Towards Multi-modal Explainable Video Understanding

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Sciences

by

Kashu Yamazaki
University of Arkansas
Bachelor of Science in Mechanical Engineering, 2020

August 2023
University of Arkansas

This thesis is approved for recommendation to the Graduate Council.

Ngan Le, Ph.D.
Thesis Director

Khoa Luu, Ph.D.
Committee Member

Lu Zhang, Ph.D.
Committee Member

ABSTRACT

This thesis presents a novel approach to video understanding by emulating human perceptual processes and creating an explainable and coherent storytelling representation of video content. Central to this approach is the development of a Visual-Linguistic (VL) feature for an interpretable video representation and the creation of a Transformer-in-Transformer (TinT) decoder for modeling intra- and inter-event coherence in a video.

Drawing inspiration from the way humans comprehend scenes by breaking them down into visual and non-visual components, the proposed VL feature models a scene through three distinct modalities. These include: (i) a global visual environment, providing a broad contextual understanding of the scene; (ii) local visual main agents, focusing on key elements or entities in the video; and (iii) linguistic scene elements, incorporating semantically relevant language-based information for a comprehensive understanding of the scene. By integrating these multimodal features, the VL representation offers a rich, diverse, and interpretable view of video content, effectively bridging the gap between visual perception and linguistic description.

To ensure the temporal coherence and narrative structure of the video content, we introduce an autoregressive Transformer-in-Transformer (TinT) decoder. The TinT design consists of a nested architecture where the inner transformer models the intra-event coherency, capturing the semantic connections within individual events, while the outer transformer models the inter-event coherency, identifying the relationships and transitions between different events. This dual-layer transformer structure facilitates the generation of accurate and meaningful video descriptions that reflect the chronological and causal links in the video content.

Another crucial aspect of this work is the introduction of a novel VL contrastive loss function. This function plays an essential role in ensuring that the learned embedding features are semantically consistent with the video captions. By aligning the embeddings with the ground truth captions, the VL contrastive loss function enhances the model's performance and contributes to the quality of the generated descriptions.

The efficacy of our proposed methods is validated through comprehensive experiments on popular video understanding benchmarks. The results demonstrate superior performance in terms of both the accuracy and diversity of the generated captions, highlighting the potential of our approach in advancing the field of video understanding.

In conclusion, this thesis provides a promising pathway toward building explainable video understanding models. By emulating human perception processes, leveraging multimodal features, and incorporating a nested transformer design, we contribute a new perspective to the field, paving the way for more advanced and intuitive video understanding systems in the future.

ACKNOWLEDGEMENTS

I would like to acknowledge a number of people that supported, in various ways, the creation of this work.

First and foremost, I would like to express my greatest gratitude towards my supervisor *Ngan Le* for accepting me as her research student at AICV lab. I am profoundly grateful for her continuous support and motivation throughout the course of this research. Her insightful feedback, patience, and intellectual rigor have consistently driven this work to higher standards. Her guidance helped me navigate the complexities of this research and her mentorship was invaluable in not just the academic, but also in the personal aspects of this journey. Thank you, Ngan, for being a mentor of the highest caliber.

I am deeply grateful to all the members of the AICV lab whose invaluable assistance made this work possible. In particular, I would like to extend my heartfelt appreciation to *Khoa Vo* for his pioneering contributions to video understanding, which provided significant insights that greatly influenced the direction and outcomes of my research. Without his foundational work, this project would not have been as enriching and enlightening. I am sincerely thankful for his guidance and inspiration throughout this journey.

My special appreciation goes to my examination committee members, who have contributed their time and thoughtful critique, which played a significant role in shaping the final form of this thesis.

I would like to thank my family, whose love, understanding, and constant faith in my capabilities have been my pillars of strength. Their endless optimism and unwavering belief in my abilities have been the motivation that drove me to persist and persevere, even when faced with daunting challenges.

To all of you who have walked this journey with me, in ways big and small, seen and unseen, I express my profound gratitude. This accomplishment would not have been possible without you all. Thank you.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

Number *Page*

Chapter 1

INTRODUCTION AND BACKGROUND

This chapter lays the foundation for our exploration into the world of video captioning, providing both an introduction and a comprehensive background context. The materials presented here are crucial for understanding the progression of ideas and the core methodologies applied in the ensuing discussions and analyses. In Section 1.1, we delve into the foundational concepts of deep learning, including problem formulation, optimization techniques, and architectural components such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and Transformers. In Section 1.2, we delve into related research areas in video representation and its various approaches. We also discuss the advantages and challenges of this technique. In Section 1.3, we present the applications of video understanding, with a focus on video captioning. We highlight the benefits of using this technique in each of these domains. Finally, in Section 1.4, we outline our research contributions Our study is committed to augmenting our understanding of video captioning, a goal we aspire to achieve by enhancing the model's explainability and incorporating an inductive bias within the model. We believe that our research can contribute to the advancement of deep learning and its applications. We hope that the insights gleaned from this work will not only serve academic interest but also foster practical improvements in the wider realm of video understanding.

1.1 Deep Learning

Deep learning is a method of machine learning that leverages deep neural networks with multiple linear and non-linear layers[1] to learn complex patterns in data. With a hierarchical connection of neurons that transform input data into progressively more abstract representations, deep learning models can approximate highly nonlinear functions with unprecedented accuracy. In general, deep learning models try to model the data distribution p_{data} with a probability model p_θ with a learnable parameter θ. Although we don't have direct access, we assume the existence of the data distribution and we substitute it with empirical distribution $\hat{p}_{data}^{|\mathcal{D}|}$ of dataset $\mathcal{D}$. An empirical

[1]The most commonly used activation functions are $\sigma(x) = \frac{1}{1+\exp(-x)}$, $\tanh(x) = \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)}$, and $\mathrm{ReLU}(x) = \max(0, x)$.

distribution will converge to the data distribution as we increase the number of data samples: $\lim_{|\mathcal{D}|\to\infty} \hat{p}_{data}^{|\mathcal{D}|} = p_{data}$. To model the data distribution via an empirical distribution, we use a statistical distance and minimize the difference between the probability model and the empirical distribution. To this end, a family of f-divergence or integral probability metrics is often utilized in the context of deep learning. Kullback-Leibler (KL) divergence is one of the most commonly used instances of f-divergence[2] as a measure of two probability distributions over the same random variable. Consider two probability distributions P and Q. Usually, P represents the data or the observations, and Q represents a model or an approximation of P. Then, the KL divergence from Q to P is given as:

$$D_{KL}(P||Q) = \mathbb{E}_{x\sim P}\left[\log \frac{P(x)}{Q(x)}\right] = \mathbb{E}_{x\sim P}[\log P(x) - \log Q(x)] \tag{1.1}$$

We need to note that the KL divergence is not a true distance measure because it is asymmetric, i.e., $D_{KL}(P||Q) \neq D_{KL}(Q||P)$, as we notice from the formula.

As we want to find a parameter θ^* that minimizes the distance between the empirical distribution and the model, the training objective can be formulated as a minimization of KL divergence. For a discriminative model, we will model the conditional probability distribution $p(y|x)$ given an annotated dataset $\mathcal{D} = \{(x^{(n)}, y^{(n)})\}_{n=1}^{|\mathcal{D}|}$, which consists of a set of data x and its corresponding labels y.

$$\begin{aligned} \theta^* &= \operatorname*{argmin}_\theta \mathbb{E}_{x,y\sim\hat{p}_{data}^{|\mathcal{D}|}}\left[\log \hat{p}_{data}^{|\mathcal{D}|}(y|x) - \log p_\theta(y|x)\right] \\ &= \operatorname*{argmax}_\theta \sum_{i=1}^{|\mathcal{D}|} \log p_\theta(y_i|x_i) \end{aligned} \tag{1.2}$$

For a regression problem, we usually model the label y with a normal distribution with a mean determined by a function f with parameter θ and some fixed variance of σ_y^2, i.e., $y_i \sim \mathcal{N}(f_\theta(x_i), \sigma_y^2)$.

[2]KL divergence is the f-divergence generated by $f(x) = x \ln x$

Considering the probability density function of the normal distribution[3], we can rewrite the objective function as follows:

$$\underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{|\mathcal{D}|} \log p_\theta(y_i|x_i) = \underset{\theta}{\operatorname{argmin}} \frac{1}{2\sigma_y^2} \sum_{i=1}^{|\mathcal{D}|} (f_\theta(x_i) - y_i)^2 \tag{1.3}$$

Therefore, the maximization of the log-likelihood of the regression model is equivalent to the minimization of the squared error between the predicted and ground-truth values.

For a classification problem, we consider that the label y is drawn from a categorical distribution that follows class probabilities $\pi_1, \ldots, \pi_k$, i.e., $y_i \sim Cat(\pi_i)$, which can be represented by the output of the softmax function. The softmax function takes a vector $z \in \mathbb{R}^k$ with k values as its input and normalizes it into a probability distribution consisting of k probabilities. We apply this function to the output of f to model the categorical distribution of y, i.e., $p_\theta(y_i|x_i) = \operatorname{softmax}(f_\theta(x_i))$. The softmax function is defined as follows for $k \geq 1$:

$$\operatorname{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} \text{ for } i = 1, \ldots, k \tag{1.4}$$

Then, the objective function of the model can be rewritten as follows according to the probability mass function of a categorical distribution[4]:

$$\underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{|\mathcal{D}|} \log p_\theta(y_i|x_i) = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{k} -y_{i,j} \log \operatorname{softmax}(f_\theta(x_{i,j})) \tag{1.5}$$

This shows that the maximization of the log-likelihood of the classification model is equivalent to the minimization of the cross-entropy loss.

When trying to find the θ^* that satisfies Equation 1.3 or Equation 1.5, depending on the problem setup, we will find it very difficult to obtain an analytical solution because of the complexity of the

[3]The probability density function of normal distribution is: $f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$

[4]The probability mass function of a categorical distribution is: $f(x|p) = \prod_{i=1}^{k} p_i^{x_i}$

function f implemented with a neural network. Therefore, we instead use a numerical method called stochastic gradient descent (SGD). SGD is an iterative optimization technique used to optimize an objective function. It can be seen as a stochastic approximation of gradient descent optimization because it replaces the true gradient, which is calculated using the entire data set, with an estimated gradient calculated from a randomly selected subset of the data. As we consider the problem of empirical risk minimization, the value $L_i(\theta)$, which is the loss function at i^{th} observation in the dataset to approximate the true gradient of the empirical risk $L(\theta)$, is used to update the parameter θ:

$$\theta \leftarrow \theta - \eta \nabla L_i(\theta) \tag{1.6}$$

During the execution of the algorithm, it iteratively applies the above update for every training sample in the dataset. Multiple passes over the training set can be performed until the algorithm reaches convergence. To avoid cyclic patterns, the data can be shuffled before each pass. In practice, the gradient against more than one training sample, called a mini-batch, is used to alleviate a large deviation in gradient and slow convergence.

In the following subsections, we will introduce some widely-adopted network architectures in deep learning.

Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) refers to a category of fully connected, feedforward artificial neural networks. It is composed of a series of linear and non-linear operations arranged in an alternating fashion. Following is an instance of MLP that can be commonly seen in a Transformer block, which will be described in detail in the subsequent subsection:

$$\mathrm{MLP}(x) = \sigma(xW_1 + b_1)W_2 + b_2 \tag{1.7}$$

where W_1, W_2 are the linear transformation weights, b_1, b_2 are bias, and σ is a non-linearity, which is usually set to Rectified Linear Unit (ReLU) (Agarap, 2018) activation or Gaussian Error Linear Unit (GELU) (Hendrycks and Gimpel, 2016).

Convolutional Neural Networks

Convolutional Neural Networks (CNN) are a neural network architecture that is specialized for processing data with local structure. This includes time-series data like audio, which can be thought of as a 1D sequence with regular temporal sampling interval, image data, which can be thought of as a 2D grid of pixels, and video data, which can be thought of as a 3D grid of pixels with temporal consistency. CNN achieves this by using a convolutional layer that introduces locality and translation equivariant[5] maps by sharing the parameters of the linear transformation in every location. This property brings a strong *inductive bias* about the data structure into the model. The operation of a 2D convolutional layer can be summarized as:

$$x_{i,j}^{(l+1)} = \sum_{h=1}^{k_h} \sum_{w=1}^{k_w} W_{h,w} x_{(i+h),(j+w)}^{(l)} + b \tag{1.8}$$

where $W \in \mathbb{R}^{k_h \times k_w \times C^{(l)} \times C^{(l+1)}}$ is the convolutional weights that will be shared throughout the spacial dimension and b is a bias term.

Residual Network

Residual network (ResNet) (He, X. Zhang, et al., 2016) is a special instance of CNN introduced to train deep neural networks. ResNet explicitly reformulates the layers as learning residual functions with reference to the layer inputs by a residual connection. A residual connection is an identity mapping that allows for the direct flow of information from the earlier layer to the later layer, bypassing intermediate layers. The purpose of introducing residual connections is to facilitate the gradient flow during training and alleviate the vanishing gradient problem. Residual connections enable the network to effectively learn the desired mapping by focusing only on residual mapping.

[5]A function f is translation equivariant if it commutes with translations, i.e. if $f(TX) = T[f(X)]$

This mechanism facilitates improved learning and optimization within the network architecture and widely adopted in many advanced network architectures in recent years. The residual connection is formulated as:

$$x^{(l+1)} = f\left(x^{(l)}\right) + x^{(l)} \tag{1.9}$$

where $f(\cdot)$ is a subnetwork and l indicates the layer index.

Recurrent Neural Networks

Recurrent Neural Networks (RNN) is a neural network architecture that can deal with the arbitrary lengths of the input sequence. Given a input tensor with length T, RNN calculates the l^{th} hidden states recursively as:

$$h_t^{(l)} = \sigma\left(W^{(l)}\left[h_t^{(l-1)}; h_{t-1}^{(l)}\right] + b^{(l)}\right) \tag{1.10}$$

where the network parameters W and b are shared across the temporal dimension, and the first hidden state $h_0^{(1)}$ is the pseudo hidden state, which is usually initialized as zero-vector.

Gated RNNs

Since RNNs are deep neural networks in the temporal domain, they face difficulties in propagating errors that occurred at temporally distant time steps. Consequently, they tend to struggle with learning long-term dependencies, such as the relationship between the beginning and end of a sentence and instead focus on learning only immediate dependencies. To address this issue, gates were introduced to enable the balanced learning of short-term and long-term memories.

LSTM: long short-term memory (LSTM) is a representative architecture for gated RNNs (Hochreiter and Schmidhuber, 1997). LSTM utilizes three gates including input gate i, forget gate f, and output gate o besides the cell c to store long-term information.

$$\begin{bmatrix} \bar{h}_t^{(l)} \\ i_t^{(l)} \\ f_t^{(l)} \\ o_t^{(l)} \end{bmatrix} = \begin{bmatrix} \tanh \\ \sigma \\ \sigma \\ \sigma \end{bmatrix} \left(\begin{bmatrix} W_{\bar{h}}^{(l)} \\ W_i^{(l)} \\ W_f^{(l)} \\ W_o^{(l)} \end{bmatrix} \begin{bmatrix} h_t^{(l-1)} \\ h_{t-1}^{(l)} \end{bmatrix} + \begin{bmatrix} b_{\bar{h}}^{(l)} \\ b_i^{(l)} \\ b_f^{(l)} \\ b_o^{(l)} \end{bmatrix} \right) \tag{1.11}$$

$$c_t^{(l)} = i_t^{(l)} \odot \bar{h}_t^{(l)} + f_t^{(l)} \odot c_{t-1}^{(l)} \tag{1.12}$$

$$h_t^{(l)} = o_t^{(l)} \odot \tanh\left(c_t^{(l)}\right) \tag{1.13}$$

where $\odot$ denotes Hadamard product and σ refers to sigmoid function.

