International Journal of Computer Vision (2025) 133:8617–8641
https://doi.org/10.1007/s11263-025-02577-2

Language-guided Recursive Spatiotemporal Graph Modeling for Video Summarization

Jungin Park[1] · **Jiyoung Lee**[2] · **Kwanghoon Sohn**[1,3]

Received: 6 March 2024 / Accepted: 25 August 2025 / Published online: 4 September 2025
© The Author(s) 2025

Abstract

Video summarization aims to select keyframes that are visually diverse and can represent the whole story of a given video. Previous approaches have focused on global interlinkability between frames in a video by temporal modeling. However, fine-grained visual entities, such as objects, are also highly related to the main content of the video. Moreover, language-guided video summarization, which has recently been studied, requires a comprehensive linguistic understanding of complex real-world videos. To consider how all the objects are semantically related to each other, this paper regards video summarization as a language-guided spatiotemporal graph modeling problem. We present recursive spatiotemporal graph networks, called *VideoGraph*, which formulate the objects and frames as nodes of the spatial and temporal graphs, respectively. The nodes in each graph are connected and aggregated with graph edges, representing the semantic relationships between the nodes. To prevent the edges from being configured with visual similarity, we incorporate language queries derived from the video into the graph node representations, enabling them to contain semantic knowledge. In addition, we adopt a recursive strategy to refine initial graphs and correctly classify each frame node as a keyframe. In our experiments, VideoGraph achieves state-of-the-art performance on several benchmarks for generic and query-focused video summarization in both supervised and unsupervised manners. The code is available at https://github.com/park-jungin/videograph.

Keywords Language-guided video summarization · Recursive graph refinement · Spatiotemporal graph convolutional networks

1 Introduction

Over the years, online video platforms have grown enormously with the development of portable devices and the internet. Although this has made a considerable amount of data widely available, video data still needs to be accessed. Moreover, the lengths of uploaded videos continue to increase, and it is impractical to watch the videos in full to obtain their valuable information. In response to these issues, many recent computer vision researchers have investigated techniques to browse enormous video data efficiently (Zhang et al., 2019; Wu et al., 2019). In particular, video summarization has become a prominent research topic to automatically create summary videos that describe the original video contents, helping to handle the overwhelming amount of video data (Zhang et al., 2016a; Mahasseni et al., 2017; Zhang et al., 2016b).

Variants of recurrent neural networks (RNNs) have been introduced (Zhang et al., 2016b; Mahasseni et al., 2017; He et al., 2019; Yuan et al., 2019; Zhao et al., 2020; Ji et al., 2021) to treat video summarization as a sequence labeling problem, achieving substantial accuracy due to their robustness in dealing with sequential data. Although RNNs effectively capture long-range dependencies among frames, the signals are progressively propagated through the frames by repeatedly applying recurrent operations. Thus, those approaches need to be carefully considered to optimize the models

Communicated by Jingdong Wang.

✉ Jiyoung Lee
lee.jiyoung@ewha.ac.kr

✉ Kwanghoon Sohn
khsohn@yonsei.ac.kr

Jungin Park
newrun@yonsei.ac.kr

[1] Yonsei University, Seoul, Korea

[2] Ewha Womans University, Seoul, Korea

[3] Korea Institute of Science and Technology (KIST), Seoul, Korea



Fig. 1 Keyframes (red boxes) and background frames (black boxes) from the QFVS benchmark (Sharghi et al., 2017). To distinguish visually similar frames, we leverage fine-grained visual entities in each frame, enabling a more accurate and robust video summarization

due to this inherent specificity (He et al., 2016). They also can cause a multihop dependency modeling problem where delivering messages back and forth between distant positions is difficult (Wang et al., 2018). Convolutional neural networks (CNNs) based approaches (Rochan et al., 2018; Rochan & Wang, 2019; Park et al., 2019a; Zhu et al., 2020), which have been proposed to tackle these limitations, generally outperform recurrent models. However, CNN-based approaches inevitably failed to address semantic relevance between keyframes with varying time distances properly. Most recently, transformer (Vaswani et al., 2017) has been introduced for video summarization (Li et al., 2022). While they achieved promising results, thanks to the transformer's strong representation capabilities, they require a huge amount of computational cost and a lot of inference time to produce summary videos.

Meanwhile, graphical models have been used specifically to model semantic interactions (Krähenbühl & Koltun, 2011; Chen et al., 2018; Kipf & Welling, 2017; Zeng et al., 2019; Jain et al., 2016; Park et al., 2019b), and they have been revisited with graph convolutional networks (GCNs) (Kipf & Welling, 2017), which generalize convolutions from grid-like data to non-grid structures. Thus, GCNs have received increasing interest in various computer vision applications, such as object detection (Yuan et al., 2017; Nassar et al., 2020; Shi & Rajkumar, 2020), video classification (Wang & Gupta, 2018; Shi et al., 2019a), video object tracking (Gao et al., 2019; Li et al., 2020), and action localization (Zeng et al., 2019, 2021). These works modeled knowledge graphs based on relationships between different entities, such as images, objects, and temporal proposals. Although GCNs have shown promising results, they generally use a fixed graph directly obtained from affinities between feature representations (Jiang et al., 2019), where nodes are strongly connected when they have similar entities. Video scene graph generation methods (Teng et al., 2021; Wang et al., 2022; Gao et al., 2022) have more focused on contextual information for the relationships between objects appearing in the video. However, they assumed that the objects, whose relationships are to be inferred, should directly interact with each other. Therefore, it is difficult to employ those methods for video summarization directly, which requires identifying comprehensive connections over the whole story to infer the summary video.

Language-guided video summarization (Narasimhan et al., 2021; Park et al., 2022; Li et al., 2023a) has been introduced to produce a generic or query-focused summary. Language captions generated from the video captioning model (Iashin & Rahtu, 2020) or user-defined language queries are leveraged into video summarization using transformer (Vaswani et al., 2017) and provide customized summaries corresponding to given language queries. Despite their flexibility, previous methods (Narasimhan et al., 2021; Park et al., 2022; Li et al., 2023a) used global visual representations (i.e., frame-level features), leading to inconsistency between visual and language information. The language query consists of compositional semantics (Liu et al., 2020; Park et al., 2021), while global visual representations contain not only relevant context, but also irrelevant noise (Zhu et al., 2021). In practice, as shown in Fig. 1, keyframes ((a), (c), (e)) and background frames ((b), (d), (f)) are often visually similar in global representations but can be distinguished based on fine-grained object details, such as the car handle, the man holding the shoe, and the object held by the hands. This inconsistency between the two modality representations makes it difficult to fully utilize the compositional semantics of language queries.

In addition, the types of alternative language queries remain underexplored, requiring additional parameters and time for caption generation.

In this paper, we formulate a semantic relationship between frames for video summarization as *a unified relationship linked over time based on the entire video story* and present a novel framework, termed VideoGraph. Our VideoGraph incorporates the advantages of deep spatiotemporal graph modeling and a language-guided framework into video summarization by leveraging object-object and frame-frame interactions, as shown in Fig. 2. Unlike conventional methods (Zhang et al., 2016b; Rochan et al., 2018; Park et al., 2020; Narasimhan et al., 2021; Li et al., 2022) solely using global frame feature representations (e.g. GoogleNet (Szegedy et al., 2015) features), VideoGraph works with object feature representations. We formulate a frame as a set of objects within it and construct a spatial graph,

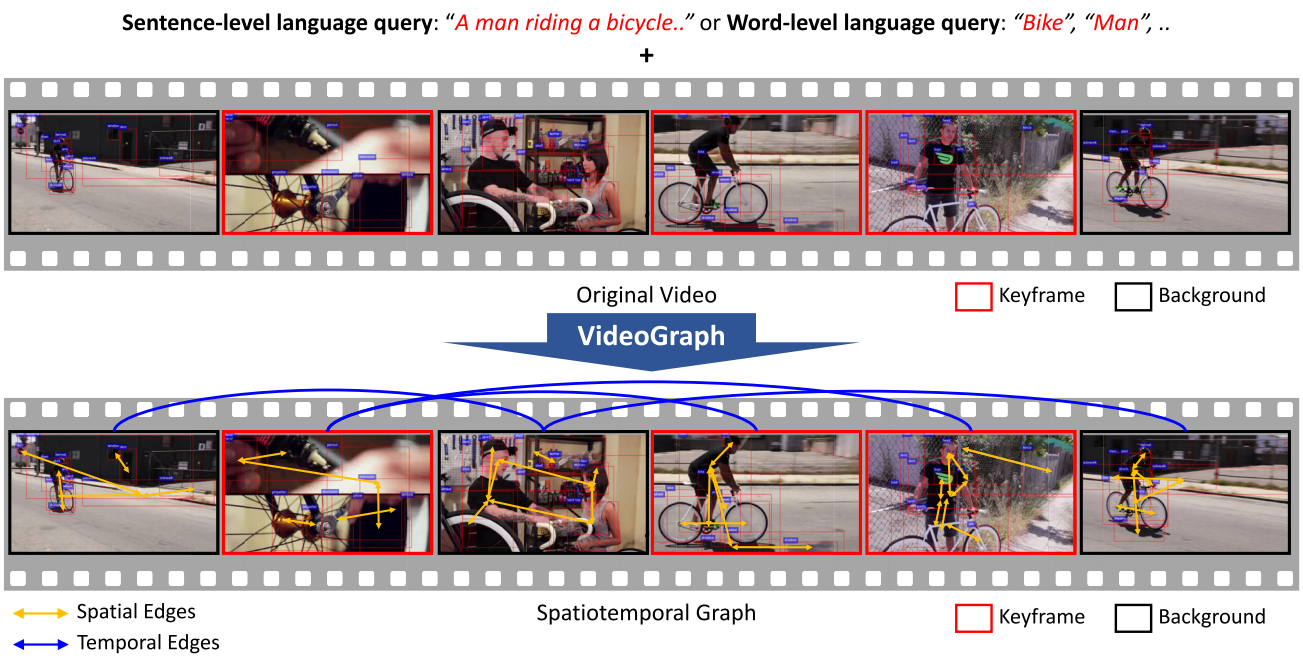


Fig. 2 Proposed VideoGraph approach, regarding summarization as a language-guided spatiotemporal graph modeling problem. Given video and language queries, richer video summaries are obtained by constructing the language-guided spatiotemporal graphs that represent object-object relationships in each frame and frame-frame relationships in the video

where objects are regarded as nodes. The object node representations in each frame are connected based on semantic similarity, where the node representations are based on spatially localized features from the object detector (Ren et al., 2015) with language features. Then the nodes are aggregated along spatial edges through a graph convolution operation. The temporal relationships are estimated across different frames by constructing a temporal graph where the averaged object nodes are regarded as nodes. The aggregated object representations are propagated frame-to-frame by passing the message through temporal edges, which represent semantic relationships between pairs of frames in the temporal graph.

In addition, we propose a recursive graph refinement learning scheme that recursively refines the graphs by estimating the residuals of the adjacency matrices to model the semantic similarity (i.e., keyframe or not). Unlike the previous works (Gao et al., 2019; Zeng et al., 2019), whose constructed relation graphs depend heavily on visual similarity, VideoGraph refines the graphs and enhances feature representations according to the updated graphs (Park et al., 2020), as shown in Fig. 3. At each iteration, the temporal graph is used to estimate the residual spatial adjacency matrix, and the refined spatial graph is used to estimate the residual temporal adjacency matrix to refine the temporal graph. With the recursive graph refinement on the language-guided node representations, a frame included in the summary is connected to other keyframe nodes with high affinity weights in the final temporal graph, such that semantic connections between frames can be modeled through the spatial and temporal graphs. We extensively examine the proposed method through ablation studies and comparisons with state-of-the-art methods in both supervised and unsupervised manners on the SumMe (Gygli et al., 2014), TVSum (Song et al., 2015), and QFVS (Sharghi et al., 2017) benchmarks.

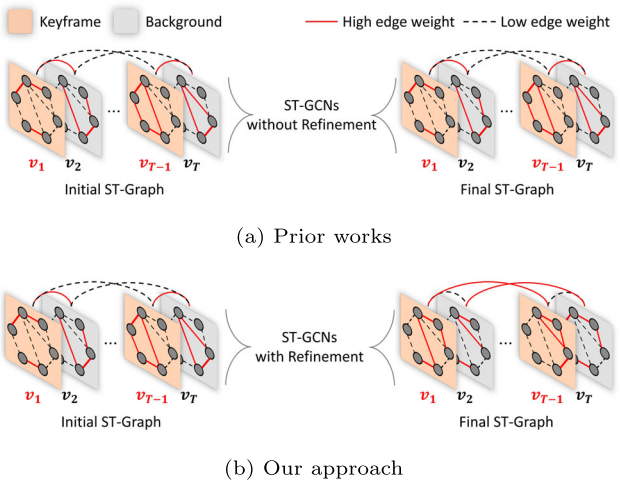


Fig. 3 Comparing the proposed VideoGraph with previous ST-GCNs approaches. Circles denote object node representations, red and black squares represent keyframes and backgrounds, respectively: (a) ST-GCNs based methods without graph refinement and (b) VideoGraph refines the initial graph by recursively estimating the graph, hence leveraging semantic relationships

A preliminary version of this work appeared in Park et al. (2020). The current version (1) extends the graph topology to the spatiotemporal domain to reason the semantic relationships between fine-grained visual entities (i.e., objects) in the video; (2) presents the language-guided graph construction framework to leverage semantic knowledge of language queries and further enhance semantic relationships between the graph nodes by exploring two different types of language queries; (3) extends the recursive learning scheme to refine feature representations from spatial to temporal levels; and (4) contains extensive experiments on the SumMe (Gygli et al., 2014), TVSum (Song et al., 2015), and QFVS (Sharghi et al., 2017) datasets for the generic and query-focused video summarization.

2 Related Work

2.1 Video Summarization

Video summarization condenses a given video into a short synopsis, including video synopses (Pritch et al., 2007); time lapse clips (Joshi et al., 2015; Poleg et al., 2015); montages (Kang et al., 2006; Sun et al., 2014), or storyboards (Gong et al., 2014; Gygli et al., 2015). Early video summarization relied primarily on handcrafted criteria (Kang et al., 2006; Lee et al., 2012; Liu & Kender, 2002; Ngo et al., 2003; Lu & Grauman, 2013; Potapov et al., 2014), such as importance, relevance, representativeness, and diversity to produce a summary video.

