal{T} \in \mathbb{R}^{(L_{vi}+L_{mo}+L_{au}+3)\times d_{model}}, \tag{10}$$

where $\boldsymbol{T}_{pos} = \sum_{j=0}^{d_{model}} T_{pos,j}$, and L_{vi}, L_{mo}, L_{au} represent the length of visual, motion, and audio features, respectively (excluding global features). The shape of $\mathcal{T}$ is same as $\mathcal{F}$ and $\mathcal{E}$.

3.3. Stepwise Multimodal Encoder

The SME uses the Transformer's encoder as the backbone. Because the vanilla Transformer's encoder only adopts a single-modality input, and although variants [1,21] introduce multi-modality into it, they have not sufficiently considered the adaptability of the encoding depth between different modalities. This paper strengthens the model by assigning an appropriate encoding depth to each modality. Assume that the *i-th* modality requires N_i encoder layers and the maximum of N_i is N_{max}. Figure 3 illustrates three of our early ideas:

The *late-fusion* method (Figure 3a) fuses modalities at the last layer, and each modality must go through $N_i - 1$ separate layers before fusion. In this way, different modalities cannot attend to each other until last.

The *inter-fusion* method (Figure 3b) allows the fusion in the middle between modalities that reach the $(N_{max} - N_i + 1)$-*th* layer. This increases the occasion of modal fusion and reduces the parameters compared to the late-fusion method.

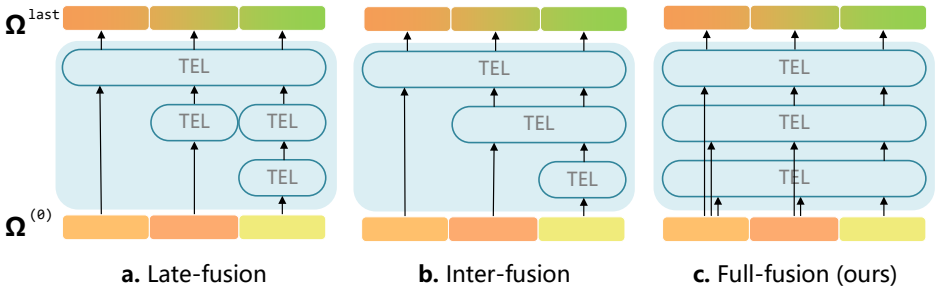

Figure 3. Three modal fusion strategies. TEL is short for Transformer Encoder Layer. The inputs of the TEL are concatenated at temporal dimension. The features of different modalities correspond to different colors. We assign one layer for the first modality, two for the second and three for the third as for example in this figure.

Finally, we devise a *full-fusion* method (Figure 3c) that further enhances the fusion and maintains a minimum of parameters. We assign one layer for the first modality, two for the second and three for the third as example. The first layer combines the information of all modalities and outputs the encoded feature of the third modality. Likewise, the second layer takes the original vector of the first two modalities and the encoded feature of the third modality as inputs. It then outputs the encoded feature of the last two modalities. By this point, the second modality is encoded with one layer, while the third modality is encoded with two layers. Since this process is step-like, we name it the Stepwise Multimodal Encoder. We formally describe our model as follows:

In the previous steps, we have obtained the video representation vector $\mathbf{\Omega}^{(0)}$. We represent the different parts of this vector with different subscripts:

$$\mathbf{\Omega}^{(0)} = [\ \overbrace{\mathbf{\Omega}_0, \ldots, \underbrace{\mathbf{\Omega}_{L_v}}_{\mathbf{\Omega}_{\{global\_v\}}}}^{\mathbf{\Omega}_{\{visual\}}}, \overbrace{\mathbf{\Omega}_{L_v+1}, \ldots, \underbrace{\mathbf{\Omega}_{L_v+L_m+1}}_{\mathbf{\Omega}_{\{global\_m\}}}}^{\mathbf{\Omega}_{\{motion\}}}, \overbrace{\mathbf{\Omega}_{L_v+L_m+2}, \ldots, \underbrace{\mathbf{\Omega}_{L_v+L_m+L_a+2}}_{\mathbf{\Omega}_{\{global\_a\}}}}^{\mathbf{\Omega}_{\{audio\}}}\ ]. \quad (11)$$

As shown in Equation (11), the length of $\mathbf{\Omega}^{(0)}$ is $L_v + L_m + L_a + 3$, and we denote the first $L_v + 1$ vectors as $\mathbf{\Omega}_{\{visual\}}$, the middle $L_m + 1$ vectors as $\mathbf{\Omega}_{\{motion\}}$, and the last $L_a + 1$ vectors as $\mathbf{\Omega}_{\{audio\}}$. We also denote the global feature vector as $\mathbf{\Omega}_{\{global\_v\}}, \mathbf{\Omega}_{\{global\_m\}}, \mathbf{\Omega}_{\{global\_a\}}$, whose position corresponds to the position of global features in $\boldsymbol{F}$.

