GEM-VPC: A dual Graph-Enhanced Multimodal integration for Video Paragraph Captioning

Anonymous ACL submission

Abstract

Video Paragraph Captioning (VPC) aims to generate paragraph captions that summarises key events within a video. Despite recent advancements, challenges persist, notably in effectively utilising multimodal signals inherent in videos and addressing the long-tail distribution of words. The paper introduces a novel multimodal integrated caption generation framework for VPC that leverages information from various modalities and external knowledge bases. Our framework constructs two graphs: a '*video-specific*' temporal graph capturing major events and interactions between multimodal information and commonsense knowledge, and a '*theme graph*' representing correlations between words of a specific theme. These graphs serve as input for a transformer network with a shared encoder-decoder architecture. We also introduce a node selection module to enhance decoding efficiency by selecting the most relevant nodes from the graphs. Our results demonstrate superior performance across benchmark datasets.

1 Introduction

Dense video captioning (DVC) (Krishna et al., 2017) is a sub-branch of video captioning, which requires the model to first localise the important events in the video and then generate the associated captions. Video paragraph captioning (VPC) (Park et al., 2019) is a simplified version of DVC where the event segments in a video are assumed given; therefore, the event proposal generation step is not needed, and the ultimate goal is to generate better paragraph captions with the known events. While research in video captioning is recently becoming more popular, numerous challenges still persist. Firstly, most VPC works solely use visual information for generating captions (Park et al., 2019; Song et al., 2021). However, they overlook that videos naturally contain rich content with multimodal signals such as additional speech text and an audio soundtrack. Incorporating these extra modalities and unravelling their interactions can provide vital cues for video understanding. Another challenge is overcoming the long-tail distribution of words, whereby the model tends to overfit on frequent terms while neglecting objects, properties or behaviours that rarely appear in the training data. Past natural language generation works have shown that exploiting external data from knowledge graphs can alleviate this issue and encourage more diverse generated text (Zhou et al., 2019b). Finally, existing studies (Iashin and Rahtu, 2020b; Lei et al., 2020) simply feed the video's feature embeddings into the captioning model directly, leading to two problems: 1) the model cannot effectively handle the long sequence, and 2) it struggles to select the relevant context from the long input stream.

As such, we address the aforementioned challenges by introducing GEM-VPC, a graph-based novel framework for VPC that integrates information from various modalities. Unlike past works (Iashin and Rahtu, 2020b,a), rather than purely feeding in the raw features as a long input stream, we first convert the videos into a graphical structure to capture high-level salient features and context. We construct two types of graphs. The first is a '*video-specific*' temporal graph, which aims to depict the major events of the video in chronological order whilst simultaneously representing interactions between various multimodal information and related commonsense knowledge. In particular, nodes are represented using language class labels to provide key details about the video contents instead of using raw feature embeddings, which may contain noisy information. To this end, we leverage pretrained action/audio/object recognition models and text parsers to extract linguistic information such as the action label, sound label or object label from the visual features, audio features and speech transcript to be used as nodes in the graph. To alleviate the long-tail problem, we further enhance the graph by incorporating language features from an external knowledge data source. While other VPC studies (Gu et al., 2023) using knowledge graphs typically employ static graphs like ConceptNet (Speer et al., 2017), we use a neural knowledge model trained on existing commonsense knowledge graph datasets to generate diverse commonsense about human everyday experiences on-demand. These nodes are then connected with informative edge labels. We utilise sentences from the corpus to create a '*theme graph*' to represent correlations between words relating to a specific theme with the motivation of providing corpus-level information for each sample during training. In the model training stage, both graphs are finally fed as supporting information into a transformer network. As some nodes in the graph may be noisy, we propose a node selection module to select only the most useful nodes from the video-specific and theme graphs when decoding the caption.

The main contributions are to: 1) introduce a novel framework for VPC that leverages multi-modal commonsense knowledge to enhance video understanding. It incorporates heterogeneous video and theme graphs derived from various modalities, including visual, audio, and textual data, along with commonsense knowledge. 2) demonstrate the superior performance of our model compared to state-of-the-art methods on two widely used benchmarks. 3) conduct a comprehensive ablation analysis to dissect the contribution of different components.

2 Related Work[1]

Video Paragraph Captioning: Earlier works for VPC often employ an LSTM-based model for generating the captions (Xiong et al., 2018; Zhang et al., 2018; Zhou et al., 2019a). Park et al. (2019) adopts adversarial training in their LSTM model by proposing a hybrid discriminator to measure the language characteristics, relevance to a video segment, and coherence of their generated captions. Transformer-based (Vaswani et al., 2017) methods have become increasingly popular (Ging et al., 2020; Wang et al., 2021; Yamazaki et al., 2023; Gu et al., 2023). This was first introduced by (Zhou et al., 2018) for DVC and VPC, and each event in the video is decoded separately, resulting in context fragmentation and poor inter-event coherency. Later works have tried to alleviate this issue such as in MART (Lei et al., 2020), which modified Transformer-XL (Dai et al., 2019) and proposed a memory module for remembering the video segments and the sentence history to improve future caption predictions with respect to coherence and repetition aspects. Yamazaki et al. (2023) extracts local and global visual features and linguistic scene elements and leverages a Transformer to simultaneously model the long-range dependencies between features at an intra- and inter-event level.

Multimodal Video Captioning: Existing studies have integrated multimodal features as extra information for video captioning. Most works consider the audio modality, with their frameworks first encoding the modalities separately with modality-specific encoders, followed by a fusion unit to combine the multiple streams together (Xu et al., 2017; Rahman et al., 2019; Iashin and Rahtu, 2020a). Other than video and audio modalities, previous studies have suggested that considering speech features can enhance model outputs (Iashin and Rahtu, 2020b). In Hessel et al. (2019) and Shi et al. (2019), automatic speech recognition (ASR) was used to extract human speech from narrated instructional cooking videos for DVC while in Gu et al. (2023), commonsense from knowledge graphs was incorporated into their captioning model where the ASR was used as a source for constructing the graph. Inspired by these methods, we consider the audio and speech modality as model inputs. Unlike the aforementioned approaches, we convert the videos into a heterogeneous graph from language labels extracted from the raw modality segments to represent relationships between key temporal events and different modality information, and propose a novel approach for explicitly incorporating the external commonsense knowledge into the graph.

Some studies propose pretraining tasks to explicitly align the different modalities for improving feature representation, after which the model is fine-tuned to the captioning task. Common pretraining objectives involve predicting whether an ASR and video segment are aligned or predicting masked speech segments and frames (Huang et al., 2020; Luo et al., 2020; Li et al., 2020). Generative pretraining objectives have been explored in (Yang et al., 2023) and (Seo et al., 2022), which proposed predicting the transcribed speech given related video frames to jointly train the visual encoder and text decoder. Our framework requires no pretraining, but can achieve comparable scores to VPC models that utilise such methods.

[1]The main integration methods of past works are highlighted in Table 1 and 2

Graphs for Video Analysis: Graph structures have been widely used in video-related tasks from video scene graph classification (Arnab et al., 2021), temporal action localisation (Zeng et al., 2019) to video question and answering (Jiang and Han, 2020). Several studies have delved into 'spatio-temporal' graphs that try to represent interactions of features at a static time and relations between features across time. For the spatial component, numerous works connect objects and regions together within a timeframe and then connect identical or similar objects across time for the temporal component (Pan et al., 2020; Zhang et al., 2020; Jin et al., 2021; Min et al., 2022). In VPC, (Ji et al., 2022) proposed a multimodal heterogeneous graph that connects visual and text features within the same event. While they use the raw feature embeddings for node representation, which create large graphs with noisy information, we utilise the linguistic labels to provide a more high-level representation of the key semantic contents of the video and further propose a node selection module to filter out irrelevant nodes.

3 Method

Problem Definition: Given an untrimmed video v with temporally ordered events $E = \{e_{v1}, e_{v2}, ..., e_{vN}\}$ where e_{vt} is the event at timestep t defined by a starting and ending timestamp (e^s_{vt}, e^e_{vt}) and N is the total number of events in the video, the task of VPC is to generate $Y = \{y_{v1}, y_{v2}, ..., y_{vN}\}$ where y_{vt} is a matching textual description for e_{vt}.

We first describes constructing the graphs as input for our VPC model. Two graphs (Section 3.1 and 3.2) are built: 1) a commonsense-enhanced video-specific graph (VG), representing the main sequential events in the video with related commonsense and contextual information, and 2) a theme graph (TG) representing relationships between vocabulary of a specific theme. For the video-specific graphs, we propose two ways to construct the primary nodes: 1) Utilising the video's visual information ('VF-method') and 2) extracting information from the speech transcript ('ASR-method').

3.1 Video-Specific Graph Creation

3.1.1 Creating the Nodes - VF-Method

Graphs created using the VF-method have 3 main node types: action, context (consisting of location, object, audio nodes), and commonsense nodes.

Action Nodes: The action nodes describe the main actions at each key event and are represented using linguistic action class labels. To obtain these labels, we download the video frames at 5fps. For each event e_{vt}, we uniformly sample frames between the event's starting and ending frames with a step size of 10 and then feed every 16 frames into a pretrained video action classification model for each 16-frame segment. As the agent does not always perform a specific action (e.g. just standing or no human agent in the video segment), we replace the class label with '*no action*' if the predicted class probability is less than a threshold. When less than the threshold and speech is detected by the audio node, we replace the label with '*speaking*'.

Context Nodes: For extra scene context, we include location, object and audio nodes. For the location and object nodes, we take the centre and last frame of each event and leverage a Visual Question Answering (VQA) model to extract open-ended answers about the images. For the location node, we ask the VQA model *'what is the location?'* for each of the 3 images and take the most common answer as the location for each event. For the object nodes, we obtain the object labels by asking 3 questions: *'what objects are in this image?'*, *what is in the background?'* and *'who is in this image?'*. We further expand this object set by employing an object detection model to detect objects from the frames. Finally, the audio nodes represent the sound information and can provide vital cues for video understanding in addition to the visual information. We sample 10 second segments of audio data from the video and obtain the the top 2 predicted audio classes by confidence score for each segment via a pretrained audio classifier.