Deep networks have recently been applied with considerable success (Zhang et al., 2016b; Rochan et al., 2018; Zhang et al., 2018; Rochan & Wang, 2019; Park et al., 2020; Zhao et al., 2020; Ji et al., 2021); Recurrent models are widely used to capture the variable range dependencies between frames (Zhang et al., 2016b; Mahasseni et al., 2017; Zhang et al., 2018; He et al., 2019; Yuan et al., 2019; Zhao et al., 2020; Ji et al., 2021; Kaseris et al., 2022). Although RNN-based approaches have been applied to video data, the recurrent operations are sequential, limiting the simultaneous processing of all the frames. To overcome this limitation, CNN-based approaches have been proposed by formulating video summarization as a binary label prediction problem (Rochan et al., 2018) and a temporal boundary regression problem (Zhu et al., 2020). However, the consideration of modeling the relationships among the frames, which provides significant cues as to how best to summarize the video, is not addressed in this approach. Transformer (Vaswani et al., 2017) has recently been exploited for video summarization (Li et al., 2022). While they successfully incorporated the transformers into a cross-video attention module to take into consideration the semantic dependencies across videos, the high computational requirement of the transformers arises as a new problem.

Some existing methods have been extended into an unsupervised training scheme (Yuan et al., 2019; Mahasseni et al., 2017; Rochan & Wang, 2019; He et al., 2019). Mahasseni et al. (2017) extended an LSTM-based framework with a discriminator network without human-annotated summary videos. Their frame selector uses a variational autoencoder LSTM to decode the output for reconstruction through selected frames. The discriminator is another LSTM network that learns to distinguish between the input video and its reconstruction. Rochan and Wang (2019) trained a network from unpaired data using adversarial learning. He et al. (2019) produced weighted frame features to predict importance scores with attentive conditional generative adversarial networks, whereas Yuan et al. (2019) proposed a cycle-consistent learning objective to relieve the difficulty of unsupervised learning. Although these approaches resolved the problems caused by insufficient data, the locality of recurrent and convolutional operations, which does not directly compute relevance between any two frames, is still problematic. To address this problem, Park et al. (2020) introduced the graph convolutional networks (GCNs) to the video summarization. They formulated the video summarization as a temporal graph modeling problem and proposed a recursive graph refinement framework in which keyframes are connected by semantic affinities. Similarly, there has been an attempt to leverage the spatiotemporal graph for video summarization (Wu et al., 2021). Although they demonstrated that fine-grained visual entities (i.e., objects) can be an alternative to complement frame information, less progress has been made in constructing the graph with semantic relationships between entities.

Most recently, language-guided video summarization methods (Narasimhan et al., 2021; Wu et al., 2022) have been proposed to produce the video summary based on the given language queries. They demonstrated the effectiveness of the language-guided video summarization framework by learning the interactions between the video and language queries. However, they require manually provided keywords from a benchmark or an additional model and computations for caption generation, which limits their applicability. While we share the same spirit with their works, we explore more in-depth studies according to types of language queries with fine-grained visual representations.

The recursive learning scheme in video summarization has also appeared in Li et al. (2023a).

They proposed to iteratively refine frame scores by enhancing the frame representations based on the frame scores predicted from the previous stage. Due to their iterative direct prediction process of subsequent stages, the incorrect biases from an early stage can be propagated to the following stages (Carreira et al., 2016; Mukhoty et al., 2019). In contrast, we define refinement on semantic connections between

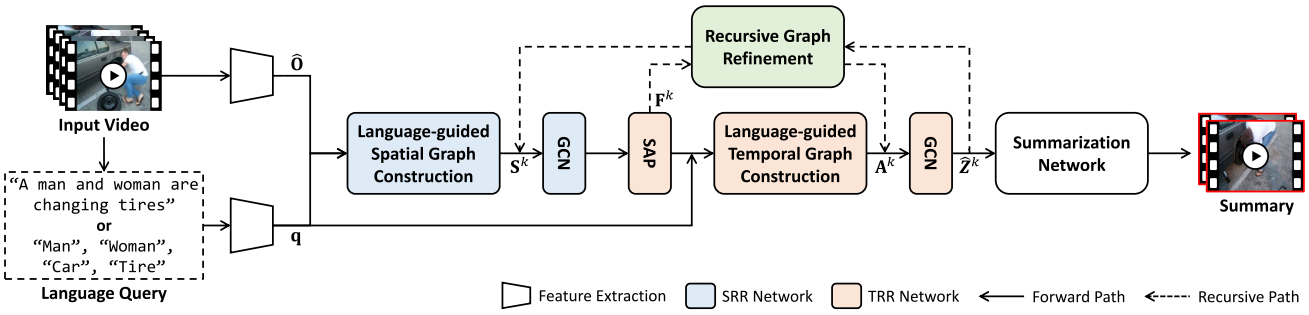


Fig. 4 Overall framework of VideoGraph comprising feature extraction, spatial relation reasoning (SRR) network, temporal relation reasoning network (TRR), and summarization network in a recurrent structure

nodes within graphs, where interactions between nodes are continuously updated to reflect semantic relationships.

2.2 Graphical Models

Our notion of modeling a graph from a video is partly related to recent research in graphical models. One popular direction is using conditional random fields (CRF) (Lafferty et al., 2001), especially for semantic segmentation (Krähenbühl & Koltun, 2011; Chen et al., 2018), where the CRF model is applied to all pairs of pixels in an image to infer mean-field with high confidence.

An attempt at modeling pairwise spatiotemporal relations has been made in non-local neural networks (Wang et al., 2018). However, the model is not explicitly defined on graphs. Moreover, the non-local operator is applied to every pixel in the feature space, from lower to higher layers incurring a high computational cost. GCNs (Kipf & Welling, 2017) have been used in several computer vision areas, such as video classification (Jain et al., 2016; Wang & Gupta, 2018; Shi et al., 2019a), visual object tracking (Gao et al., 2019; Li et al., 2020), and temporal action localization (Zeng et al., 2019, 2021) to model the temporal relations in a video. They have demonstrated the effectiveness of GCNs by exploiting the relations between input data, showing satisfactory performance on each task. Similarly, spatiotemporal graph convolutional networks (ST-GCNs) (Yu et al., 2018) have been employed in several computer vision tasks, including video relation detection (Qian et al., 2020), video captioning (Pan et al., 2020; Zhang et al., 2020a), spatiotemporal video grounding (Zhang et al., 2020b; Li et al., 2023b), emotion recognition (Bhattacharya et al., 2020), person re-identification (Yang et al., 2020), and skeleton-based action recognition (Yan et al., 2018; Shi et al., 2019b; Ye et al., 2020). They have effectively modeled temporal dynamics and spatial dependencies of videos by incorporating spatiotemporal information. However, they constructed the spatial graph based on visual and positional similarities between entities (Zhang et al., 2020b; Pan et al., 2020). From this perspective, their temporal modeling mainly focused on capturing dynamics in the same entity over time (usually in a few seconds) (Zhang et al., 2020b; Li et al., 2023b; Pan et al., 2020; Zhang et al., 2020a). In addition, since their graphs are obtained from input feature representations, the connections of graphs may not be optimal (Jiang et al., 2019) and are heavily biased to the visual similarities. Instead of using fixed initial graphs, we recursively refine the graphs by learning the relationships between nodes to obtain the optimal relation graphs in which edges are connected by a semantic story. While some works have attempted to learn the sample-wise spatial graph (Shi et al., 2019b; Ye et al., 2020), their temporal graph is still fixed, accomplishing information propagation along the temporal dimension without considering any interactions between spatial and temporal graphs. Contrary to this, the spatial and temporal graphs in VideoGraph are fully trainable and recursively refined from the output of the other graph (e.g. refining the adjacency matrix of the spatial graph from the output of the temporal graph).

Our work is partially similar to video scene graph generation (Teng et al., 2021; Wang et al., 2022; Gao et al., 2022) that produces semantic graphs for given videos. However, there are several major differences. First, they require a complicated object tracking framework to capture the consistently appearing objects in a video. Second, they construct graphs for co-occurrence objects that have direct interactions in consecutive frames. We simply reuse the pretrained object detector and construct semantic connections for indirectly interacted objects based on the whole story of the video.

3 Preliminaries

This section briefly reviews GCNs (Kipf & Welling, 2017). Given a graph $\mathcal{G}$ represented by a tuple $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of unordered vertices, and $\mathcal{E}$ is the set of edges representing the connectivity between vertices $v \in \mathcal{V}$, GCNs aim to extract richer features at a vertex by aggregating the features from neighboring vertices. Suppose vertices v_i and

v_j are connected by an edge $e_{ij} \in \mathcal{E}$. GCNs represent vertices by associating each vertex v with a feature representation h_v. The adjacency matrix $\mathbf{A}$ is derived as a $T \times T$ matrix with $A_{ij} = e_{ij}$ representing similarity between vertices v_i and v_j for a fully connected graph, or $A_{ij} = 1$ if $e_{ij} \in \mathcal{E}$, and $A_{ij} = 0$ if $e_{ij} \notin \mathcal{E}$ for a binary graph.

In standard GCNs, the output of the graph convolution operation is as follows:

$$\mathbf{Z} = \sigma(\mathbf{D}^{-1/2}\hat{\mathbf{A}}\mathbf{D}^{-1/2}\mathbf{X}\mathbf{W}), \tag{1}$$

where $\sigma(\cdot)$ denotes an activation function such as ReLU, $\mathbf{X}$ and $\mathbf{Z}$ are input and output features, and $\hat{\mathbf{A}} = \mathbf{A}+\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. $\mathbf{D}$ is a diagonal matrix in which a diagonal entry is a sum of the row elements of $\hat{\mathbf{A}}$, and $\mathbf{W}$ is a trainable weight matrix in the graph convolution layer, respectively. Given the graph representation, we can perform reasoning on the graph by applying GCNs rather than applying CNNs or RNNs, which have limited capability to represent relationships between features. We briefly denote the consecutive graph convolutions as $\mathcal{F}$ in the following sections for the sake of simplicity.

4 VideoGraph

4.1 Problem Formulation and Overview

Inspired by Ngo et al. (2003), VideoGraph learns to model the input video as a graph $\mathcal{G} = \{\mathcal{G}_S, \mathcal{G}_T\}$ guided by the language query to select a set of keyframes, $\mathcal{K}$, where $\mathcal{G}_S$ and $\mathcal{G}_T$ are spatial and temporal graphs, respectively. The main improvement over (Park et al., 2020) is extending the graph topology to the spatiotemporal domain and leveraging semantic knowledge of language queries. The proposed model incorporates the relationships between objects in each frame with edges of the spatial graph and propagates them across frames with edges of the temporal graph. Specifically, we regard objects in each frame as nodes and their relationships as edges of the spatial graph, such that the spatial graph can be represented as

$$\mathcal{G}_S = \{(\mathcal{V}_S^t, \mathcal{E}_S^t)\}_{t=1}^T, \quad \{v_S^{t,n}\}_{n=1}^N \subset \mathcal{V}_S^t, \tag{2}$$

where $\mathcal{V}_S^t$ and $\mathcal{E}_S^t$ are the set of nodes and edges in the t-th frame, and $v_S^{t,n} \in \mathcal{V}_S^n$ is the n-th node in each frame. T is the number of video frames and N is the number of objects in each frame. The object nodes in each frame are averaged and added by the temporal positional encodings to derive a frame node $v_t \in \mathcal{V}_T$, and the relationships between frames are used as edges $\mathcal{E}_T$ for the temporal graph, such that

$$\mathcal{G}_T = (\mathcal{V}_T, \mathcal{E}_T), \quad v_t = < \mathcal{V}_S^t > \oplus \text{PE}, \tag{3}$$

where v_t is the t-th node of $\mathcal{G}_T$, $\oplus$ is the elementwise summation, PE denotes temporal positional encoding, and $< \cdot >$ is the average operation. In each graph construction, the node representations are enhanced with a language query representation using a multi-head cross-attention to transfer semantic knowledge of the language query to the node representations.

In conventional GCNs, the graph is initially constructed from the input data through explicit graphical modeling and fixed in its initial form. In contrast to the prior works that capture the visual similarity by using fixed graphs (Gao et al., 2019; Zeng et al., 2019), we seek to refine the initial graph iteratively such that the nodes are connected by the story of the video. To accomplish this, we formulate the recursive spatiotemporal graph relation networks that gradually complete the semantic graph by iteratively estimating the residuals of the adjacency matrices of the graphs.

As shown in Fig. 4, our networks are split into four parts: 1) *feature extraction networks* to extract initial object node representations and a language query representations from pretrained models; 2) a *spatial relation reasoning network* to build the language-guided spatial graph $\mathcal{G}_S$ and estimate the residual temporal adjacency matrix $d\mathbf{A}$, where $\mathbf{A}$ is a temporal adjacency matrix; 3) a *temporal relation reasoning network* to construct the language-guided temporal graph $\mathcal{G}_T$ from the spatial graph and estimate the residual spatial adjacency matrix $d\mathbf{S}_t$, where $\mathbf{S}_t$ is a spatial adjacency matrix for the t-th frame; and 4) a *summarization network* to classify the final frame representations into keyframes for the summary video.

4.2 Feature Extraction Network

4.2.1 Visual feature representation

In contrast to Park et al. (2020), which constructs the temporal relation graph directly from the frame feature representations, we first construct the language-guided spatial graph $\mathcal{G}_S$ in the spatial relation reasoning (SRR) network, where objects in each frame are regarded as nodes and connected by their semantic relationships guided by the language query. Although every object node can be connected over frames, we separately construct the spatial graph for each frame to account for computational efficiency. Specifically, we extract a set of object feature representations $\mathbf{O}_t = \{\mathbf{o}_t^n\}_{n=1}^N$ for the t-th frame using an off-the-shelf object detector, where N is the number of objects. The extracted feature is fed into a fully-connected (FC) layer to project $\mathbf{o}_t^n$ to a d-dimensional embedding space as follows:

$$\hat{\mathbf{o}}_t^n = \mathbf{W}_o\mathbf{o}_t^n + \mathbf{b}_o, \tag{4}$$

where $\hat{\mathbf{O}}_t$ is the initial visual representations for the t-th frame, $\mathbf{W}_o$ and $\mathbf{b}_o$ are learnable parameters of the FC layer.