Here, $\bar{h}_t$ corresponds to the hidden state of an RNN, and its value is adjusted by the input gate i to update the cell c, which aggregates long-term information. Additionally, the forget gate f reduces the value of the past cell. In other words, we can say that the cell value is updated by adjusting the balance between short-term and long-term information using the input gate and the forget gate. Finally, the updated cell value is used to determine the final hidden state by adjusting it with the output gate.

GRU: gated recurrent unit (GRU) integrates long-term information into hidden state h, allowing it to operate with only two gates to manipulate the forgetting and updating of the state (Cho et al., 2014). The two gates are a reset gate r and an update gate z, which will be used to update the hidden states as follows:

$$\begin{bmatrix} r_t^{(l)} \\ z_t^{(l)} \end{bmatrix} = \sigma \left(\begin{bmatrix} W_r^{(l)} \\ W_z^{(l)} \end{bmatrix} \begin{bmatrix} h_t^{(l-1)} \\ h_{t-1}^{(l)} \end{bmatrix} + \begin{bmatrix} b_r^{(l)} \\ b_z^{(l)} \end{bmatrix} \right) \tag{1.14}$$

$$\widetilde{h}_t^{(l)} = \tanh\left(\begin{bmatrix} h_t^{(l-1)} \\ r_t^{(l)} \odot h_{t-1}^{(l)} \end{bmatrix}\right) \tag{1.15}$$

$$h_t^{(l)} = \left(1 - z_t^{(l)}\right) \odot \widetilde{h}_t^{(l)} + z_t^{(l)} \odot h_{t-1}^{(l)} \tag{1.16}$$

The performance of RNNs varies depending on the task, and it is not clear whether LSTM or GRU is superior (Jozefowicz, Zaremba, and Sutskever, 2015). However, GRU has fewer gates and does not require a cell, which means that under conditions where state variables are aligned, GRU can perform calculations similar to LSTM with less computational and memory requirements than LSTM.

Transformers

Transformers are a type of neural network architecture that leverage attention mechanisms to capture long-range dependencies in sequential data, such as text, speech, and image-patch sequences, with a cost of memory complexity. The vanilla Transformer (Vaswani et al., 2017) consists of two main components, namely, multi-head attention and a point-wise feed-forward network (FFN) with the residual connection. The multi-head attention is the parallel operation of Scaled Dot-Product Attention, which computes the softmax of the dot products of the query with all keys scaled by the inverse squared root of the keys' dimension to obtain the weights on the values. The operation of Scaled Dot-Product Attention can be summarized as follows using the query, key, and value matrices Q, K, V as below:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{1.17}$$

where the superscript $\top$ refers to the transpose operation, d_k is a dimension of key vector.

By projecting queries, keys, and values onto multiple subspaces using n different linear projections, multi-head attention can jointly attend to information from different representation sub-spaces at different positions. In contrast, using a single attention head averages over different subspaces and inhibits the model's ability to capture fine-grained patterns in the data. Considering the multi-head attention operation with h heads, the feature dimension d is split into h identical blocks, i.e., $\mathbb{R}^{L\times\frac{d}{h}\times h}$. Then, we can formulate the multi-head attention operation as:

$$\mathrm{MHA}(Q, K, V) = [\mathrm{head}_1; \dots; \mathrm{head}_h]W^O \tag{1.18}$$

where $[;]$ denotes the channel wise concatenation of tensor, $W^O \in \mathbb{R}^{d\times d}$ is the output projection weights, and each head head_i is calculated as:

$$\mathrm{head}_i = \mathrm{softmax}\left(\frac{Q_i K_i^\top}{\sqrt{d/h}} + M\right) V_i \tag{1.19}$$

where $Q_i = QW_i^Q, K_i = KW_i^K, V_i = VW_i^V \in \mathbb{R}^{L\times\frac{d}{h}}$ are the query, key, and value tensors, which are created by linearly projecting the input with learnable weights of W^Q, W^K, and $W^V \in \mathbb{R}^{\frac{d}{h}\times\frac{d}{h}}$. $M \in \mathbb{R}^{L\times L}$ is a mask matrix that assigns $-\infty$ to those elements we want to mask out after the softmax. The mask matrix is set to zero if no masking is required.

The attention mechanism is referred to by different names, depending on the route through which Q, K, and V are prepared. We provide a typical classification as follows:

1. *Self-Attention*: This prepares Q, K, and V by applying respective transformation matrices to a common input X, such that $Q = XW^Q$, $K = XW^K$, and $V = XW^V$. We can extract the inter-token relationship of the given sequence.

2. *Masked Self-Attention*: When used for autoregressive generation tasks, it is necessary to prevent the attention mechanism from referring to "future" elements. By applying a triangular mask M, we can prevent each element from accessing future elements, allowing the model to learn to predict future information using only past and current information.

3. *Cross-Attention*: This prepares Q, K, and V from different input matrices X and Y, such that $Q = XW^Q$, $K = YW^K$, and $V = YW^V$. This can be interpreted as a process where X extracts information from a different source Y.

Feed Forward Net (FFN) constitutes two-thirds of a Transformer model's parameters. Geva et al. demonstrated that the feed-forward layers in transformer-based language models function as key-value memories. Each key corresponds to textual patterns within the training examples, and each value triggers a distribution over the output vocabulary. In practical applications, a simple Multi-Layer Perceptron (MLP), comprising two linear transformations interspersed with a ReLU activation function, is typically used to implement the FFN. This is mathematically represented as:

$$\mathrm{FFN}(x) = \mathrm{ReLU}(xW_1 + b_1)W_2 + b_2 \tag{1.20}$$

Here, W_1 and W_2 denote the weight matrices, and b_1 and b_2 are the bias vectors. The inner dimension of the linear transformation is generally four times the transformer dimension. This is then reduced back to the original size, e.g., from $512 \rightarrow 2048 \rightarrow 512$, which preserves the input-output dimensions while still allowing the model to learn more complex representations.

Extending to Vision Tasks

Traditionally, Convolutional Neural Networks (CNNs) have been the dominant architecture for visual tasks such as image classification, object detection, and segmentation. However, the success of Transformers in natural language processing prompted researchers to explore their potential in computer vision as well. Transformers were first introduced to the vision tasks by embedding 16×16 pixels into a single patch that is treated equally as the language tokens (Dosovitskiy et al., 2021). By applying self-attention mechanisms to capture global dependencies and modeling relationships between patches, Vision Transformers (ViTs) can learn powerful representations for image understanding. However, this advantage comes at the expense of requiring larger amounts of data for effective training. In other words, when dealing with small-scale dataset, the inductive bias that we bring to the model plays a crucial role.

Scaling Laws and Potential of Transformers

The Scaling Laws of Transformers present a fascinating aspect of their functioning. J. Kaplan et al. delineated a power law relationship that can predict the test loss of a Transformer which models language in an autoregressive fashion. This relationship becomes relevant when the Transformer's performance is restrained by certain factors, namely the number of non-embedding parameters (N), the size of the dataset (D), or the optimally allocated compute budget (C_{min}).

$$L(x) = L_\infty + \left(\frac{x_0}{x}\right)^{\alpha_x}, \ x \in \{N, D, C_{min}\} \tag{1.21}$$

This demonstrates the potential for enhancing language model performance in a predictable manner through the scaling up of models, datasets, and computational resources. Later, this Scaling Law was shown to also apply to Transformers (autoregressive generative models) in domains such as images, videos, image-text, and mathematical formulas (Henighan et al., 2020). Following this, various companies and research groups have undertaken efforts to upscale their models. These large-scale models are referred to as foundational models, and there is a growing trend of using these pre-trained models in zero-shot or few-shot manner.

1.2 Spatiotemporal Feature Learning

The effective extraction of spatiotemporal features from video data is fundamental to any video understanding task. It enables the recognition of complex activities occurring over time while considering both spatial and temporal information. To perform video understanding tasks successfully, it is vital to incorporate both spatial and temporal dimensions, which contain visual and motion information, respectively. As the field has advanced, various approaches have been developed to handle this intricate process, each with its unique strengths and limitations.

This section delves into several of the prominent methods used in spatiotemporal feature learning,

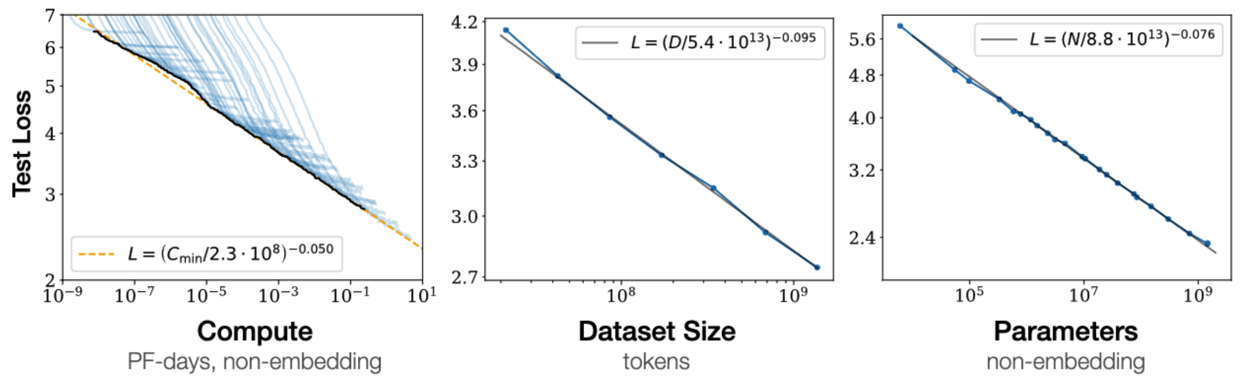


Figure 1.1: Power-law scaling laws of autoregressive language model (adopted from J. Kaplan et al., 2020)

covering a range of techniques from 3D Convolutional Networks (C3D), Inflated 3D ConvNet (I3D), and SlowFast Networks, to Cooperative Hierarchical Transformer (COOT) - from traditional 3D convolution-based methods to more recent transformer-based architectures.

3D Convolutional Networks (C3D)

C3D (Tran et al., 2014) is a video representation method that also uses 3D convolutions to learn spatio-temporal features from videos. Unlike 2D convolutions that operate on individual frames, 3D convolutions operate on video clips, allowing the model to learn temporal information. C3D demonstrates strong generalization capability and can serve as a versatile feature extractor for a range of video processing tasks. Many researchers have chosen to use C3D primarily as a feature extraction tool for varying applications rather than opting for network modifications or fine-tuning.

Inflated 3D ConvNet (I3D)

I3D (Carreira and Zisserman, 2017) extends the traditional 2D CNNs to 3D, allowing them to learn spatio-temporal features directly from videos. The I3D model is pre-trained on a large dataset of videos and then fine-tuned for specific tasks. This method has shown excellent performance in video classification and action recognition tasks.

SlowFast Networks (SlowFast)

The key idea behind SlowFast (Feichtenhofer et al., 2018) is to process a video through two parallel pathways: a slow pathway and a fast pathway. The slow pathway operates at a low frame rate and captures spatial semantics, while the fast pathway operates at a high frame rate to capture motion at fine temporal resolution. The outputs of these two pathways are fused to produce the final representation. This method is computationally efficient and has achieved state-of-the-art results on several video understanding benchmarks.

Cooperative Hierarchical Transformer (COOT)

COOT (Ging et al., 2020) is a transformer-based method for video understanding. It uses a hierarchical transformer architecture that operates at two levels: clip-level and video-level. The clip-level transformer captures local temporal dependencies, while the video-level transformer captures global temporal dependencies. The key feature of COOT is its cooperative learning mechanism, which allows the clip-level and video-level transformers to interact and learn from each other. This method has shown promising results in video understanding tasks, including video captioning and video question answering.

1.3 Applications in Video Understanding - Video Captioning

The current era is one of rapid technological development, where data takes the shape of not only texts, figures, or individual images but extends to complex, dynamic entities like videos. The ability to interpret and analyze videos has been a subject of intense study, primarily in areas such as surveillance, entertainment, healthcare, and autonomous driving. In this section, we will explore the applications of video understanding with a particular focus on captioning tasks.

Video captioning is an interdisciplinary research area that combines computer vision and natural language processing. The primary objective is to generate descriptive and meaningful sentences that accurately represent the content of a video. This task is challenging due to the inherent complexity of understanding and interpreting video content, which includes recognizing objects, actions, and events, and then translating these visual elements into coherent and contextually appropriate language. The task is further complicated by the temporal dimension of videos, which requires understanding the sequence and progression of events.

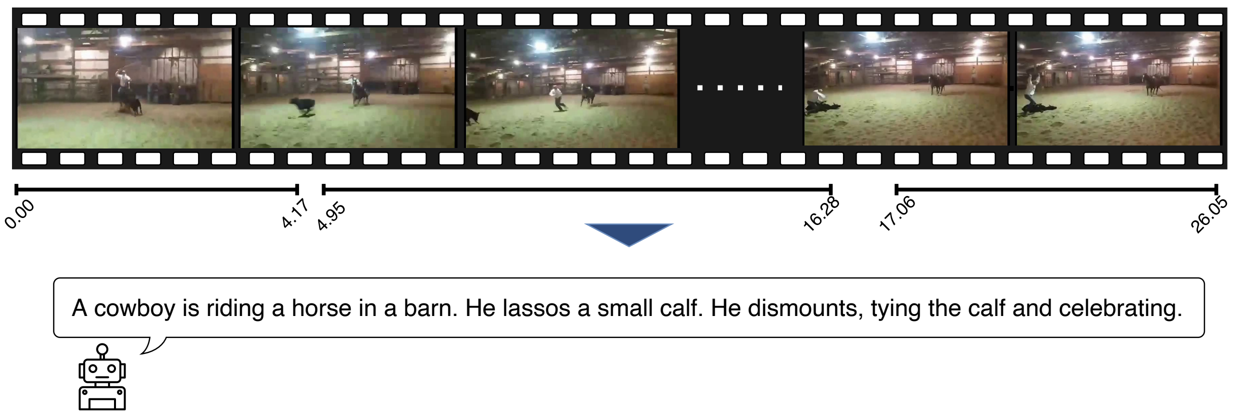

0.00
4.17
4.95
16.28
17.06
26.05
A cowboy is riding a horse in a barn. He lassos a small calf. He dismounts, tying the calf and celebrating.

Figure 1.2: An illustration of the video paragraph captioning (VPC) task. Given the video stream and the corresponding action intervals (top), we aim to produce coherent video paragraph captions that describe the video content according to the action intervals (bottom).

Dense Video Captioning

An important branch of video captioning is dense video captioning (DVC) (Krishna et al., 2017), which requires generating a list of temporal event proposals and the associated sentence description of each event to form a coherent paragraph description of a video. DVC has been implemented by various approaches including visual feature only (Krishna et al., 2017; Li et al., 2018; L. Zhou, Y. Zhou, et al., 2018; Mun et al., 2019; Deng et al., 2021) or multimodal features such as audio (Rahman, B. Xu, and Sigal, 2019), speech (Shi et al., 2019; Iashin and Rahtu, 2020), and both Iashin and Rahtu, 2020).

Video Paragraph Captioning

As a simplified version of DVC, video paragraph captioning (VPC) (Park et al., 2019) concentrates on generating better paragraph captions given a set of event segments in a video, which eases the requirement of event proposal generation. In general, a VPC model consists of two main components: an encoder to represent each event segment as a feature; and a decoder to generate captions while maintaining the consistency within each event and the coherence among all sentences of the generated paragraph. L. Zhou, Y. Zhou, et al. first introduced the transformer to the VPC task known as Vanilla Transformer, where each event is decoded individually without knowing the coherence between sentences. To address this limitation, Lei et al. modified the Transformer-XL (Z. Dai et al., 2019) and proposed MART. MART decodes the caption to learn word-level dependencies by a transformer while modeling the paragraph coherence based on GRU (Chung et al., 2014). Different from the existing VPC methods which utilize pre-trained backbone networks to extract features, Yamazaki et al. inherits the merits of both vision and language models and proposed VLCap. However, all previous works rely on RNN-based setup for the inter-event modeling and are limited in capturing long-range dependencies as well as prone to suffer from the problem of gradient vanishing (Pascanu, Mikolov, and Bengio, 2013).