Commonsense Nodes We also add external commonsense knowledge for richer graphs. Comet-ATOMIC2020 (Hwang et al., 2021), a *neural knowledge model* capable of dynamically generating commonsense about everyday events is adopted. Given a head phrase and relation (e.g. cut a cake `CapableOf`), Comet-ATOMIC2020 can produce a tail phrase on-demand (e.g. celebrate birthday). We use the action node class labels as the head phrase and append 11 different relation tokens to generate 5 commonsense inferences per relation. The relation is described in Appendix E.

3.1.2 Creating the Nodes - ASR-Method

For videos where the speech modality is considered vital for video understanding, we introduce the ASR-method for creating the VG nodes. This is useful for how-to or cooking videos, where actions are explicitly described in the speech transcript, and visual information such as the location/scene may not be as important. There are 3 node types:

Action Nodes: We extract the ASR between each event and use a pretrained Open Information Extraction (OpenIE) model to breakdown the syntactically complex speech sentences into a list of verbs (V) and related arguments (ARG). Given the sentence *'I chop the onions and put the meat in the frying pan'*, OpenIE can extract related arguments for the 2 verbs (*'chop'* and *'put'*): <ARG0, V, ARG1> = <I, chop, onions> and <ARG0, V, ARG1, ARG2> = <I, put, meat, in the frying pan>. The extracted verb and argument tuples from the speech segments within each event are then used as the action nodes for event e_i. As the speech may contain irrelevant content, we tag the verbs in the ground-truth annotations and only retain tuples if the extracted verb has a high word embedding similarity score with at least one of the tagged verbs in the annotations. Moreover, we only retain words from the extracted arguments if it is a noun/adverb in the training annotations.

Context Nodes: Instead of location nodes as introduced in the VF-method, we concatenate the action node labels within the same event to form a 'contextual phrase node'. This represents similar information to the action nodes, but at a less fine-grained level with more context about surrounding actions. For the object nodes, we tag the nouns from the ASR segment, retaining only the tagged nouns if they appear in the training ground-truth annotations. The audio nodes are retrieved in the same way as the VF-method except we filter out any irrelevant sound labels. For example, with cooking videos, we retain cooking-related sounds ('*boiling*', '*sizzling*', '*frying*', '*chopping*' etc).

Commonsense Nodes We follow the VF-method but instead of using the action node information as the head phrase, we find that better commonsense is generated when using the linguistic information inside the contextual phrase node to query Comet-ATOMIC2020.

3.1.3 Connecting the VG Nodes

For event e_{vt}, let $AC_t = \{ac_{t1}, ..., ac_{tk}\}$ be the action nodes, l_t be the corresponding location node when the VF-method is used, or cp_t be the contextual phrase node when the ASR-method is used, $CK_t = \{ck_{t1}, ..., ck_{tm}\}$ are the commonsense nodes, $O_t = \{o_{t1}, ..., o_{tn}\}$ are the object nodes, and $AU_t = \{au_{t1}, ..., au_{tp}\}$ are the audio nodes.

To form the graph, all action nodes are first connected in temporal order. To capture forward information, we add a directed edge with the label `occursAfter` between each consecutive action node and further capture backwards information by adding a reversed edge with the label `occursBefore`. Each location node l_t or contextual phrase node cp_t is then connected to all the nodes in AC_t with the edge label `atLocation` or `hasContext`. Next, commonsense nodes from CK_t are connected to the corresponding action nodes from AC_t that were used to generate the commonsense, using the commonsense relation token as the edge label. For the object and audio nodes, each node in O_t and AU_t is connected with l_t or cp_t with the edge label `inScene` and `hasSound` respectively. For the VF-method, we additionally filter out any irrelevant commonsense if the predicted action class confidence score used to generate that commonsense does not exceed a particular threshold. Noisy audio or object labels are disregarded at each timestep by converting the class labels to a text embedding and only retaining those that have a high cosine similarity score with any of the nodes in AC_t, CK_t or l_t. A depiction of the final graphs using the VF- and ASR-method is in Appendix I.

3.2 Theme Graph Creation

We also create a theme graph for each action class to incorporate corpus-level information. Given an action predicted at e_{vt}, we collect the corresponding ground-truth training sentence at e_{vt} and tag the nouns, verbs and adverbs to build a vocabulary for each action class. With the ASR-method, the action classes are created by the k-means algorithm to cluster the text embeddings of the action nodes. We retain the top-n most frequent words for each action class vocabulary and following Yao et al. (2019), the individual words are connected based on word co-occurrence statistics to form a graph.

$$PMI(i,j) = log\frac{p(i,j)}{p(i)p(j)} \tag{1}$$

$$NPMI = \frac{PMI}{-log(p(i,j))} \tag{2}$$

We utilise the normalised point-wise mutual information score (NPMI), where a positive score implies high semantic correlation between words. Here, $p(i,j) = \frac{\#S(i,j)}{\#S}$, $p(i) = \frac{\#S(i)}{\#S}$ and $p(j) =$

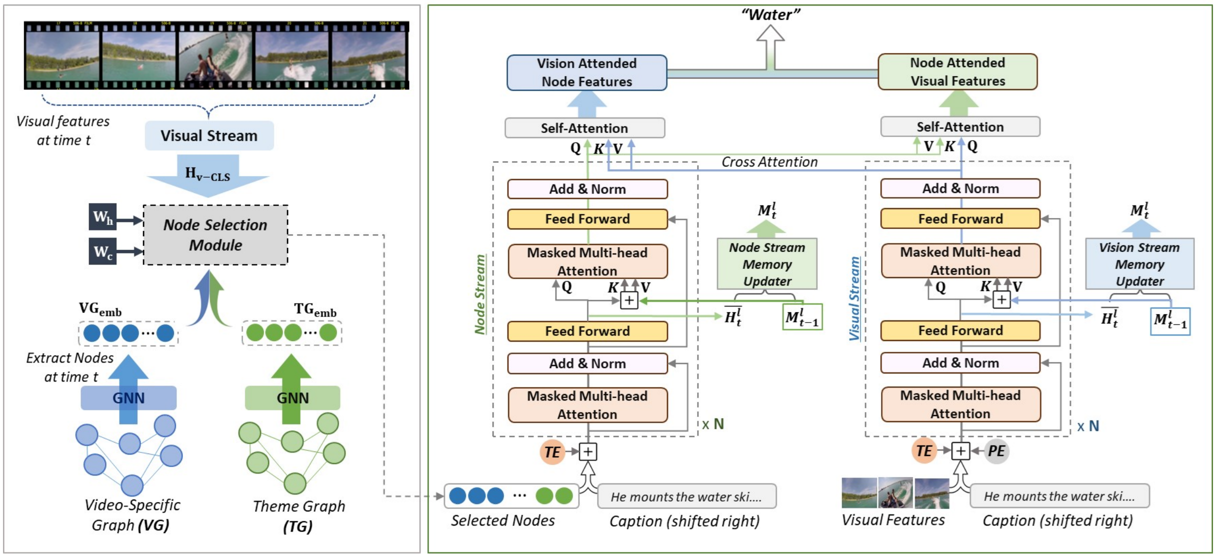


Figure 1: Architecture of GEM-VPC. At time t, the entire video-specific (VG) and theme graph (TG) corresponding to the action at time t is fed into separate Graph Neural Networks. In the visual stream, visual features summed with positional (PE) and token type embeddings (TE) are inputted into a Recurrent Transformer and the sequence representation (H_{v-CLS}) is then used to select nodes from VG and TG in the node selection module. The selected nodes plus TE are fed into another Recurrent Transformer in the node stream. Cross-attention is employed between the visual and node stream and cross-attended features are finally fed into an MLP to predict the next word.

$\frac{\#S(j)}{\#S}$ where $\#S(i)$ is the number of sentences in the corpus that contain word i, $\#S(i,j)$ is the number of sentences that contain both words and $\#S$ is the number of sentences in the corpus. For the corpus, we use the ground-truth sentences from external datasets (see Section 4). A word-to-word connection is made only if the NPMI score exceeds 0.10. A theme graph example is in Appendix F.

3.3 VPC Model

GEM-VPC (Figure 1) adopts a transformer-based shared encoder-decoder augmented with an external memory module to model temporal dependencies between events.

1) Visual Stream: At each timestep t related to event e_t, we concatenate the visual features F_V and predicted video captions F_C from e_t. A [CLS] token is also prepended to learn the sequence representation. We denote the concatenated sequence as $F_{VC} = concat(F_V, F_C)$. F_{VC} is fed into a transformer with learnt positional and token type embeddings (for indicating the token's modality type), which applies multi-head self attention (MHA):

$$MHA(Q,K,V) = softmax(\frac{QK}{\sqrt{d_k}} + M)V \quad (3)$$

where $Q = XW^Q$, $K = XW^K$, $V = XW^V$, W^Q, W^K, and W^V are learnable parameters, $X = F_{VC}$ and M is a masked matrix to prevent the model from attending to future words. The outputted intermediate hidden state $\bar{H}^l_t$ is then fed into another attention layer that performs MHA between $\bar{H}^l_t$ and past memory states for capturing history information.

2) Node Stream: For each event (timestep), a representative action is extracted by using the predicted action label with the highest confidence score out of the predicted actions from e_{vt}. The matching theme graph for that action class is then obtained and fed through a Graph Attention Network (GAT) to learn theme node embeddings. For encoding the video-specific graph information, we feed the entire graph into another GAT and extract the node embeddings corresponding to timestep t. We denote the theme and video-specific graph node embeddings as $TG_{emb} \in R^{N \times d}$ and $VG_{emb} \in R^{M \times d}$, where N, M are the number of nodes and d is the embedding dimension. Specifically, we compute:

$$H_{v-CLS} = VisualStream(F_{VC}) \quad (4)$$

$$p_s = softmax(W_h H^t_{v-CLS})^T W_c N_{emb} \quad (5)$$

where H_{v-CLS} is the [CLS] representation from the visual stream at time t, W_h and W_c are learnable, N_{emb} is either TG_{emb} or VG_{emb} and p_s contains probability scores for each node. The top-n nodes yielding the highest probabilities from each

TG_{emb} and VG_{emb} are then selected to be inputs for the node stream. Finally, we concatenate the selected nodes F_N with the predicted captions F_C and feed $F_{NC} = concat(F_N, F_C)$ through another transformer analogous to the one used in the visual stream. We do not add positional embeddings here as the selected nodes have no temporal order.