4.2.2 Language query representation

To explore the effectiveness of language queries, we introduce two different types of language queries.

Sentence-level language query. We employ an off-the-shelf dense video captioning model (Iashin & Rahtu, 2020) to generate the sentence corresponding to a given video, following the prior work (Narasimhan et al., 2021). In this case, we can obtain multiple language captions for each video. The generated captions are projected into the feature space through the text encoder (e.g. the pretrained CLIP (Radford et al., 2021) model). We concatenate all caption features and feed them into a multi-layer perceptron (MLP) to extract the final language query representation, such that

$$\mathbf{q} = \text{MLP}(\text{Concat}(l_1, l_2, ..., l_M)), \tag{5}$$

where M is the number of generated captions and l_m is the m-th caption feature from the text encoder.

Word-level language query. As an alternative to the sentence-level language query, we directly use the object classes detected from the object detector (e.g. car, tire, and person in Fig. 4) as word-level language queries. We select W object classes based on the detected frequency over the whole video, assuming that frequently detected objects are closely related to the main contents of the video. We embed each word using the text encoder and the MLP layer to low-dimensional embedding space, and project the concatenated word embeddings using the MLP for the final language query representation:

$$\mathbf{q} = \text{MLP}(\text{Concat}(c_1, c_2, ..., c_W)), \tag{6}$$

where c_w is the w-th low-dimensional word embedding.

While the sentence-level language query produces more elaborate descriptions for the video, it requires additional parameters for the dense video captioning model. In contrast, the word-level language query can be obtained without any additional captioning model, but it only provides simple keywords. We compare the results according to the type of language query in Sec. 5 in terms of the summarization performance, the number of model parameters, and the inference time.

4.3 Spatial Relation Reasoning Network

We construct the initial language-guided spatial graph $\mathcal{G}_S$ for the t-th frame as an undirected fully connected graph, as shown in Fig. 5. To obtain the set of language-guided object node representations $\tilde{\mathbf{O}}_t$, we set the language query representation $\mathbf{q}$ as a query Q_S, and the set of object node representations $\hat{\mathbf{O}}_t$ as a key K_S and a value V_S of the modified multi-head cross-attention (MHCA) (Narasimhan et al., 2021) as follows:

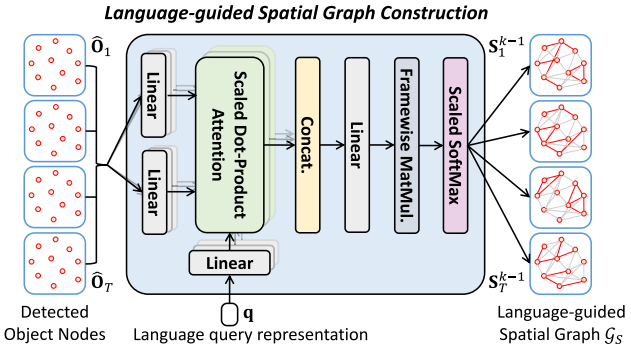


Fig. 5 Language-guided spatial graph construction module

$$\begin{aligned} Q_S = \mathbf{q}, \quad K_S, V_S = \hat{\mathbf{O}}_t, \\ \tilde{\mathbf{O}}_t = \text{Concat}(\text{head}_1, ..., \text{head}_{H_S})\mathbf{W}_S, \\ \text{head}_h = \text{Attention}(Q_S\mathbf{W}_h^{SQ}, K_S\mathbf{W}_h^{SK}, V_S\mathbf{W}_h^{SV}) \\ \text{Attention}(Q, K, V) = \text{softmax}(QK^\top/\sqrt{d_K})V, \end{aligned} \tag{7}$$

where H_S denotes the number of heads in the MHCA, $\mathbf{W}_S, \mathbf{W}_h^{SQ}, \mathbf{W}_h^{SK}, \mathbf{W}_h^{SV}$ are the learnable parameter matrices, and d_K is the dimension of K. Different from the prior work (Narasimhan et al., 2021), which used the visual features as the query, and the language feature as the key and value, we set the language feature as the query to learn the *language-guided visual representations*. The spatial adjacency matrix $\mathbf{S}_t$, which represents connections of $\tilde{\mathbf{o}}_t$ and their edge weights, is obtained by computing the affinity between language-guided node representations,

$$s_{t,ij} = \frac{\exp(\lambda_o(\tilde{\mathbf{o}}_t^i)^\top \tilde{\mathbf{o}}_t^j)}{\sum_{j=1}^N \exp(\lambda_o(\tilde{\mathbf{o}}_t^i)^\top \tilde{\mathbf{o}}_t^j)}, \tag{8}$$

where $s_{t,ij}$ indicates the edge weight between the i-th and j-th nodes in the t-th spatial graph, and λ_o is a scaling factor.

The language-guided object node representations and the adjacency matrix for the t-th frame are fed into consecutive graph convolution layers to aggregate each representation to its neighbor nodes along spatial edges, such that the output of the SRR network is derived by the following equation:

$$\mathbf{Z}_t = \mathcal{F}(\tilde{\mathbf{O}}_t, \mathbf{S}_t | \mathbf{W}_z), \tag{9}$$

where $\mathbf{Z}_t$ is the output of the SRR network for the t-th frame and $\mathcal{F}$ is the feed-forward process for consecutive graph convolution layers with the set of learnable parameters $\mathbf{W}_z$. The aggregated object node representations $\{\mathbf{Z}_t\}_{t=1}^T$ are used to construct an initial language-guided temporal graph at the first feed-forward pass and estimate the residual temporal adjacency matrix in the recursive graph refinement process.

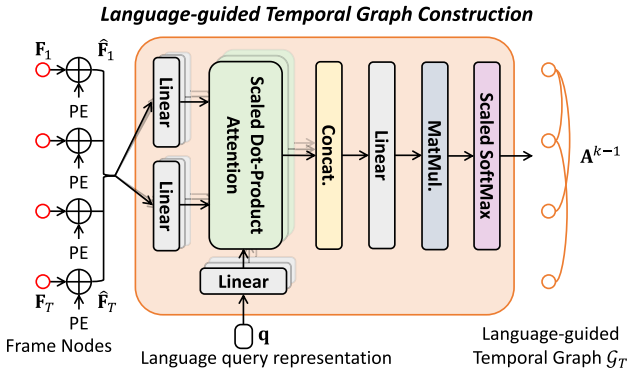


Fig. 6 Language-guided temporal graph construction module

4.4 Temporal Relation Reasoning Network

Although relationships between entities in the same frame can be learned in the SRR network, temporal relations that represent semantic relationships between frames are not captured. To take temporal dependencies of the video frames into account, we construct the temporal graph $\mathcal{G}_T$ in the temporal relation reasoning (TRR) network in which language-guided frame representations are regarded as nodes and fully connected by their semantic affinities, as shown in Fig. 6. Concretely, the frame representation is derived from the output of the SRR network by averaging the aggregated object node representations in a corresponding frame, such that the t-th frame node representation can be represented as

$$\mathbf{f}_t = \frac{1}{N}\sum_{n=1}^{N}\mathbf{z}_t^n, \tag{10}$$

where N is the number of objects in the t-th frame. Similar as in the SRR network, we set $\mathbf{q}$ as the query, and the frame representations with the positional encoding (PE) as the key and value of the additional MHCA to obtain the language-guided frame node representations $\hat{\mathbf{F}}$:

$$\begin{aligned} Q_T = \mathbf{q}, \quad K_T, V_T = \mathbf{F} + \mathrm{PE}, \\ \hat{\mathbf{F}} = \mathrm{Concat}(\mathrm{head}_1, ..., \mathrm{head}_{H_T})\mathbf{W}_T, \\ \mathrm{head}_h = \mathrm{Attention}(Q_T\mathbf{W}_h^{FQ}, K_T\mathbf{W}_h^{FK}, V_T\mathbf{W}_h^{FV}), \end{aligned} \tag{11}$$

where H_T denotes the number of heads, $\mathbf{W}_h^{FQ}$, $\mathbf{W}_h^{FK}$, and $\mathbf{W}_h^{FV}$ are the learnable parameters to embed the query, key, and value, respectively. The temporal adjacency matrix $\mathbf{A}$ is obtained in the same way as in (8), and the edge weight a_{ij} between the i-th and j-th frame nodes is

$$a_{ij} = \frac{\exp(\lambda_f \hat{\mathbf{f}}_i^\top \hat{\mathbf{f}}_j)}{\sum_{j=1}^{T}\exp(\lambda_f \hat{\mathbf{f}}_i^\top \hat{\mathbf{f}}_j)}, \tag{12}$$

where λ_f is a scaling factor for the temporal graph.

Although the frame node representation can be directly propagated to its neighbor nodes through graph convolution, the conventional node propagation cannot be applied to the recursive graph refinement formulation that derives the residual spatial adjacency matrix from frame node representations, since the average operation in (10) discards the object coordinates. Therefore, we reformulate the graph convolution operation in the TRR network to reason temporal relationships by propagating frame nodes to object nodes over frames along temporal edges rather than through frame-to-frame propagation. Formally, the n-th object node representation in the t-th frame after applying a single graph convolution layer in the TRR network can be expressed as

$$\begin{aligned} \hat{\mathbf{z}}_t^n &= \sigma(\mathbf{W}_f(\mathbf{z}_t^n + \sum_{k\in\mathcal{N}_t}\frac{a_{tk}}{N}\sum_{m=1}^{N}\mathbf{z}_k^m) + \mathbf{b}_f) \\ &= \sigma(\mathbf{W}_f(\mathbf{z}_t^n + \sum_{k\in\mathcal{N}_t} a_{tk}\hat{\mathbf{f}}_k) + \mathbf{b}_f), \end{aligned} \tag{13}$$

where $\mathbf{W}_f$ and $\mathbf{b}_f$ are learnable parameters of the graph convolution layer, and $\mathcal{N}_t$ is a set of neighboring nodes to the t-th frame node. This form of aggregation not only maintains the advantages of the graph convolution operation but also retains the object coordinates, allowing the residual spatial adjacency matrix to be estimated from frame nodes. The updated object node representations are the d to estimate the residual spatial adjacency matrix $d\mathbf{S}$ in the recursive graph refinement process and fed into the summarization network for graph reasoning after the final iteration.

4.5 Recursive Graph Refinement

Although the visual representations are guided by the language query, the initial spatial and temporal graphs are highly biased towards visual similarity since the initial object node representations are extracted from the pretrained visual network. In addition, constructing graph edges by connecting all objects and frames aggregates redundant and noisy information for video summarization. Therefore, unrelated node pairs should be disconnected. To this end, the SRR and TRR networks interact with each other by iteratively estimating relative residual adjacency matrices to refine the spatial and temporal graphs. Formally, intermediate object node representations for the t-th frame $\mathbf{Z}_t^k$ are derived from (9) such that, $\mathbf{Z}_t^k = \mathcal{F}(\mathbf{Z}_t^{k-1}, \mathbf{S}_t^{k-1}|\mathbf{W}_z)$. The averaged object node representations $\mathbf{F}^k$, i.e., the frame node representations at the k-th iteration, are used to estimate the residual between the

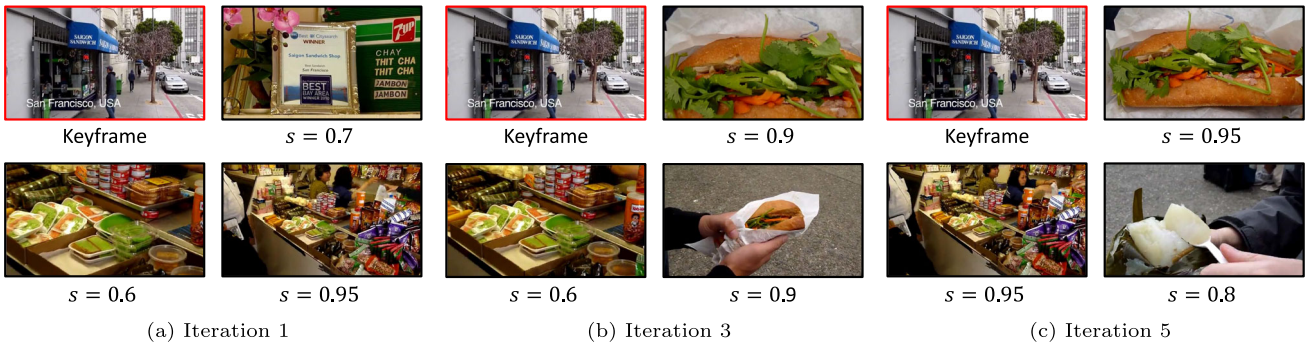


Fig. 7 Convergence of VideoGraph: We depict frames that have top3 affinity value with a selected keyframe at (a) iteration 1, (b) iteration 3, and (c) iteration 5; where s denotes the averaged and normalized user-annotated importance scores which range from 0 to 1. The keyframes are progressively connected with high affinity as graph refinement repeats in VideoGraph

previous and current temporal adjacency matrices as

$$\begin{aligned}\mathbf{A}^k - \mathbf{A}^{k-1} &= d\mathbf{A}^k \\ &= \sigma\left(\frac{(\mathbf{W}_{\theta_S}\mathbf{F}^k)^\top(\mathbf{W}_{\phi_S}\mathbf{F}^k)}{||\mathbf{W}_{\theta_S}\mathbf{F}^k||_2 \cdot ||\mathbf{W}_{\phi_S}\mathbf{F}^k||_2}\right),\end{aligned} \tag{14}$$

where $\mathbf{W}_{\theta_S}$ and $\mathbf{W}_{\phi_S}$ are two different weight matrices to be learned to enable computing the affinity in the linear embedding space, and $\sigma(\cdot)$ is an activation function. The refined temporal adjacency matrix $\mathbf{A}^k = \mathbf{A}^{k-1} + d\mathbf{A}^k$ aggregates the object node representations over frames in the TRR network, and the updated object node representations $\hat{\mathbf{Z}}_t$ are obtained following (13).