1.4 Research Questions and Contributions

In this thesis, our primary objective is to enhance the understanding of video captioning. We seek to accomplish this by focusing on two critical areas: bolstering the model's explainability and integrating an inductive bias within the model. The key research questions and contributions that guide and characterize this work are presented below:

- *Question 1*: How can we represent video content in an explainable manner making it beneficial not only for the task of video captioning but also for humans to interpret?

  In Section 2.2, we present a novel video representation framework called VL Encoder that models a video with (i) environment, (ii) main agents, (iii) scene elements, and their interactions. The VL Encoder learns to selectively utilize features of each modality via Hybrid Attention Mechanism (HAM) that will be introduced in 2.1. The HAM plays a crucial role in VL Encoder enabling us to examine and comprehend which components have the most influence on video representation.

- *Question 2*: How can we decode the salient features into comprehensive captions while simultaneously modeling dependencies within and between events to generate a coherent narrative?

  In Section 3.3, we introduce our tailored Transformer-in-Transformer (TinT) decoder. Uniquely designed, this TinT decoder is capable of capturing both the intra- and inter-dependencies of events found within a video. This intricate process allows it to generate captions that maintain coherence and accuracy, effectively encapsulating the content of the video.

- *Question 3*: What strategies can we employ to better align both visual and linguistic information, thereby improving the overall effectiveness of video captioning?

  In Section 3.4, we proposed a VL contrastive loss function designed to ensure the learned embedding features exhibit semantic consistency with their corresponding video captions. This function works by aligning the feature embeddings with the ground-truth captions.

Much of the works in this thesis appear in the following publications:

1. **K. Yamazaki**, K. Vo, S. Truong, B. Raj, N. Le "VLTinT: Visual-Linguistic Transformer-in-Transformer for Coherent Video Paragraph Captioning," Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI) - (**Oral Presentation**), 2023.

2. **K. Yamazaki**, S. Truong, K. Vo, M. Kidd, C. Rainwater, K. Luu, N. Le "VLCap: Vision-Language with Contrastive Learning for Coherent Video Paragraph Captioning," IEEE International Conference on Image Processing (ICIP), 2022.

3. K. Vo, S. Truong*, **K. Yamazaki***, B. Raj, M. Tran, N. Le "AOE-Net: Entities Interactions Modeling with Adaptive Attention Mechanism for Temporal Action Proposals Generation," International Journal of Computer Vision, 2022.

Chapter 2
EXPLAINABLE VIDEO REPRESENTATION

In this chapter, we will introduce our effort towards explainable video representation. Video representation aims to transform raw video data into meaningful and manageable forms, providing a way for systems to understand the embedded information. In the pursuit of accurate understanding, models have become increasingly sophisticated and complex. However, this complexity often comes with a price, rendering these models as "black boxes", challenging to understand and interpret. The quest for explainability is thus born out of the need to decipher these black boxes and illuminate their inner workings. Explainable video representation, consequently, is not just about how videos are interpreted and processed, but about how the models used to do so can be understood. In other words, how can we not only make our models perform accurately but also make them transparent, allowing humans to understand their decision-making processes?

In this chapter, we will explore this exciting frontier, starting with an important building block called **Hybrid Attention Mechanism (HAM)** and then we introduce our proposed **Visual-Linguistic (VL) Encoder**. We then delve into a comprehensive discussion of each of the Encoder's components, providing a description that elucidates their design and functionality.

2.1 Hybrid Attention Mechanism (HAM)

The Hybrid Attention Mechanism (HAM) combines the advantages of both hard attention (Patro and Namboodiri, 2018) and self-attention (Vaswani et al., 2017). It judiciously selects a subset of representative features from a given set of input features, examines their mutual relationships, and integrates them into a unified representation. The concept of HAM was first introduced by (Vo, Joo*, et al., 2021), and since then, it has found successful applications in video analysis domains such as action localization (Vo, Yamazaki, Nguyen, et al., 2022; Vo, Truong*, et al., 2023). A visual representation of the HAM workflow is presented in Fig. 2.1, and its mathematical formulation is as follows:

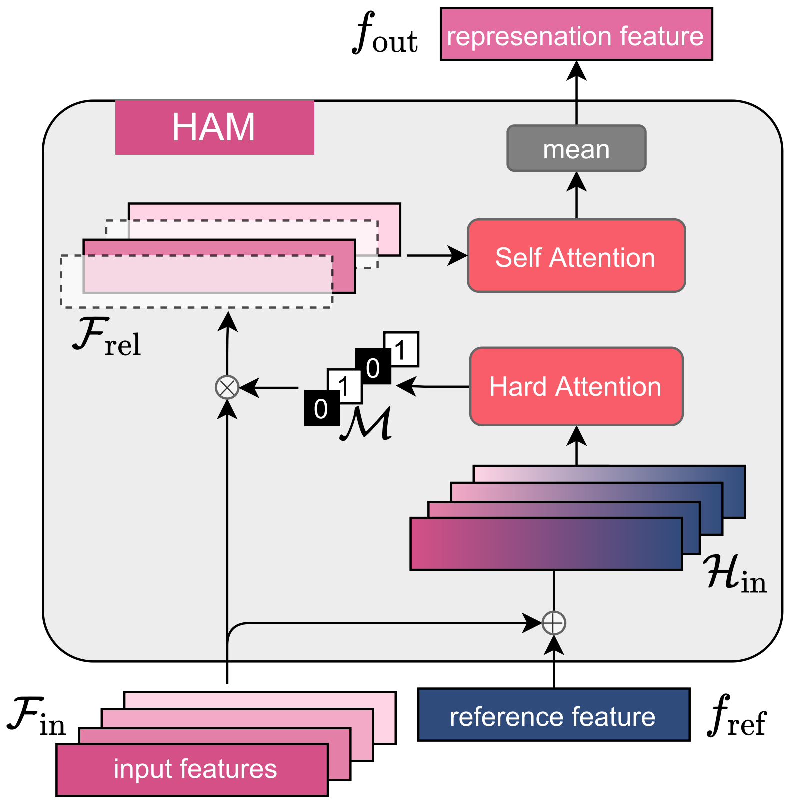


Figure 2.1: Illustration of Hybrid Attention Mechanism (HAM). HAM is capable of selecting and representing an arbitrary number of representative features from the input features $\mathcal{F}_{in}$ with guidance from reference feature f_{ref}.

$$\mathcal{H}_{\text{in}} = \mathcal{F}_{\text{in}} \oplus f_{\text{ref}} \tag{2.1a}$$

$$C = \text{softmax}(||\mathcal{H}_{\text{in}}||_2) \tag{2.1b}$$

$$\mathcal{M} = C > \frac{1}{N_{in}} \tag{2.1c}$$

$$f_{\text{out}} = \text{mean}(\text{MHA}(\mathcal{F}_{\text{in}} \odot \mathcal{M})) \tag{2.1d}$$

Eq.2.1a propagates the reference feature f_{ref} across all input features in $\mathcal{F}_{\text{in}}$, employing element-wise addition to establish a set of hidden features $\mathcal{H}_{\text{in}}$. Subsequently, Eq.2.1b calculates the $L2$-norm value for each hidden feature within $\mathcal{H}_{\text{in}}$ and re-normalizes these values using softmax to ensure their sum equals 1.0. As a result, the value associated with each input feature signifies its probability

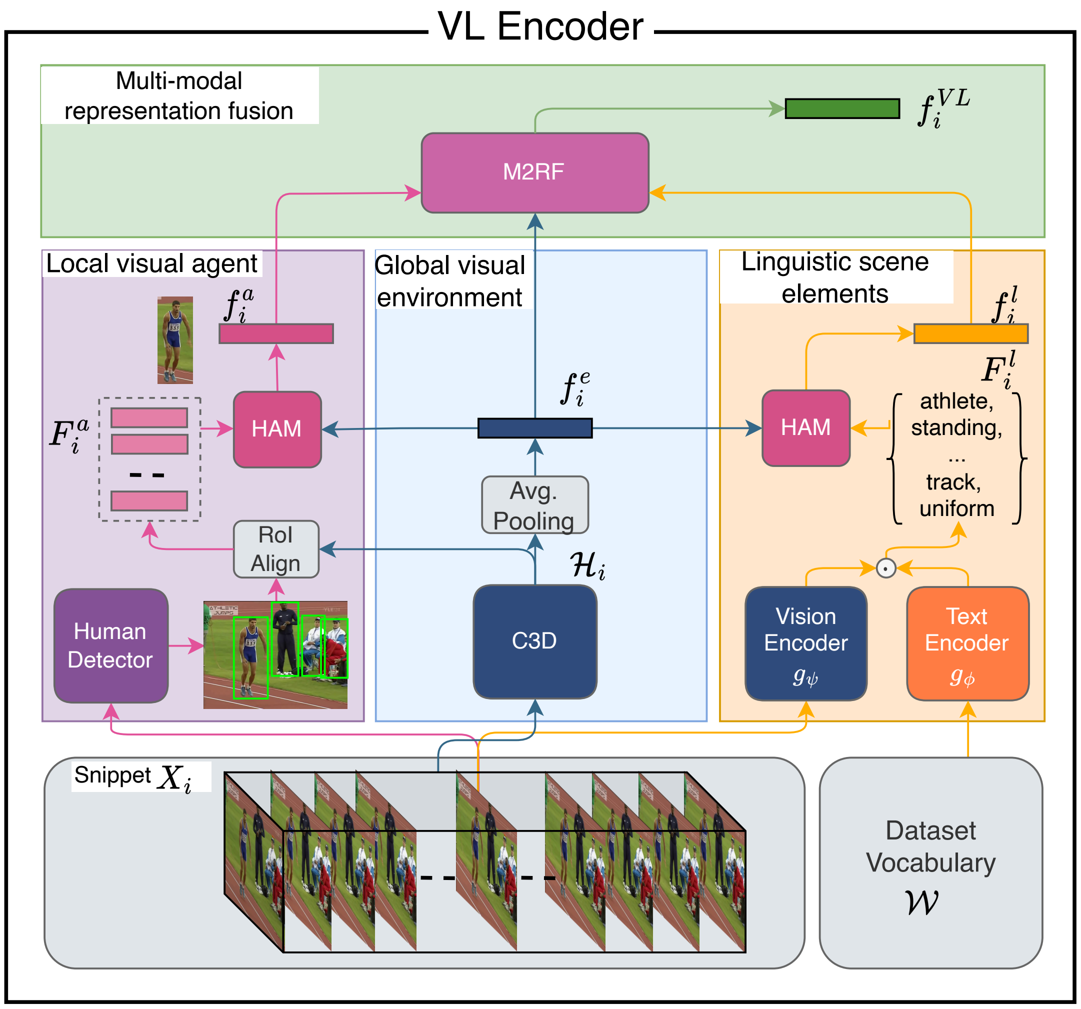


Figure 2.2: Depiction of our Visual-Linguistic (VL) Encoder's overall architecture: Provided a snippet X_i, the encoder concurrently derives local visual features from primary agents, global visual features from the surrounding environment, and linguistically relevant scene elements. It subsequently models the interactions amongst these three modalities using our Multi-modal Representation Fusion (M2RF) module.

of being a representative feature of the entire input feature set.

The adaptive threshold defined in Eq.2.1c generates a mask $\mathcal{M}$, which screens out features deemed non-representative. Finally, Eq.2.1d employs self-attention to discern mutual relationships amongst the selected input features, which are then averaged to yield a single representation feature $f_{\text{out}} \in \mathbb{R}^{d_{\text{in}}}$ representing the entire set of input features.

2.2 Visual-Linguistic (VL) Encoder

Our VL encoder is responsible for comprehensively representing each snippet X_i of an event into a representative feature to compose a sequence of snippet features for the decoder. Given an event $e = (e^b, e^e)$ and its corresponding video frames $\mathcal{V}_e = \{v_i | e^b \leq i \leq e^e\}$, we follow the standard settings from existing works (L. Zhou, Y. Zhou, et al., 2018; Lei et al., 2020; Song, S. Chen, and Jin, 2021) and divide $\mathcal{V}_e$ into a sequence of δ-frame *snippets* $\{X_i\}_{i=1}^L$. Each snippet X_i consists of δ consecutive frames and $\mathcal{V}_e$ has a total of $L = \lceil \frac{|\mathcal{V}_e|}{\delta} \rceil$ snippets. The VL encoder module encodes each snippet X_i to a VL representation f_i^{VL} as shown in Fig.2.2. Therefore, video segment $\mathcal{V}_e$ is encoded into VL representation $\{f_i^{VL}\}_{i=1}^L$.

The VL encoder first models a video with the three modalities, (i) global visual environment, (ii) local visual main agents, and (iii) linguistic relevant scene elements, and then fuses them into one representation based on the interactions between them. Given a snippet X_i, it is encoded into these three modalities, corresponding to f_i^e, f_i^a, and f_i^l, respectively. The final feature f_i^{VL} representing the interaction is extracted by fusing f_i^e, f_i^a, and f_i^l through our Multi-modal Representation Fusion (M2RF) module. We will provide the details of each module in the following subsections.

Global Visual Environment

This modality provides the visual semantic information from the entire spatial scene of input snippet X_i. To obtain such target, we adopt a backbone 3D-CNN network (Ji et al., 2013) to X_i to extract feature map $\mathcal{H}_i$ at the last convolutional block of the network. Then, we obtain the global environment visual feature $f_i^e \in \mathbb{R}^{d_{\text{emb}}}$ by processing $\mathcal{H}_i$ with an average pooling operation to reduce the entire spatial dimension followed by channel MLP. The procedure is summarized as follows:

$$f_i^e = \text{MLP}_{\theta_e}(\text{Avg.Pooling}(\mathcal{H}_i)) \tag{2.2}$$

Local Visual Main Agents

This modality provides the visual features of the main human agents, who actually contribute to the formation of the event being described. Even though most of the events are associated with

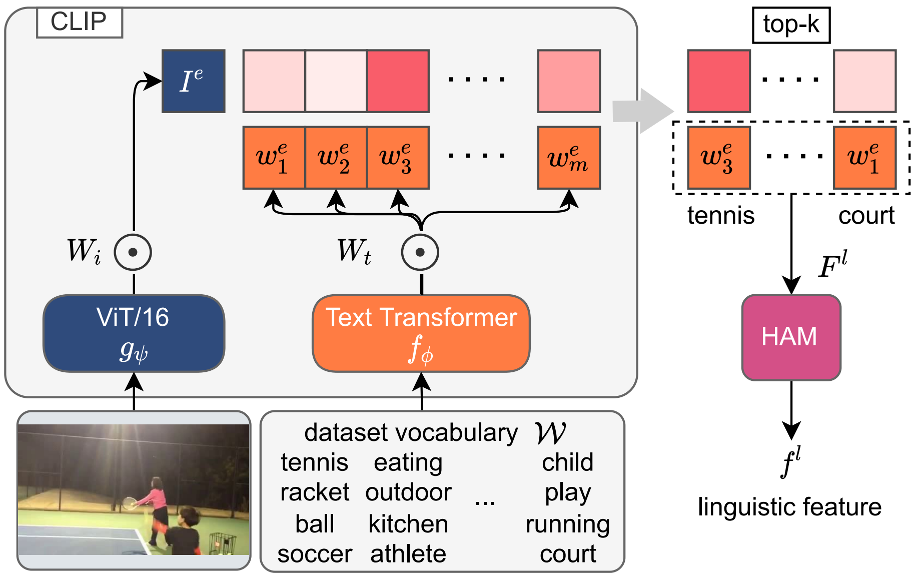


Figure 2.3: Illustration of relevant scene elements extraction process where we utilized ViT/16 and Text Transformer pre-trained with CLIP (Radford et al., 2021).