3) Decoding the Caption Visual and node streams exchange information with cross attention:

$$H_{v-CA}, H_{n-CA} = CrossAttention(H_v, H_n) \tag{6}$$

Here, H_v and H_n are the outputs from the visual and node stream respectively at time t while H_{v-CA} and H_{n-CA} are node attended visual features and visual attended node features respectively. The concatenation of H_{v-CA} and H_{n-CA} is finally fed into a linear (MLP) layer and the next word predicted word is the $argmax$ of the output.

4) Encoding Recurrence To capture temporal dependencies between events from previous timesteps, recent methods for encoding recurrence into transformer models are adopted for our visual and node stream. **A) MART:** memory augmented recurrent transformer (Lei et al., 2020), using multi-head attention to encode the memory state. Given the intermediate hidden state $\bar{H}_t^l$, the memory updated intermediate hidden state H_t^l is computed:

$$H_t^l = MHA(M_{t-1}^l, \bar{H}_t^l, \bar{H}_t^l) \tag{7}$$

where M_{t-1} is the past memory calculated by:

$$C_t^l = tanh(W_{mc}^l M_{t-1}^l + W_{sc}^l S_t^l + b_c^l) \tag{8}$$

$$Z_t^l = sigmoid(W_{mz}^l M_{t-1}^l + W_{sz}^l S_t^l + b_z^l) \tag{9}$$

$$M_t^l = (1 - Z_l^t) \otimes C_t^l + Z_t^l \otimes M_{t-1}^l \tag{10}$$

where $\otimes$ is the Hadamard product, W_{mc}^l, W_{sc}^l, W_{mz}^l, W_{sz}^l are trainable weights, b_c^l and b_z^l are trainable bias, C_t^l is the internal cell state and Z_t^l is the update gate that controls which information to retain from previous memory states. **B) TinT:** proposed by Yamazaki et al. (2023), utilising Hybrid Attention Mechanism (HAM) (Vo et al., 2021) to select information from previous hidden states:

$$M_t^l = [M_{t-1}^l; \bar{H}_t^l] \tag{11}$$

$$Z_t^l = HAM(M_{t-1}^l, \bar{H}_t^l) \tag{12}$$

$$H_t^l = MLP(mean(Att([\bar{H}_t^l; Z_t^l]))) + \bar{H}_t^l \tag{13}$$

Here, ';' denotes concatenation along a new dimension, $mean(Att((\cdot))$ is self-attention applied on the new dimension and reduced by the mean operation, M_t is the memory information at time t ($M_0^l = \emptyset$) and $\bar{H}_t^l$ is defined as above.

4 Evaluation Setup[2]

Data: **1) ActivityNet Captions** (Krishna et al., 2017) consists of 10,009 training and 4,917 validation videos on people performing complex activities. On average, each video contains 3.65 event segments covering 36 seconds. We follow previous works (Lei et al., 2020) and split the original validation set into *ae-val* and *ae-test*. **2) YouCook2** (Zhou et al., 2018) is for dense video procedural captioning in the recipe domain. It contains 1,333 training and 457 validation samples comprised specifically of instructional cooking videos. On average, videos are 5.26 minutes long with 7.7 event segments and each annotation for an event is a language description of the procedure's step covering 1.96 seconds. We report our results on the validation set ('*yc2-val*'). **3)RecipeNLG** (Bień et al., 2020) is for recipe generation, consisting of 2,231,142 cooking recipes and food entities from the recipes extracted using Named Entity Recognition. We use RecipeNLG as a supporting dataset to compute the NPMI scores when constructing the theme graphs for the YouCook2.

Evaluation Metrics: We follow previous VPC works and evaluate with: BLEU-4 (B4) (Papineni et al., 2002), METEOR (M) (Banerjee and Lavie, 2005), CIDEr (C) (Vedantam et al., 2015), and ROUGE-L (R) (Lin, 2004). We also analyse the repetitiveness and diversity of the captions by measuring 2-gram diversity (Div2) (Shetty et al., 2017) and 4-gram repetition (Xiong et al., 2018).

5 Results[3]

5.1 Performance Against SOTA

We compare GEM-VPC with prior SOTA on ActivityNet Captions's *ae-test* split (Table 1) and YouCook2's validation split (Table 2).

Our best model (GEM-VPC w/ TinT decoder) evidently outperforms most of the existing baselines. VLTinT w/ CL and w/o CL is the VLTinT model trained with their novel contrastive loss (in addition to the classic MLE loss) and without their

[2] Implementation details can be found in Appendix G

[3] Appendix J shows qualitative examples of generated captions from our model versus state-of-the-art

ae-test									
Model	**Conference**	**Year**	**Modalities**	**Integration Method**	**B4 ↑**	**M ↑**	**C ↑**	**R ↑**	**R4**
VTrans (Zhou et al., 2018)	CVPR	2018	V+F	Concatenation	9.31	15.54	21.33	28.98	-
Trans-XL (Dai et al., 2019)	ACL	2019	V+F	Concatenation	10.25	14.91	21.71	30.25	8.54
MDVC (Iashin and Rahtu, 2020b) †	CVPR	2020	V+S+A	Concatenation	8.50	14.28	17.57	25.48	-
BMT (Iashin and Rahtu, 2020a) †	BMVC	2020	V+A	CM Attention	8.42	14.08	15.41	25.44	-
MART (Lei et al., 2020)	ACL	2020	V+F	Concatenation	9.78	15.57	22.16	-	5.44
MART-COOT (Ging et al., 2020)	NeurIPS	2020	V+L	Joint CM Space	10.85	15.99	28.19	-	-
Trans-XLRG (Lei et al., 2020)	ACL	2020	V+F	Concatenation	8.85	10.07	14.58	20.34	-
Motion-Aware (Hu et al., 2023)	ICASSP	2023	V+O	CM Attention	11.90	16.54	30.13	-	4.12
Memory Trans. (Song et al., 2021)	CVPR	2021	V+F	Concatenation	11.74	15.64	26.55	-	**2.75**
VLTinT w/ CL (Yamazaki et al., 2023)	AAAI	2023	V+L+O	CM Attention	14.50	17.97	31.13	**36.56**	4.75
VLTinT w/ CL* (Yamazaki et al., 2023)	AAAI	2023	V+L+O	CM Attention	14.32	17.84	31.83	36.51	5.16
VLTinT w/o CL (Yamazaki et al., 2023)	AAAI	2023	V+L+O	CM Attention	13.80	17.72	30.59	36.11	-
GEM-VPC w/ No Recurrence	-	2024	V+G(V+A+C)	CM Attention	12.82	17.4	26.97	33.45	7.28
GEM-VPC w/ MART decoder	-	2024	V+G(V+A+C)	CM Attention	13.47	17.38	30.38	35.8	5.93
GEM-VPC w/ TinT decoder	-	2024	V+G(V+A+C)	CM Attention	**14.54**	**17.99**	**32.62**	36.51	5.17

Table 1: Automatic scores for ActivityNet *ae-test*. In 'Modalities', V=visual, F=optical flow, O=bounding box object visual features, A=audio, S=speech, L=language, G(V+A+C)=graph built with visual, audio modality and commonsense. † indicates results computed by ourselves. ∗ are computed by rerunning the model with our own environment. 'Integration Method'=how to integrate the distinct modalities (see Appendix H for specific meanings).

yc2-val										
Model	**Conference**	**Year**	**Modalities**	**Pretraining**	**Integration Method**	**B4 ↑**	**M ↑**	**C ↑**	**R ↑**	**R4**
VTrans (Zhou et al., 2018)	CVPR	2018	V+F	✗	Concatenation	7.62	15.65	32.26	-	7.83
Trans-XL (Dai et al., 2019)	ACL	2019	V+F	✗	Concatenation	6.56	14.76	26.35	-	6.30
MART (Lei et al., 2020)	ACL	2020	V+F	✗	Concatenation	8.00	15.90	35.74	-	4.39
MART-COOT (Ging et al., 2020)	NeurIPS	2020	V+L	✗	Joint CM Space	9.44	18.17	46.06	-	6.30
Trans-XLRG (Lei et al., 2020)	ACL	2019	V+F	✗	Concatenation	6.63	14.74	25.93	-	6.03
VLTinT (Yamazaki et al., 2023)	AAAI	2023	V+L	✗	CM Attention	9.40	17.94	48.70	34.55	**4.29**
DECEMBERT (Tang et al., 2021)	NAACL	2021	V+L+S	✓	CM Pretraining	**11.92**	**20.01**	58.02	**40.22**	-
MTrans+COOT+MIL-NCE PT (Tang et al., 2021)	NAACL	2021	V+L	✓	Joint CM Space	11.05	19.79	55.57	37.51	-
MART+COOT+MIL-NCE PT(Tang et al., 2021)	NAACL	2021	V+L	✓	Joint CM Space	11.30	19.85	57.24	37.94	-
GEM-VPC w/ No Recurrence	-	2024	V+G(S+A+C)	✗	CM Attention	11.03	**20.01**	**58.49**	36.89	4.64
GEM-VPC w/ MART decoder	-	2024	V+G(S+A+C)	✗	CM Attention	11.01	19.86	54.84	36.81	4.47
GEM-VPC w/ TinT decoder	-	2024	V+G(S+A+C)	✗	CM Attention	11.47	19.72	56.00	37.48	4.91

Table 2: Automatic scores for baselines and GEM-VPC on YouCook2. The 'Modalities' and 'Integration Method' columns are the same as Table 1. Additionally, 'G(S+A+C)' is graph built with speech/audio modality and commonsense, 'CM Pretaining' indicates the use of pretraining objectives like masked language modelling. The 'Pretraining' column indicates whether the model has been pretrained on an external video dataset.

contrastive loss respectively. Specifically, GEM-VPC w/ TinT decoder outperforms VLTinT w/ CL on BLEU-4, METEOR and CIDEr and all metrics when considering the VLTinT w/o CL variant which is optimised using the same MLE loss as our model. For a more accurate comparison, we rerun VLTinT w/ CL (with their optimal parameters) in our own environment and record the results under VLTinT w/ CL*. As shown, GEM-VPC w/ TinT decoder yields higher BLEU-4, METEOR and CIDEr scores than VLTinT w/ CL* with similar ROUGE and R4. While R4 does not outperform some baselines, the lower repetition does not necessarily mean good caption quality as lower repetition can be simply achieved by generating words unrelated to the video content. Hence, a strong model should have a balance of high n-gram metrics and a low repetition score. Examining YouCook2, our model variants achieve higher n-gram scores with relatively low repetition of 4.6-4.9 compared to baselines with no pretraining (first 6 baselines). Even when comparing with the last 3 baselines with pretraining methods and a large separate instructional video dataset (HowTo100M Miech et al. (2019)), we achieve similar scores with our best CIDEr score (58.49) outperforming all baselines.