In the case of Figure 3, the video representation vector is encoded by three Transformer Encoder Layers (the *i-th* layer is denoted as $TEL^{(i)}$). The output of the first layer is calculated as

$$\mathbf{\Omega}^{(1)} = TEL^{(1)}([\mathbf{\Omega}^{(0)}_{\{visual\}}, \mathbf{\Omega}^{(0)}_{\{motion\}}, \mathbf{\Omega}^{(0)}_{\{audio\}}]). \quad (12)$$

The input of the second layer uses part of the output of the first layer and part of the original input:

$$\mathbf{\Omega}^{(2)} = TEL^{(2)}([\mathbf{\Omega}^{(0)}_{\{visual\}}, \mathbf{\Omega}^{(0)}_{\{motion\}}, \mathbf{\Omega}^{(1)}_{\{audio\}}]), \quad (13)$$

The third level is similar to the second level as

$$\mathbf{\Omega}^{last} = \mathbf{\Omega}^{(3)} = TEL^{(3)}([\mathbf{\Omega}^{(0)}_{\{visual\}}, \mathbf{\Omega}^{(1)}_{\{motion\}}, \mathbf{\Omega}^{(2)}_{\{audio\}}]). \quad (14)$$

This method requires only N_{max} layers in total and the modality that needs to be encoded can attend to all other modalities in every layer. In this example $\mathbf{\Omega}^{last}$ is derived from the procedure described above; however, the encoding depth of each mode can be freely assigned, and we denote the final output as $\mathbf{\Omega}^{last}$. The assignment of encoding depth will be the hyperparameter of our method, whose ablation experiments can be found in Section 5.2.3.

3.4. Multi-Task Learning

3.4.1. Video-Captioning Decoder

We leverage the decoder of the Transformer as our caption generator's backbone (including embedding and positional encoding, denoted by *Decoder*). The caption generator takes $\mathbf{\Omega}^{last}$ as the input and generates captions autoregressively. Given the one-hot vector of previously generated words ($\boldsymbol{s}_{<t}$), the one-hot vector of the current time step ($\boldsymbol{s}_t$) is obtained by:

$$\boldsymbol{e}_t = Decoder(\mathbf{\Omega}^{last}, \boldsymbol{s}_{<t}), \tag{15}$$

$$\boldsymbol{p}_t = softmax(\boldsymbol{W}_p \boldsymbol{e}_t + \boldsymbol{b}_p), \tag{16}$$

$$\boldsymbol{s}_t = argmax(\boldsymbol{p}_t), \tag{17}$$

where $\boldsymbol{W}_p$, $\boldsymbol{b}_p$ and the parameters in *Decoder* are learnable.

Because the MSR-VTT and MSVD datasets contain a large amount of noise [46], and the same video often has multiple reasonable but different captions for this task, we use the SCE-loss [40] as our decoder loss to increase the robustness of our model. The traditional cross-entropy loss for input x and K-class dataset is:

$$\ell_{ce} = -\sum_{k=1}^{K} q(k|\boldsymbol{x}) \log p(k|\boldsymbol{x}), \tag{18}$$

where $q(k|\boldsymbol{x})$ is the ground-truth distribution over labels, and $p(k|\boldsymbol{x})$ is the probability distribution of the classifier. The reverse cross-entropy is then calculated as

$$\ell_{rce} = -\sum_{k=1}^{K} p(k|\boldsymbol{x}) \log q(k|\boldsymbol{x}). \tag{19}$$

The SCE-loss is defined as

$$\ell_{dec} = \alpha \ell_{ce} + (1-\alpha)\ell_{rce}, \tag{20}$$

where α is a hyperparameter that ranges in $[0, 1]$.

3.4.2. Video–Text Semantic Matching

Video–text semantic matching is an auxiliary task that additionally uses a pretrained text encoder. Considering the difficulty of training this task, we select the text encoder of CLIP to obtain the features of the ground-truth captions. For the same reason as using CLIP's visual encoder, we believe that the text encoder of VLP model is also beneficial in obtaining text features closer to the video space. We use the official code where the text encoder is a 12-layer 512-wide 8-head Transformer, and the feature vector is a fixed number of 512. To adapt to our model, we additionally add a linear projection layer to project this 512-d vector to a d_{model}-d vector. Formally, we use ψ to denote the off-the-shelf pretrained text encoder, $\boldsymbol{s}$ to denote the one-hot vector of the sentence, and the subscript *proj* to denote the linear projection layer. We first obtain the vector $\hat{\boldsymbol{s}}$ that represents the whole sentence as

$$\hat{\boldsymbol{s}} = \boldsymbol{W}_{proj}\psi(\boldsymbol{s}) + \boldsymbol{b}_{proj}. \tag{21}$$

Then we obtain the vector $\hat{\boldsymbol{\omega}}$ representing the whole video by

$$\hat{\boldsymbol{\omega}} = \frac{1}{3}(\mathbf{\Omega}^{last}_{\{global\_v\}} + \mathbf{\Omega}^{last}_{\{global\_m\}} + \mathbf{\Omega}^{last}_{\{global\_a\}}), \tag{22}$$

where *global_v*, *global_m* and *global_a* represent the global feature of visual, motion and audio modality.