Similarly, the spatial adjacency matrix $\mathbf{S}_t^k$ for the t-th frame at the k-th iteration is refined from $\hat{\mathbf{Z}}_t$ as

$$\begin{aligned}\mathbf{S}_t^k &= \mathbf{S}_t^{k-1} + d\mathbf{S}_t^k \\ &= \mathbf{S}_t^{k-1} + \sigma\left(\frac{(\mathbf{W}_{\theta_T}\hat{\mathbf{Z}}_t^{k-1})^\top(\mathbf{W}_{\phi_T}\hat{\mathbf{Z}}_t^{k-1})}{||\mathbf{W}_{\theta_T}\hat{\mathbf{Z}}_t^{k-1}||_2 \cdot ||\mathbf{W}_{\phi_T}\hat{\mathbf{Z}}_t^{k-1}||_2}\right),\end{aligned} \tag{15}$$

where $\mathbf{W}_{\theta_T}$ and $\mathbf{W}_{\phi_T}$ are weight matrices to be learned.

The final spatial and temporal adjacency matrices are then formulated in a recurrent manner as

$$\mathbf{A}^K = \mathbf{A} + \sum_{k=1}^{K} d\mathbf{A}^k, \quad \mathbf{S}_t^K = \mathbf{S}_t + \sum_{k=1}^{K} d\mathbf{S}_t^k, \tag{16}$$

where K denotes the maximum iteration, and $\mathbf{A}$ and $\mathbf{S}_t$ are the initial adjacency matrices. In contrast to Wang and Gupta (2018) where the adjacency matrix is constructed from the input feature representations and fixed, we obtain the refined adjacency matrices through iterative refinement with the output representations. Repeatedly inferring the relative residual adjacency matrices facilitates fast and stable convergence for video summarization and gradually incorporates the spatial and temporal relationships. Furthermore, residual shortcut connections can transfer information easily (directly propagating information) by addition rather than accumulation by multiplication. In practice, the object and frame nodes initially connected by the visual similarity are progressively linked into the subset of nodes based on semantic relationships, as shown in Fig. 7.

4.6 Summarization Network

Our final goal is to classify each node in the temporal graph. As illustrated in Fig. 4, the final object node representations $\hat{\mathbf{Z}}^K$ and temporal adjacency matrix $\mathbf{A}^K$ are fed into the summarization network to create a video summary. We first feed the final object node representations $\hat{\mathbf{Z}}^K$ into the average pooling layer to produce final frame node representations $\tilde{\mathbf{F}}^K$. We then append the graph convolutional layers, followed by an activation function to obtain a binary summary score $\mathbf{Y}$ indicating the probability that each node is a keyframe or background, such that,

$$\mathbf{Y} = \mathcal{F}(\tilde{\mathbf{F}}^K, \mathbf{A}^K|\mathbf{W}_s), \tag{17}$$

where $\mathbf{W}_S$ is a set of learnable parameters of the summarization network. If the probability corresponding to the keyframe is higher than that of the background, the node is contained in the set of keyframes $\mathcal{K}$.

4.7 Loss Functions

To utilize our model in a practical environment, we verify VideoGraph in supervised and unsupervised settings. Specifically, we train VideoGraph with four loss functions for a supervised setting, including the classification, sparsity, reconstruction, and diversity loss functions; and three loss functions (omitting the classification loss) for an unsupervised setting.

For the classification loss $\mathcal{L}_c$, we apply a weighted binary cross entropy loss between the output of the summarization network and the groundtruth label to address imbalances between the number of keyframes and background frames, following the previous works (Rochan et al., 2018; Zhang et al., 2016b):

$$\mathcal{L}_c^{\text{bin}} = -\frac{1}{T}\sum_{t=1}^{T} w_t[y_t^* \log(y_t) + (1-y_t^*)\log(1-y_t)], \quad (18)$$

where y_t^* is the groundtruth binary label of the t-th frame. Each node is weighted by $w_t = \omega_m/\omega_t$, where $\omega_t = \frac{|\mathcal{K}|}{T}$ for keyframes and $\omega_t = 1 - \frac{|\mathcal{K}|}{T}$ for the background frames with the number of keyframes $|\mathcal{K}|$. Since there are two classes (keyframe or not), we set the median frequency ω_m to 0.5.

In addition to the binary label, the importance score for each frame has been utilized to train the model (Zhao et al., 2020, 2021; Li et al., 2022, 2023a). Since the score label represents the relative importance of each frame, it provides more fine information to the model. We alternatively apply the mean squared error between the groundtruth importance score y_t^* and the predicted score y_t as the classification loss:

$$\mathcal{L}_c^{\text{sco}} = \frac{1}{T}\sum_{t=1}^{T} ||y_t - y_t^*||_2^2. \quad (19)$$

In practice, a long video can be summarized into a sparse subset of keyframes. Based on this intuition, our model learns the parameters to construct sparse spatial and temporal graphs for video summarization. We employ an entropy function (Grandvalet & Bengio, 2004) of the spatial and temporal adjacency matrices to enforce this constraint as follows:

$$\begin{aligned} \mathcal{L}_s^s &= -\sum_{t=1}^{T}\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} s_{t,ij}^K \log s_{t,ij}^K, \\ \mathcal{L}_s^t &= -\sum_{i=1}^{T}\sum_{j=1,j\neq i}^{T} a_{ij}^K \log a_{ij}^K, \end{aligned} \quad (20)$$

where $s_{t,ij}^K \in \mathbf{S}_t^K$ and $a_{ij}^K \in \mathbf{A}^K$. The sparsity loss is then given by the combination of two entropy functions as

$$\mathcal{L}_s = \mathcal{L}_s^s + \rho \cdot \mathcal{L}_s^t, \quad (21)$$

where ρ controls the balance between the two entropy functions. By minimizing uncertain and noisy edge values, the entropy function enables learning more discriminative connections in fully connected graphs than (Park et al., 2020), which uses L_1 normalization as the sparsity loss.

The keyframes should be visually diverse to ensure the summary represents the entire video story (Zhang et al., 2016b; Rochan et al., 2018). The reconstruction and diversity loss functions are employed to facilitate diverse keyframe selection. To reconstruct the original frame node representations $\mathbf{X}_{\mathcal{K}}$, we decode the final frame node representations of the selected keyframes $\hat{\mathbf{F}}_{\mathcal{K}}^K$ by applying consecutive 1×1 convolutions so that the dimensionality of the decoded representation is the same as the original node representations. The decoded and original frame node representations are concatenated and fed into an additional 1×1 convolution layer to obtain the final reconstructed frame node representations. The reconstruction loss $\mathcal{L}_r$ is defined as the mean squared error between the reconstructed and original node representations, such that:

$$\mathcal{L}_r = \frac{1}{|\mathcal{K}|}\sum_{t\in\mathcal{K}} ||\mathbf{x}_t - \hat{\mathbf{x}}_t||_2^2, \quad (22)$$

where $\hat{\mathbf{x}}$ denotes the final reconstructed features. Note that, the original frame node representations are derived by averaging the input object node representations in each frame as $\mathbf{x}_t = \frac{1}{N}\sum_{n=1}^{N} \mathbf{o}_t^n$, to reduce the computational complexity.

To ensure the visual diversity of the selected keyframes, we apply a repelling regularizer (Zhao et al., 2017) to the reconstructed frame node representations as the diversity loss:

$$\mathcal{L}_d = \frac{1}{|\mathcal{K}|(|\mathcal{K}|-1)}\sum_{i\in\mathcal{K}}\sum_{j\in\mathcal{K},j\neq i} \frac{\hat{\mathbf{x}}_i^T\hat{\mathbf{x}}_j}{||\hat{\mathbf{x}}_i||_2^2 \cdot ||\hat{\mathbf{x}}_j||_2^2}. \quad (23)$$

Since the reconstruction and diversity loss functions complement each other, they are applied simultaneously to learn the model.

The final loss function for supervised learning is then,

$$\mathcal{L}_{sup} = \mathcal{L}_c^* + \alpha \cdot \mathcal{L}_s + \beta \cdot \mathcal{L}_d + \gamma \cdot \mathcal{L}_r, \quad (24)$$

where $\mathcal{L}_c^*$ is the classification loss according to the groundtruth types (i.e., binary or importance score), α, β, and γ control the trade-off between the four loss functions. The graph modeling scheme in VideoGraph has also been extended for unsupervised video summarization with a simple modification to the loss functions. Since the groundtruth summary cannot be used as supervision in an unsupervised manner, the final loss function for unsupervised learning is represented as

$$\mathcal{L}_{unsup} = \alpha \cdot \mathcal{L}_s + \beta \cdot \mathcal{L}_d + \gamma \cdot \mathcal{L}_r, \quad (25)$$

where α, β, and γ are balancing parameters to control the trade-off between the three terms.

Table 1 The format of produced groundtruth (GT) and converted training and testing GT for different datasets. In training, we convert from multiple user frame-level annotations to a single set of the keyframes for the SumMe (Gygli et al., 2014) and TVSum (Song et al., 2015) datasets. In testing, frame-level scores are converted to the interval-based keyshot annotations for the TVSum dataset following (Zhang et al., 2016a, b). Note that OVP and YouTube datasets are not used in the testing phase

Dataset	Produced GT	Training GT	Testing GT
SumMe	frame scores	keyframes	keyshots
TVSum	frame scores	keyframes	keyshots
OVP	keyframes	keyframes	-
YouTube	keyframes	keyframes	-

5 Experiments

5.1 Experimental Settings

5.1.1 Datasets

We evaluated our model on two standard video summarization datasets (i.e., SumMe (Gygli et al., 2014) and TVSum (Song et al., 2015)), and on the egocentric video dataset (UT Egocentric (Lee et al., 2012)). The SumMe dataset consists of 25 videos capturing multiple events, including cooking and sports, and the lengths of the videos vary from 1.5 to 6.5 minutes. The TVSum dataset contains 10 categories from the MED task, including 5 videos per category sampled from YouTube. The contents of the videos are diverse, similar to the SumMe, and the video lengths vary from 1 to 5 minutes. These datasets provide frame-level importance scores annotated by several users. We used additional datasets, the YouTube (De Avila et al., 2011) and OVP [43] datasets, to augment the training data. The UT egocentric (Lee et al., 2012) dataset consists of four videos recorded by wearable first-person view cameras. Each video contains 3-5 hours long multiple diverse events captured in uncontrolled environments. The QFVS (Sharghi et al., 2017) dataset provides generic summaries and query-focused summaries for each video in the UT egocentric dataset. For the generic summaries, each video is divided into multiple clips and annotated by selecting clips for the generic contents of each video by three users and averaging all the annotations. For the query-focused summaries, three users selected clips corresponding to given language queries (e.g. book and street). We use the groundtruth summaries to evaluate the generic and query-focused summarization performance.

5.1.2 Groundtruth

The groundtruth annotations of the datasets used in the experiments (i.e., SumMe, TVSum, OVP, and YouTube) are provided in various formats, as shown in Table 1. Following previous works (Gong et al., 2014; Zhang et al., 2016b; Rochan et al., 2018), we generated a single set of keyframes for each training video from multiple user annotations. Specifically, we labeled the frames selected for summary as 1 and others as 0. We converted the predicted keyframes and groundtruth keyframes to the interval-based keyshot summaries for the testing videos used in the performance evaluations. Since the TVSum dataset only provides keyframe annotations, we convert the keyframe annotations to keyshot-based summaries following the steps in Zhang et al. (2016b): 1) apply kernel temporal segmentation (KTS) (Potapov et al., 2014) to generate temporal segments in the form of disjoint intervals; 2) compute the average importance score for each segment and assign the score to the frames in each interval; and 3) apply the knapsack algorithm (Song et al., 2015) to select frames such that the length of the keyshot groundtruth is not exceeded a certain threshold. Consequently, the keyshot-based annotations are represented by a binary vector (i.e., 1 for keyframes and 0 for the background) where the number of elements is equal to the number of frames in the video.

5.1.3 Data configuration

Generic video summarization. We evaluate our model on the SumMe and TVSum datasets with three different data configurations for generic video summarization: standard, augmented, and transfer data settings. The standard data setting selects training and testing videos from the same dataset by splitting 80% of the videos for training and the remainder for testing. The augmented data setting uses the other three datasets to augment training data. For example, 80% videos of the SumMe and whole videos of the TVSum, OVP, and YouTube datasets are used for training, and 20% videos of the SumMe are used to evaluate the performance on SumMe. Not surprisingly, augmenting the training data derives improved performances similar to recent works (Zhang et al., 2018; Rochan et al., 2018). The more challenging setting, the transfer data setting introduced in Zhang et al. (2016b), uses other available datasets to train the model, and the target dataset is only used to evaluate the performance. In addition, we evaluate the model on the QFVS dataset using the generic summary groundtruth. We run four rounds of experiments to evaluate the generic video summarization performance by splitting four videos into 2, 1, and 1 for training, validation, and testing, respectively, following prior works (Sharghi et al., 2017; Narasimhan et al., 2021). We train the model with three different levels of the language query: using video data only (w/Video only), the word-level language query (w/Word-level query), and the sentence-level language query (w/Sentence-level query).

Query-focused video summarization. We provide the query-focused video summarization performance evaluated

on the QFVS dataset. Each video in the dataset is divided into 45 different language queries (e.g. book_chair, car_market), and the groundtruth corresponding to each query is provided. We use two videos for training, one video for validation, and the remaining one video for evaluation for query-focused video summarization. Different from generic video summarization, we train the model with captions provided by the dataset in a supervised manner.

5.1.4 Evaluation metrics

We evaluated our method using the keyshot-based metrics commonly used in recent works (Rochan et al., 2018; Rochan & Wang, 2019). Let $\mathbf{Y}$ and $\mathbf{Y}^*$ be the predicted keyshot summary and groundtruth summary created by multiple users, respectively. The precision (P) and recall (R) are defined as

$$\begin{aligned} P &= \frac{\text{overlap between } \mathbf{Y} \text{ and } \mathbf{Y}^*}{\text{total duration of } \mathbf{Y}}, \\ R &= \frac{\text{overlap between } \mathbf{Y} \text{ and } \mathbf{Y}^*}{\text{total duration of } \mathbf{Y}^*}. \end{aligned} \tag{26}$$

We compute the F-score to measure the quality of the summary with precision and recall:

$$F\text{-}score = \frac{2 \times P \times R}{P + R}. \tag{27}$$

For datasets with multiple groundtruth summaries, we follow the standard approaches (Zhang et al., 2016b; Gygli et al., 2014, 2015) to calculate the metrics for the videos.