agents, not all agents committing movements are related to the main content of the event segment. Using a similar assumption as in (Vo, Le, et al., 2021; Vo, Yamazaki, Truong, et al., 2021), we apply a human detector to the center frame of X_i to obtain the bounding boxes of all human agents. Afterward, we align each of the detected bounding boxes $\mathcal{B}_i$ onto the feature map $\mathcal{H}_i$, which is obtained by the previous modality, using RoIAlign (He, Gkioxari, et al., 2017). Then, features overlapped by each agent bounding box are averagely pooled into a single feature vector to represent visual information of the agent inside that box. Finally, we obtain a set of local agent visual features $F_i^a \in \mathbb{R}^{N_a \times d_a}$, where N_a and d_a are the number of detected agents and agent embedding dimension, respectively. Finally, we apply HAM to adaptively select an arbitrary number of main agents from N_a detected agents and extract their mutual relationships to form a unified agent-aware visual feature $f_i^a \in \mathbb{R}^{d_{\text{emb}}}$ as follows:

$$f_i^a = \text{HAM}(\text{MLP}_{\theta_a}(F_i^a), f_i^e) \tag{2.3}$$

Linguistic Relevant Scene Elements

This modality provides additional contextual details of the scene. While the two former modalities capture visual information of spatial appearances and temporal motions, their features may overlook some of the scene components because of the spacial reduction in the pooling operation from the feature map $\mathcal{H}_i$. Furthermore, non-visual features could hardly be captured by a normal vision backbone model. Recent studies (Patashnik et al., 2021; Yang, T. Zhang, and Zou, 2022) have shown the extreme zero-shot capability of Contrastive Language-Image Pre-training (CLIP) where the model can estimate the semantic similarity between a set of words and an image. Trained on large-scale text-image pairs, CLIP can correlate not only the visual words but also the non-visual words to the given image. We thus leverage CLIP as a linguistic feature extractor to obtain top k scene elements (i.e., k texts) that are highly correlated with the middle frame of the input snippet X_i. Specifically, we construct a vocabulary $\mathcal{W} = \{w_1, \dots w_m\}$ based on the groundtruth captions of our training dataset. Each vocabulary $w_i \in \mathcal{W}$ is encoded by a transformer network f_ϕ into a text feature f_i^w. Let W_t be a text projection matrix pre-trained by CLIP, the embedding text vocabulary is computed as

$$w^e = W_t \cdot f_\phi(\mathcal{W}) = W_t \cdot f^w \text{where } f^w = \{f_i^w\}_{i=1}^m. \tag{2.4}$$

Let W_i be an image projection matrix pre-trained by CLIP, the center frame I of the input snippet X_i is first encoded by a pre-trained ViT g_ψ to extract visual feature f^I, and then embedded by W_i as below:

$$I^e = W_i \cdot g_\psi(I) = W_i \cdot f^I \tag{2.5}$$

The pairwise cosine similarities between embedded I^e and w^e are then computed. The text feature with top k similarity scores are chosen as linguistic categorical concept features $F_i^l \in \mathbb{R}^{k \times d_l}$. This feature is also subjected to the HAM module to select only the most relevant representative linguistic features and merge them into a single representation $f_i^l \in \mathbb{R}^{d_{\text{emb}}}$ as follows:

$$f_i^l = \text{HAM}(\text{MLP}_{\theta_l}(F_i^l), f_i^e) \tag{2.6}$$

The flowchart of extracting f_i^l is illustrated in Fig.2.3.

Multi-modal Representation Fusion (M2RF)

This component aims to fuse features from the three modalities. While concatenation or summation are the two common fusion mechanisms, they treat all modalities equally. To better model the impact of each individual modality, we propose M2RF as a function g_γ, which takes the features f_i^e, f_i^a, and f_i^l as its input. We extract the inter-feature relationships by utilizing a self-attention (MHA) layer (Vaswani et al., 2017) followed by a mean operation. The final representation $f_i^{VL} \in \mathbb{R}^{d_\text{emb}}$ of a given snippet X_i is defined as follows:

$$f_i^{VL} = g_\gamma([f_i^e; f_i^a; f_i^l]) = \text{mean}(\text{MHA}([f_i^e; f_i^a; f_i^l])) \tag{2.7a}$$

where $[;]$ represents the concatenation of features in a new dimension, where self-attention is applied on the new dimension and reduced by the mean operation to account for permutation invariance.

2.3 Analysis of Local Visual Main Agents

The effectiveness of our proposed local visual main agents modality is demonstrated through the qualitative results depicted in Fig.2.4. Our example illustrates that this novel modality is capable of distinguishing and eliminating non-essential agents, whilst retaining those critical agents that actively perform actions within the scene. This underscores its potential in honing the focus on principal elements in video analysis. This approach has further implications for the model's explainability. By focusing on the main agents and their actions, our model becomes more transparent and its operation more understandable. This attribute is vital as it allows for easier interpretation of the model's outputs and facilitates debugging and optimization efforts. With a clear understanding of how our model processes and evaluates video content, we can more efficiently adjust and fine-tune its parameters to achieve superior video captioning performance. Hence, the Local Visual Main Agents modality contributes not only to the accuracy of the model but also significantly enhances

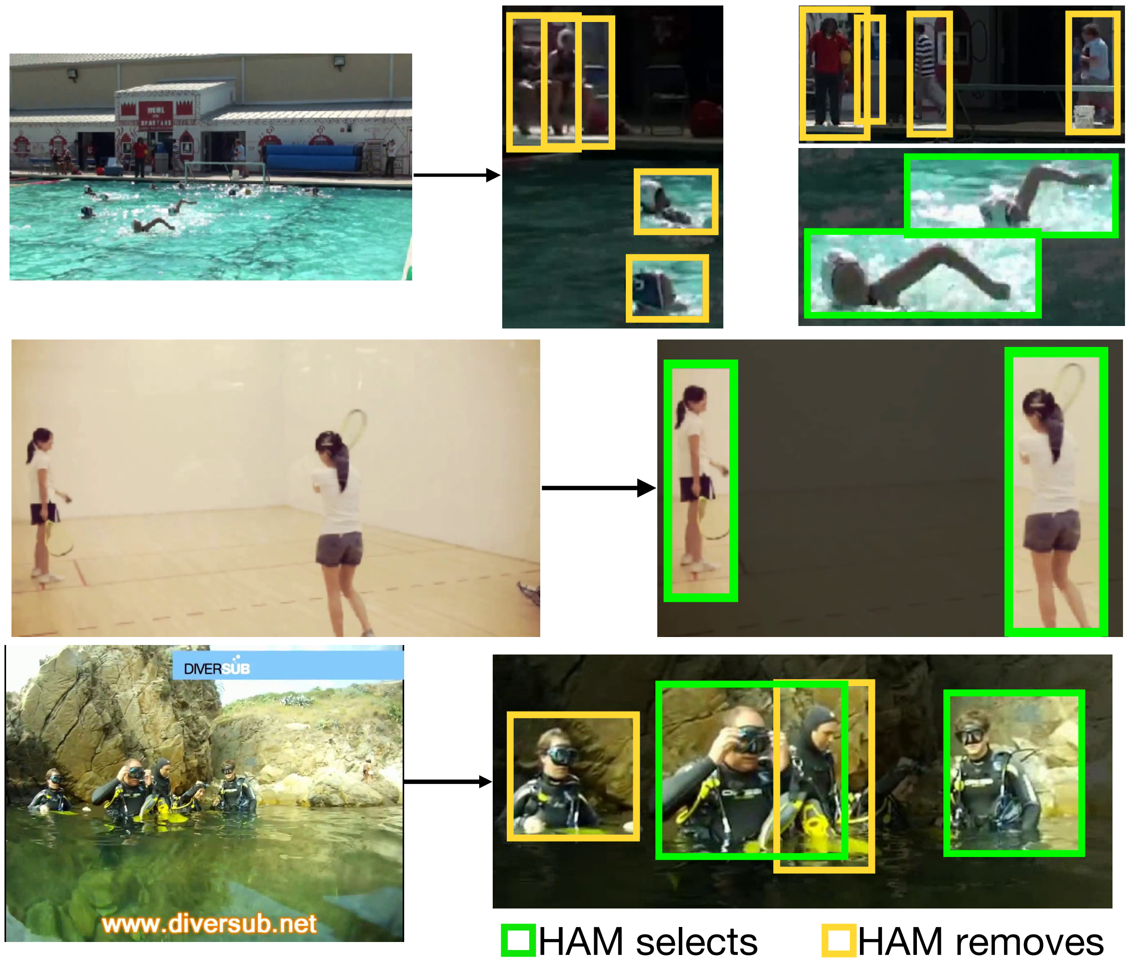


Figure 2.4: Qualitative results of our local visual main agent modality. 🟩 indicates main agents selected by our local main agent modality, and 🟨 indicates eliminated trivial agents. Left: Input image. Right: selected and eliminated agents.

its explainability.

2.4 Analysis of Linguistic Relevant Scene Elements

In the linguistic relevant scene elements modality, the linguistic scene elements are first extracted by CLIP (Radford et al., 2021). Subsequently, the HAM assists in selecting the most pertinent elements. Fig.2.5 first illustrates the qualitative results derived from CLIP, followed by a presentation of the most relevant linguistic scene elements as determined by HAM. As demonstrated in Fig.2.5, CLIP effectively recognizes both visual and non-visual scene elements. Among all the scene elements captured by CLIP, only a subset is genuinely relevant to the action at hand. With the aid of HAM,

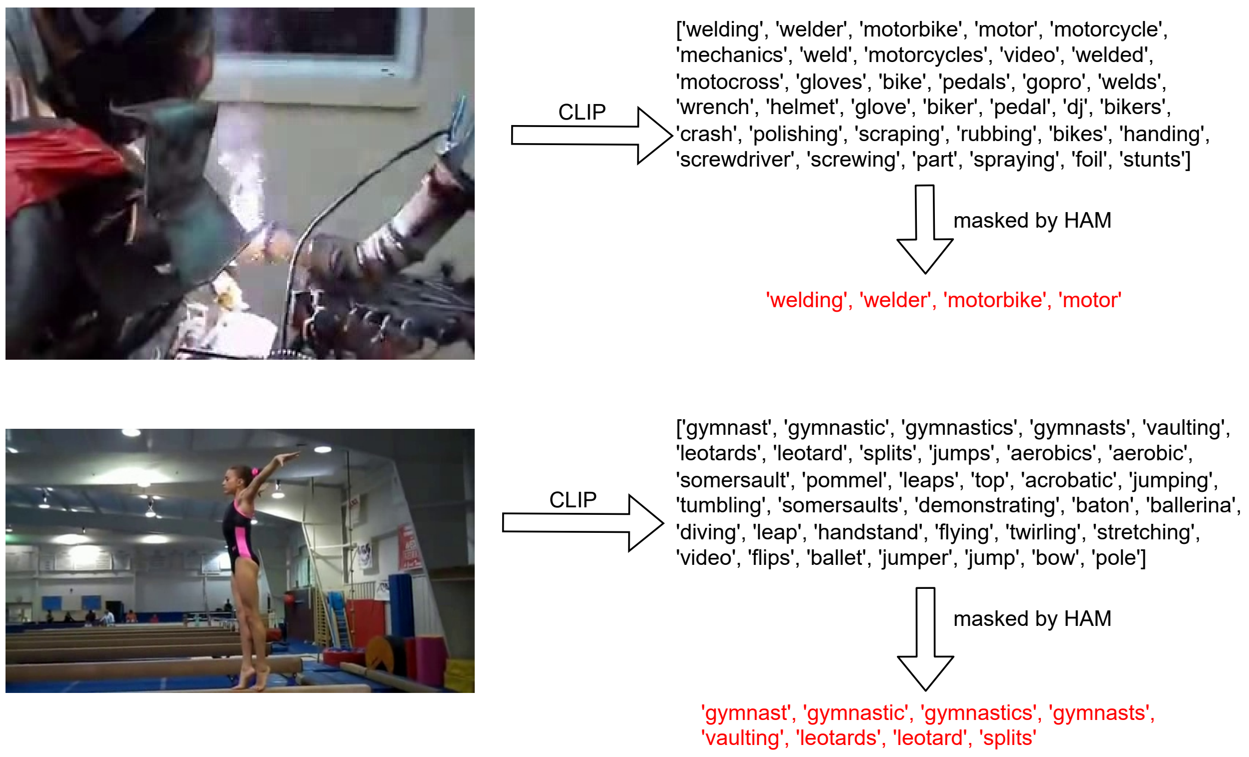

CLIP
['welding', 'welder', 'motorbike', 'motor', 'motorcycle', 'mechanics', 'weld', 'motorcycles', 'video', 'welded', 'motocross', 'gloves', 'bike', 'pedals', 'gopro', 'welds', 'wrench', 'helmet', 'glove', 'biker', 'pedal', 'dj', 'bikers', 'crash', 'polishing', 'scraping', 'rubbing', 'bikes', 'handing', 'screwdriver', 'screwing', 'part', 'spraying', 'foil', 'stunts']
masked by HAM
'welding', 'welder', 'motorbike', 'motor'
CLIP
['gymnast', 'gymnastic', 'gymnastics', 'gymnasts', 'vaulting', 'leotards', 'leotard', 'splits', 'jumps', 'aerobics', 'aerobic', 'somersault', 'pommel', 'leaps', 'top', 'acrobatic', 'jumping', 'tumbling', 'somersaults', 'demonstrating', 'baton', 'ballerina', 'diving', 'leap', 'handstand', 'flying', 'twirling', 'stretching', 'video', 'flips', 'ballet', 'jumper', 'jump', 'bow', 'pole']
masked by HAM
'gymnast', 'gymnastic', 'gymnastics', 'gymnasts', 'vaulting', 'leotards', 'leotard', 'splits'

Figure 2.5: Examples of the scene elements that are initially extracted by CLIP (notated in black text) and the most pertinent ones selected by HAM (red text). For a given image, the top *k* related terms from the dataset vocabulary are drawn out as scene elements via CLIP, and these are then refined using HAM.

we can efficiently identify and select these pertinent scene elements.

Scene elements are often presented as objects in the scene. Thus, we further compare the effectiveness of our CLIP and HAM against Mask R-CNN (He, Gkioxari, et al., 2017) in extracting the most relevant scene elements. We observe that object detectors like Mask R-CNN can only extract a limited amount of visual scene elements, whereas CLIP provides much richer information on scene concepts including visual and non-visual scene elements. For example, given an image of people playing tennis as shown in Fig. 2.6, it is unfeasible to detect a small object such as a tennis ball using an object detector. For instance, given an image of people engaging in a game of tennis as displayed in Fig. 2.6, detecting a small object like a tennis ball using an object detector can prove to be difficult. Fig. 2.6 (bottom) shows that Mask-R-CNN can only identify humans and a tennis racket, failing to capture the tennis ball. On the other hand, CLIP is able to encode the concept of

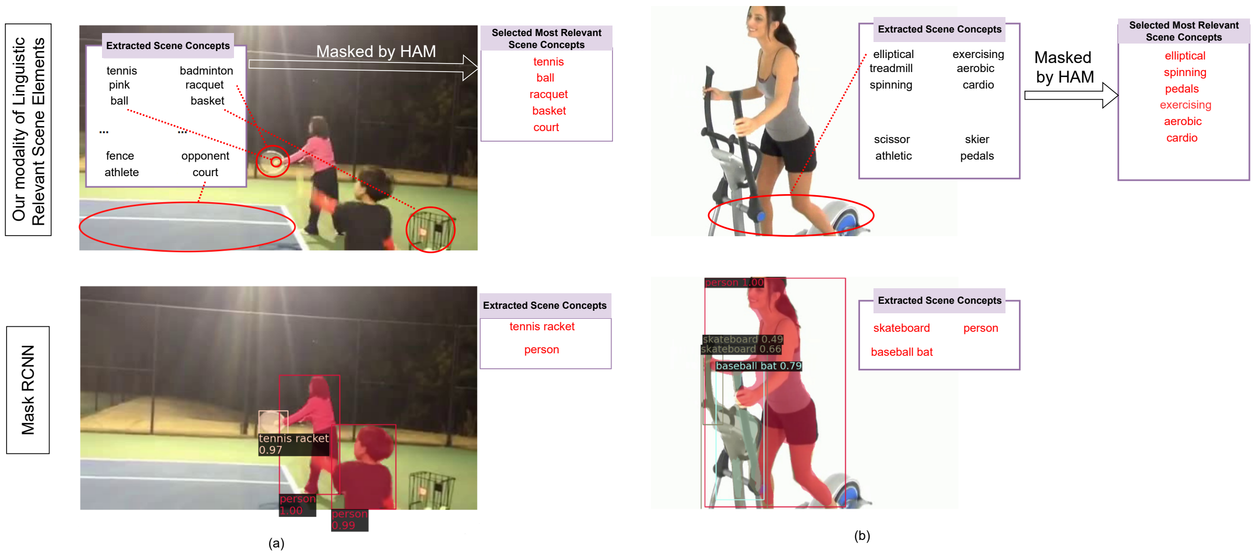


Figure 2.6: Examples of scene elements derived qualitatively from our proposed CLIP + HAM (top row) versus Mask-RCNN (bottom row). Despite lacking specific location information, the key scene elements selected by HAM, highlighted in red text (top row), successfully reveal image-based semantic associations.

a tennis game, including the presence of a tennis ball and other related objects, such as a basket, court, fence, etc., as shown in Fig. 2.6 (top). This demonstrates the effectiveness of utilizing CLIP model to extract linguistically relevant information from the scene.