We match the video and text by minimizing the InfoNCE loss [47] in a batch. Supposing the batch size is B, we denote the batch of text feature by $\hat{\mathbf{S}}$ and the batch of video feature by $\hat{\mathbf{\Omega}}$. The cosine similarity score is calculated by

$$\boldsymbol{W}^{score} = \frac{\hat{\mathbf{S}}}{Norm(\hat{\mathbf{S}})} \cdot \frac{\hat{\mathbf{\Omega}}}{Norm(\hat{\mathbf{\Omega}})}. \tag{23}$$

where $Norm$ function calculates the Frobenius norm. We then calculate the losses of video-to-text and text-to-video:

$$\ell_{v2t} = -\sum_{j=1}^{B} \log \frac{\boldsymbol{W}^{score}_{(i,j)}}{\sum_{i=1}^{B} \boldsymbol{W}^{score}_{(i,j)}}, \tag{24}$$

$$\ell_{t2v} = -\sum_{i=1}^{B} \log \frac{\boldsymbol{W}^{score}_{(i,j)}}{\sum_{j=1}^{B} \boldsymbol{W}^{score}_{(i,j)}}. \tag{25}$$

Finally, the video–text semantic match loss is obtained by adding the above two:

$$\ell_{match} = \ell_{v2t} + \ell_{t2v}. \tag{26}$$

3.4.3. Object Function

Finally, we set a hyperparameter β to combine the loss of two tasks:

$$\ell_{model} = \beta \ell_{dec} + (1 - \beta)\ell_{match}. \tag{27}$$

4. Experiment Setup

4.1. Datasets

We employ MSVD [22] and MSR-VTT [23] datasets for our experiments. The MSVD dataset consists of 1970 videos on the YouTube website and 80,827 English captions collected by the Amazon Mechanical Turk (AMT). Following the standard, we divide the videos into three parts: 1200 videos for training, 100 videos for validation, and 670 videos for testing. The MSR-VTT dataset is a more extensive dataset collected by AMT that contains 10,000 videos with 20 captions for every video clip. This dataset is divided into three parts: 6513 videos for training, 497 videos for validation, and 2990 videos for testing. A short natural sentence can describe every video clip in these two datasets.

4.2. Evaluation Metrics

The performances of generated captions are evaluated by four metrics in this paper: BLEU@4 [48], METEOR [49], ROUGE-L [50], CIDEr [51], which are abbreviated as B@4, M, R and C, respectively. The B@4 and R metrics calculate the n-gram overlap between the predicted result and the reference captions, with larger metrics indicating a more complete fit of the result to the reference captions. The difference is that the former calculates the accuracy while the latter calculates the recall. In natural language, the same meaning can be expressed by different words and different syntaxes. Therefore, the M metric considers synonyms while the C metric focuses more on whether the predicted result contains key information of the references rather than a complete overlap. We use the standard evaluation protocol from the Microsoft COCO evaluation server [52]. For all four metrics, a larger value is better.

4.3. Implementation Details

4.3.1. Environment

Our implementation is based on Python3 and Pytorch1.10. We train our model on four NVIDIA GTX TITAN X GPU with 12 G memory.

4.3.2. Pretrained Models

For video feature extraction, we select three pretrained models for visual, motion and audio modality. The 512-dimension visual features are extracted by CLIP-ViT-B/32 [41], which is a pretrained model that achieved the best results in all CLIP models. The 1024-dimension motion features are extracted by I3D [26] with RGB frames and dense optical flow images, where the optical flow images are extracted by Denseflow [53] using TV-L1 algorithm [54]. This model is widely used to extract motion features in similar tasks[1,2,19]. Specifically for the MSR-VTT dataset, following the details in [21], the 128-dimension audio features are extracted by VGGish [55] trained on YT8M dataset, which transfers CNN from vision to audio, and its pretrained models trained on large-scale dataset can effectively extract audio features. For text feature extraction in a text encoder, we use CLIP-ViT-B/32 model in the CLIP's official codes.

4.3.3. Preprocess

We extract visual features by sampling the videos at 2 fps, and extract motion features for every 64 consecutive frames from all video frames without overlap. For text preprocessing in the video-captioning task, we follow the method in BERT [56]: we use *[CLS]* as the start token and *[SEP]* as the end token of a sentence. The case of the sentence is ignored.

4.3.4. Learning Settings

During the training process, all parameters from pretrained models are frozen, and a dropout with a rate of 0.3 is adopted for regularization. For the Transformer-like architecture, the model dimension d_{model} is set as 768, the hidden state size of the feed-forward layer is set as 2048, and the number of decoder's layer is set as 3. We use the Adam optimizer [57] with a learning rate of 0.0001 and apply a cosine annealing schedule where the minimum learning rate is 0.00001, and the maximum number of iterations is 10. The training process stops when it reaches 30 epochs, or the sum of four metrics is not increased on the validation set for six epochs. The hyperparameter α is set to 0.5 empirically, and β is set to 0.4 according to the ablation study. During the inference, only the video-captioning task is processed.

5. Results And Analysis

5.1. Comparison to the State-of-the-Art

Table 1 shows the performances of our proposed model and several state-of-the-art methods on the MSVD dataset and MSR-VTT dataset. Due to the diversity of modalities, we list the pretrained model for visual feature extraction of those methods and other modalities they use. We did not use audio features in the MSVD dataset because it does not contain audio information.

Table 1. Performance comparisons on MSVD and MSR-VTT benchmarks. The best results and corresponding features are listed. B@4 is short for Bleu@4, M is short for METEOR, R is short for ROUGE, and C is short for CIDEr. MT-SME is short for our model's name—Multi-Task Stepwise Multimodal Encoder. The maximum value of each column is marked in bold.