We also evaluated our method using rank-based metrics, Kendall's τ (Kendall, 1945) and Spearman's ρ (Zwillinger & Kokoska, 1999) correlation coefficients, following (Otani et al., 2019). To compute the correlation coefficients, we first rank the video frames according to their probability of being a keyframe and the annotated importance scores. After that, we compare the generated ranking with each annotated ranking. The correlation scores are then computed by averaging the individual results.

5.2 Implementation Details

5.2.1 Network Architecture

The VideoGraph is composed of four main parts: a *feature extraction network* to extract the initial object node representations and the language query representation from the pretrained feature extractors; *spatial and temporal relation reasoning networks* that construct language-guided spatial and temporal graphs, propagate the object and frame nodes along spatial and temporal edges, and refine the spatial and temporal graphs by recursively estimating relative adjacency matrices; and a *summarization network* to predict keyframes.

Table 2 Network configuration of VideoGraph. 'Lan-emb.' denote linear layers to embed the sentence-level queries and 'Lan-emb.*' is for the word-level queries. 'MHCA-emb.' is a linear layer in the MHCA, 'G-conv*ks*' and 'G-conv*kt*' denote the graph convolutional layer in the SRR and TRR networks, respectively. 'Common Layers' are used in the feature extraction network and MHCA

Layer	Ch I/O	BN	Activation
Common Layers			
Obj-emb.	2048/1024	-	-
Lan-emb.	2048/1024	-	-
Lan-emb.*	$128 \times W/1024$	-	-
MHCA-emb.	1024/512	-	-
Graph Layer 1			
G-conv1s	512/256	✓	ELU(·)
G-conv1t	256/512	✓	ELU(·)
Graph Layer 2			
G-conv1s	512/256	✓	ELU(·)
G-conv2s	256/256	✓	ELU(·)
G-conv1t	256/256	✓	ELU(·)
G-conv2t	256/512	✓	ELU(·)
Graph Layer 3			
G-conv1s	512/256	✓	ELU(·)
G-conv2s	256/256	✓	ELU(·)
G-conv3s	256/256	✓	ELU(·)
G-conv1t	256/256	✓	ELU(·)
G-conv2t	256/256	✓	ELU(·)
G-conv3t	256/512	✓	ELU(·)
Graph Layer 4			
G-conv1s	512/256	✓	ELU(·)
G-conv2s	256/256	✓	ELU(·)
G-conv3s	256/256	✓	ELU(·)
G-conv4s	128/128	✓	ELU(·)
G-conv1t	128/128	✓	ELU(·)
G-conv2t	128/256	✓	ELU(·)
G-conv3t	256/256	✓	ELU(·)
G-conv4t	256/512	✓	ELU(·)
Summarization Layers			
Conv-sum1	512/256	✓	ReLU(·)
Conv-sum2	256/2	✓	Softmax(·)
Conv-recon1	512/2048	✓	ELU(·)
Conv-recon2	4096/2048	-	-

Although we use three graph convolution layers for the spatial and temporal graphs in each relation reasoning network, we can use fewer or more graph convolution layers. The overall configuration of VideoGraph corresponding to the number of graph convolution layers is shown in Table 2.

5.2.2 Feature extraction

Visual feature extraction. To train VideoGraph, we uniformly sampled frames for every video at 2 frames per second, as described in Zhang et al. (2016b). We fed the sampled frames into Faster R-CNN (Ren et al., 2015) that uses ResNet-101 (He et al., 2016) as the backbone with bottom-up attention (Anderson et al., 2018) pretrained on Visual Genome (Krishna et al., 2017). The objects in each frame are detected by the model and applied non-maximum suppression (Hosang et al., 2017) for each class with an intersection over union (IoU) threshold of 0.7. We select the top N region of interests (RoIs) with the highest detection confidence scores and extract 2048-dimensional activations as an object node feature representation $\mathbf{o}_t^n$ for each selected RoI after average pooling. Although the maximum number of objects N in each frame is set to 36, we evaluated the model primarily using the top 16 object feature representations based on the ablation study. Note that the object detector is frozen while training.

Language feature extraction. In generic video summarization, we generate two types of language queries, i.e., sentence-level queries and word-level queries. To obtain the sentence-level language queries, we use the bi-modal transformer dense video captioning model (Iashin & Rahtu, 2020), which generates a multiple sentence description, following (Narasimhan et al., 2021) for fair comparisons. We first embed each sentence of the description using the CLIP (Radford et al., 2021) text encoder and fuse them using the MLP layer. For the word-level language queries, we select W words from the detected object classes based on the detected frequency. We set W differently according to the dataset through the ablation study. Each word is embedded by the CLIP text encoder and the MLP layer and is fused using an additional MLP layer. In the case of the user-defined queries, we identically use the CLIP text encoder for each query and fuse all query embeddings using the MLP layer. As the same with the visual feature
extraction, the caption generator and pretrained CLIP encoder are frozen while training.

5.2.3 Training and optimization

Although VideoGraph can be trained independently of the lengths of the videos, we uniformly sampled frames with fixed temporal length $T = 320$ from each training video for TVSum and SumMe to learn the network parameters with a mini-batch size of 5. We train VideoGraph for 200 epochs with the Adam optimizer (Kingma & Ba, 2015). The learning rate is set to 10^{-4} and decayed by a factor of 0.1 every 80 epochs. We use four heads each in the MHCA of the SRR and TRR networks, the projected query, key, and value have 256 dimensions. The scaling factors λ_o and λ_f in (8) and (12) are set to 1.6 and 30, respectively. We set $\rho = 5$ in (21), $\alpha = 0.0001$, and $\beta = \gamma = 0.1$ in (24) for supervised learning, and $\alpha = 0.001$ and $\beta = \gamma = 10$ in (25) for unsupervised learning. The maximum number of iterations K is fixed at five throughout the ablation study (see Fig. 12). The quantitative performance for the experiments on TVSum and SumMe is evaluated five times on five random splits of the datasets and reported as the average values.

5.3 Quantitative Results

5.3.1 Evaluation on SumMe and TVSum datasets

We train the model for generic video summarization with three different levels of language queries, no language query, word-level language queries, and sentence-level language queries. In Table 3, we show our results in comparison to supervised methods (Zhang et al., 2016a, b; Mahasseni et al., 2017; Zhao et al., 2017; Zhang et al., 2018; Rochan et al., 2018; Rochan & Wang, 2019; He et al., 2019; Ji et al., 2020; Park et al., 2020; Ji et al., 2021; Zhu et al., 2020; Zhao et al., 2021; Jiang & Mu, 2022; Narasimhan et al., 2021) and unsupervised video summarization (Song et al., 2015; Mahasseni et al., 2017; Rochan et al., 2018; Yuan et al., 2019; Rochan & Wang, 2019; He et al., 2019; Park et al., 2020; Wu et al., 2021; Narasimhan et al., 2021) on both SumMe (Gygli et al., 2014) and TVSum (Song et al., 2015) datasets.

The comparison between the approaches that not use any language queries shows that our model achieves state-of-the-art performance of 54.7% on SumMe and 65.2% on TVSum in the standard and transfer settings. While the performance of Jiang and Mu (2022) in a supervised manner with the augment setting is slightly better than our method, we outperform (Jiang & Mu, 2022) in other settings (i.e., the standard and transfer settings). Especially, the result in the transfer setting proves the generalizability of the proposed method, showing 6.2% and 2.1% performance improvements on the SumMe and TVSum datasets, respectively. Our VideoGraph reduces the gap between the supervised and unsupervised approaches. It further improves the performance even without using the unpaired data used as additional information for the summary video in Rochan and Wang (2019).

The results of the model with language queries demonstrate that the language-guided approach consistently further improves the performance regardless of the types of queries. In addition, the comparisons with the state-of-the-art language-guided approach (Narasimhan et al., 2021) validate the effectiveness of the language-guided graph modeling, showing significant performance improvements with large margins in supervised and unsupervised manners. We observe that the model trained with the word-level query, surprisingly, outperforms the model with the sentence-level query on both datasets despite the simplicity of the word-level

Table 3 Comparison with state-of-the-art methods on the SumMe (Gygli et al., 2014) and TVSum (Song et al., 2015) datasets, with various data configurations including standard data, augmented data, and transfer data settings. † used partial supervision. We denote the usage of language query on the column of 'Language'

Supervision	Method	Language	SumMe			TVSum		
			Standard	Augment	Transfer	Standard	Augment	Transfer
Unsupervised	Mahasseni et al. (2017)	✗	39.1	43.4	-	51.7	59.5	-
	Rochan et al. (2018)(SUM-FCN$_{unsup}$)	✗	41.5	-	39.5	52.7	-	-
	Rochan et al.†(2019)	✗	-	48.0	41.6	-	56.1	55.7
	He et al. (2019)	✗	46.0	47.0	44.5	58.5	58.9	57.8
	Yuan et al. (2019)	✗	41.9	-	-	57.6	-	-
	Park et al. (2020)	✗	49.8	52.1	47.0	59.3	61.2	57.6
	Wu et al. (2021)	✗	23.2	-	-	58.0	-	-
	Li et al. (2022)	✗	50.6	51.7	46.4	60.9	61.9	58.9
	Narasimhan et al. (2021)	✓	52.5	54.7	50.0	63.0	65.7	62.8
	VideoGraph w/Video only	✗	51.2	54.0	48.5	61.1	62.3	58.7
	VideoGraph w/Word-level query	✓	**55.0**	**56.9**	**53.0**	**66.8**	**68.1**	**65.5**
	VideoGraph w/Sentence-level query	✓	53.4	55.3	52.9	65.6	67.2	64.7
Supervised	Zhang et al. (2016a)	✗	40.9	41.3	38.5	-	-	-
	Zhang et al. (2016b)	✗	38.6	42.9	41.8	54.7	59.6	58.7
	Mahasseni et al. (2017) (SUM-GAN$_{sup}$)	✗	41.7	43.6	-	56.3	61.2	-
	Zhao et al. (2017)	✗	44.3	-	-	62.1	-	-
	Rochan et al. (2018)(SUM-FCN)	✗	47.5	51.1	44.1	56.8	59.2	58.2
	Rochan et al. (2018)(SUM-DeepLab)	✗	48.8	50.2	45.0	58.4	59.1	57.4
	Zhou et al. (2018)	✗	42.1	43.9	42.6	58.1	59.8	58.9
	Zhang et al. (2018)	✗	44.9	-	-	63.9	-	-
	He et al. (2019)	✗	47.2	-	-	59.4	-	-
	Ji et al. (2020)	✗	44.4	46.1	-	61.0	61.8	-
	Zhao et al. (2020)	✗	42.5	-	-	58.4	-	-
	Park et al. (2020)	✗	51.4	52.9	48.7	63.9	65.8	60.5
	Ji et al. (2021)	✗	46.1	47.6	-	64.5	65.8	-
	Zhu et al. (2020)	✗	51.2	53.3	47.6	61.9	62.2	58.0
	Zhao et al. (2021)	✗	45.0	45.7	44.0	60.1	61.1	60.0
	Jiang and Mu (2022)	✗	54.5	56.9	49.2	63.4	64.2	59.8

Table 3 continued

Supervision	Method	Language	SumMe			TVSum		
			Standard	Augment	Transfer	Standard	Augment	Transfer
	Narasimhan et al. (2021)	✓	54.2	56.4	51.9	66.3	69.0	65.5
	Li et al. (2023a)	✓	50.7	-	-	60.4	-	-
	VideoGraph w/Video only	✗	54.7	55.6	55.4	65.2	67.3	61.9
	VideoGraph w/Word-level query	✓	**58.1**	**59.4**	**55.6**	**69.5**	**71.4**	**67.7**
	VideoGraph w/Sentence-level query	✓	57.0	58.6	54.3	68.4	70.7	67.1

Table 4 Kendall's τ (Kendall, 1945) and Spearman's ρ (Zwillinger & Kokoska, 1999) correlation coefficients computed by VideoGraph with the word-level query and state-of-the-art approaches on the TVSum benchmark (Song et al., 2015). We report the types of the groundtruth for training, i.e., binary (B) and frame score (S) labels

Method	Label	τ	ρ
Human	-	0.177	0.204
Groundtruth	S	0.364	0.456
Zhang et al. (2016b)	B	0.042	0.055
Zhou et al. (2018)	-	0.020	0.026
Park et al. (2020)	B	0.094	0.138
Narasimhan et al. (2021)	B	0.108	0.147
Zhao et al. (2020)	S	0.082	0.088
Zhao et al. (2021)	S	0.083	0.090
Li et al. (2022)	S	0.097	0.105
Li et al. (2023a)	S	0.181	0.238
VideoGraph	B	0.162	0.188
VideoGraph	S	**0.204**	**0.242**

query. We insist on this due to a characteristic of the datasets: the SumMe and TVSum datasets consist of short videos that contain no complicated themes without multiple events so that the simple keywords can more suitably represent the whole video than the multiple complicated sentences.

Table 4 compares the correlation coefficients of the proposed VideoGraph, deepLSTM (Zhang et al., 2016b), DR-DSN (Zhou et al., 2018), humans reported in Otani et al. (2019), SumGraph (Park et al., 2020), HSA-RNN (Zhao et al., 2020), RSGN (Zhao et al., 2021), VJMHT (Li et al., 2022), and SSPVS (Li et al., 2023a). To fairly compare the performance, we report the performance corresponding to the groundtruth types, i.e., binary labels with (18) and score labels with (19), respectively. VideoGraph trained with the binary labels outperforms the several baselines (except for Li et al. (2023a)) by large margins despite the coarsely annotated labels. The importance score labels significantly improve the performance of VideoGraph, outperforming the previous best performance (Li et al., 2023a) by 0.023 for Kendall's τ and 0.004 for Spearman's ρ.

5.3.2 Evaluation on QFVS dataset

We evaluate the performance on the QFVS (Sharghi et al., 2017) dataset for generic and query-focused video summarization. In Table 5, we report the generic video summarization performance and compare our model with state-of-the-art supervised (Gygli et al., 2015; Sharghi et al., 2016; Narasimhan et al., 2021) and unsupervised (Zhao & Xing, 2014; Narasimhan et al., 2021) approaches. The results demonstrate that our VideoGraph consistently outperforms the prior works in supervised and unsupervised manners. Specifically, our model with a sentence-level query can be fairly compared with Narasimhan et al. (2021) trained with the generated captions (Gen. Caption). The proposed method outperforms (Narasimhan et al., 2021) by 5.37% and 5.47% in the average performances in supervised and unsupervised settings, respectively. The comparisons between the performance of our model trained with word-level and sentence-level queries validate the effectiveness of sentence-level queries for generic summaries of long videos. Since videos in the QFVS dataset contain multiple events with long running times, describing the videos using multiple captions is more effective than using the set of keywords. With the groundtruth captions obtained from VideoSet (Yeung et al., 2014), our model achieves state-of-the-art performances throughout all videos.