5.2 Ablation Studies

Different Input Modalities: Our model is examined with different modality settings in Table 3. Using visual features alone (Exp # ①) for both datasets yields the worst performance with the lowest scores across all n-gram metrics. Using nodes only (Exp # ②) can substantially improve the scores, although this produces higher repetition and lower diversity. We also find that the setting using visual features combined with node features results in significant performance improve-

Exp #	V	VG	TG	A	S	B4 ↑	M ↑	C ↑	Div2 ↑	R4 ↓
ActivityNet (*ae-test*)										
①	✓	✗	✗	✗	✗	12.90	16.92	28.27	75.65	6.00
②	✗	✓	✓	✗	✗	10.63	16.51	20.75	74.83	7.66
③	✓	✓	✗	✗	✗	<u>13.27</u>	17.24	28.99	74.29	6.93
④	✓	✗	✓	✗	✗	13.12	17.09	27.97	75.02	7.01
⑤	✓	✓	✓	✗	✗	**13.47**	<u>17.38</u>	**30.38**	<u>75.74</u>	<u>5.93</u>
⑥	✓	✓	✓	✓	✗	13.16	**17.40**	<u>29.88</u>	**76.24**	**5.80**
YouCook2 (*yc2-val*)										
①	✓	✗	✗	✗	✗	7.12	15.25	30.12	<u>70.75</u>	3.66
②	✗	✓	✓	✗	✗	9.91	18.65	44.50	65.38	6.33
③	✓	✓	✗	✗	✗	10.82	19.42	54.73	67.11	4.65
④	✓	✗	✓	✗	✗	8.06	16.35	36.68	69.96	3.72
⑤	✓	✓	✓	✗	✗	<u>11.03</u>	**20.01**	<u>58.49</u>	67.08	4.64
⑥	✓	✓	✓	✓	✗	9.73	18.50	53.33	68.22	4.36
⑦	✓	✗	✗	✗	✓	10.94	19.90	57.45	**71.55**	**1.94**
⑧	✓	✓	✓	✗	✓	**11.56**	<u>19.98</u>	**58.70**	70.46	<u>2.61</u>

Table 3: GEM-VPC performance with different input modalities. ActivityNet (with MART decoder); YouCook2 (with No Recurrence setting). Exp # is the experiment number and V, VG, TG, A, S stand for visual, video-specific graph, theme graph, audio and speech.

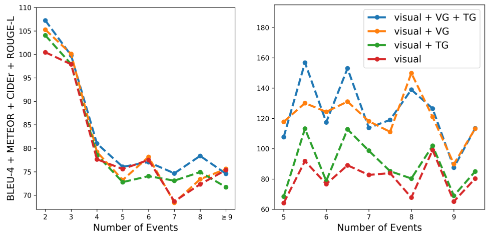


Figure 2: Sum of n-gram metrics on ActivityNet (*ae-val+ae-test*) (left) and YouCook2 (*yc2-val*) (right) across samples with different number of events.

ment across all metrics (Exp # ⑤). Comparing ③ and ④, inputting visual+VG features exhibit higher n-gram metrics than using visual+TG features for both datasets, indicating that the VG provide more useful information representative of the video content. R4 and Div2 scores remain similar for ActivityNet, but that for YouCook2 yields lower repetition/higher diversity. However as previously noted, lower repetition/higher diversity does not mean good caption quality if the n-gram metrics are also low. Overall, we show that incorporating video-specific information and the TG corpus-level information (Exp # ⑤) is superior. We further experimented by adding a separate stream to process the raw audio features. Comparing ⑤ and ⑥ for both datasets, adding audio information slightly improves the repetition/diversity at the cost of lower B4 and CIDEr. This could be due to a misalignment in the audio track and the video's topic e.g. there are cases where users upload background music unrelated to the video contents. Moreover, we examine noisy background audio that could potentially confuse the model. For YouCook2, by examining unprocessed speech features (Exp # ⑦), inputting the visual and speech transcript can produce competitive performance. However, this can be further enhanced by incorporating node information from VG and TG as seen in ⑧ which yields the highest B4 and CIDEr out of all the settings whilst maintaining competitive Div2 and R4.

Different Decoders: We evaluated different methods for encoding recurrence using MART, TinT, and a 'No Recurrence' setting, as in the last three rows of Table 1/2 for ActivityNet/YouCook2. For ActivityNet, the TinT decoder achieved the best results across all metrics, followed by MART, with the No Recurrence setting performing the worst, indicating the importance of a recurrent memory module. Conversely, YouCook2 results showed that the No Recurrence setting yielded the highest METEOR (20.0) and CIDEr (58.5) scores, while TinT improved BLEU-4 and ROUGE-L but had the lowest METEOR and R4. This suggests that encoding recurrence benefits captioning if the current timestep relies on past information. We analysed samples by their total timesteps and plotted the average sum of n-gram metrics for each group in Figure 2. For YouCook2, even without recurrence, decoding captions for samples with more timesteps wasn't necessarily more complex. However, for ActivityNet, scores decreased with more timesteps, highlighting the need for recurrent information. This aligns with the MART paper's findings on ActivityNet, though YouCook2 wasn't tested in their study (Lei et al., 2020).

6 Conclusion

We introduced GEM-VPC, a novel framework for video captioning (VPC) that leverages multimodal information and external knowledge. We construct a commonsense-enhanced video-specific graph for key events and context, and a theme graph from ground-truth captions to represent word relationships. These graphs are processed by separate GNNs, and a node selection module identifies useful nodes for caption decoding. The selected nodes and supporting information (visual, audio, etc.) are fed into a transformer with multiple streams for different modalities, followed by a cross-attention module for inter-stream information exchange. Experiments on benchmark datasets demonstrate that GEM-VPC outperforms existing baselines, generating coherent and visually-grounded captions.

7 Limitations

Our model pipeline requires the use of pretrained models and pre-built methods like action detectors, audio classifier, text parsers and OpenIE models in the data pre-processing stage. As to date, these models and methods are mainly available for the English language but not for low-resource languages that have relatively less data available for training natural language processing systems. Moreover, we emphasise that the metrics used for evaluation are also only capable of judging English-written language. Nevertheless, our framework and pipeline can still be reproduced and as such, for future studies, experiments can be re-run on other languages once models and data become readily available.

References

Anurag Arnab, Chen Sun, and Cordelia Schmid. 2021. Unified graph structured models for video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8117–8126.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Gedas Bertasius, Heng Wang, and Lorenzo Torresani. 2021. Is space-time attention all you need for video understanding? In *ICML*, volume 2, page 4.

Michał Bień, Michał Gilski, Martyna Maciejewska, Wojciech Taisner, Dawid Wisniewski, and Agnieszka Lawrynowicz. 2020. Recipenlg: A cooking recipes dataset for semi-structured text generation. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 22–28.

Shaked Brody, Uri Alon, and Eran Yahav. 2021. How attentive are graph attention networks? In *International Conference on Learning Representations*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. 2017. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 776–780. IEEE.

Simon Ging, Mohammadreza Zolfaghari, Hamed Pirsiavash, and Thomas Brox. 2020. Coot: Cooperative hierarchical transformer for video-text representation learning. *Advances in neural information processing systems*, 33:22605–22618.

Yuan Gong, Yu-An Chung, and James Glass. 2021. Ast: Audio spectrogram transformer. *Interspeech Conference*.

Xin Gu, Guang Chen, Yufei Wang, Libo Zhang, Tiejian Luo, and Longyin Wen. 2023. Text with knowledge graph augmented transformer for video captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18941–18951.

Kaiming He, 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*, pages 770–778.

Jack Hessel, Bo Pang, Zhenhai Zhu, and Radu Soricut. 2019. A case study on combining asr and visual features for generating instructional video captions. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 419–429.

Yimin Hu, Guorui Yu, Yuejie Zhang, Rui Feng, Tao Zhang, Xuequan Lu, and Shang Gao. 2023. Motion-aware video paragraph captioning via exploring object-centered internal knowledge. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Gabriel Huang, Bo Pang, Zhenhai Zhu, Clara Rivera, and Radu Soricut. 2020. Multimodal pretraining for dense video captioning. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 470–490.

Jena D Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. (comet-) atomic 2020: on symbolic and neural commonsense knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6384–6392.

Vladimir Iashin and Esa Rahtu. 2020a. A better use of audio-visual cues: Dense video captioning with bi-modal transformer. In *The 31st British Machine Vision Virtual Conference*. BMVA Press.

Vladimir Iashin and Esa Rahtu. 2020b. Multi-modal dense video captioning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 958–959.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr.

Lei Ji, Rongcheng Tu, Kevin Lin, Lijuan Wang, and Nan Duan. 2022. Multimodal graph neural network for video procedural captioning. *Neurocomputing*, 488:88–96.

Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 2012. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231.

Pin Jiang and Yahong Han. 2020. Reasoning with heterogeneous graph alignment for video question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11109–11116.

Weike Jin, Zhou Zhao, Xiaochun Cao, Jieming Zhu, Xiuqiang He, and Yueting Zhuang. 2021. Adaptive spatio-temporal graph enhanced vision-language representation for video qa. *IEEE Transactions on Image Processing*, 30:5477–5489.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 706–715.

Jie Lei, Liwei Wang, Yelong Shen, Dong Yu, Tamara Berg, and Mohit Bansal. 2020. Mart: Memory-augmented recurrent transformer for coherent video paragraph captioning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR.

Linjie Li, Yen-Chun Chen, Yu Cheng, Zhe Gan, Licheng Yu, and Jingjing Liu. 2020. Hero: Hierarchical encoder for video+ language omni-representation pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Jason Li, Taroon Bharti, and Ming Zhou. 2020. Univl: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*.