Models	**Modality**		**MSVD**				**MSR-VTT**			
	Visual	**Others**	**B@4**	**M**	**R**	**C**	**B@4**	**M**	**R**	**C**
SibNet [30]	GoogleNet	-	54.2	34.8	71.7	88.2	40.9	27.5	60.2	47.5
OA-BTG [35]	ResNet-200	Object	**56.9**	36.2	-	90.6	41.4	28.2	-	46.9
ORG-TRL [15]	InceptionResnetV2	Motion & Object	54.3	36.4	73.9	95.2	43.6	28.8	62.1	50.9
GRU-EVE	InceptionResnetV2	Motion & Object	47.9	35	71.5	78.1	38.3	28.4	60.7	48.1
SBD [18]	ResNet152	-	47.7	34.3	69.4	81.6	39.2	27.8	59.4	44.5
TVT [19]	NasNet	Motion & Object	-	-	-	-	42.5	28.2	61.1	48.5
TVT	NasNet	Motion	53.2	35.2	-	86.8	40.1	27.9	59.6	47.7
CLIP4Caption [58]	CLIP	-	-	-	-	-	46.1	30.7	63.7	57.7
MT-SME (ours)	CLIP	Motion & Audio	-	-	-	-	**46.9**	**31.3**	**64.7**	**61.2**
MT-SME (ours)	CLIP	Motion	56.5	**39.2**	**76.9**	**112.7**	44.2	30.3	62.6	57.4

According to the results, our model outperforms most existing methods on both datasets and all metrics. Compared to some methods using object features, our method has better results without that. In addition, for the methods using same modalities (i.e., visual, motion and audio), our method outperforms them especially on CIDEr, where we achieve a 29.8% increase on the MSVD dataset and a 20.3% increase on the MSR-VTT dataset. For a fair comparison, we also compare with the most recent method that uses CLIP features, and our method is still better.

5.2. Ablation Study

The ablation study is mainly performed on the MSR-VTT dataset as it has more adequate and diverse data. In this section, we show the impact of SME and multi-task design in our model.

5.2.1. Effect of Multi-Modality

To validate the effectiveness of multi-modality, we report our experiments with different combinations of three selected modalities. In Table 2 and Figure 4, we only apply a simple Transformer's encoder as the baseline to control variables. The result shows that using multiple modalities is better than using any single one. Moreover, the audio feature performs better than the motion feature when combined with the visual feature, which might be attributable to the higher similarity between the visual and motion. We achieved the best results with three modalities; however, this is not much of an improvement over using two features.

Table 2. Ablation study of modalities on MSR-VTT dataset. The modalities are fused by the baseline encoder, a simple Transformer's encoder. The maximum value of each column is marked in bold.

Visual	Motion	Audio	B@4	M	R	C
✓			37.5	28.5	59.8	47.4
	✓		38.0	27.2	59.2	41.1
✓	✓		44.9	30.0	63.0	53.8
✓		✓	**46.9**	**31.0**	**64.8**	58.6
✓	✓	✓	45.7	30.9	64.3	**60.7**

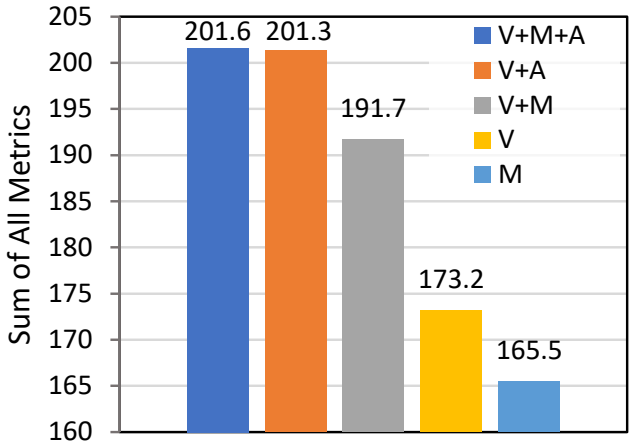

Figure 4. Ablation study of modalities on MSR-VTT dataset, using the sum of all metrics as overall metric. A larger value is better.

5.2.2. Effect of SME and Multi-Task Learning

We carried out this ablation study by adding modules on a single-modality and a three-modality baseline. Because the visual features contain the essential information in the video-captioning task compared with other features (e.g., motion and audio features), the single-modality baseline uses only visual features extracted by CLIP [41]. For a more explicit comparison, we name the single-modality model as the CLIP baseline and the

three-modality model as the multimodal baseline in Table 3. The baseline models use a simple Transformer's encoder, as in Table 2. The same hyper-parameters are used in two modules, which are obtained from subsequent ablation experiments.

In both single-modal and multimodal cases, multi-tasking brings an improvement. It improves the SUM metric by 2.7 and 2.0, respectively. By introducing the SME module, the SUM metric is improved by 29.5 compared to CLIP baseline and 1.1 compared to multimodal baseline. By applying both modules, the SUM metric is improved by 30.9 compared to CLIP baseline and 2.5 compared to multimodal baseline, which reaches the state-of-the-art result. Hence, we conclude that simply using the latest CLIP features does not achieve satisfactory results, and the performance is improved after adding our proposed modules both separately and together.

Table 3. Compared results without multi-task or stepwise design on the MSR-VTT dataset. SUM represents the sum of four metrics. The maximum value of each column is marked in bold. $\odot$ and $\otimes$ refer to two baselines.