Table 6 presents the results of query-focused video summarization for baselines (Gong et al., 2014; Sharghi et al., 2016, 2017; Narasimhan et al., 2021) and ours. Note that we train the model only in a supervised manner with the groundtruth queries provided from the dataset. Similar to generic video summarization, our model shows superior per-

Table 5 Comparison with state-of-the-art methods on the QFVS (Sharghi et al., 2017) dataset for generic video summarization in terms of F-score

Supervision	Method	Vid 1	Vid 2	Vid 3	Vid 4	Avg.
Unsupervised	Zhao and Xing (2014)	53.06	53.80	49.91	22.31	44.77
	Narasimhan et al. (2021) (Gen. Caption)	67.02	59.48	66.43	44.19	59.28
	Narasimhan et al. (2021) (GT Caption)	73.90	66.83	75.44	52.31	67.12
	VideoGraph w/Word-level query	69.03	62.11	68.93	48.60	62.17
	VideoGraph w/Sentence-level query	71.94	65.08	71.25	50.71	64.75
	VideoGraph w/GT. Caption	**76.34**	**69.52**	**76.86**	**55.29**	**69.50**
Supervised	Gygli et al. (2015)	49.51	51.03	64.52	35.82	50.22
	Sharghi et al. (2017)	62.66	46.11	58.85	33.50	50.29
	Narasimhan et al. (2021) (Gen. Caption)	74.13	63.44	75.86	50.23	65.92
	Narasimhan et al. (2021) (GT Caption)	84.98	71.26	82.55	61.46	75.06
	VideoGraph w/Word-level query	76.78	67.18	77.24	53.80	68.75
	VideoGraph w/Sentence-level query	79.55	69.86	78.30	57.43	71.29
	VideoGraph w/GT. Caption	**87.02**	**74.62**	**85.47**	**64.19**	**77.83**

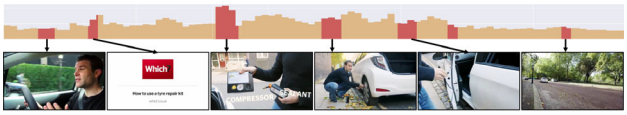
Table 6 Comparison with state-of-the-art methods on the QFVS (Sharghi et al., 2017) dataset for query-focused video summarization in terms of F-score

Method	Vid 1	Vid 2	Vid 3	Vid 4	Avg.
Gong et al. (2014)	36.59	43.67	25.26	18.15	30.92
Sharghi et al. (2016)	35.67	42.74	36.51	18.62	33.38
Sharghi et al. (2017)	48.68	41.66	56.47	29.96	44.19
Narasimhan et al. (2021)	57.13	53.60	66.08	41.41	54.55
VideoGraph	**60.21**	**57.50**	**68.71**	**45.38**	**57.95**

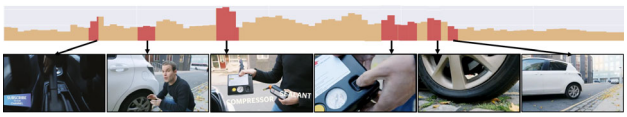
formance, outperforming the strongest baseline (Narasimhan et al., 2021) by 3.4% on the average performance.

5.4 Qualitative Results

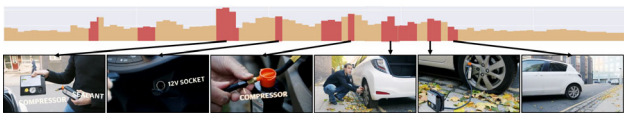
In Fig. 8, we provide qualitative results for the challenging video comprising visually similar frames. To visualize the summaries, we sampled frames from the keyshots selected by Park et al. (2020), VideoGraph without the language queries, and with the word-level language queries. The main differences between the first two methods are the use of fine-grained frame features (i.e., object feature representations) and the SRR network. The summary generated by VideoGraph without the language queries captures almost all of the peak regions of the groundtruth scores, and the performance is also improved by 6.8% in terms of F-score on the example video. This observation indicates that leveraging fine-grained frame features and considering object relationships enable more informative and meaningful summaries to be generated. The word-level language queries with VideoGraph derive an additional 8.1% performance improvement, demonstrating the effectiveness of the language-guided video summarization framework.



(a) SumGraph [Park et al., 2020] (F-Score: 50.3)



(b) VideoGraph w/Video only (F-Score: 62.1)



Word-level queries: *Man, Passenger Car, Hand, Tire, Tires, Roadway, Socket, Words*

(c) VideoGraph w/Word-level query (F-Score: 70.2)

Fig. 8 Qualitative results on Video 2 from the TVSum benchmark (Song et al., 2015): (a) SumGraph (Park et al., 2020), (b) VideoGraph without the language query, and (c) VideoGraph with the word-level language queries. Brown bars show frame-level user annotations and red bars are selected subset shots for the summary video. Best viewed in color

In Fig. 9, we visualize the generic and query-focused summarization results on Video 1 from the QFVS dataset. The input video is about four hours long and contains multiple events related with several language queries. As shown in Fig. 9-(b), the generic summarization, which is generated with the word-level language queries, contains various scenes that represent the whole video. Fig. 9-(c) and (d) show that different summaries can be yield from the same video for the given different queries “Sky and Tree" and “Food and Drink".

In video summarization, the importance of each object contained in a frame is not defined. We hypothesize that



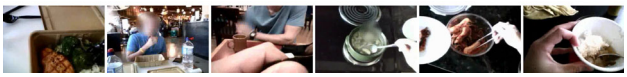
(a) Input Video from QFVS (P01)



(b) Generic Summarization



(c) Query-focused Summarization (Sky, Tree)



(d) Query-focused Summarization (Food, Drink)

Fig. 9 Qualitative results on the QFVS benchmark (Sharghi et al., 2017): (a) input video, (b) generic summarization, (c) query-focused summarization with the queries "Sky and Tree", and (d) query-focused summarization with the queries "Food and Drink". We sample frames from the generated summaries

dominant objects contributing to the final summary video are highly correlated with large edge weights as object nodes propagate their information through the edges of the spatial graph in VideoGraph. Consequently, we define the dominance[1] of each object as the normalized sum of its edge weights, representing the degree to which an object interacts with other objects within a frame. Specifically, we first aggregate the n-th object's edge weights:

$$\mathbf{d}_t = \{d_{t,n} | d_{t,n} = \sum_{i=1, i\neq n}^{N} s_{t,ni}^{K}, \quad n = 1, ..., N\}, \tag{28}$$

where $s_{t,ni}^{K}$ is the edge weight between the n-th and i-th object nodes in the *final* spatial graph of the t-th frame (i.e., S_t^K in Equation 16 of the main paper). The dominance score $\bar{\mathbf{d}}_t$ can be computed by applying the min-max normalization to $\mathbf{d}_t$, such that,

$$\bar{\mathbf{d}}_t = \frac{\mathbf{d}_t - \min \mathbf{d}_t}{\max \mathbf{d}_t - \min \mathbf{d}_t}. \tag{29}$$

We visualize the dominance score for objects within selected keyframes in Fig. 10. The results show that objects pivotal to the video's story tend to have higher dominance scores, as they are more strongly connected to other objects. This observation supports our hypothesis and suggests that VideoGraph effectively identifies and emphasizes information from key objects through the graph structure, enhancing summarization quality without manually assigning dominance scores.

[1] We use the term "dominance" instead of importance since "importance score" is a predefined term that represents the importance of each frame in a video summarization task.

Table 7 Ablation study for each component in VideoGraph on the SumMe (Gygli et al., 2014), TVSum (Song et al., 2015), and QFVS (Sharghi et al., 2017) datasets. We report the F-score

Graph Topology		Language	SumMe	TVSum	QFVS
Temporal	Spatial				
✓	✗	✗	51.4	63.9	-
✓	✗	✓	55.8	67.1	67.27
✗	✓	✗	45.2	55.6	-
✗	✓	✓	47.6	58.9	57.33
✓	✓	✗	54.7	65.2	-
✓	✓	✓	**58.1**	**69.5**	**71.29**

Table 8 Ablation study for the various combinations of loss functions in VideoGraph on the TVSum benchmark (Song et al., 2015)

Classification	Sparsity	Diversity + Reconstruction	F-Score
✓			65.4
✓	✓		67.8
✓		✓	67.5
✓	✓	✓	**69.5**

5.5 Ablation Study

5.5.1 Component analysis

We analyze the contribution corresponding to the graph topology and language query of the proposed method on the SumMe, TVSum, and QFVS datasets, as shown in Table 7. Note that the first row of the table corresponds to the SumGraph (Park et al., 2020). The results show that the language query (i.e., sentence-level query) improves the performance of the SumGraph on all three datasets. On the other hand, the spatial graph topology without the temporal graph significantly degrades the performance, demonstrating the lack of temporal modeling capacity of the spatial graph topology. While the language query improves the performance even with the spatial graph, the performance is still insufficient compared to the model with the temporal graph. The full model of VideoGraph achieves significant performance improvement, consistently outperforming other baselines by large margins on three datasets.

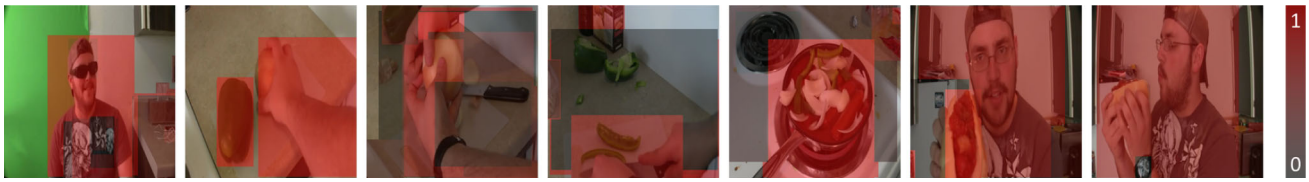


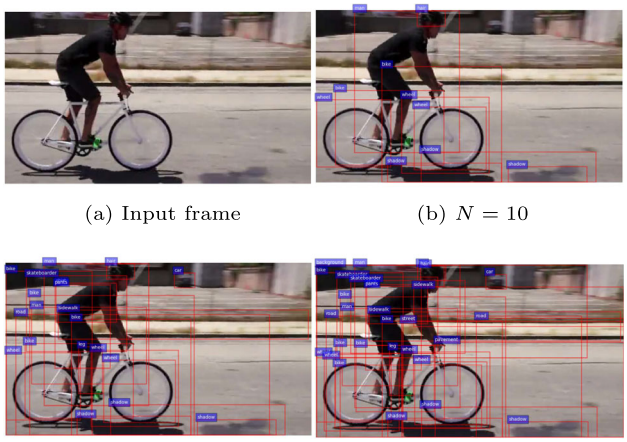
Fig. 10 Visualization for the dominance score of detected objects in the selected keyframes

5.5.2 Loss functions

We also investigate the effectiveness of the loss functions. The four loss terms, (i.e., classification, sparsity, reconstruction, and diversity losses), are employed for supervised learning. While the reconstruction loss forces the reconstructed features to be similar to the original features, the diversity loss enforces the diversity of the reconstructed features. Therefore, we analyze the effectiveness of the reconstruction and diversity losses together. As shown in Table 8, the sparsity loss and the sum of the diversity and reconstruction losses improve the performance by 2.4% and 2.1%, respectively, on the TVSum benchmark. The full use of loss functions contributes to a performance improvement of 4.1%.

5.5.3 Number of objects

We evaluated our model according to the number of objects used in the spatial graph. Following the original method (Anderson et al., 2018), we extract the top 36 confident region features in each frame and select N objects in the order of the confidence score. The value of N is set to 1, 2, 4, 8, 16, 20, and 36. Note that, when N is 1, the SRR network is not used so that the network structure is the same as in our previous work (Park et al., 2020). The only difference is that the input feature is the most salient region feature, not the GoogleNet (Szegedy et al., 2015) feature for the entire frame. As shown in Table 9, the performance increases as N increases and decreases as N becomes greater than 16. Not surprisingly, the performance of $N = 1$, 2, and 4 shows lower performance than our previous work (Park et al., 2020) since a small value of N does not include the entire frame information. On the other hand, a large value of N significantly degrades the summarization performance (e.g., $N = 20$ and 36). Since the feature extractor (Anderson et al., 2018) is pretrained on the Visual Genome (Krishna et al., 2017) dataset containing a relatively large number of annotations per image, the features for many objects might contain redundant and noisy information. To validate this, we depict the detection results according to N for the video frame from the TVSum (Song et al., 2015) dataset in Fig. 11. Based on this result, subsequent ablation studies are investigated by setting the number of objects N to 16 with the standard supervision configuration.



(a) Input frame (b) $N = 10$

(c) $N = 20$ (d) $N = 30$

Fig. 11 (a) Input frame and (b)-(d) detected objects with respect to the number of objects N. Noisy and redundant objects are detected as N increases

Table 9 Performance comparison evaluated on the TVSum benchmark (Song et al., 2015) corresponding to the number of objects used in the SRR network

# objects	Precision	Recall	F-score
1	60.7	59.7	60.2
2	61.7	60.8	61.2
4	65.7	64.8	65.2
8	68.5	65.3	66.9
16	**71.3**	**67.8**	**69.5**
20	70.7	66.4	68.5
36	64.1	64.4	66.2

5.5.4 Number of words

We analyze the effect according to the number of words, W, used as word-level language queries by measuring the generic video summarization performance on the SumMe, TVSum, and QFVS datasets. As shown in Table 10, the performance consistently increases until the number of words is 8 on the SumMe and TVSum datasets and decreases after that. The results validate that a small number of keywords can describe

the videos in datasets and a larger number of language queries provide noisy information.

Meanwhile, we attain the best performance on QFVS with a much larger number of language queries (i.e., $W = 25$) than SumMe and TVSum, implying that a large number of keywords are required to describe long videos containing multiple events. In the main experiments, we thus use 8 words on SumMe and TVSum, and 25 words on QFVS, respectively.