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. 2019. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2630–2640.

Kyle Min, Sourya Roy, Subarna Tripathi, Tanaya Guha, and Somdeb Majumdar. 2022. Learning long-term spatial-temporal graphs for active speaker detection. In *European Conference on Computer Vision*, pages 371–387. Springer.

Boxiao Pan, Haoye Cai, De-An Huang, Kuan-Hui Lee, Adrien Gaidon, Ehsan Adeli, and Juan Carlos Niebles. 2020. Spatio-temporal graph for video captioning with knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10870–10879.

Kishore Papineni, 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 of the Association for Computational Linguistics*, pages 311–318.

Jae Sung Park, Marcus Rohrbach, Trevor Darrell, and Anna Rohrbach. 2019. Adversarial inference for multi-sentence video description. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6598–6608.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pages 28492–28518. PMLR.

Tanzila Rahman, Bicheng Xu, and Leonid Sigal. 2019. Watch, listen and tell: Multi-modal weakly supervised dense event captioning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8908–8917.

Paul Hongsuck Seo, Arsha Nagrani, Anurag Arnab, and Cordelia Schmid. 2022. End-to-end generative pre-training for multimodal video captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17959–17968.

Rakshith Shetty, Marcus Rohrbach, Lisa Anne Hendricks, Mario Fritz, and Bernt Schiele. 2017. Speaking the same language: Matching machine to human captions by adversarial training. In *Proceedings of the IEEE international conference on computer vision*, pages 4135–4144.

Botian Shi, Lei Ji, Yaobo Liang, Nan Duan, Peng Chen, Zhendong Niu, and Ming Zhou. 2019. Dense procedure captioning in narrated instructional videos. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 6382–6391.

Yuqing Song, Shizhe Chen, and Qin Jin. 2021. Towards diverse paragraph captioning for untrimmed videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11245–11254.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and I. Dagan. 2018. Supervised open information extraction. In *NAACL-HLT*.

Zineng Tang, Jie Lei, and Mohit Bansal. 2021. Decembert: Learning from noisy instructional videos via dense captions and entropy minimization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2415–2426.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Khoa Vo, 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. *arXiv preprint arXiv:2110.11474*.

Teng Wang, Ruimao Zhang, Zhichao Lu, Feng Zheng, Ran Cheng, and Ping Luo. 2021. End-to-end dense video captioning with parallel decoding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6847–6857.

Yilei Xiong, Bo Dai, and Dahua Lin. 2018. Move forward and tell: A progressive generator of video descriptions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 468–483.

Jun Xu, Ting Yao, Yongdong Zhang, and Tao Mei. 2017. Learning multimodal attention lstm networks for video captioning. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 537–545.

Kashu Yamazaki, Khoa Vo, Quang Sang Truong, Bhiksha Raj, and Ngan Le. 2023. Vltint: Visual-linguistic transformer-in-transformer for coherent video paragraph captioning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 3081–3090.

Antoine Yang, Arsha Nagrani, Paul Hongsuck Seo, Antoine Miech, Jordi Pont-Tuset, Ivan Laptev, Josef Sivic, and Cordelia Schmid. 2023. Vid2seq: Large-scale pretraining of a visual language model for dense video captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10714–10726.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7370–7377.

Runhao Zeng, Wenbing Huang, Mingkui Tan, Yu Rong, Peilin Zhao, Junzhou Huang, and Chuang Gan. 2019. Graph convolutional networks for temporal action localization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7094–7103.

Bowen Zhang, Hexiang Hu, and Fei Sha. 2018. Cross-modal and hierarchical modeling of video and text. In *Proceedings of the european conference on computer vision (ECCV)*, pages 374–390.

Zhu Zhang, Zhou Zhao, Yang Zhao, Qi Wang, Huasheng Liu, and Lianli Gao. 2020. Where does it exist: Spatio-temporal video grounding for multi-form sentences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10668–10677.

Luowei Zhou, Yannis Kalantidis, Xinlei Chen, Jason J Corso, and Marcus Rohrbach. 2019a. Grounded video description. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6578–6587.

Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. 2018. End-to-end dense video captioning with masked transformer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8739–8748.

Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. 2022. Detecting twenty-thousand classes using image-level supervision. In *European Conference on Computer Vision*, pages 350–368. Springer.

Yimin Zhou, Yiwei Sun, and Vasant Honavar. 2019b. Improving image captioning by leveraging knowledge graphs. In *2019 IEEE winter conference on applications of computer vision (WACV)*, pages 283–293. IEEE.

Andrew Zisserman, Joao Carreira, Karen Simonyan, Will Kay, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, et al. 2017. The kinetics human action video dataset.

A Video-Specific Graph Statistics

Total count for the different node types for the ActivityNet and YouCook2 video-specific graphs. On average, the ActivityNet and YouCook2 graphs have 57.04 and 127.83 nodes respectively with the largest graphs containing 259 and 304 nodes respectively.

Node Type	ActivityNet	YouCook2
Action	74,017	43,555
Location	54,802	-
Contextual Phrase	-	13,464
Object	198,905	64,379
Audio	49,496	2,848
Commonsense	472,534	97,147

Table 4: Count of different node types for both the ActivityNet and YouCook2 video-specific graphs.

B Node Type Importance

To examine the importance of different node types in the video-specific graph, for each video sample during inference, we extract the top-10 selected nodes chosen by our node selection module at each event timestep. Then, for each node type, we count the frequency of selected nodes across the timesteps and divide the count by the total number of nodes (of that same type) that are present in the video-specific graph to get the proportion (normalised count). Finally, the proportions computed from each video sample is averaged across the validation set. This number reflects the expected probability that a node of a specific type will be selected to be part of the top-10 nodes. The results for both datasets is shown in the table below. For example, 48.99% in the table means that for ActivityNet video-specific graphs, if the node is an action node, then it has 48.99% chance to be in the top-10 nodes as ranked by our node selection module.

Node Type	ActivityNet (%)	YouCook2 (%)
Action	48.99	51.48
Location	54.30	-
Contextual Phrase	-	55.56
Object	39.79	43.61
Audio	54.24	59.76
Commonsense	30.80	10.74

Table 5: Average proportion of selected nodes for each node type in the video-specific graphs for both ActivityNet and YouCook2.

Examining Table 5, for both ActivityNet and YouCook2, the node types with the highest average selected proportions were the location/contextual phrase, audio and action nodes, indicating that these node types tend to be more vital for video understanding. Action nodes having a high chance of being selected is not surprising, as this node captures information closely aligned with the VPC task where the aim is to generate captions describing the action and events in the video segment. Similarly for ActivityNet, the location nodes may be important as the action/events happening in the video are often closely related to location e.g. videos about water skiing often happen in locations with water. Moreover, for YouCook2, the contextual phrase nodes are most likely significant as they provide similar information to the action nodes. The large percentage of audio nodes selected for both datasets may be unexpected at first as raw video sounds tend to contain noisy background information. However, as mentioned in Section 3.1, we already perform extra post-processing in an attempt to retain only the relevant audio labels. For both datasets, the node type with the second least selection probability are object nodes with on average, 39-44% considered as important. This is however still a relatively large proportion, suggesting that object nodes are significant for VPC. Finally, we observe that a majority of the commonsense nodes were not useful, especially for YouCook2, despite the large count of commonsense nodes in both graphs (see Appendix A). This is perhaps attributed to the fact that Comet-ATOMIC2020 focuses on generating social commonsense such as people's reactions, intents and desires relating to a specific event. However, we find that the ground-truth captions are often limited in detail whereby annotators do not always describe such information but mainly just simply focus on stating what is visually happening in the video. Nevertheless, a relatively large proportion of 30.8% is still selected from the ActivityNet video-specific graphs, suggesting that this social commonsense knowledge can still provide useful contextual cues for videos that are similar in nature to the ones in ActivityNet.

C Number of Nodes Selected

We report the performance of our model when changing the different maximum number of nodes that can be selected from each video-specific graph and each theme graph per timestep. Results for the

ActivityNet and YouCook2 dataset are displayed in Table 6 and Table 7 respectively. For ActivityNet, the best n-gram and repetition scores can be achieved when using 20 nodes (10 nodes selected from each of the video-specific and theme graphs at each timestep) or 40 nodes (20 nodes selected from each graph at each timestep). For YouCook2, we find that the best performance stabilises at around 60-80 total nodes.

	ActivityNet					
# Nodes	**B4 ↑**	**M ↑**	**C ↑**	**R ↑**	**Div2 ↑**	**R4 ↓**
10	13.80	17.40	31.21	35.88	75.22	6.50
20	**13.91**	17.40	**31.45**	**36**	75.19	6.41
40	**13.91**	**17.47**	30.68	35.97	**75.75**	**6.18**
60	13.51	17.38	30.74	35.82	75.21	6.42

Table 6: Performance of our model by setting different maximum number of nodes that can be selected from our node selection module at each timestep on ActivityNet. All results are reported using the model w/ MART decoder with video and node input features.

	YouCook2					
# Nodes	**B4 ↑**	**M ↑**	**C ↑**	**R ↑**	**Div2 ↑**	**R4 ↓**
10	10.39	18.82	52.24	35.52	65.87	5.47
20	10.73	19.27	54.21	35.97	67.63	4.80
40	10.88	19.58	57.38	36.69	66.03	5.40
60	11.03	20.01	**58.49**	**36.89**	67.08	**4.64**
80	**11.23**	**20.04**	57.84	36.78	**67.77**	4.75

Table 7: Performance of our model by setting different maximum number of nodes that can be selected from our node selection module at each timestep on YouCook2. All results are reported using the model w/o Recurrence with video and node input features.

D Performance Across Different Video Categories

To examine how our model performs across different types of videos, we compute the average sum of BLEU-4, METEOR, CIDEr and ROUGE-L across 14 different categories for the ActivityNet validation and testing split. These categories are provided by the user when uploading the video and roughly represent the video's main topic. For this experiment, 3 different types of input modalities are tested: 1) using video visual features only (visual), 2) using visual features combined with node features chosen by the node selection module (visual + nodes), and 3) using visual features combined with node features and audio features (visual + nodes + audio).