Chapter 3

PARAGRAPH CAPTIONING MODEL

In this chapter, we will introduce our video paragraph captioning model that utilizes the Visual-Linguistic (VL) Encoder presented in the previous chapter and the Transformer-in-Transformer (TinT) Decoder developed specifically for generating a coherent paragraph caption of a video. We trained our VL Encoder and TinT decoder in an end-to-end fashion using our Visual-Linguistic (VL) Loss to better align the learned features.

The primary focus of the chapter will be on the description and demonstration of the mechanics of the model, including its intricate design and various functionalities. We delve into the technical aspects of the model, exploring the algorithms and techniques that power it, as well as the novel solutions developed to overcome challenges faced during its development. A comparative analysis will be presented to highlight the benefits and improvements our model offers over existing video captioning methodologies.

3.1 Problem Setup

In Video Paragraph Captioning (VPC), we are given an untrimmed video $\mathcal{V} = \{v_i\}_{i=1}^{|\mathcal{V}|}$, where $|\mathcal{V}|$ is the number of frames, and a list of its important events $\mathcal{E} = \{e_i = (e_i^b, e_i^e)\}_{i=1}^{|\mathcal{E}|}$, where $|\mathcal{E}|$ is the number of events within a video and an event e_i is defined by a pair of beginning and ending timestamps (e_i^b, e_i^e). Our objective is to generate a coherent paragraph that matches the ground truth paragraph $\mathcal{P} = \{\mathbf{s}_i\}_{i=1}^{|\mathcal{E}|}$ that describes the whole video $\mathcal{V}$. In this setup, i^{th} sentence $\mathbf{s} = \{s_1 \dots s_N\}$ that consists of N words is the description of its corresponding event e_i.

3.2 Baselines

In the following, we first describe the baseline models that has been presented for video paragraph captioning.

Vanilla Transformer

L. Zhou, Y. Zhou, et al. proposed a baseline model for dense video captioning utilizing the Transformer model proposed by Vaswani et al. whose architecture was detailed in Section 1.1. This model, often referred to as the Vanilla Transformer, is a standard transformer model equipped with an auxiliary proposal generation module for dense video captioning, i.e., for the event proposal generation. As our focus within this discussion is strictly confined to the paragraph caption generation, we disregard this auxiliary module for our baseline. Fundamentally, this model is designed to accept a single video segment as its input. It then independently produces a single sentence that succinctly describes the content of the given segment. We illustrate the model in Fig. 3.1 (left).

Transformer-XL

Initially proposed by Z. Dai et al., the Transformer-XL model was developed to handle long-term dependencies in natural language. Transformer-XL modifies the input to the multi-head self-attention where the query is $Q = H_t$ and key and value takes the concatenation of the previous and current hidden states $K, V = [SG(H_{t-1}); H_t]$. Here, $SG(\cdot)$ indicates a stop gradient operation, which was introduced as a strategy to conserve GPU memory and computation resources. Lei et al. modified this model to adopt to VPC task. For a more balanced comparison with other VPC models, Lei et al. proposed a variant of the Transformer-XL that allows the gradient to propagate through different recurrent steps, which is known as Transformer-XLRG. We illustrate the model in Fig. 3.1 (right).

MART

Lei et al. proposed Memory-Augmented Recurrent Transformer (MART) to better utilize the video segments and sentence history information by augmenting the transformer with recurrent memory. A graphical representation of this memory module can be found in Fig. 3.2. The memory module is formulated as the modified GRU (Chung et al., 2014), where the multi-head attention is used to encode the memory state. The memory update process is as follows:

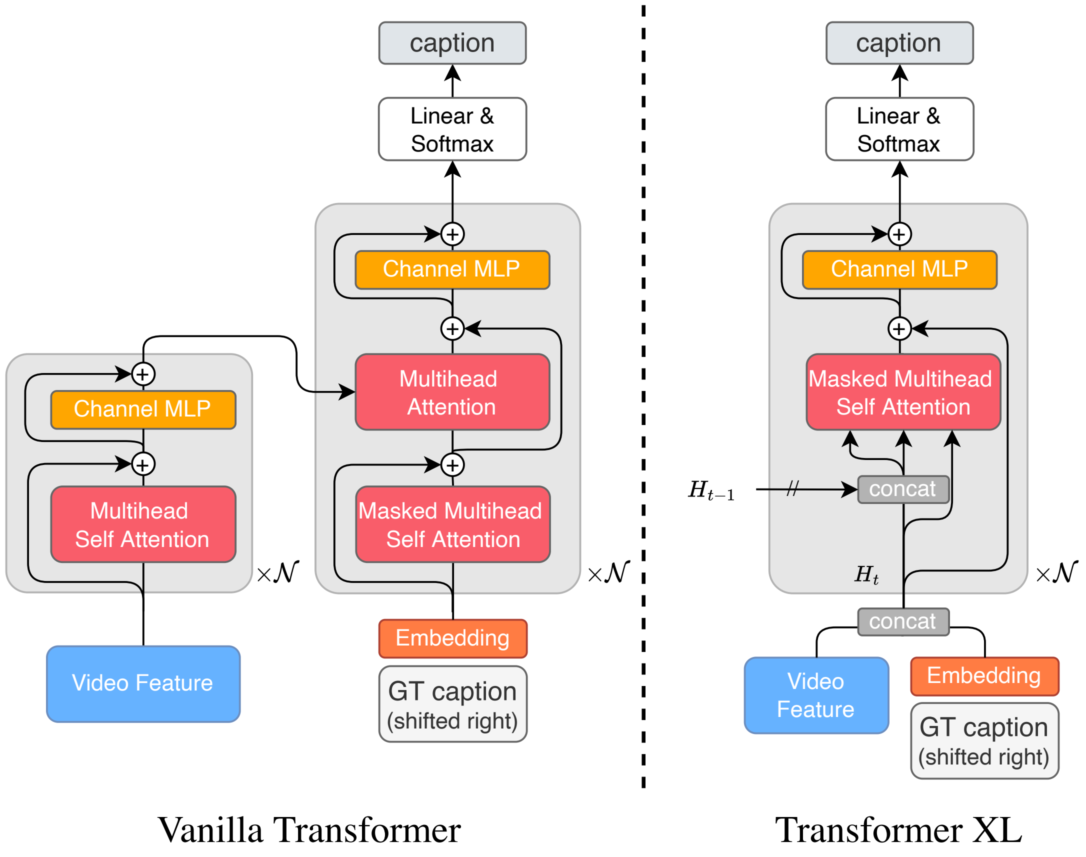


Figure 3.1: Illustration of the baseline video paragraph captioning models. (left) Vanilla Transformer (L. Zhou, Y. Zhou, et al., 2018) (right) Transformer-XL (Z. Dai et al., 2019) // in the Transformer-XL hidden state concatenation denotes stop-gradient. Transformer-XLRG (Lei et al., 2020) does not apply this stop-gradient.

$$S_t^l = \text{MHA}\left(M_{t-1}^l, \bar{H}_t^l, \bar{H}_t^l\right) \tag{3.1a}$$

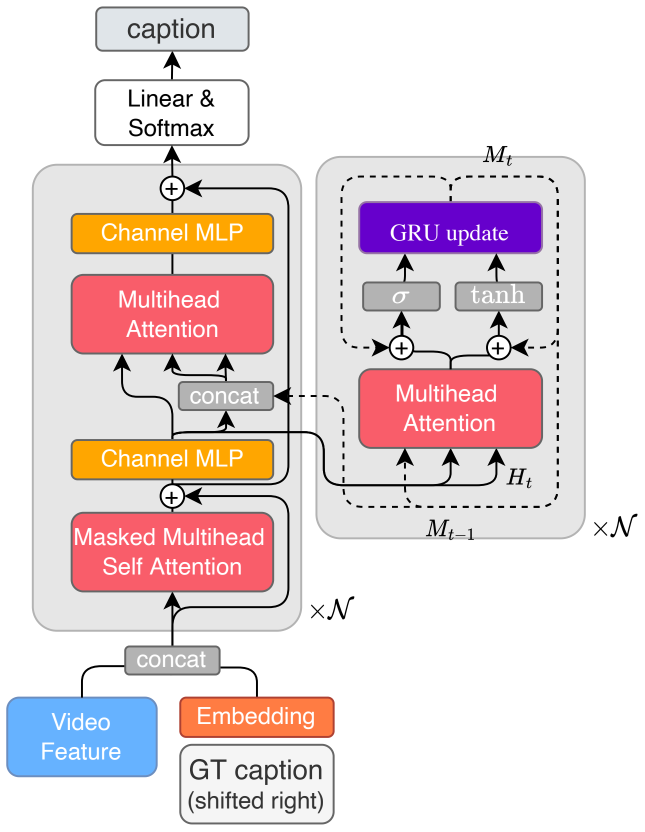
$$C_t^l = \tanh\left(W_{mc}^l M_{t-1}^l + W_{sc}^l S_t^l + b_c^l\right) \tag{3.1b}$$

$$Z_t^l = \sigma\left(W_{mz}^l M_{t-1}^l + W_{sz}^l S_t^l + b_z^l\right) \tag{3.1c}$$

$$M_t^l = \left(1 - Z_t^l\right) \odot C_t^l + Z_t^l \odot M_{t-1}^l \tag{3.1d}$$

where $W_{mr}, W_{ur}, W_{mz}, W_{uz}$ are network linear weights and b_r^l, b_z^l are bias.

In a manner akin to Transformer-XL, the calculated memory, denoted as M_t, is merged with the



MART

Figure 3.2: Illustration of MART (Lei et al., 2020) architecture. MART augments the transformer with the GRU-like memory module to exploit the video segments and sentences history information to better model the inter-event relationship.

model's hidden states and then supplied as key and value in the multi-head attention mechanism for the subsequent decoding step. Thus, the input query matrix becomes $Q = H_{t+1}$, while the key and value matrices become $K, V = [M_t; H_{t+1}]$.

However, a comparison between the MART and Transformer-XL reveals an advantage of the former. While Transformer-XL employs all the hidden states from the preceding step to enable recurrence, MART adopts a more strategic approach, in which its integrated memory module is designed to filter out information that is less relevant or repetitive. This feature of MART ensures more effective utilization of historical information, enhancing the model's performance and output quality in video paragraph captioning tasks.

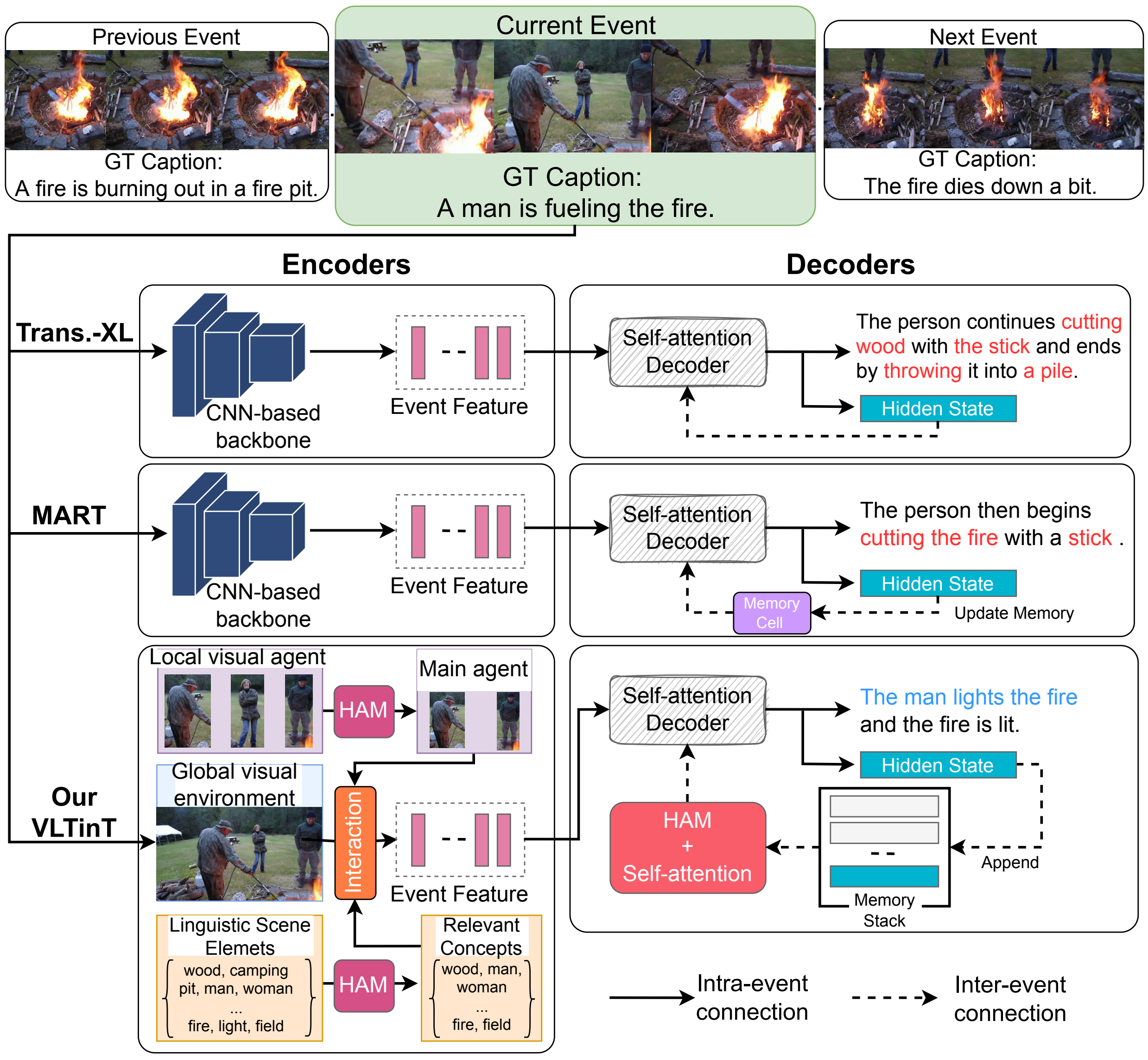


Figure 3.3: A high-level comparison between our VLTinT and recent SOTA VPC methods. In the encoder, both Transformer-XL (Z. Dai et al., 2019) and MART (Lei et al., 2020) encode visual features by applying a 3D CNN-based backbone network whereas our VLTinT encodes visual-linguistic features by (i) global visual environment, (ii) local visual main agents, (iii) linguistic scene elements, and a fusion mechanism. In the decoder, Transformer-XL uses recurrence to address context fragmentation, MART uses a highly summarized memory to remember history information whereas we propose to utilize a transformer to model the contextual dependencies at both intra- and inter-levels.