Models	B@4	M	R	C	SUM
CLIP baseline ($\odot$)	37.5	28.5	59.8	47.4	173.2
$\odot$ + multi-task	38.4	28.8	60.4	48.3	175.9
multimodal baseline ($\otimes$)	45.7	30.9	64.3	60.7	201.6
$\otimes$ + stepwise	46.5	31.1	64.6	60.5	202.7
$\otimes$ + multi-task	**47.0**	30.8	**64.7**	61.1	203.6
$\otimes$ + stepwise & multi-task	46.9	**31.3**	**64.7**	**61.2**	204.1

5.2.3. The Evaluation of SME

Our stepwise design allows the assignment of different depths for each modality, which brings some hyper-parameters. Table 4 illustrates the experimental results of various settings. We carried out the experiment by gradually adding the depth of each modality. The learnable parameters of the model are related to the maximum depth. Since multiple metrics are used, each with a different focus as described in Section 4.2, it is not convincing to focus on any one metric alone. For example, a high C metric but low other metrics may indicate that the model captures key information but has poor fluency, a high B@4 metric but low other values may indicate that the model is good at learning fixed sentence patterns (e.g., "a man is . . . " or "talking about something"), but ignores key information. Since it has been an unsolved problem to find a metric that could accurately evaluate language-generating tasks, we simply used the sum of the four metrics to select the best model. As a result, we found the model performs best when assigning a deeper depth to motion modality. Additionally, we found that the heavier model tends to have higher B@4 and lower C, which we analyzed because more parameters led to overfitting. The model with more parameters tends to learn fixed sentence patterns in the dataset rather than key information.

Table 4. Experimental results of assigning different depths to each modality by SME. V, M, A represent the layer number of video, motion, and audio, respectively. SUM represents the sum of four metrics. The maximum value of each column is marked in bold.

V	M	A	B@4	M	R	C	SUM	Params
1	1	1	45.7	30.9	64.3	**60.7**	201.6	77.3 M
1	1	2	46.0	30.7	64.5	59.6	200.8	82.9 M
1	1	3	45.8	30.8	64.2	58.8	199.6	88.4 M
1	2	1	46.5	**31.1**	**64.6**	60.5	**202.7**	82.9 M
1	3	1	45.9	30.8	64.3	57.2	198.2	88.4 M
2	1	1	47.1	30.7	64.4	56.9	199.1	82.9 M
2	2	2	**47.4**	30.8	64.5	58.7	201.4	82.9 M
2	4	2	46.8	30.9	64.5	58.7	200.9	97.3 M

5.2.4. The Evaluation of Multi-Task Learning

As shown in Figure 5, we investigate the effect of assigning different weights to the auxiliary task on the results. The larger β is, the more weight the video-captioning task takes up, and the condition of $\beta = 1$ is the result of no multi-task learning. We can observe that the model performs better when β is set to 0.3 or 0.4.

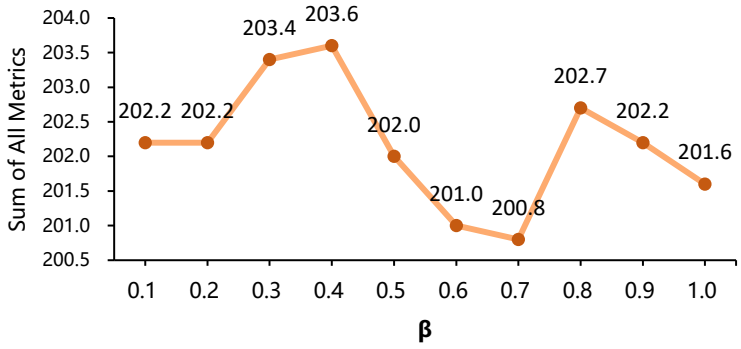

Figure 5. Experimental results of the hyperparameter of multi-task. Use the sum of all metrics as the overall metric. A larger value is better.

5.3. Qualitative Results

Several examples of the MSR-VTT dataset generated by our model are shown in Figure 6. Our model successfully captures the details of the video, such as the black shirt, glasses, desk, and Lego man. In addition, our model does not over-describe the video. For example, the baseline model describes in the top-left video that the man is talking about a computer while he is just using it. In addition, on the top-right video, the baseline model describes that the content is about a recent movie, but it is not. It could be seen as an example of overfitting, while our multi-task design mitigates it. However, when encountering some confusing videos (e.g., the bottom-right video, which is a still video of two images of Lego toys), our model accurately identified some elements (i.e., Lego) but incorrectly added relationships (i.e., talking).