Table 10 Ablation study for the number of the word-level language queries in VideoGraph on the SumMe (Gygli et al., 2014), TVSum (Song et al., 2015), and QFVS (Sharghi et al., 2017) benchmarks

Dataset	# of words						
	1	2	4	8	10	25	30
SumMe	54.6	55.8	57.2	**58.1**	57.8	56.1	55.9
TVSum	65.4	66.3	67.9	**69.5**	68.7	68.2	68.0
QFVS	65.1	65.4	66.4	67.9	68.3	**68.8**	68.5

5.5.5 Number of iterations

To validate the effectiveness of the recursive graph refinement procedure used in VideoGraph, we examine the F-score corresponding to the number of iterations, as shown in Fig. 12. Specifically, VideoGraph with the fixed initial graph (iteration 0) shows the performance of 49.8% and 59.6% in terms of F-score on the SumMe and TVSum datasets, respectively. The performance is gradually improved as the number of iterations increases, achieving 58.1% and 69.5% with 5 iterations on two datasets, and converging in three to five iterations. Fig. 13 illustrates the groundtruth importance scores and keyframes selected by VideoGraph corresponding to the number of iterations. The selected keyframes are marked as red bars and the backgrounds are indicated as brown bars. The results show that VideoGraph selects frames with higher importance scores as the number of iterations increases.

We further analyze the importance scores predicted by VideoGraph at each iteration, as shown in Fig. 14. The results emphasize the importance of our graph refinement mechanism for effectively capturing semantic relationships between frames. Since the predictions in the initial graph (yellow lines) primarily rely on visual similarity, importance scores are relatively uniform across frames. However, as refinement iterations progress, the graph learns to establish meaningful semantic connections (i.e., discrimination for keyframes from non-keyframes).

5.5.6 Memory efficiency and computational cost

To investigate the efficiency of our model, we compare the total number of parameters, number of learnable parameters, runtime corresponding to each process for inference, and performance evaluated on SumMe (Gygli et al., 2014) with the CNNs-based (Rochan et al., 2018; Park et al., 2020) and transformer-based methods (Narasimhan et al., 2021; Park et al., 2022) in Table 11. We measure runtimes using a single 24GB TITAN RTX GPU.

We repeat the inference 300 times for all videos, which have 320 frames, and record the average value as runtime. We follow the procedure of the official implementation of BMT (Iashin & Rahtu, 2020)[2] to generate video captions. For GoogleNet (Szegedy et al., 2015) and Faster R-CNN (Ren et al., 2015), we use open-source models provided by torchvision library and Detectron2 repository[3], respectively. For CLIP image and text encoders (Radford et al., 2021), we use the CLIP ViT-B/32 pretrained on LAION-400M (Schuhmann et al., 2021)[4].

For the baseline network, we used 16 object features, three graph convolution layers each for the SRR and TRR networks, and two graph convolution layers for the summarization network. The number of iterations for graph refinement is set to five.

For only summarization with pre-extracted features. For inference, while (Rochan et al., 2018) requires approximately 84ms using about 116.5M network parameters for a video containing 320 frames, our model without the language query takes 11ms using about 4.7M network parameters under the same settings. This result indicates that VideoGraph surpasses (Rochan et al., 2018) in terms of memory efficiency and summarization performance. Our model uses 0.8M fewer parameters and takes 2ms less time than (Park et al., 2020), with 2.3% and 1.3% performance improvement on the SumMe and TVSum datasets, respectively, demonstrating the effectiveness of learning fine-grained relationships between frames.

On the other hand, a large increase in the size of backbone networks is remarkable in comparison with the transformer-based methods (Narasimhan et al., 2021; Park et al., 2022). Specifically, they employed the CLIP ViT-B/32 image and text encoders (Radford et al., 2021), which have 85.8M and 63.0M parameters, for visual and language representations, respectively. For caption generation, they identically used the bi-modal transformer (BMT) (Iashin & Rahtu, 2020). Our VideoGraph achieves better performance than (Narasimhan et al., 2021), showing compatible runtime and the number of learnable parameters.

[2] Official code: https://github.com/v-iashin/BMT

[3] Official code: https://github.com/facebookresearch/detectron2

[4] Official code: https://huggingface.co/docs/transformers/model_doc/clip

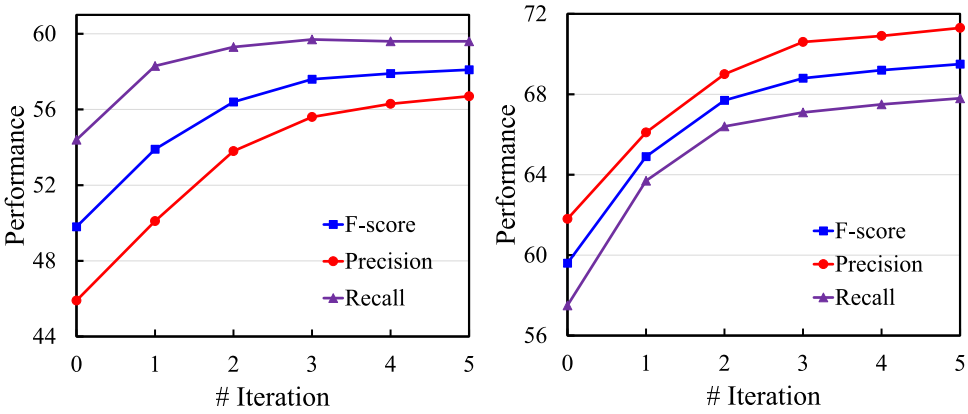


Fig. 12 Convergence analysis of VideoGraph with respect to the number of iterations on (a) SumMe benchmark (Gygli et al., 2014) and (b) TVSum (Song et al., 2015) benchmarks

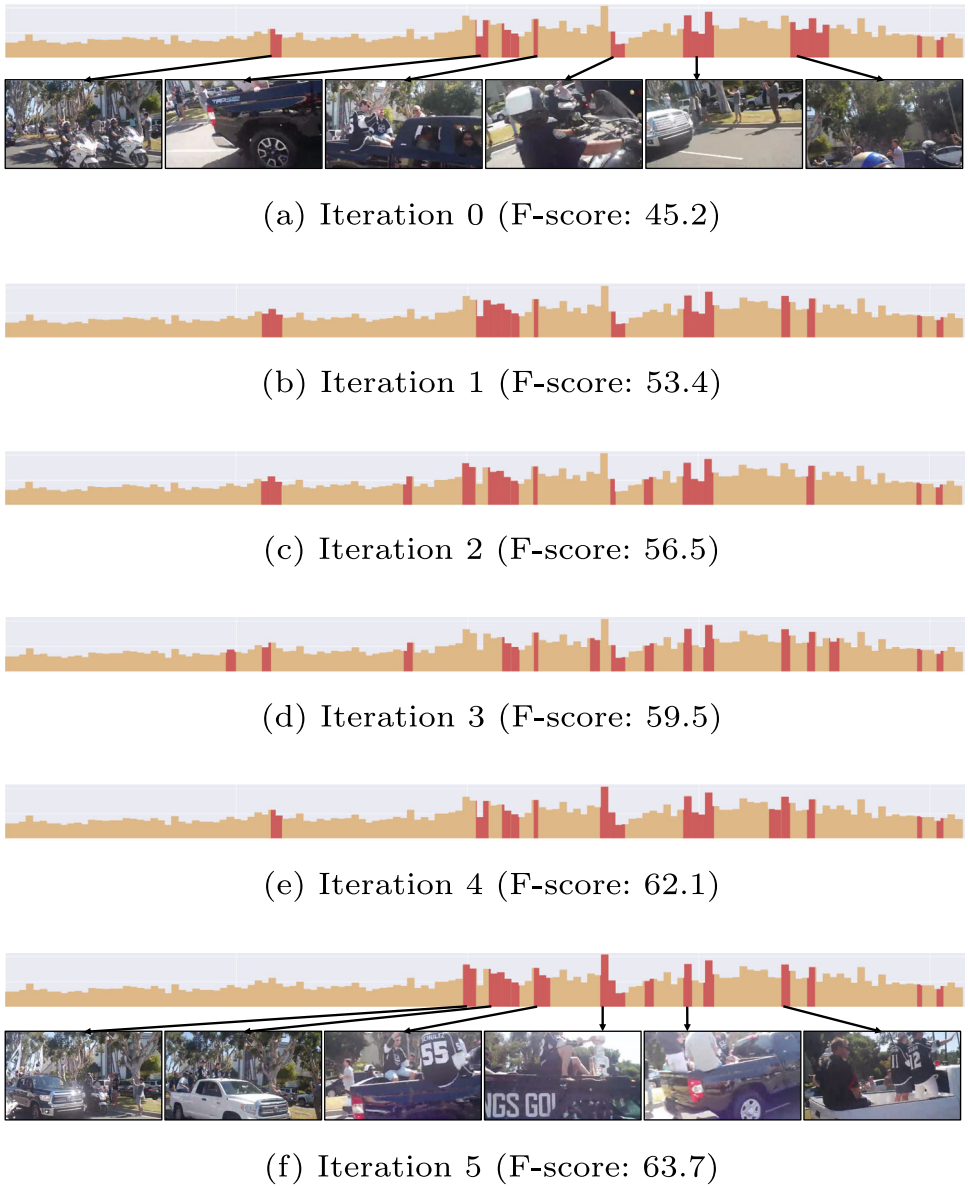


Fig. 13 Qualitative results for Video 28 on the TVSum benchmark (Song et al., 2015) with respect to the number of iterations. Brown bars show frame-level user annotations, and red bars are selected subset shots for the summary

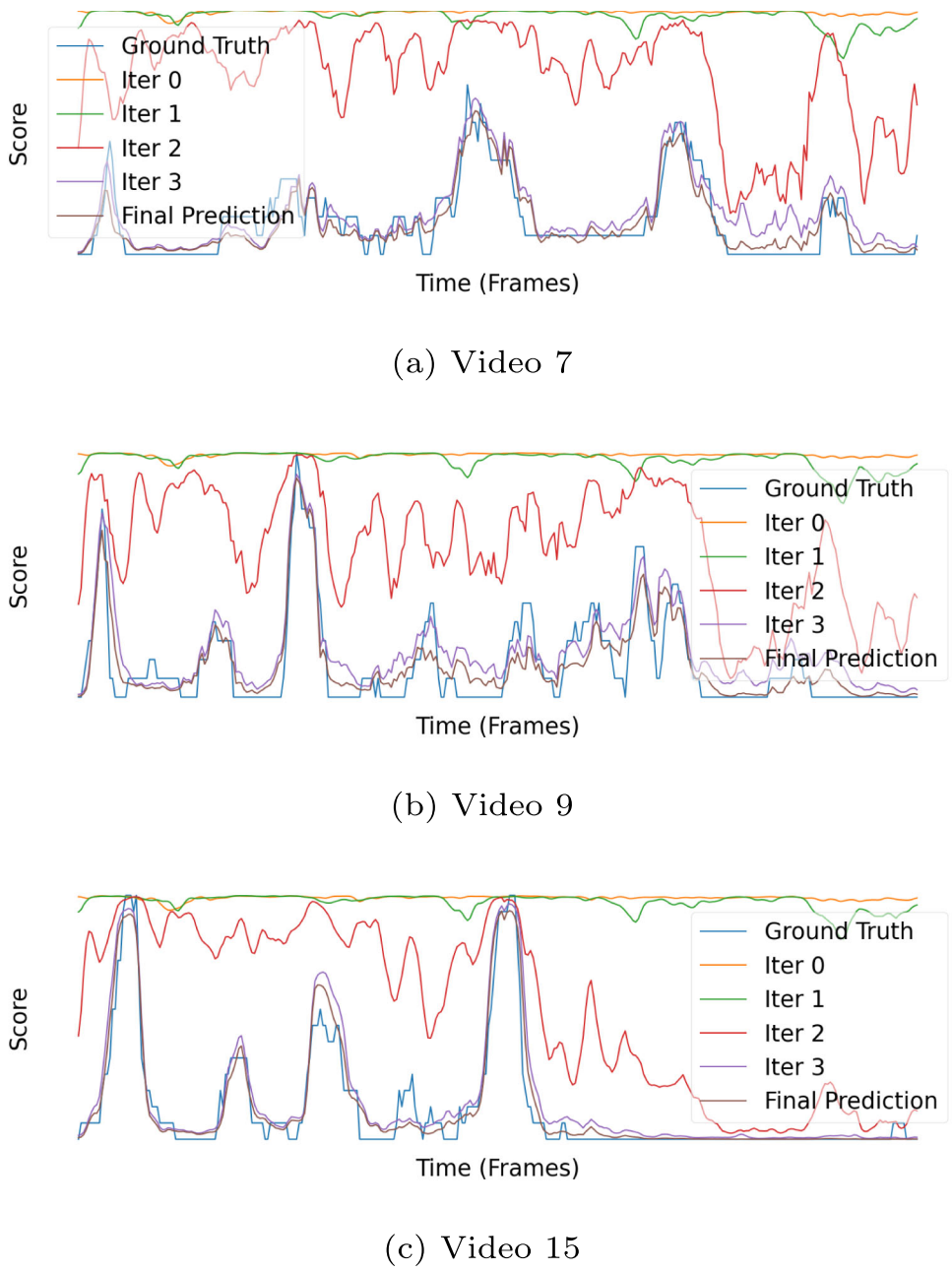


Fig. 14 Predicted importance scores according to refinement iteration for videos from the TVSum benchmark (Song et al., 2015)

Table 11 Use of language queries (Lan.) and captioning model (Cap. model), total number of parameters (Total params.), number of learnable parameters (Learn. params.), and runtimes for caption generation (Cap.), feature extraction (FE), and summarization (Sum.)

Method	Lang.	Cap. model	Total params.	Learn. params.	Runtime			SumMe (F-score)
					Cap.	FE	Sum.	
Rochan et al. (2018)	✗	✗	123.1M	116.5M	-	129ms	84ms	47.5
Park et al. (2020)	✗	✗	12.1M	5.5M	-	129ms	13ms	51.4
Narasimhan et al. (2021)	✓	✓	205.6M	5.8M	250s	317ms	23ms	54.2
Park et al. (2022) w/o Audio	✓	✓	280.1M	31.7M	250s	1057ms	69ms	54.2
VideoGraph w/Video only	✗	✗	49.3M	4.7M	-	474ms	11ms	54.7
VideoGraph w/Sentence	✓	✓	176.3M	17.7M	250s	485ms	32ms	57.0
VideoGraph w/Word	✓	✗	124.3M	16.7M	-	484ms	31ms	**58.1**

Park et al. (2022) additionally exploited the VideoCLIP (Xu et al., 2021) for visual representations, and they learn a large number of trainable parameters, showing a low memory efficiency and taking about 2.2x more runtime than our VideoGraph for inference.