3.3 Transformer-in-Transformer (TinT) Decoder

In this section, we introduce our proposed model - the Transformer-in-Transformer (TinT) Decoder. This novel architecture employs a Transformer to model the relationships between events. Fig. 3.3 provides a comparative overview of our model and the current state-of-the-art (SOTA) baseline models, namely MART and Transformer-XL. In terms of the decoder design, Transformer-XL uses recurrence to address context fragmentation, while MART employs a highly summarized memory to remember historical information. Contrastingly, our proposed TinT Decoder advances a different approach. It employs a Transformer model to capture the contextual dependencies at both intra- and inter-event levels. This dual-level modeling of contextual dependencies promises an enhanced comprehension and representation of the sequence, improving the quality of video paragraph captioning.

Inspired by the recent transformer-based vision-language models (Y.-C. Chen et al., 2020; Lei et al., 2020), we adopt the unified encoder-decoder transformer structure as a foundation for the caption generator, namely an inner transformer. The inner transformer's input is described as follows. In this setup, video features $\mathcal{F}^{VL}$ is formed by concatenating all f_i^{VL} obtained by applying VL Encoder into each snippet X_i, i.e., $\mathcal{F}^{VL} = \{f_i^{VL}\}_{i=1}^{L} \in \mathbb{R}^{L \times d_{\text{emb}}}$. Textual tokens $\mathcal{F}^{text}$ is encoded by a pre-trained text transformer g_ϕ from CLIP and a MLP layer, i.e., $\mathcal{F}^{text} = MLP(g_\phi(\text{Shifted GT text})) \in \mathbb{R}^{N \times d_{\text{emb}}}$, where N is the sequence length of the text tokens. Following (Lei et al., 2020), learnable token type embeddings $\mathcal{F}^{type} \in \mathbb{R}^{(L+N) \times d_{\text{emb}}}$ are introduced to inform the location of the video and the caption representations. $\mathcal{F}^{type}$ is initialized as 0/1 vectors, i.e., video as 0 and text as 1. For the t^{th} event, an intermediate hidden state $\bar{H}_t^l \in \mathbb{R}^{(L+N) \times d_{\text{emb}}}$ is computed in Eq. 3.2b as canonical inner transformer encoder, where $\tilde{H}_t^l$ is the internal states after Masked Multihead Self Attention (MSA).

$$H_t^0 = [\mathcal{F}^{VL}; \mathcal{F}^{text}] + \mathcal{F}^{type} \in \mathbb{R}^{(L+N) \times d_{\text{emb}}} \tag{3.2a}$$

$$\bar{H}_t^l = \text{MLP}(\tilde{H}_t^l) + \tilde{H}_t^l, \tilde{H}_t^l = \text{MSA}(H_t^l) + H_t^l \tag{3.2b}$$

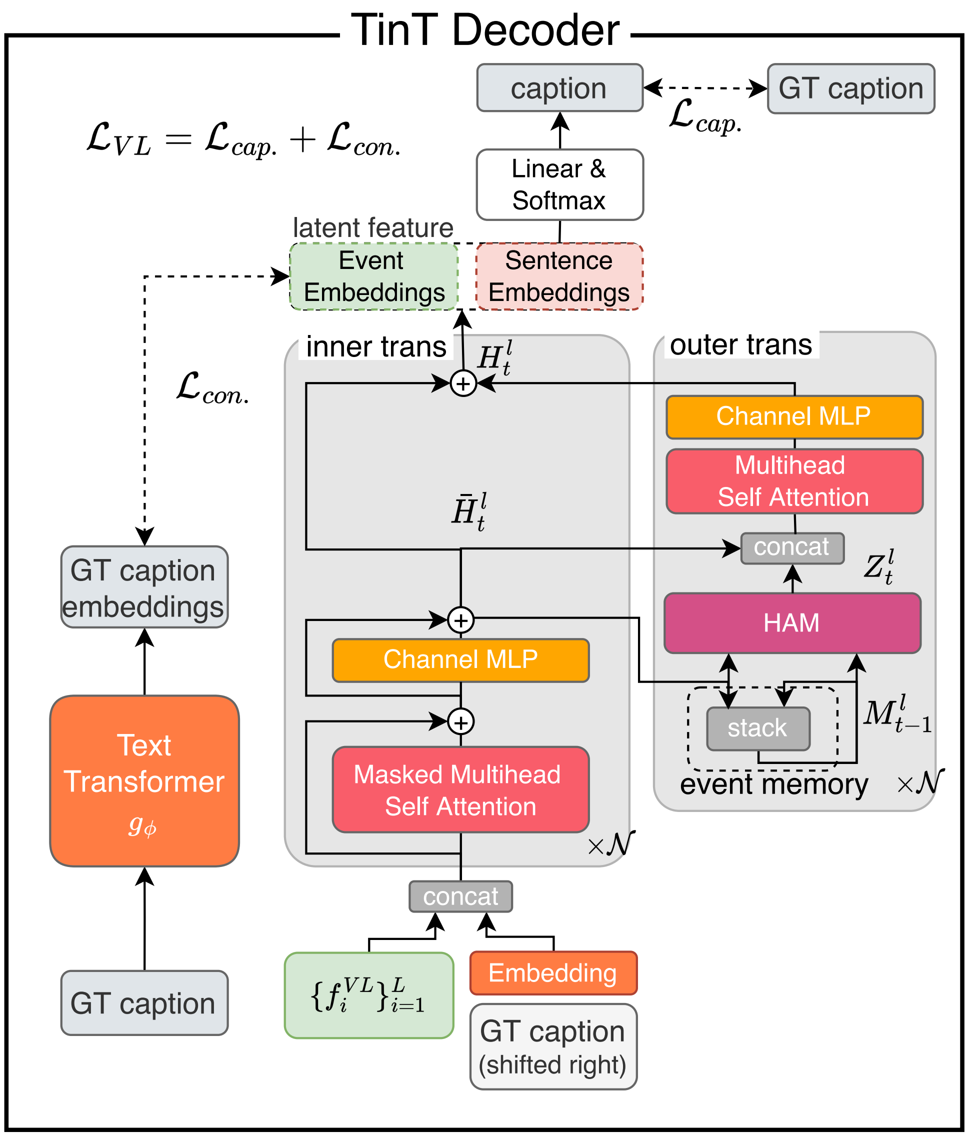


Figure 3.4: TinT Decoder: the canonical transformer encoder is extended by an autoregressive outer transformer that can selectively access the 1^{st} to $t-1^{th}$ hidden states, which are stored in the event memory, at the t^{th} event captioning step.

While the inner transformer can effectively model intra-event coherency, it cannot handle the contextual relationship of inter-event. To address this limitation, we introduce an autoregressive outer transformer. The outer transformer selectively utilizes the activations of the inner transformer from the previous time steps for generating a coherent paragraph.

Specifically, we take advantage of HAM to select only the most relevant hidden states of all previous events stored in event memory with respect to the current one. The outer transformer process is formulated below:

$$M_t^l = [M_{t-1}^l; \bar{H}_t^l] \tag{3.3a}$$

$$Z_t^l = \text{HAM}(M_{t-1}^l, \bar{H}_t^l) \tag{3.3b}$$

$$H_t^l = \text{MLP}(g_\gamma([\bar{H}_t^l; Z_t^l])) + \bar{H}_t^l \tag{3.3c}$$

For the t^{th} event, an intermediate hidden states $\bar{H}_t^l$ is stacked to the event memory $M_t^l \in \mathbb{R}^{t\times(L+N)\times d_{\text{emb}}}$, where $M_0^l = \varnothing$ as in Eq. 3.3a. Eq. 3.3b computes the context Z_t^l from the previous states of the event memory and the current intermediate hidden states $\bar{H}_t^l$ using HAM. Finally, in Eq. 3.3c, the context is integrated with the intermediate hidden states $\bar{H}_t^l$ using g_γ, which was introduced in Eq. 2.7a, and the hidden states are updated via the residual connection. After the last layer, video token positions in H_t^N are ignored, and only the text token positions are fed to a feed-forward layer followed by softmax to predict a caption for the t^{th} event.

3.4 Visual-Linguistic (VL) Loss

Typically, the existing VPC methods exploit the MLE loss to train their models. The MLE loss serves the objective of increasing the likelihood of predicted captions to be matched with the groundtruths. However, it is unable to address the question of how well the learnt event embedding features represent the groundtruth captions. To this end, we leverage the recent advantages of contrastive learning (Wu et al., 2018; T. Chen et al., 2020) and propose $\mathcal{L}_{con}$ to pull all snippets of the same event and push snippets of different events. Our VL Loss consists of two terms corresponding to captioning loss ($\mathcal{L}_{cap.}$) and a contrastive contextual loss ($\mathcal{L}_{con.}$). While $\mathcal{L}_{cap.}$ aims to decode captions that match with groundtruths, $\mathcal{L}_{con.}$ guarantees the learnt latent features are close to the semantic information encoded in the groundtruth captions.

Captioning Loss $\mathcal{L}_{cap.}$

Kullback–Leibler (KL) divergence is commonly utilized to minimize the divergence between empirical distribution $p(\mathbf{s}|\mathcal{V}_e)$ and predicted distribution $p_\theta(\mathbf{s}|\mathcal{V}_e)$ for a video segment $\mathcal{V}_e$. However, this objective easily makes the captioning model overfit high-frequency tokens and phrases, which results in repetitive phrases. In order to enhance the smoothness of the predicted sentence, a regularization term τ is introduced to the training objective with hyper-parameter λ as:

$$\theta^{*} = \underset{\theta}{\operatorname{argmin}} \mathbb{E}_{\mathbf{s} \sim p(\mathbf{s})} \left[\log \left(\frac{p(\mathbf{s})}{p_{\theta}(\mathbf{s})} \right) + \lambda \tau(\mathbf{s}) \right] \tag{3.4}$$

The second term τ imposes a token-level high-frequency penalty as Song, S. Chen, and Jin. Based on the observation that the model tends to generate words that have been generated before, we penalize the previously appeared words in the regularization term:

$$\tau(\mathbf{s}) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c \in \{s|s_{<i}\}} \log \left(1 - p_{\theta}(c|s_{<i}, \mathcal{E})\right) \tag{3.5}$$

where c is the candidate word at n to be penalized.

Our $\mathcal{L}_{cap.}$ is defined as follows:

$$\mathcal{L}_{cap.} = -\frac{1}{N} \sum_{i=1}^{N} (\log p_{\theta}(s_i|s_{<i}, \mathcal{V}_e)) + \lambda \tau(\mathbf{s}) \tag{3.6}$$

where θ is the model parameters, $s_{1:N}$ is the target ground truth sequence.

Contrastive Contextual Loss $\mathcal{L}_{con.}$

We propose contrastive contextual loss to optimize the latent feature of the input event to be highly correlated with its groundtruth description. This loss function implicitly encourages our model to learn better representations of the events and enhance its overall performance without extra computational cost.

Specifically, contrastive contextual loss processes the entire mini-batch of training examples $\mathcal{B} = \{(\mathcal{V}_b, \mathbf{s}_b)\}_{b=1}^{|\mathcal{B}|}$, where $\mathcal{V}_b$ is a set of snippets within the same event and $\mathbf{s}_b$ is its corresponding groundtruth description sentence. On the one hand, video snippets in $\mathcal{V}_b$ are processed through our proposed VLTinT to obtain the event embeddings, which corresponds to the video token position $\mathcal{F}_b^{\mathcal{N}} \in \mathbb{R}^{L \times d_{\text{emb}}}$ of the final hidden state $H_b^{\mathcal{N}}$. On the other hand, we process each groundtruth caption sentence $\mathbf{s}_b$ through the transformer g_ϕ of CLIP (Radford et al., 2021) to obtain a representation feature $f_b^{\mathcal{T}} \in \mathbb{R}^{d_{\text{emb}}}$. Then, $\mathcal{L}_{con.}$ processes $\mathcal{F}_b^{\mathcal{N}}$ and $f_b^{\mathcal{T}}$ as follows:

$$\begin{aligned} \mathcal{L}_{con.} = -\sum_{b_1=1}^{|\mathcal{B}|}\sum_{b_2=1}^{|\mathcal{B}|} [\mathbb{1}_{b_1=b_2} \log(e^{\rho}(f_{b_1}^{\mathcal{N}} \cdot f_{b_2}^{\mathcal{T}})) \\ + (1 - \mathbb{1}_{b_1=b_2})(1 - \log(e^{\rho}(f_{b_1}^{\mathcal{N}} \cdot f_{b_2}^{\mathcal{T}})))] \end{aligned} \tag{3.7a}$$

where $f_b^{\mathcal{N}} = \text{mean}(\mathcal{F}_b^{\mathcal{N}})$. $\mathbb{1}_{b_1=b_2}$ returns 1 when samples come from the same event, i.e., $b_1 = b_2$ and 0 when samples come from the different events i.e., $b_1 \neq b_2$. ρ is a learnable temperature parameter initialized as $\log(1/0.07)$, to prevent scaling of the dot product values and stabilize the training.

Finally, our proposed VL contrastive loss $\mathcal{L}_{VL}$ is defined as:

$$\mathcal{L}_{VL} = \mathcal{L}_{cap.} + \mathcal{L}_{con.} \tag{3.8}$$

3.5 Experiments

This section is dedicated to presenting the experimental framework that we have set up to validate our model's effectiveness.

Datasets and Evaluation Metrics

Our model is benchmarked on two widely used datasets: ActivityNet Captions (Krishna et al., 2017) and YouCookII (L. Zhou, C. Xu, and Corso, 2018). The ActivityNet Captions dataset comprises 10,009 training videos and 4,917 validation videos. In the training set, each video is associated with a single reference paragraph, while each video in the validation set has two reference paragraphs. The average count of event segments for each video stands at 3.65. Following previous works (Lei et al., 2020), the original validation set is split into two subsets: *ae-val* (for validation, with 2,460 videos) and *ae-test* (for testing, with 2,457 videos).

The YouCookII dataset includes 1,333 training videos and 457 validation videos. Each video in this dataset is coupled with a single reference paragraph, with an average of 7.7 event segments for each video. Our results are reported on the validation sets of both datasets.

We constructed a vocabulary based on the words found in the training set captions, supplemented with a few special tokens. As a result, the vocabulary encompasses 10,648 words for ActivityNet Captions and 2,310 words for YouCookII.

To evaluate performance, we employ four standard metrics: BLEU@4 (B@4) (Papineni et al., 2002), METEOR (M) (Denkowski and Lavie, 2014), CIDEr (C) (Vedantam, Zitnick, and Parikh, 2015), and ROUGE (R) (C.-Y. Lin, 2004). Furthermore, to assess the diversity of the generated captions, we utilize two additional metrics: 2-gram diversity (Div@2) (Shetty et al., 2017) and 4-gram repetition (R@4) (Xiong, B. Dai, and D. Lin, 2018). These latter metrics serve to benchmark the uniqueness and non-repetitiveness of the generated captions.

Implementation Details

For visual feature extraction, we utilize a C3D model, as outlined by Ji et al., which is pretrained on the Kinetics-400 dataset (Kay et al., 2017). This backbone network serves as a primary conduit for the interpretation of environmental visuals. The agent visual feature is extracted by Faster-RCNN (Ren et al., 2015) that is pre-trained on the COCO dataset (T.-Y. Lin et al., 2014). To obtain linguistic features of the scene elements, we use the Contrastive Language–Image Pretraining (CLIP) model (Radford et al., 2021) based on ViT-B/16, made publicly available by OpenAI. In the CLIP model, the text and image features are encoded by the Transformer (Vaswani et al., 2017) and the Vision Transformer (Dosovitskiy et al., 2021), respectively.