Examining Figure 3, when comparing video versus visual + nodes, we find that visual + nodes does better than visual only in all categories except for 'Travel & Events', 'Autos & Vehicles', and 'Science & Technology'. In particular, the largest gap occurs in the 2 latter categories. A reason for this may be due to a lack of action classes related to these categories in which the TimeSformer model is capable of predicting, which subsequently affects the quality of the nodes in the video-specific graph. For instance, there are no specific action classes that are related to 'Science & Technology' in the Kinetics600 dataset in which the TimeSformer model was trained on, while there are only 4 action classes that are related to 'auto maintenance' (*'changing oil'*, *'changing wheel'*, *'checking tires'*, *'pumping gas'*). Furthermore, we observe that adding audio features to the model does not necessarily provide useful context cues for all categories. This is perhaps due to a misalignment between the audio track and video's topic. For example, people will often put a sound track with music even when the video itself is not about 'Music'. However, we do find that audio helps in improving performance for categories related to 'Education', 'Travel & Events', 'Howto & Style', and 'Comedy'.

In summary, visual + nodes performs the best in general, outperforming the other 2 model variants for 7 out of the 14 categories. This aligns with the findings from Section 5.2. Visual + nodes + audio is the second-best with superior performance in 5 categories. This is finally followed by the visual only setting, whereby visual features alone clearly does not provide enough contextual information to generate high quality captions and thus, only benefits 2 out of the 14 categories.

Relation	Description	Example (<**head**><`relation`><**tail**>)
ObjectUse	describes everyday affordances or uses of objects	put into pan `ObjectUse` frying
MadeUpOf	describes a part, portion or makeup of an entity	making cake `MadeUpOf` eggs
HasProperty	describes entities' general characteristics	boiling water `HasProperty` heat
CapableOf	describe abilities and capabilities of everyday living entities	cut cake `CapableOf` celebrate birthday
isAfter	events that can follow an event	mop the floor `isAfter` sweep the floor
HasSubEvent	provides the internal structure of an event	boil the dumplings `HasSubEvent` boils water
isBefore	events that can precede an event	opens a gift `isBefore` rips wrapping paper
xNeed	describes a precondition for an agent to achieve the event	give a gift `xNeed` buys the presents
xAttr	describes personas or attributes perceived by others given an event	decorates Christmas tree `xAttr` festive
xEffect/oEffect	actions that happen to an agent that may occur after the event	gives a present `xEffect` gets thanked
xReact/oReact	emotional reactions of participants in an event	gives a present `xReact` feels happy
xWant/oWant	postcondition desires after an event	gives a present `xWant` wants to hug
xIntent	defines the likely intent of an agent	pour sauce on food `xIntent` add flavour

Table 8: Relations in Comet-ATOMIC2020 used to generate the commonsense nodes for the video-specific graph and their corresponding descriptions. The 'head' indicates the input phrase that is fed into Comet-ATOMIC2020 and the 'tail' is the possible generated commonsense.

Figure 3: Sum of BLEU-4, METEOR, CIDEr and ROUGE-L scores for the ActivityNet predicted captions across the different video categories using 3 different input modalities (visual only, visual + nodes, visual + nodes + audio). The scores are obtained from the combined validation (*ae-val*) and testing set (*ae-test*).

E Relation Description

The relation tokens used to extract knowledge from the Comet-ATOMIC2020 neural knowledge model for the commonsense nodes in the video-specific graphs and their corresponding descriptions are detailed in Table 8. The descriptions are taken from the official Comet-ATOMIC2020 paper (Hwang et al., 2021). For the ActivityNet graphs, all relations below were used except for `isAfter`, `isBefore`, `MadeUpOf`, `ObjectUse` and `HasProperty`. Although `isAfter` and `isBefore` relations may be useful, we find that the commonsense generated using these relations for the ActivityNet data tends to produce similar results to `xNeed` and `xEffect/oEffect` and so we disregard these relations to help reduce the number of commonsense nodes in the graphs. `MadeUpOf`, `ObjectUse` and `HasProperty` are further ignored as information about properties, compositions or characteristics of entities are not closely aligned with the content in the ActivityNet captions. For the YouCook2 graphs, all relations below were used except for `xReact/oReact`, `xAttr` and `xWant/oWant` as we believe information about an event's attributes and individual's subjective reactions/desires may not be useful for captioning instructional cooking videos.

F Theme Graph Example

The image below shows an example of what a snippet from the theme graph corresponding to the action class *carving pumpkins*' would look like. Nodes represent tagged nouns, verbs, and adverbs from the ground-truth training annotations. All edges in the graph are undirected and weighted by normalised point mutual information scores.

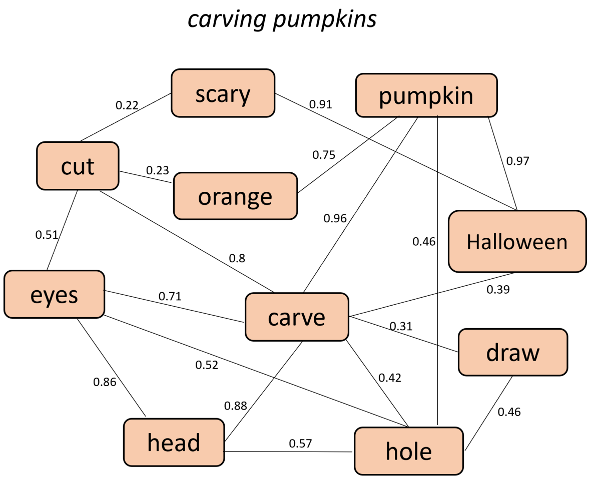


Figure 4: Visual example of a sub-graph for the theme graph corresponding to the ActivityNet action class *carving pumpkins*.

G Implementation Details

Graph Construction: The TimeSformer (Bertasius et al., 2021) pretrained on the Kinetics600 dataset (Zisserman et al., 2017) was used as the action classification model for constructing the action nodes for the VF-method. The model is capable of predicting 600 unique action classes. We leveraged the Audio Spectrogram Transformer (Gong et al., 2021) pretrained on AudioSet (Gemmeke et al., 2017) (capable of predicting 632 audio event classes) as the audio classification model to create the audio nodes for the VF and ASR-method. Commonsense nodes are generated by Comet-ATOMIC2020 (Hwang et al., 2021) using the *'comet_atomic2020_bart'* implementation. Object and location nodes for the VF-method are generated by the BLIP-VQA base model as proposed in (Li et al., 2022), with the object nodes further expanded using Detic's (Zhou et al., 2022) object detection model. ASR from the YouCook2 videos was extracted using OpenAI's Whisper (Radford et al., 2023) while we used AllenNLP's OpenIE model (Stanovsky et al., 2018) for creating the action nodes in the ASR-method. All part-of-speech tagging is done with the NLTK toolkit.

For each set of commonsense knowledge generated by its corresponding action node, we filter out any similar generated commonsense to avoid adding duplicate commonsense into the video-specific graph at the same timestep. Specifically, we removed any similar commonsense if its Levenshtein Distance ratio with another commonsense is greater than 0.70. As mentioned in Section 3.1, we also did not add the commonsense into the graph if the action class used to generate that commonsense had a confidence score of less than 0.5 so as to avoid incorporating irrelevant external knowledge. The threshold for filtering out any noisy object and audio labels was 0.25 and 0.3 respectively while the threshold to determine whether an action node contained *'no action'* was 0.35. For creating the theme graphs in the case when the ASR-method is used, k-means clustering with $k = 300$ and 10 repetitions was used to create the action classes. The theme graphs contain the top-100 most occurring words within that action class/theme.

Model Training: The 2048D visual features for the ActivityNet were extracted using a 3D-CNN backbone (Ji et al., 2012). For YouCook2, we used 2048D ResNet-200 (He et al., 2016) visual features concatenated with 1024D optical flow features from BNInception (Ioffe and Szegedy, 2015). The node/edge linguistic features for the video-specific and theme graphs are represented using CLIP textual embeddings (Radford et al., 2021).

We train the modules in an end-to-end fashion with teacher forcing to optimise the Kullback–Leibler divergence loss with the best model using a label smoothing of 0.3. The word embedding matrix of the models is initialised with GloVe embeddings of dimension 300 (Pennington et al., 2014). Inputs into each transformer stream are added with fixed positional embeddings (only for the visual stream) and learnt token type embeddings. The token type embedding matrix was size 10 to incorporate for different modality types such as visual, audio or type of node e.g. location, commonsense etc. We use 2 hidden transformer layers with 12 attention heads where the hidden and intermediate size was 768. For the theme graph encoder, 2 GATv2Conv layers (Brody et al., 2021) were used while the video-specific graph encoder used 1 GATv2Conv layer with all layers using 4 attention heads. Adam optimizer was used to train our model with an initial learning rate of 1e-4, $\beta_1 = 0.9$ and $\beta_2 = 0.999$, L_2 weight decay of 0.01, learning rate warmup over the first 5 epochs and batch size of 2. Early stopping was applied after no improvement was seen in the validation CIDEr score in 3 consecutive epochs. For decoding the caption at inference, nucleus sampling with 0.6 top-p and 0.5 temperature was used.