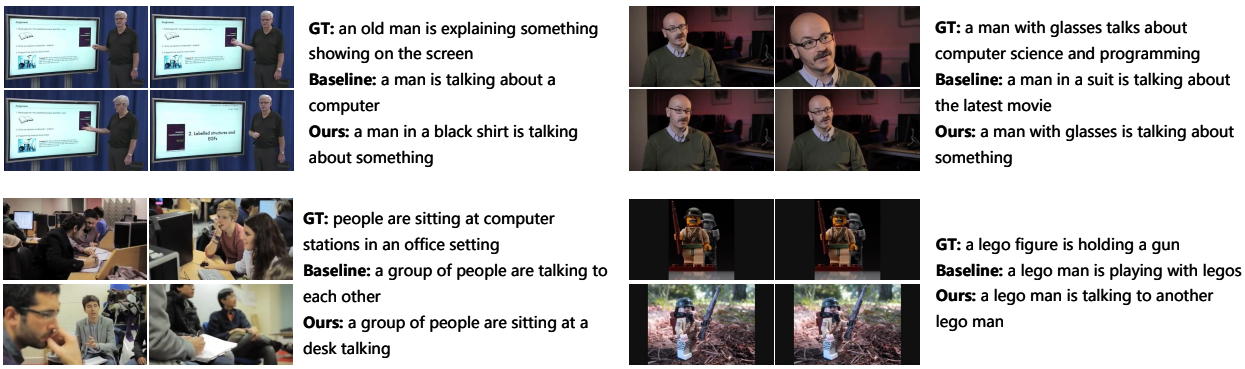

Figure 6. Examples of the MSR-VTT dataset. GT is short for ground truth, which we randomly select for one from candidates. Baseline uses only CLIP features without multi-modality and multi-task, while ours applies both.

The visualization of attention weights is shown in Figure 7. Our model accurately describes this music video. The model focused primarily on the visual features, and the motion features are given more attention when generating two verbs (i.e., singing, dancing). Additionally, as the verbs are related to audio, the audio features are used when generating two verbs.

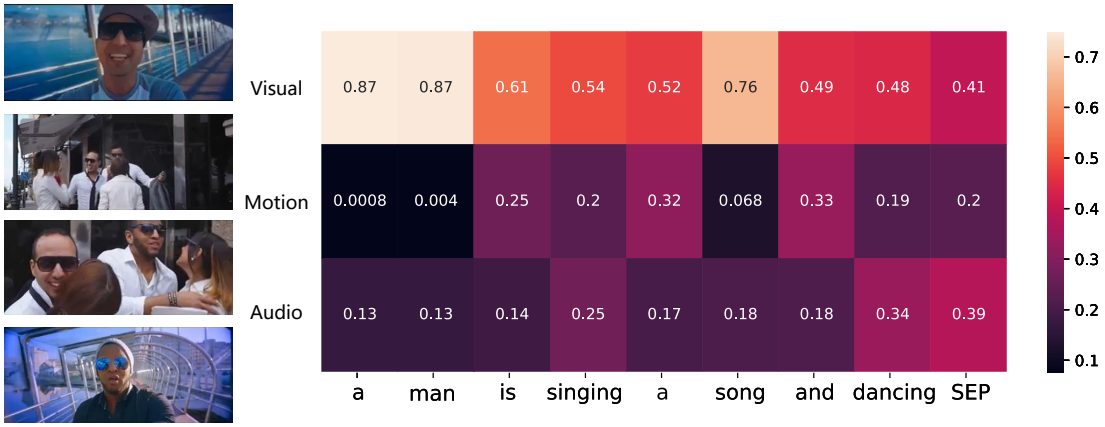

Figure 7. Visualization of attention weights of the last layer of caption generator. Each row represents a modality, and each column represents the attention weights of three modalities when generating a word. SEP is the end symbol. The example video is a music video of a pop song from MSR-VTT dataset.

6. Conclusions

This paper presents a video-captioning method using a novel Stepwise Multimodal Encoder (SME) and a multi-task design. In this method, SME exploits the multimodal nature of the video and considers the difference between various modalities. In addition, the multi-task design leverages the T2V flow, which mitigates overfitting problems. In general, this paper improves the quality of the output captions of the model and takes a step toward actually helping the visually impaired. The experiments show that our method achieves competitive results on MSVD and MSR-VTT datasets. We also provide an ablation study to verify each module and qualitative examples to visualize the results.

Although the number of parameters of the model in this paper is not too large, the number of parameters is still too large for practical use when the parameters of the feature-extraction model are included. We hope to improve it by using lighter feature extractors or performing model distillation in the future. We expect this work may further inspire more future studies for video captioning.

Author Contributions: Conceptualization, Z.L. and X.W.; methodology, Z.L.; software, Z.L.; validation, Z.L.; formal analysis, Z.L. and X.W.; investigation, Z.L. and X.W.; resources, X.W. and Y.Y.; data curation, Z.L.; writing—original draft preparation, Z.L.; writing—review and editing, Z.L., X.W. and Y.Y.; visualization, Z.L.; supervision, X.W. and Y.Y.; project administration, X.W.; funding acquisition, X.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work is funded by National Key R&D Program of China (No. 2021YFF0900701, No. 2021YFF0602103), National Natural Science Foundation of China (No. 61801441). We also thank for the research fund from the High-quality and Cutting-edge Disciplines Construction Project for Universities in Beijing (Internet Information, Communication University of China).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in https://www.microsoft.com/en-us/research/publication/msr-vtt-a-large-video-description-dataset-for-bridging-video-and-language/ and https://www.microsoft.com/en-us/download/details.aspx?id=52422&from=https%3A%2F%2Fresearch.microsoft.com%2Fen-us%2Fdownloads%2F38cf15fd-b8df-477e-a4e4-a4680caa75af%2F (accessed on 6 July 2022).