In terms of model specifications, the hidden size is set to 768, and we opt for a configuration of 3 Transformer layers with 12 attention heads each. The model is trained using the Adam optimizer with an initial learning rate of 1e-4, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and an L_2 weight decay of 0.01. The learning rate is warmed up over the initial 5 epochs. During the training phase, we employ label v_PAGuZzrzSO4

VTrans: A man is **riding a horse** **down a river**. The man then gets up and **throws the calf down** and **grabs the horse** and **runs back to the horse**. He **gets back on his horse** and **gets back on his horse** .
MART: A man is seen **standing on a horse** and throws a rope around. The man **throws the calf down** and the man **chases after it**. He **ties the calf** up and walks back to the horse .
VLTinT: A man is **riding a horse** in a rodeo ring. **He lassos a calf**. He **ties the calf** up and **ties it up**.
GT: A cowboy is riding a horse in a barn. He lassos a small calf. He dismounts, tying the calf and celebrating.

v_G8dCenteoT0

VTrans: The person then **puts eye on the contact lens**. The woman **puts the contact lens in her eye**. The person **puts a contact lens in the eye**.
MART: A woman is seen looking at the camera. She holds up a contact lens and **puts it in her eye**. She then **puts the contact into the camera**.
VLTinT: A close up of a eye is shown with a person's eye. A person is then seen **putting a contact lens in her eye**. The person then **takes a contact lens out** of her eye.
GT: A woman holds a contact lens on her finger. She puts the contact lens into her eye. She opens her eye with her fingers and takes the contact lens out.

v_UxlSiLBleX4

VTrans: A man **is playing a guitar**. He **is playing the guitar**. He **stops playing the guitar** .
MART: A man is seen sitting on **a stool** holding **a guitar** and playing **a guitar**. The man continues playing the guitar while the camera captures his movements. The man **finishes** the song and smiles .
VLTinT: A man is **sitting down playing an acoustic guitar**. He is playing the guitar. He **finishes playing the guitar** and smiles .
GT: A man is sitting down in a chair. He begins to play an acoustic guitar. He finishes playing the guitar and standing up.

Figure 3.5: Qualitative comparison on ActivityNet Captions *ae-test* split. Red text indicates the captioning mistakes, purple text indicates repetitive patterns, and blue text indicates some distinct expressions.

smoothing with a value of 0.1 and set $\lambda = 0.1$. All these computations and training routines were conducted on a single NVIDIA RTX 3090 (24GB) GPU.

Qualitative Analysis

Fig. 3.5 offers a comparison of our model (VLTinT), the Vanilla Transformer (VTrans) (L. Zhou, Y. Zhou, et al., 2018), and MART (Lei et al., 2020). Overall, VLTinT demonstrates superior performance in generating more descriptive captions with finer details. A key observation is that while VTrans and MART tend to rely on high-frequency words in their captions, VLTinT has a propensity to use more expressive but less frequently occurring words. For instance, in the first example, VLTinT chooses "lassos" over the more commonplace "throws". We attribute this enhanced expressiveness to our VL Encoder, which incorporates relative scene elements, thus

Table 3.1: Performance comparison of our model with other SOTA models on ActivityNet Captions *ae-val* split. † denotes results by us.

Methods	Venue	Input	B4 ↑	M ↑	C ↑	R ↑	Div2 ↑	R4 ↓
Vanilla Trans.	CVPR	Res200/Flow	9.75	15.64	22.16	28.90†	<u>77.40</u>†	7.79
AdvInf	CVPR	C3D/Object	10.04	<u>16.60</u>	20.97	–	–	5.76
GVD	CVPR	Res200/Flow/Object	11.04	15.71	21.95	–	–	8.76
Trans.-XL	ACL	Res200/Flow	10.39	15.09	21.67	30.18†	75.96†	8.54
Trans.-XLRG	ACL	Res200/Flow	10.17	14.77	20.40	–	–	8.85
MART	ACL	Res200/Flow	10.33	15.68	23.42	<u>30.32</u>†	75.71†	<u>5.18</u>
PDVC	ICCV	C3D/Flow	<u>11.80</u>	15.93	<u>27.27</u>	–	–	–
VLTinT (ours)	AAAI	C3D/Ling	**14.93**	**18.16**	**33.07**	**36.86**	**77.72**	**4.87**

Table 3.2: Performance comparison of our model with other SOTA models on ActivityNet Captions *ae-test* split. † denotes results by us.

Methods	Venue	Input	B4 ↑	M ↑	C ↑	R ↑	Div2 ↑	R4 ↓
Vanilla Trans.	CVPR	Res200/Flow	9.31	15.54	21.33	28.98†	77.29†	7.45
Trans.-XL	ACL	Res200/Flow	10.25	14.91	21.71	30.25†	76.17†	8.79
Trans.-XLRG	ACL	Res200/Flow	10.07	14.58	20.34	–	–	9.37
MART	ACL	Res200/Flow	9.78	15.57	22.16	<u>30.85</u>†	75.69†	5.44
MART^{COOT}	NIPS	COOT	10.85	<u>15.99</u>	<u>28.19</u>	–	–	6.64
Memory Trans.	CVPR	I3D	<u>11.74</u>	15.64	26.55	–	**83.95**	**2.75**
VLTinT (ours)	AAAI	C3D/Ling	**14.50**	**17.97**	**31.13**	**36.56**	<u>77.72</u>	<u>4.75</u>

Table 3.3: Performance comparison of VLTinT with other SOTA models on YouCookII validation set.

Methods	Venue	Input	B@4 ↑	M ↑	C ↑	R ↑	R@4 ↓
Vanilla Trans.	CVPR	Res200/Flow	4.38	11.55	38.00	–	–
MART	ACL	Res200/Flow	8.00	15.90	35.74	–	<u>4.39</u>
MART^{COOT}	NIPS	COOT	**9.44**	**18.17**	<u>46.06</u>	–	6.30
VLTinT (ours)	AAAI	C3D/Ling	<u>9.40</u>	<u>17.94</u>	**48.70**	**34.55**	**4.29**

enriching the model's language usage. Another observed issue is caption repetitiveness in VTrans and MART. This is effectively tackled by our proposed TinT Decoder. Notably, when dealing with similar actions (e.g., inserting and removing a contact lens), our VLTinT is able to distinguish between the two, while the other baselines fail to correctly caption these actions. This distinction capability can be credited to the rich spatial information provided by the VL Encoder and the strong temporal coherency facilitated by the TinT Decoder.

Quantitative Analysis

Our model, VLTinT, is benchmarked and compared with previous state-of-the-art (SOTA) video paragraph captioning (VPC) models on ActivityNet Captions (*ae-val*, *ae-test*) and YouCookII, as shown in Tables 3.1, 3.2, and 3.3 respectively. In these tables, we underline the second-best scores and bolden the best scores for each metric. When compared with the SOTA models MART (Lei et al., 2020), MART w/COOT (a variant of MART using COOT feature (Ging et al., 2020), and PDVC (Wang et al., 2021), our VLTinT significantly outperforms them across both accuracy and diversity metrics on ActivityNet Captions. For instance, on the *ae-val* split, our model achieves gains of 3.13%, 1.56%, 5.80%, and 6.54% on the B@4, M, C, and R accuracy metrics respectively, when compared to the second-best performance. Furthermore, diversity increases by 0.32% on Div@2 and reduces by 0.32% on R@4.

On the *ae-test* split, the accuracy gains are 3.65%, 1.98%, 2.94%, and 5.71% on B@4, M, C, and R metrics respectively, while diversity increases by 0.43% on Div@2 and reduces by 0.67% on R@4, all compared to the second-best performance.

On YouCookII, our model sets the best scores on C, R, and R@4 metrics by considerable margins while achieving competitive performance on B@4 and M metrics. This demonstrates the overall effectiveness of our VLTinT in video paragraph captioning tasks across diverse videos.

3.6 Ablation Studies

This section aims to investigate the impact and significance of individual components or factors within the proposed system.

Contribution of each modality in VL Encoder

We examine VLTinT on ActivityNet Captions with different modality settings as given in Table 3.4 and 3.5. The first three rows show the performance on each individual modality whereas the last three rows show the performance on different combinations. Even though the best performance on overall is obtained by combining all three modalities of both vision (environment and agent) and language (scene elements), the performance on only linguistic feature is promising with notable

Table 3.4: Ablation study on the contribution of each modality in VL Encoder on ActivityNet Captions dataset for *ae-val* split. Env., Agt., and Ling. denote the global visual environment, local visual main agents, and linguistic relevant scene elements, respectively.

Env.	Agt.	Ling.	B@4 ↑	M ↑	C ↑	R ↑	Div@2 ↑	R@4 ↓
√	×	×	14.02	17.58	30.31	36.20	76.11	6.08
×	√	×	12.13	16.57	24.98	34.36	79.18	4.24
×	×	√	14.00	17.88	31.64	35.95	**80.44**	**3.22**
√	√	×	14.12	17.78	31.15	36.12	78.02	4.56
√	×	√	14.84	17.97	31.86	36.80	76.41	5.67
√	√	√	**14.93**	**18.16**	**33.07**	**36.86**	77.72	4.87

Table 3.5: Ablation study on the contribution of each modality in VL Encoder on ActivityNet Captions dataset for *at-test* split. Env., Agt., and Ling. denote the global visual environment, local visual main agents, and linguistic relevant scene elements, respectively.

Env.	Agt.	Ling.	B@4 ↑	M ↑	C ↑	R ↑	Div@2↑	R@4 ↓
√	×	×	13.62	17.41	29.09	35.96	76.14	5.97
×	√	×	11.83	16.22	21.39	33.97	79.20	4.16
×	×	√	13.38	17.69	30.30	35.63	**80.50**	**3.32**
√	√	×	13.77	17.52	30.05	35.93	77.78	4.69
√	×	√	**14.53**	17.79	30.83	**36.67**	76.47	5.60
√	√	√	14.50	**17.97**	**31.13**	36.56	77.72	4.75

performance, especially on diversity metrics. This should be included in our future investigation.

Effectiveness of linguistic relevant scene elements

To demonstrate the effectiveness of our chosen method for extracting scene elements, we compare the performance of VLTinT with two scenarios, as detailed in Table 3.6: (i) where scene elements are extracted using Mask-RCNN trained on the COCO dataset with 80 classes (He, Gkioxari, et al., 2017), and (ii) where scene elements are extracted using CLIP. Our ablation study highlights the superiority of CLIP-extracted scene elements over those extracted using Mask-RCNN. Scene elements encompass a broad variety of factors, including human and non-human elements (such as animals and vehicles), and visual and non-visual elements (such as relations and activities). Mask R-CNN, however, is only capable of capturing a limited subset of these elements because it has

Table 3.6: Performance comparison between two cases trained on TinT network without visual feature: (i) scene elements extracted by Mask R-CNN (M RCNN) (ii) scene elements extracted by CLIP. The experiment is done on the ActivityNet Captions dataset *at-test* split.

	B@4↑	M↑	C↑	R↑	R@4↓
M RCNN	1.35	9.09	12.29	23.06	18.52
CLIP	**13.38**	**17.69**	**30.30**	**35.63**	**3.32**

Table 3.7: Comparison between RNN and Transformer to model inter-event dependencies in TinT decoder on ActivityNet Captions with C3D (env+agent) is visual feature in the encoder. Linguistic feature (Ling.) is considered as an option.

	use of ling.	inter-event modeling	B@4↑	M↑	C↑	R↑
ae-val	×	RNN	11.68	16.79	25.86	33.97
		Trans.	**14.12**	**17.78**	**31.15**	**36.12**
	√	RNN	13.75	17.63	28.01	36.21
		Trans.	**14.93**	**18.16**	**33.07**	**36.86**
ae-test	×	RNN	11.10	15.72	27.67	31.75
		Trans.	**13.77**	**17.52**	**30.05**	**35.93**
	√	RNN	13.45	17.42	29.68	36.09
		Trans.	**14.50**	**17.97**	**31.13**	**36.56**

been trained on a relatively small number of visual objects/classes. Consequently, its diversity and scene understanding performance are poorer compared to CLIP. The results demonstrate the improved effectiveness and utility of CLIP for extracting linguistically relevant scene elements in our VLTinT.

Effectiveness of VL Loss $\mathcal{L}_{VL}$

We examine the effectiveness of the VL Loss by replacing it with the MLE loss, a commonly employed loss function in VPC. The performance of the VLTinT model on the ActivityNet Captions *ae-test* dataset, when implemented with these two different loss functions, is presented in Table 3.8. Our analysis confirms that aligning the semantics of the latent feature with the groundtruth caption indeed contributes to enhancing the model's performance.

Table 3.8: Effectiveness of $\mathcal{L}_{VL}$ compared to the standard MLE loss on ActivityNet Captions *ae-test* split.

Loss	B@4↑	M ↑	C ↑	R ↑
MLE	13.80	17.72	30.59	36.11
$\mathcal{L}_{VL}$ (ours)	**14.50**	**17.97**	**31.13**	**36.56**

Table 3.9: Computational cost vs. accuracy between VLTinT (Env./Agent./Ling.) with different settings and SOTA VPC models.

	Computational cost			Accuracy	
Models	Params↓	Comp. ↓	Inf.↓	M↑	C↑
MART	36.25	6.32	0.025	15.57	22.16
Mem Trans	29.69	256.44	0.706	16.10	27.36
Env.	36.01	17.69	0.028	17.41	29.09
Env./Agent.	40.37	22.70	0.032	17.52	30.05
Env./Agent./Ling.	43.40	40.37	0.038	17.97	31.13

Computational Complexity

Finally, we evaluate the computational complexity of our proposed VLTinT model in comparison to state-of-the-art (SOTA) VPC models on the ActivityNet *ae-test* dataset. The evaluation metrics include trainable parameters (measured in millions), computation (GFLOPs), average inference time (seconds) over 100 random videos, and accuracy metrics. These results are compiled in Table 3.9. We present evaluations of our VLTinT model under different settings for a comprehensive understanding. When compared to the SOTA models, our model, when implemented with only environmental features, exhibits compatible parameter count and inference time while demonstrating improved performance. Furthermore, when our model is implemented with environmental, agent, and linguistic features, it achieves a substantial improvement in accuracy while maintaining reasonable computational complexities.

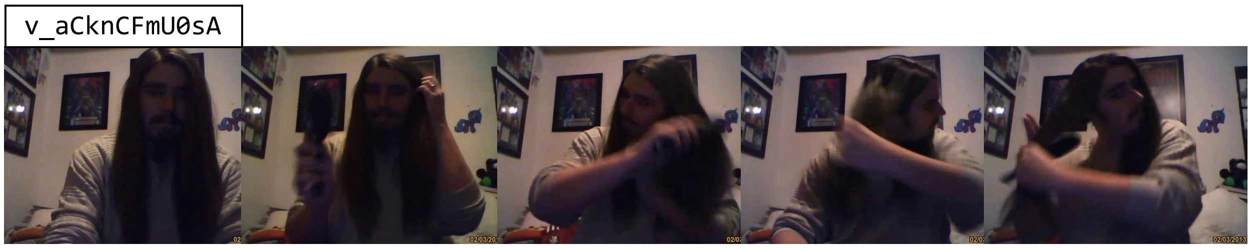
Analysis of TinT Decoder

Our TinT Decoder is structured as a nested transformer architecture, in which the inner transformer addresses intra-event coherence and the outer transformer manages inter-event coherence. Quantitative analysis of the TinT Decoder, replacing the outer transformer with an RNN-based network

RNN: A group of people are **on a beach playing volleyball**. They lob the ball back and forth over the net. They hit **the ball back and forth over the net**.

Trans: A group of people are **playing volleyball on a beach**. They lob the ball back and forth over the net. The game continues on with the ball, and the other teammates play.

GT: A group of girls are on a sandy beach. They are engaged in a game of volleyball. They lob the ball back and forth over the net.



RNN: **A young woman** is seen sitting in front of a camera and begins brushing her hair. **She** then brushes **her hair** down and **begins brushing her hair**. **She** continues brushing the hair and looking off into the camera.

Trans: **A young man** is seen speaking to the camera while holding up a brush. **The man** then begins brushing **his hair** and **looking** back **to the camera**. **He** continues brushing his hair and looking off into the distance.

GT: A man with long hair is seen looking at the camera and begins brushing his hair. The man brushes his hair all around while still looking down at the camera. The man turns around to finish brushing his hair and ends by waving to the camera.