			ae-val						
Model	**Conference**	**Year**	**Modalities**	**Integration Method**	**B4** ↑	**M** ↑	**C** ↑	**R** ↑	**R4**
VTrans (Zhou et al., 2018)	CVPR	2018	V+F	Concatenation	9.75	15.64	22.16	28.9	7.79
HSE (Zhang et al., 2018)	ECCV	2018	V	-	9.84	13.78	18.78	-	
AdvInf (Park et al., 2019)	CVPR	2019	V+F+O	Concatenation	10.04	15.93	27.27	-	5.76
GVD (Zhou et al., 2019a)	CVPR	2019	V+F+O	CM Attention	11.04	15.71	22.95	-	8.76
GVDsup (Zhou et al., 2019a)	CVPR	2019	V+F+O	CM Attention	11.30	16.41	22.94	-	7.04
Trans-XL (Dai et al., 2019)	ACL	2019	V+F	Concatenation	10.39	15.09	21.67	30.18	8.79
Trans-XLRG (Lei et al., 2020)	ACL	2020	V+F	Concatenation	10.17	14.77	20.40	-	
MDVC (Iashin and Rahtu, 2020b) †	CVPR	2020	V+S+A	Concatenation	9.12	14.69	17.57	25.85	-
BMT (Iashin and Rahtu, 2020a) †	BMVC	2020	V+A	CM Attention	9.00	14.49	16.46	26.11	-
MART (Lei et al., 2020)	ACL	2020	V+F	Concatenation	10.33	15.68	23.42	-	5.18
PDVC (Wang et al., 2021)	ICCV	2021	V+F	Concatenation	11.8	15.93	27.27	-	-
Motion-Aware (Hu et al., 2023)	ICASSP	2023	V+O	CM Attention	12.07	16.81	29.32	-	**4.28**
Text-KG (Gu et al., 2023)	CVPR	2023	V+O+S+G(S+C)	CM Attention	11.30	16.50	26.60	-	6.30
VLTinT w/ CL (Yamazaki et al., 2023)	AAAI	2023	V+L+O	CM Attention	**14.93**	**18.16**	**33.07**	**36.86**	<u>4.87</u>
VLTinT w/ CL* (Yamazaki et al., 2023)	AAAI	2023	V+L+O	CM Attention	<u>14.89</u>	<u>18.09</u>	**33.07**	<u>36.76</u>	5.11
GEM-VPC w/ No Recurrence	-	2024	V+G(V+A+C)	CM Attention	13.16	17.56	27.50	33.85	7.86
GEM-VPC w/ MART decoder	-	2024	V+G(V+A+C)	CM Attention	13.91	17.47	30.68	35.97	6.18
GEM-VPC w/ TinT decoder	-	2024	V+G(V+A+C)	CM Attention	14.73	18.02	<u>32.93</u>	36.71	5.41

Table 9: n-gram metrics and repetition scores of baselines and our model (GEM-VPC) for the ActivityNet *ae-val* split. In the 'Modalities' column, the abbreviations are defined as follows: V=visual, F=optical flow, O=bounding box object visual features, A=audio, S=speech, L=language, G(V+A+C)=graph built with visual, audio modality and commonsense, G(S+C)=graph build with speech modality and commonsense. † indicates results computed by ourselves using VPC evaluation mode.∗ indicates results computed from the model that was reran with our own environment. The 'Integration Method' column indicates the model's main approach for integrating the distinct modalities. 'Concatenation' refers to a simple concatenation of different modality vectors which are then fed into a single stream, 'CM Attention' refers to cross-modal attention employed between modules processing different modality inputs, and 'Joint CM Space' indicates that the model attempts to learn a common space for different modalities.

H ActivityNet Validation Set Quantitative Results

Table 9 shows the n-gram metrics and repetition scores of baselines and GEM-VPC for the ActivityNet *ae-val* split. In the 'Modalities' column, the abbreviations are defined as follows: V=visual, F=optical flow, O=bounding box object visual features, A=audio, S=speech, L=language, G(V+A+C)=graph built with visual, audio modality and commonsense, G(S+C)=graph build with speech modality and commonsense. † indicates results computed by ourselves using VPC evaluation mode.∗ indicates results computed from the model that was reran with the same environment as this research. The 'Integration Method' column indicates the model's main approach for integrating the distinct modalities. 'Concatenation' refers to a simple concatenation of different modality vectors which are then fed into a single stream, 'CM Attention' refers to cross-modal attention employed between modules processing different modality inputs, and 'Joint CM Space' indicates that the model attempts to learn a common space for different modalities.

Our best model (GEM-VPC w/ TinT decoder) achieves comparable performance with the strongest baselines (VLTinT w/ CL and VLTinT w/ CL*). Note that while we underperform slightly on the validation set, we outperform VLTinT in a majority of the metrics when evaluating on the testing set (see Table 1 of the main paper).

Please note that the appendix continues on the next page.

I Video-Specific Graph Visual Examples

Visual depiction of what the video-specific graphs would look like using the VF and ASR-method for an example ActivityNet and YouCook2 video. Blue nodes represent the action nodes, red nodes are the location/contextual phrase nodes, green nodes are object nodes, purple nodes are audio nodes and orange nodes are the commonsense nodes. Note that due to size of the graphs, not all nodes are presented and graphs would be larger in reality. Sentences under the video frames are the matching ground-truth captions.

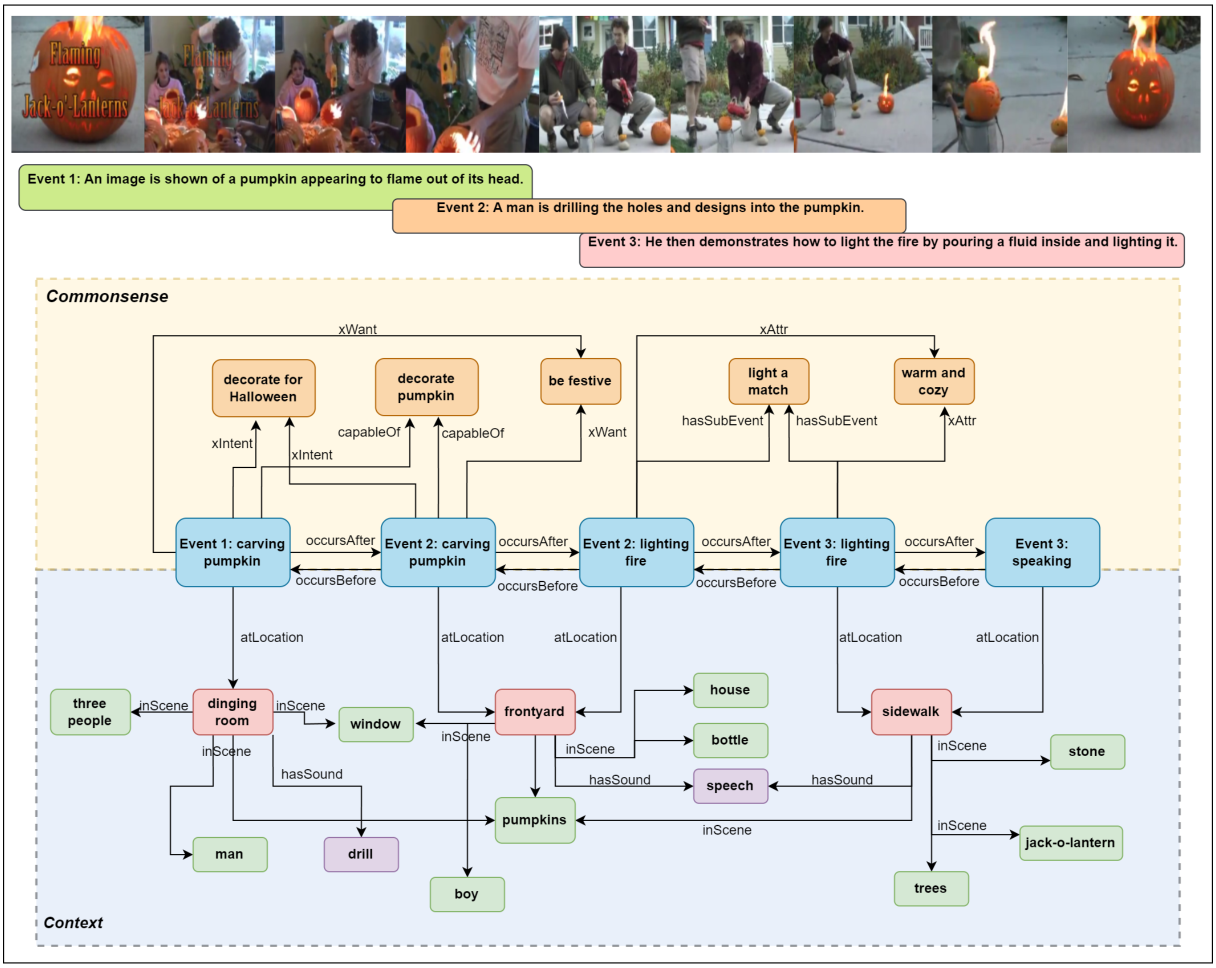


Figure 5: Video-specific graph for an example video in the ActivityNet dataset using the VF-method for the first 3 timesteps.

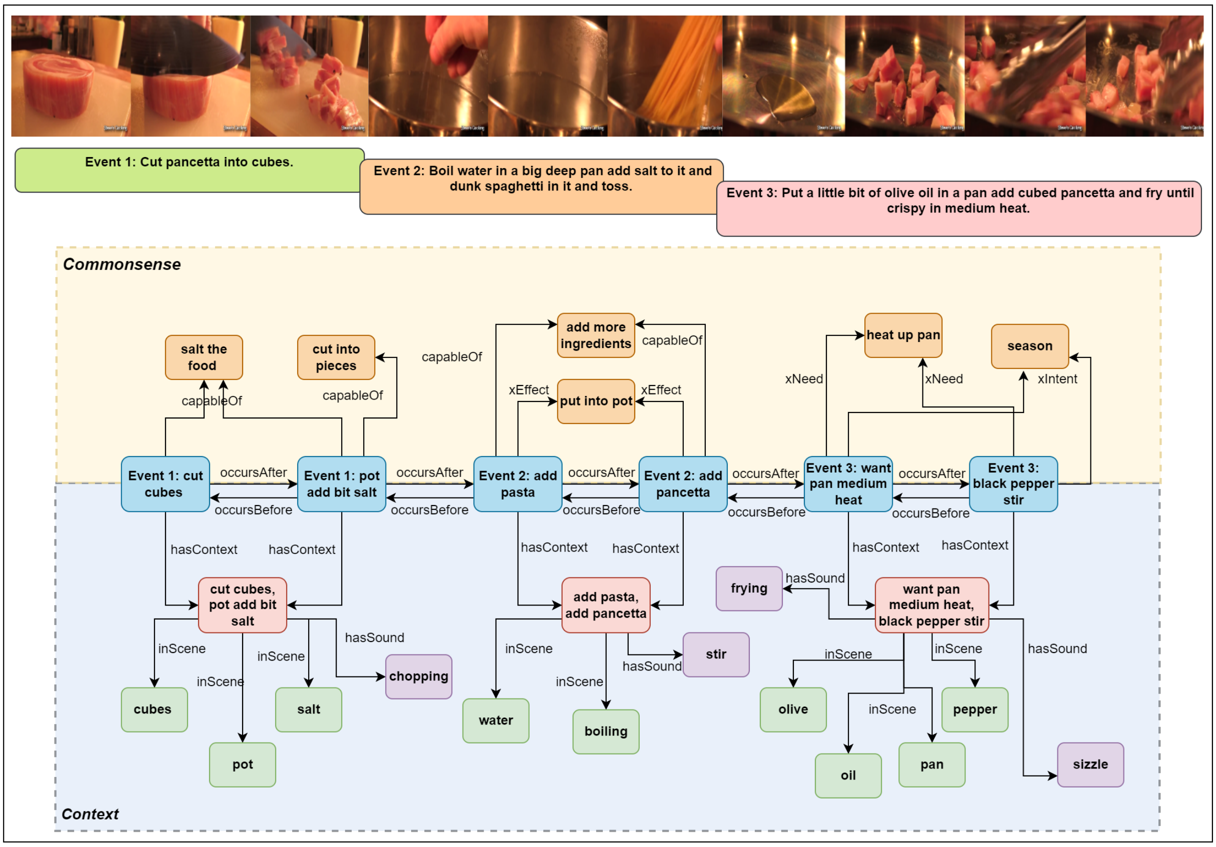


Figure 6: Video-specific graph for an example video in the YouCook2 dataset using the ASR-method for the first 3 timesteps.