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

Abbreviations

The following abbreviations are used in this manuscript:

B@4	4-gram Bleu metric
CNN	Convolutional Neural Network
C	CIDEr metric
ICP	Image-Classification Pretrained
LSTM	Long Short-Term Memory network
M	METEOR metric
MHA	Multi-Head Attention
MT-SME	Multi-Task Stepwise Multimodal Encoder
R	ROUGE metric
RNN	Recurrent Neural Network
SME	Stepwise Multimodal Encoder
Seq2Seq	Sequence to Sequence
SA	Self-Attention
T2V	Video to Text
V2T	Text to Video
VLP	Vision–Language Pretrained
ViT	Vision Transformers

References

1. Iashin, V.; Rahtu, E. A better use of audio-visual cues: Dense video captioning with bi-modal transformer. In Proceedings of the 31st British Machine Vision Conference, Virtual, UK, 7–10 September 2021.
2. Iashin, V.; Rahtu, E. Multi-modal dense video captioning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 958–959.
3. Wang, T.; Zhang, R.; Lu, Z.; Zheng, F.; Cheng, R.; Luo, P. End-to-end dense video captioning with parallel decoding. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 6847–6857.
4. Voykinska, V.; Azenkot, S.; Wu, S.; Leshed, G. How blind people interact with visual content on social networking services. In Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing, San Francisco, CA, USA, 27 February 2016; pp. 1584–1595.
5. Rohrbach, A.; Torabi, A.; Rohrbach, M.; Tandon, N.; Pal, C.; Larochelle, H.; Courville, A.; Schiele, B. Movie description. *Int. J. Comput. Vis.* **2017**, *123*, 94–120. [CrossRef]
6. Mukhiddinov, M.; Cho, J. Smart glass system using deep learning for the blind and visually impaired. *Electronics* **2021**, *10*, 2756. [CrossRef]
7. Chen, S.; Yao, D.; Cao, H.; Shen, C. A novel approach to wearable image recognition systems to aid visually impaired people. *Appl. Sci.* **2019**, *9*, 3350. [CrossRef]
8. Spandonidis, C.; Spyropoulos, D.; Galiatsatos, N.; Giannopoulos, F.; Karageorgiou, D. Design of Smart Glasses that Enable Computer Vision for the Improvement of Autonomy of the Visually Impaired. *J. Eng. Sci. Technol. Rev.* **2021**, *14*, 113–118. [CrossRef]
9. Ding, S.; Qu, S.; Xi, Y.; Wan, S. A long video caption generation algorithm for big video data retrieval. *Future Gener. Comput. Syst.* **2019**, *93*, 583–595. [CrossRef]
10. Zeng, K.H.; Chen, T.H.; Chuang, C.Y.; Liao, Y.H.; Niebles, J.C.; Sun, M. Leveraging video descriptions to learn video question answering. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017.
11. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
12. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
13. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
14. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017.
15. Zhang, Z.; Shi, Y.; Yuan, C.; Li, B.; Wang, P.; Hu, W.; Zha, Z.J. Object relational graph with teacher-recommended learning for video captioning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 13278–13288.