Figure 3.6: Qualitative analysis of inter-event modeling by RNN (the first row) and our outer transformer (the second row), whereas the groundtruth is shown in the last row. Red text indicates the captioning mistakes, purple text indicates repetitive patterns, and blue text indicates some distinct expressions.

(Lei et al., 2020), is provided in Table 3.7. This comparison was made using two encoder feature settings: one including and the other excluding linguistic features, while C3D (encompassing both environment and agent) was employed as visual features. Our results indicate a significant performance boost through the modeling of inter-event relationships with our autoregressive outer transformer.

When using the same comparison settings, qualitative results are depicted in Fig. 3.6. Aside from minor captioning errors, the main issue with RNN-based inter-event coherence modeling lies in its tendency towards repetitive patterns. This indicates that the relationships between sentences are not effectively handled by the RNN-based network, highlighting the advantages of our proposed TinT Decoder. By using the outer transformer for inter-event coherence and the inner transformer for intra-event coherence, the TinT Decoder exhibits superior performance in managing complex contextual relationships.

Chapter 4

CONCLUSION & FUTURE WORK

In this thesis, we have embarked on an in-depth exploration of video understanding, focusing on Video Paragraph Captioning (VPC). This area poses unique challenges, necessitating comprehensive insights into both visual and contextual dependencies in videos.

Our main contribution is the development of VLTinT, a novel end-to-end model designed specifically for VPC tasks. The architecture of VLTinT is comprised of two integral components: the VL Encoder and the Transformer-in-Transformer (TinT) Decoder. The VL Encoder effectively extracts video features using three modalities: the global visual environment, the local visual main agents, and the linguistically relevant scene elements. The fusion of these diverse features is facilitated by a multi-modal representation fusion (M2RF) mechanism, ensuring a comprehensive representation of the video content.

As for the TinT Decoder, it capitalizes on a nested transformer design to effectively capture both intra-event and inter-event coherencies. The unified inner transformer is tasked with modeling intra-event relationships, while the autoregressive outer transformer focuses on inter-event connections.

To train our VLTinT framework, we introduce the Video-linguistic (VL) contrastive loss, $\mathcal{L}_{VL}$, designed to align the semantics of the latent feature with the ground truth caption, contributing significantly to the model's performance.

The efficacy of our proposed VLTinT was validated on several popular benchmark datasets. The model's superior performance, in terms of both accuracy and diversity of generated captions, highlights its potential in contributing to the advancements in the field of video understanding.

Below, we point out several important future directions that should be further investigated.

Utilization of Foundation models: Over the past months, we have witnessed the exponential rise of large-scale foundation models in various domains. Early examples of models include GPTs (T. Brown et al., 2020), CLIP (Patashnik et al., 2021), DALL·E (Ramesh et al., 2021), SAM

(Kirillov et al., 2023), and more. These models, pretrained on diverse and extensive data, have demonstrated extraordinary performance in a wide range of downstream tasks. In this thesis, we have utilized CLIP to obtain linguistic concepts of the scene but many other possibilities of utilizing the foundation models are yet to be explored. Our VLTinT framework, while proficient in VPC tasks, could potentially benefit from incorporating these foundation models. Some of the possible directions include: adopting pretrained LLM for the captioning decoder, exploring the internal representations of the VL models for the video representation, and employing transformer models such as SAM for understanding segment-level dependencies.

Integration of other modalities: While our VLTinT focuses primarily on the visual and linguistic modalities, there is considerable potential for enriching video understanding by incorporating other modalities. Audio, for instance, is a crucial component of many videos and can provide important contextual cues that are not always visually apparent. Similarly, the integration of text modality in the form of subtitles or transcriptions can add another layer of contextual understanding, especially in dialog-heavy videos. In future work, we aim to expand our VLTinT to become a multimodal framework that can seamlessly integrate and learn from these additional modalities. This could significantly improve the model's ability to comprehend and describe complex videos, leading to even more accurate and comprehensive video paragraph captioning.

BIBLIOGRAPHY

Agarap, Abien Fred (2018). “Deep learning using rectified linear units (relu)”. In: *arXiv preprint arXiv:1803.08375*.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. (2020). “Language models are few-shot learners”. In: *Advances in neural information processing systems* 33, pp. 1877–1901.

Carreira, Joao and Andrew Zisserman (2017). “Quo vadis, action recognition? a new model and the kinetics dataset”. In: *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308.

Chen, Ting, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton (13–18 Jul 2020). “A Simple Framework for Contrastive Learning of Visual Representations”. In: *Proceedings of the 37th International Conference on Machine Learning*. PMLR, pp. 1597–1607.

Chen, Yen-Chun, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu (2020). “UNITER: UNiversal Image-TExt Representation Learning”. In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm. Cham: Springer International Publishing, pp. 104–120. ISBN: 978-3-030-58577-8.

Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (2014). “Learning phrase representations using RNN encoder-decoder for statistical machine translation”. In: *arXiv preprint arXiv:1406.1078*.

Chung, Junyoung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio (2014). “Empirical evaluation of gated recurrent neural networks on sequence modeling”. In: *NIPS 2014 Workshop on Deep Learning, December 2014*.

Dai, Zihang, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov (July 2019). “Transformer-XL: Attentive Language Models beyond a Fixed-Length Context”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 2978–2988. DOI: 10.18653/v1/P19-1285. URL: https://aclanthology.org/P19-1285.

Deng, Chaorui, Shizhe Chen, Da Chen, Yuan He, and Qi Wu (2021). “Sketch, Ground, and Refine: Top-Down Dense Video Captioning”. In: *CVPR*, pp. 234–243. DOI: 10.1109/CVPR46437.2021.00030.

Denkowski, Michael and Alon Lavie (June 2014). “Meteor Universal: Language Specific Translation Evaluation for Any Target Language”. In: *Proceedings of the Ninth Workshop on Statistical Machine Translation*. Baltimore, Maryland, USA: Association for Computational Linguistics, pp. 376–380. DOI: 10.3115/v1/W14-3348. URL: https://aclanthology.org/W14-3348.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby (2021). “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *ICLR*.

Feichtenhofer, Christoph, Haoqi Fan, Jitendra Malik, and Kaiming He (2018). "SlowFast networks for video recognition. 2019 IEEE". In: *CVF international conference on computer vision (ICCV)*, pp. 6201–6210.

Geva, Mor, Roei Schuster, Jonathan Berant, and Omer Levy (2020). "Transformer feed-forward layers are key-value memories". In: *arXiv preprint arXiv:2012.14913*.

Ging, Simon, Mohammadreza Zolfaghari, Hamed Pirsiavash, and Thomas Brox (2020). "COOT: Cooperative Hierarchical Transformer for Video-Text Representation Learning". In: *Advances on Neural Information Processing Systems (NeurIPS)*.

He, Kaiming, Georgia Gkioxari, Piotr Dollar, and Ross Girshick (Oct. 2017). "Mask R-CNN". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Hendrycks, Dan and Kevin Gimpel (2016). "Gaussian error linear units (gelus)". In: *arXiv preprint arXiv:1606.08415*.

Henighan, Tom, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. (2020). "Scaling laws for autoregressive generative modeling". In: *arXiv preprint arXiv:2010.14701*.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780.

Iashin, Vladimir and Esa Rahtu (2020). "Multi-Modal Dense Video Captioning". In: *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 958–959.

Ji, Shuiwang, Wei Xu, Ming Yang, and Kai Yu (2013). "3D Convolutional Neural Networks for Human Action Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.1, pp. 221–231. DOI: `10.1109/TPAMI.2012.59`.

Jozefowicz, Rafal, Wojciech Zaremba, and Ilya Sutskever (2015). "An empirical exploration of recurrent network architectures". In: *International conference on machine learning*. PMLR, pp. 2342–2350.

Kaplan, Jared, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei (2020). "Scaling laws for neural language models". In: *arXiv preprint arXiv:2001.08361*.

Kay, Will, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman (2017). "The Kinetics Human Action Video Dataset". In: *CoRR* abs/1705.06950. arXiv: `1705.06950`. URL: `http://arxiv.org/abs/1705.06950`.

Kirillov, Alexander, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. (2023). "Segment anything". In: *arXiv preprint arXiv:2304.02643*.

Krishna, Ranjay, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles (2017). “Dense-Captioning Events in Videos”. In: *International Conference on Computer Vision (ICCV)*.

Lei, Jie, Liwei Wang, Yelong Shen, Dong Yu, Tamara Berg, and Mohit Bansal (July 2020). “MART: Memory-Augmented Recurrent Transformer for Coherent Video Paragraph Captioning”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 2603–2614. DOI: 10.18653/v1/2020.acl-main.233. URL: https://aclanthology.org/2020.acl-main.233.

Li, Yehao, Ting Yao, Yingwei Pan, Hongyang Chao, and Tao Mei (June 2018). “Jointly Localizing and Describing Events for Dense Video Captioning”. In: *CVPR*.

Lin, Chin-Yew (July 2004). “ROUGE: A Package for Automatic Evaluation of Summaries”. In: *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, pp. 74–81. URL: https://aclanthology.org/W04-1013.

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick (2014). “Microsoft COCO: Common Objects in Context”. In: *Computer Vision – ECCV 2014*. Ed. by David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars. Cham: Springer International Publishing, pp. 740–755. ISBN: 978-3-319-10602-1.

Mun, Jonghwan, Linjie Yang, Zhou Ren, Ning Xu, and Bohyung Han (June 2019). “Streamlined Dense Video Captioning”. In: *CVPR*.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu (2002). “BLEU: A Method for Automatic Evaluation of Machine Translation”. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. ACL ’02. Philadelphia, Pennsylvania: Association for Computational Linguistics, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: https://doi.org/10.3115/1073083.1073135.

Park, Jae Sung, Marcus Rohrbach, Trevor Darrell, and Anna Rohrbach (2019). “Adversarial Inference for Multi-Sentence Video Description”. In: *CVPR*, pp. 6591–6601. DOI: 10.1109/CVPR.2019.00676.

Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio (17–19 Jun 2013). “On the difficulty of training recurrent neural networks”. In: *Proceedings of the 30th International Conference on Machine Learning*. Ed. by Sanjoy Dasgupta and David McAllester. Vol. 28. Proceedings of Machine Learning Research 3. Atlanta, Georgia, USA: PMLR, pp. 1310–1318. URL: https://proceedings.mlr.press/v28/pascanu13.html.

Patashnik, Or, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski (Oct. 2021). “StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2085–2094.

Patro, Badri and Vinay P. Namboodiri (June 2018). “Differential Attention for Visual Question Answering”. In: *CVPR*.

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever (18–24 Jul 2021). “Learning Transferable Visual Models From Natural Language Supervision”. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 8748–8763. URL: https://proceedings.mlr.press/v139/radford21a.html.

Rahman, Tanzila, Bicheng Xu, and Leonid Sigal (Oct. 2019). "Watch, Listen and Tell: Multi-Modal Weakly Supervised Dense Event Captioning". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV).*

Ramesh, Aditya, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever (2021). "Zero-shot text-to-image generation". In: *International Conference on Machine Learning*. PMLR, pp. 8821–8831.

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun (2015). "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc.

Shetty, Rakshith, Marcus Rohrbach, Lisa Anne Hendricks, Mario Fritz, and Bernt Schiele (Oct. 2017). "Speaking the Same Language: Matching Machine to Human Captions by Adversarial Training". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV).*

Shi, Botian, Lei Ji, Yaobo Liang, Nan Duan, Peng Chen, Zhendong Niu, and Ming Zhou (July 2019). "Dense Procedure Captioning in Narrated Instructional Videos". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 6382–6391. DOI: 10.18653/v1/P19-1641. URL: https://aclanthology.org/P19-1641.

Song, Yuqing, Shizhe Chen, and Qin Jin (June 2021). "Towards Diverse Paragraph Captioning for Untrimmed Videos". In: *CVPR*, pp. 11245–11254.

Tran, D, L Bourdev, R Fergus, L Torresani, and M Paluri (2014). "Learning Spatiotemporal Features with 3D Convolutional Networks. ArXiv e-prints". In: *arXiv preprint arXiv:1412.0767.*

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Vedantam, Ramakrishna, C. Lawrence Zitnick, and Devi Parikh (2015). "CIDEr: Consensus-based image description evaluation". In: *CVPR*, pp. 4566–4575. DOI: 10.1109/CVPR.2015.7299087.

Vo, Khoa, Hyekang Joo*, Kashu Yamazaki*, Sang Truong, Kris Kitani, Minh-Triet Tran, and Ngan Le (2021). "AEI: Actors-Environment Interaction with Adaptive Attention for Temporal Action Proposals Generation". In: *32nd British Machine Vision Conference 2021, BMVC 2021, Online, November 22-25, 2021*. BMVA Press, p. 111. URL: https://www.bmvc2021-virtualconference.com/assets/papers/1095.pdf.

Vo, Khoa, Ngan Le, Kashu Yamazaki, Akihiro Sugimoto, and Minh-Triet Tran (2021). "Agent-Environment Network for Temporal Action Proposal Generation". In: *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2160–2164. DOI: 10.1109/ICASSP39728.2021.9415101.

Vo, Khoa, Sang Truong*, Kashu Yamazaki*, Bhiksha Raj, Minh-Triet Tran, and Ngan Le (2023). "AOE-Net: Entities Interactions Modeling with Adaptive Attention Mechanism for Temporal Action Proposals Generation". In: *International Journal of Computer Vision* 131.1, pp. 302–323. ISSN: 1573-1405. DOI: 10.1007/s11263-022-01702-9. URL: https://doi.org/10.1007/s11263-022-01702-9.

Vo, Khoa, Kashu Yamazaki, Phong X. Nguyen, Phat Nguyen, Khoa Luu, and Ngan Le (2022). *Contextual Explainable Video Representation: Human Perception-based Understanding*. arXiv: 2212.06206 [cs.CV].

Vo, Khoa, Kashu Yamazaki, Sang Truong, Minh-Triet Tran, Akihiro Sugimoto, and Ngan Le (2021). "ABN: Agent-Aware Boundary Networks for Temporal Action Proposal Generation". In: *IEEE Access* 9, pp. 126431–126445. DOI: 10.1109/ACCESS.2021.3110973.

Wang, Teng, Ruimao Zhang, Zhichao Lu, Feng Zheng, Ran Cheng, and Ping Luo (Oct. 2021). "End-to-End Dense Video Captioning With Parallel Decoding". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6847–6857.

Wu, Zhirong, Yuanjun Xiong, Stella X. Yu, and Dahua Lin (June 2018). "Unsupervised Feature Learning via Non-Parametric Instance Discrimination". In: *CVPR*.

Xiong, Yilei, Bo Dai, and Dahua Lin (2018). "Move Forward and Tell: A Progressive Generator of Video Descriptions". In: *Computer Vision – ECCV 2018*. Ed. by Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss. Cham: Springer International Publishing, pp. 489–505. ISBN: 978-3-030-01252-6.

Yamazaki, Kashu, Sang Truong, Khoa Vo, Michael Kidd, Chase Rainwater, Khoa Luu, and Ngan Le (2022). "VLCAP: Vision-Language with Contrastive Learning for Coherent Video Paragraph Captioning". In: *2022 IEEE International Conference on Image Processing (ICIP)*, pp. 3656–3661. DOI: 10.1109/ICIP46576.2022.9897766.

Yang, Bang, Tong Zhang, and Yuexian Zou (2022). *CLIP Meets Video Captioning: Concept-Aware Representation Learning Does Matter*. arXiv: 2111.15162 [cs.CV].

Zhou, Luowei, Chenliang Xu, and Jason Corso (Apr. 2018). "Towards Automatic Learning of Procedures From Web Instructional Videos". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1. DOI: 10.1609/aaai.v32i1.12342.

Zhou, Luowei, Yingbo Zhou, Jason J. Corso, Richard Socher, and Caiming Xiong (June 2018). "End-to-End Dense Video Captioning With Masked Transformer". In: *CVPR*.

INDEX