J Qualitative Examples (Ours vs SOTA)

Qualitative Examples for the start-of-the art methods versus ours (GEM-VPC) are shown on the next page. The first example is from YouCook2 while the last 2 are from ActivityNet. Blue words in the machine-generated captions are visually grounding to the video, while red words represent irrelevant words that are 'hallucinated' by the model.

We collect the top-10 selected nodes by confidence score at each timestep during inference and display the selected nodes and their types in the table after each example. Highlighted blue words in the table indicate information related to the theme of the video. Evidently, the commonsense-enhanced video graph and theme graph assists our model in producing concepts and phrases relevant to the video segment. For instance in the second example, our model mentions relevant phrases like '*smiling to the camera*' and '*putting ornaments on the tree*' which were perhaps derived from selected nodes such as '*happy*', '*decoration*' and '*jingle*'. Conversely, other baseline models will sometimes mention concepts irrelevant to the video such as in the last instance where Text-KG mistakens a motorcycle for a '*car*'. Likewise, BMT incorrectly outputs '*brushing his face*'in contrast to our model which is capable of recognising the action of a person shaving his beard.

Ground Truth: Cut the avocado and place in a bowl. Cut a lime squeeze on the avocado and stir. Add 1 tbsp of olive oil to it and season it with a pinch of salt and stir. Lay bacon slices over the coin rack. Cut firm artisan bread into slices

MART: Cut a **chicken** into small pieces. **Peel** and **cut** the **ginger**. Add **soy sauce** and **sesame oil** to the **pan**. **Cut** the **tomatoes** into small pieces. **Cut** a **strip** into small pieces.

Ours: **Slice** the **avocado** and **add** to the **salad**. **Squeeze** some **lemon juice** on the **salad**. **Heat** some **olive oil** in a **pan**. Take the **bacon pieces** and put them on a baking **tray** and **add** some **olive oil** and top it with **bacon**. **Slice** the **bread**.

Action	Object	Commonsense	Theme	Audio
use fork, give flavor, squeeze avocado, preserve color, release juice, add little bit	lime, fork, roll, half, time, pinch, bacon, salt, olive	hungry, make sandwich, satisfied, put in blender, buy avocados, put in glass, put in salad dressing, make juice, put in pan, buy bacon, add to batter	add, vinegar, mix, quinoa, chili, coriander, water, drain, flake, dough, oil, salt, butter, pan, bowl, heat, pepper, ingredient, dressing	chopping

Ground Truth: People are putting together a Christmas tree. They put the lights onto the Christmas tree. They pour mugs of eggnog. They finish decorating the tree with bulbs.

MART: A **woman** is sitting in a **chair** in front of a **Christmas tree**. She puts the **Christmas tree** in a chair. She is **decorating** the **tree**. She puts a **Christmas tree** in the air.

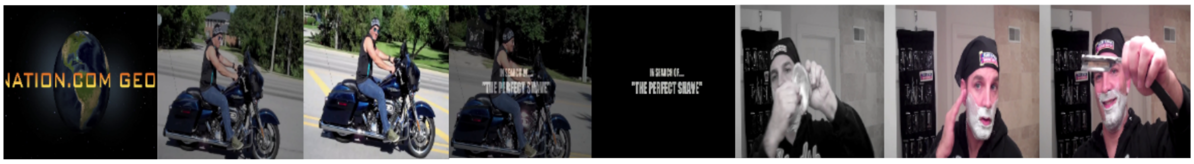
PDVC: A **woman** is seen standing in front of a **Christmas tree**. The **woman** is putting a **Christmas tree** on the **tree**. The **woman** then begins to the **tree** and the **woman** is putting the **tree** on the **tree**. The girl then takes a small **bowl** of the **tree** and the **woman** is shown.

Text-KG: Two **women** are seen sitting in a **chair** and speaking to one another. The **women** then put a **Christmas tree** into a **bowl** and end by presenting it to one another. They continue to play and end by presenting the **tree** and showing off the finished product. They put the **tree** back on the **tree** and show off the finished product .

BMT: Two **girls** are seen speaking to the camera while moving around the **tree** and leads into a **tree**. They put the **cake** on the **tree** and put it on the **tree**. They **put lights** on the **tree.** They put the **lights** on the **tree** and put them on the **tree.**

Ours: Two people are seen **standing** before a **Christmas tree** speaking to one another and leads into several **people decorating**. The people then put a bow on the **Christmas tree** and end by presenting it to the camera. The people continue to **put lights on the tree** and end by **smiling to the camera**. More people are seen **putting ornaments on the tree** and **speaking to one another**.

Action	Object	Commonsense	Theme	Audio	Location
decorating the christmas tree	three women, christmas tree, two women, sweater, christmas trees, television set	put lights on tree, to enjoy the holiday, get a gift, decorate tree, loved, buy decorations, happy	family, kiss, dad, dog, stair, see, playing, hug, bulb, decoration, continues, frame, ladder, branch	piano, music, jingle, twinkle	office



Ground Truth: A man is riding a motorcycle. He is putting shaving cream onto his face. He picks up a razor and begins shaving his beard.

MART: A **man** is sitting in a **chair** as he talks to the camera. A **woman** is sitting on a **camel**. He is then shown getting a **tattoo** on his **face** and then getting it off.

PDVC: A **man** is seen speaking to the camera and leads into a person speaking to the camera. A **man** is seen **riding** around a **bike** and leads into a **man** speaking to the camera. The **man** is seen speaking to the camera and leads into a **man** speaking to the camera.

Text-KG: A **man** is **driving** a **car**. A **man** is **riding** a **bike** in a parking lot. He is using a **brush to blow dry** his **face.**

BMT: We see a title screen, A **man** is seen **sitting** on a **bike** holding a **bike** and **riding** on a **bike**. A **man** is seen speaking to the camera and leads into him **brushing** his **face** off the sides of the mirror and speaking to the camera.

Ours: A close up of a **bike** is shown. A **man** is **riding a bike** down a **track**. He then uses a **shaver** to **shave his beard**.

Action	Object	Commonsense	Theme	Audio	Location
no action	trees, earth, stop sign, motorcycle, door, letters	go for a ride, active, learn to ride, athletic, go on a ride, exhilarating, have fun, get a razor, trimming beard, get a haircut, skilled, remove beard, to have a clean beard	time, jump, back, make, place, tree, game, head, bar, brush, talk, talking, camera, speaking, people, shown, dirt, bike, riding, see, tube, beard, face	music	bathroom

Figure 7: Qualitative Examples for the state-of-the-art methods versus ours.

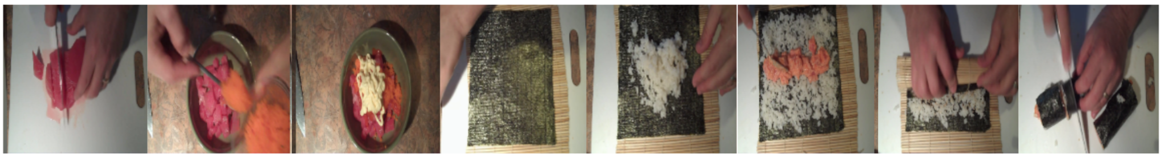
K More Qualitative Examples

Ground Truth: People drive bumper cars around a bumper car rink in an amusement park. A yellow bumper car hits another bumper car. A red bumper car hits the outer wall and stalls out.

Ours: A group of children are riding around in bumper cars. They bump into each other. The children get stuck, then they continue to ride.

Ground Truth: A group of boys are in a basketball court. The coach is showing how to play, and the boys are performing layups.

Ours: A group of boys are seen playing basketball while a coach speaks to the camera. More shots are shown of the boys playing basketball and shooting the ball into the hoop.



Ground Truth: Chop some tuna meat into small pieces. Mix the tuna pieces and some fish eggs in a bowl. Add hot sauce and mayo and then mix them together. Spread some cooked rice on seaweed sushi wrapper. Add the tuna mix on top and season with some sesame seeds. Roll it up and press firmly. Cut the roll into small rolls

Ours: Slice the tuna into pieces. Add the fish to the bowl. Add the sauce to the bowl and mix. Place the sushi rice on the seaweed. Place the tuna in the center of the sheet. Roll the sheet up. Cut the roll into pieces.

Ground Truth: Cook the macaroni in boiling water. Grease the pan with butter and fill with macaroni. Add cream milk and cheese to a bowl. Add seasoning. Mix in the eggs. Pour the sauce onto the pasta. Sprinkle cheese onto the macaroni. Bake the macaroni in the oven.

Ours: Boil the macaroni in water. Spread butter on the pan. Add some milk and cheese and mash the mixture. Season the eggs with salt and pepper. Pour the eggs into the bowl and add some lemon juice. Pour the sauce on top of the pasta. Sprinkle cheese and cheese on top. Bake the pizza in an oven.

Figure 8: Qualitative examples of generated captions using our model. Top 2 examples are from ActivityNet and bottom 2 examples are from YouCook2. Blue words in the machine-generated captions are visually grounding to the video, while red words represent irrelevant words that are 'hallucinated' by the model.