16. Deng, J.; Li, L.; Zhang, B.; Wang, S.; Zha, Z.; Huang, Q. Syntax-guided hierarchical attention network for video captioning. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *32*, 880–892. [CrossRef]
17. Ye, H.; Li, G.; Qi, Y.; Wang, S.; Huang, Q.; Yang, M.H. Hierarchical Modular Network for Video Captioning. *arXiv* **2021**, arXiv:2111.12476.
18. Qi, S.; Yang, L. Video captioning via a symmetric bidirectional decoder. *IET Comput. Vis.* **2021**, *15*, 283–296. [CrossRef]
19. Chen, M.; Li, Y.; Zhang, Z.; Huang, S. Tvt: Two-view transformer network for video captioning. In Proceedings of the Asian Conference on Machine Learning, PMLR, Beijing, China, 14–16 November 2018; pp. 847–862.
20. Jin, T.; Huang, S.; Chen, M.; Li, Y.; Zhang, Z. SBAT: Video Captioning with Sparse Boundary-Aware Transformer. *arXiv* **2020**, arXiv:2007.11888.
21. Gabeur, V.; Sun, C.; Alahari, K.; Schmid, C. Multi-modal transformer for video retrieval. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Cham, Switzerland, 2020; pp. 214–229.
22. Chen, D.; Dolan, W.B. Collecting highly parallel data for paraphrase evaluation. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; pp. 190–200.
23. 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 Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 5288–5296.
24. Lin, K.; Li, L.; Lin, C.C.; Ahmed, F.; Gan, Z.; Liu, Z.; Lu, Y.; Wang, L. SwinBERT: End-to-End Transformers with Sparse Attention for Video Captioning. *arXiv* **2021**, arXiv:2111.13196.
25. Hara, K.; Kataoka, H.; Satoh, Y. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 6546–6555.
26. Carreira, J.; Zisserman, A. Quo vadis, action recognition? A new model and the kinetics dataset. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6299–6308.
27. Venugopalan, S.; Xu, H.; Donahue, J.; Rohrbach, M.; Mooney, R.; Saenko, K. Translating videos to natural language using deep recurrent neural networks. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics, Denver, CO, USA, 31 May–5 June 2015; pp. 1494–1504.
28. 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; pp. 4534–4542.
29. Wang, B.; Ma, L.; Zhang, W.; Liu, W. Reconstruction network for video captioning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7622–7631.
30. Liu, S.; Ren, Z.; Yuan, J. Sibnet: Sibling convolutional encoder for video captioning. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *43*, 3259–3272. [CrossRef] [PubMed]
31. Aafaq, N.; Akhtar, N.; Liu, W.; Gilani, S.Z.; Mian, A. Spatio-temporal dynamics and semantic attribute enriched visual encoding for video captioning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 12487–12496.
32. Pan, P.; Xu, Z.; Yang, Y.; Wu, F.; Zhuang, Y. Hierarchical recurrent neural encoder for video representation with application to captioning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1029–1038.
33. Yu, H.; Wang, J.; Huang, Z.; Yang, Y.; Xu, W. Video paragraph captioning using hierarchical recurrent neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 4584–4593.
34. Chadha, A.; Arora, G.; Kaloty, N. iPerceive: Applying common-sense reasoning to multi-modal dense video captioning and video question answering. *arXiv* **2020**, arXiv:2011.07735.
35. Zhang, J.; Peng, Y. Object-aware aggregation with bidirectional temporal graph for video captioning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 8327–8336.
36. Pasunuru, R.; Bansal, M. Multi-Task Video Captioning with Video and Entailment Generation. *arXiv* **2017**, arXiv:1704.07489.
37. Pan, Y.; Mei, T.; Yao, T.; Li, H.; Rui, Y. Jointly modeling embedding and translation to bridge video and language. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27-30 June 2016; pp. 4594–4602.
38. Li, L.; Gong, B. End-to-end video captioning with multitask reinforcement learning. In Proceedings of the 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa Village, HI, USA, 7–11 January 2019; pp. 339–348.
39. Luo, H.; Ji, L.; Shi, B.; Huang, H.; Duan, N.; Li, T.; Li, J.; Bharti, T.; Zhou, M. UniVL: A Unified Video and Language Pre-Training Model for Multimodal Understanding and Generation. *arXiv* **2020**, arXiv:2002.06353.
40. Wang, Y.; Ma, X.; Chen, Z.; Luo, Y.; Yi, J.; Bailey, J. Symmetric cross entropy for robust learning with noisy labels. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 322–330.
41. Radford, A.; Kim, J.W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. Learning transferable visual models from natural language supervision. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 18–24 July 2021; pp. 8748–8763.
42. Desai, K.; Johnson, J. Virtex: Learning visual representations from textual annotations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 11162–11173.
43. Sariyildiz, M.B.; Perez, J.; Larlus, D. Learning visual representations with caption annotations. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Cham, Switzerland, 2020; pp. 153–170.

44. Zhu, L.; Yang, Y. Actbert: Learning global-local video-text representations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 8746–8755.
45. Yu, F.; Tang, J.; Yin, W.; Sun, Y.; Tian, H.; Wu, H.; Wang, H. ERNIE-ViL: Knowledge Enhanced Vision-Language Representations Through Scene Graph. *arXiv* **2020**, arXiv:2006.16934.
46. Chen, H.; Li, J.; Frintrop, S.; Hu, X. Annotation Cleaning for the MSR-Video to Text Dataset. *arXiv* **2021**, arXiv:2102.06448.
47. van den Oord, A.; Li, Y.; Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv* **2018**, arXiv:1807.03748.
48. 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, Stroudsburg, PA, USA, 6–12 July 2002; pp. 311–318.
49. Denkowski, M.; Lavie, A. Meteor universal: Language specific translation evaluation for any target language. In Proceedings of the Ninth Workshop on Statistical Machine Translation, Baltimore, MD, USA, 26–27 June 2014; pp. 376–380.
50. Lin, C.Y. Rouge: A package for automatic evaluation of summaries. In Proceedings of the Text Summarization Branches Out, Barcelone, Spain, 25–26 July 2004; pp. 74–81.
51. Vedantam, R.; Lawrence Zitnick, C.; Parikh, D. Cider: Consensus-based image description evaluation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 4566–4575.
52. Chen, X.; Fang, H.; Lin, T.Y.; Vedantam, R.; Gupta, S.; Dollar, P.; Zitnick, C.L. Microsoft COCO Captions: Data Collection and Evaluation Server. *arXiv* **2015**, arXiv:1504.00325.
53. Wang, S.; Li, Z.; Zhao, Y.; Xiong, Y.; Wang, L.; Lin, D. Denseflow. 2020. Available online: https://github.com/open-mmlab/denseflow (accessed on 6 July 2022).
54. Zach, C.; Pock, T.; Bischof, H. A duality based approach for realtime tv-l 1 optical flow. In Proceedings of the Joint Pattern Recognition Symposium, Heidelberg, Germany, 12–14 September 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 214–223.
55. Hershey, S.; Chaudhuri, S.; Ellis, D.P.; Gemmeke, J.F.; Jansen, A.; Moore, R.C.; Plakal, M.; Platt, D.; Saurous, R.A.; Seybold, B.; et al. CNN architectures for large-scale audio classification. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; pp. 131–135.
56. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
57. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.
58. Tang, M.; Wang, Z.; Liu, Z.; Rao, F.; Li, D.; Li, X. CLIP4Caption: CLIP for Video Caption. In Proceedings of the 29th ACM International Conference on Multimedia, Virtual, China, 20–24 October 2021; pp. 4858–4862.