The comparison between our three baselines shows that most of the computation burden is concentrated on the transformers. Although the model with the language query takes about 3× more parameters and runtime than the model without the language query, it is worth noting that considerable performance improvements and customized summaries can be obtained with the language-guided model.

For the whole inference from scratch. As shown in Table 11, we notice that most of the time for inference is concentrated on caption generation, taking about 4 minutes on average. In this regard, it is worth noting that VideoGraph with word-level queries incorporates language information into video summarization without caption generation and achieves even better performance. Although visual feature extraction is the second longest part, it takes relatively less time than caption generation. In addition, GoogleNet, which is the lightest visual encoder, takes 129ms for a video, and the transformer-based backbones, i.e., CLIP (Radford et al., 2021) and VideoCLIP (Xu et al., 2021), require 306ms and 740ms, respectively. The Faster R-CNN (Ren et al., 2015) used in our VideoGraph takes 474ms for the video, which is comparable to other encoders. Consequently, VideoGraph with word-level language queries significantly outperforms the previous works while mitigating the burden of caption generation.

6 Conclusion and Future Work

In this paper, we proposed VideoGraph, which formulates video summarization as a language-guided graph modeling problem. Contrary to previous works that employed global visual representations and sentence-level language queries, we leverage fine-grained object representations and their corresponding word-level language queries, ensuring consistent information between visual and language data.

VideoGraph constructs a language-guided spatiotemporal graph through spatial and temporal relation networks, and recursively refines the graph to enable graph nodes to be gradually connected based on semantic relationships.

We demonstrated the effectiveness of each component in VideoGraph, including graph topology, types of language queries, loss functions, and graph refinement, and achieved state-of-the-art performance in both the supervised and unsupervised settings.

There are many possible directions for future work. While we achieved a new state-of-the-art on several datasets, we use off-the-shelf pretrained models (Iashin & Rahtu, 2020; Ren et al., 2015; Radford et al., 2021) to generate and encode the language queries. This could allow inappropriate biases from the pretrained models to be propagated to our model. Moreover, the pretraining dataset of the object detector for the word-level language queries should also be carefully regarded to generate more specific summaries. In this regard, our further works will be focused on generalizing the proposed approach to deal with robust summarization performance on language queries generated from pretrained models.

Acknowledgements This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS- 2025-00515741 and No. RS-2025-02216328) and the Yonsei Signature Research Cluster Program of 2024 (2024-22-0161).

Data Availability The SumMe and QFVS datasets were publicly available on https://gyglim.github.io/me/vsum/index.html and https://www.aidean-sharghi.com/cvpr2017, respectively. However, we found that they are inaccessible now so we will provide pre-extracted features upon reasonable request. The TVSum dataset is publicly available on https://github.com/yalesong/tvsum.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., & Zhang, L. (2018). Bottom-up and top-down attention for image captioning and visual question answering. *IEEE Conf. Comput. Vis. Pattern Recog.*, 6077–6086.

Bhattacharya, U., Mittal, T., Chandra, R., Randhavane, T., Bera, A., & Manocha, D. (2020). Step: Spatial temporal graph convolutional networks for emotion perception from gaits. **AAAI Conf. Art. Intell.**, 1342–1350.

Carreira, J., Agrawal, P., Fragkiadaki, K., & Malik, J. (2016). Human pose estimation with iterative error feedback. *IEEE Conf. Comput. Vis. Pattern Recog.*, 4733–4742.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell., 40*(4), 834–848.

De Avila, S. E. F., Lopes, A. P. B., Luz, A., Jr., & Albuquerque Araújo, A. (2011). Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Patt. Rec. Letters, 32*(1), 56–68.

Gao, K., Chen, L., Niu, Y., Shao, J., & Xiao, J. (2022). Classification-then-grounding: Reformulating video scene graphs as temporal bipartite graphs. *IEEE Conf. Comput. Vis. Pattern Recog.*, 19497–19506.

Gao, J., Zhang, T., & Xu, C. (2019). Graph convolutional tracking. *IEEE Conf. Comput. Vis. Pattern Recog.*, 4649–4659.
Gong, B., Chao, W.-L., Grauman, K., & Sha, F. (2014). Diverse sequential subset selection for supervised video summarization. *Adv. Neural Inform. Process. Syst.*, 2069–2077.
Grandvalet, Y., & Bengio, Y. (2004). Semi-supervised learning by entropy minimization. *Adv. Neural Inform. Process. Syst.*, 529–536.
Gygli, M., Grabner, H., & Gool, L. V. (2015). Video summarization by learning submodular mixtures of objectives. *IEEE Conf. Comput. Vis. Pattern Recog.*, 3090–3098.
Gygli, M., Grabner, H., Riemenschneider, H., & Gool, L. V. (2014). Creating summaries from user videos. *Eur. Conf. Comput. Vis*.
He, X., Hua, Y., Song, T., Zhang, Z., Xue, Z., Ma, R., Robertson, N., & Guan, H. (2019). Unsupervised video summarization with attentive conditional generative adversarial networks. *ACM Int. Conf. Multimedia*, 2296–2304.
He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *IEEE Conf. Comput. Vis. Pattern Recog.*, 770–778.
Hosang, J., Benenson, R., & Schiele, B. (2017). Learning non-maximum suppression. *IEEE Conf. Comput. Vis. Pattern Recog.*, 4507–4515.
Iashin, V., & Rahtu, E. (2020). A better use of audio-visual cues: Dense video captioning with bi-modal transformer. *Brit. Mach. Vis. Conf.*
Jain, A., Zamir, A. R., Savarese, S., & Saxena, A. (2016). Structural-rnn: Deep learning on spatio-temporal graphs. *IEEE Conf. Comput. Vis. Pattern Recog.*, 5308–5317.
Jiang, H., & Mu, Y. (2022). Joint video summarization and moment localization by cross-task sample transfer. *IEEE Conf. Comput. Vis. Pattern Recog.*, 16388–16398.
Jiang, B., Zhang, Z., Lin, D., Tang, J., & Luo, B. (2019). Semi-supervised learning with graph learning-convolutional networks. *IEEE Conf. Comput. Vis. Pattern Recog.*, 11313–11320.
Ji, Z., Xiong, K., Pang, Y., & Li, X. (2020). Video summarization with attention-based encoder-decoder networks. *IEEE Trans. Circuit Syst. Vid. Tech., 30*(6), 1709–1717.
Ji, Z., Zhao, Y., Pang, Y., Li, X., & Han, J. (2021). Deep attentive video summarization with distribution consistency learning. *IEEE Trans. Neural Net. and Learn. Syst., 32*(4), 1765–1775.
Joshi, N., Kienzle, W., Toelle, M., Uyttendaele, M., & Cohen, M. F. (2015). Real-time hyperlapse creation via optimal frame selection. *ACM Trans. on Graphics, 34*(4), 1–9.
Kang, H.-W., Matsushita, Y., Tang, X., & Chen, X.-Q. (2006). Space-time video montage. *IEEE Conf. Comput. Vis. Pattern Recog.*, 1331–1338.
Kaseris, M., Mademlis, I., & Pitas, I. (2022). Exploiting caption diversity for unsupervised video summarization. *ICASSP*, 6519–6527.
Kendall, M. G. (1945). The treatment of ties in ranking problems. *Biometrika, 33*(3), 239–251.
Kingma, D.P., & Ba, J. (2015). Adam: A method for stochastic optimization. *Int. Conf. Learn. Represent*.
Kipf, T.N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. Int. Conf. Learn. Represent.
Krähenbühl, P., & Koltun, V. (2011). Efficient inference in fully connected crfs with gaussian edge potentials. *Adv. Neural Inform. Process. Syst.*, 109–117.
Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.-J., Shamma, D. A., Bernstein, M. S., & Fei-Fei, L. (2017). Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vis.*, 32–73.
Lafferty, J. D., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Int. Conf. Mach. Learn.*, 282–289.
Lee, Y.J., Ghosh, J., & Grauman, K. (2012). Discovering important people and objects for egocentric video summarization. *IEEE Conf. Comput. Vis. Pattern Recog*.
Li, J., Gao, X., & Jiang, T. (2020). Graph networks for multiple object tracking. *Winter Applications Comput. Vis.*, 719–728.
Li, H., Ke, Q., Gong, M., & Drummond, T. (2023). Progressive video summarization via multimodal self-supervised learning. *Winter Applications Comput. Vis.*, 5584–5593.
Li, H., Ke, Q., Gong, M., & Zhang, R. (2022). Video joint modelling based on hierarchical transformer for co-summarization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 1–14.
Li, M., Wang, H., Zhang, W., Miao, J., Zhao, Z., Zhang, S., Ji, W., & Wu, F. (2023). Winner: Weakly-supervised hierarchical decomposition and alignment for spatio-temporal video grounding. *IEEE Conf. Comput. Vis. Pattern Recog.*, 23090–23099.
Liu, T., & Kender, J. R. (2002). Optimization algorithms for the selection of key frame sequences of variable length. *Eur. Conf. Comput. Vis.*, 403–417.
Liu, C., Mao, Z., Zhang, T., Xie, H., Wang, B., & Zhang, Y. (2020). Graph structured network for image-text matching. *IEEE Conf. Comput. Vis. Pattern Recog.*, 10921–10930.
Lu, Z., & Grauman, K. (2013). Story-driven summarization for egocentric video. *IEEE Conf. Comput. Vis. Pattern Recog*.
Mahasseni, B., Lam, M., & Todorovic, S. (2017). Unsupervised video summarization with adversarial lstm networks. *IEEE Conf. Comput. Vis. Pattern Recog.*, 202–211.
Mukhoty, B., Gopakumar, G., Jain, P., & Kar, P. (2019). Globally-convergent iteratively reweighted least squares for robust regression problems. *Proceedings of Machine Learning Research, 89*, 313–322.
Narasimhan, M., Rohrbach, A., & Darrell, T. (2021). Clip-it! language-guided video summarization. *Adv. Neural Inform. Process. Syst.*, 13988–14000.
Nassar, A.S., D Aronco, S., Lefèvre, S., & Wegner, J.D. (2020). Geograph: Graph-based multi-view object detection with geometric cues end-to-end. **Eur. Conf. Comput. Vis.**, 488–504.
Ngo, C.-W., Ma, Y.-F., & Zhang, H.-J. (2003). Automatic video summarization by graph modeling. *IEEE Int. Conf. Comput. Vis*.
Open video project. https://open-video.org/.
Otani, M., Nakashima, Y., Rahtu, E., & Heikkilä, J. (2019). Rethinking the evaluation of video summaries. *IEEE Conf. Comput. Vis. Pattern Recog.*, 7596–7604.
Pan, B., Cai, H., Huang, D.-A., Lee, K.-H., Gaidon, A., Adeli, E., & Niebles, J. C. (2020). Spatio-temporal graph for video captioning with knowledge distillation. *IEEE Conf. Comput. Vis. Pattern Recog.*, 10870–10879.
Park, J., Kwoun, K., Lee, C., & Lim, H. (2022). Multimodal frame-scoring transformer for video summarization. arXiv preprint arXiv:2207.01814.
Park, J., Lee, J., & Sohn, K. (2021). Bridge to answer: Structure-aware graph interaction network for video question answering. *IEEE Conf. Comput. Vis. Pattern Recog.*, 15526–15535.
Park, J., Lee, J., Jeon, S., & Sohn, K. (2019). Video summarization by learning relationships between action and scene. *IEEE Int. Conf. Comput. Vis. Worksh.*
Park, J., Lee, J., Jeon, S., Kim, S., & Sohn, K. (2019). Graph regularization network with semantic affinity for weakly-supervised temporal action localization. *IEEE Int. Conf. Image Process.*
Park, J., Lee, J., Kim, I.-J., & Sohn, K. (2020). Sumgraph: Video summarization via recursive graph modeling. *Eur. Conf. Comput. Vis.*, 647–663.
Poleg, Y., Halperin, T., Arora, C., & Peleg, S. (2015). Egosampling: Fast-forward and stereo for egocentric videos. *IEEE Conf. Comput. Vis. Pattern Recog.*, 4768–4776.
Potapov, D., Douze, M., Harchaoui, Z., & Schmid, C. (2014). Category-specific video summarization. *Eur. Conf. Comput. Vis.*, 540–555.

Pritch, Y., Rav-Acha, A., Gutman, A., & Peleg, S. (2007). Webcam synopsis: Peeking around the world. *IEEE Int. Conf. Comput. Vis*.

Qian, X., Zhuang, Y., Li, Y., Xiao, S., Pu, S., & Xiao, J. (2020). Video relation detection with spatio-temporal graph. *ACM Int. Conf. Multimedia*, 84–93.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. *Int. Conf. Mach. Learn.*, 8748–8763.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: towards real-time object detection with region proposal networks. *Adv. Neural Inform. Process. Syst.*, 91–99.

Rochan, M., & Wang, Y. (2019). Video summarization by learning from unpaired data. *IEEE Conf. Comput. Vis. Pattern Recog.*, 7902–7911.

Rochan, M., Ye, L., & Wang, Y. (2018). Video summarization using fully convolutional sequence networks. *Eur. Conf. Comput. Vis.*, 347–363.

Schuhmann, C., Vencu, R., Beaumont, R., Kaczmarczyk, R., Mullis, C., Katta, A., Coombes, T., Jitsev, J., & Komatsuzaki, A. (2021). Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *NeurIPS Workshop*.

Sharghi, A., Gong, B., & Shah, M. (2016). Query-focused extractive video summarization. *Eur. Conf. Comput. Vis.*, 3–19.

Sharghi, A., Laurel, J. S., & Gong, B. (2017). Query-focused video summarization: Dataset, evaluation, and a memory network based approach. *IEEE Conf. Comput. Vis. Pattern Recog.*, 4788–4797.

Shi, W., & Rajkumar, R. R. (2020). Point-gnn: Graph neural network for 3d object detection in a point cloud. *IEEE Conf. Comput. Vis. Pattern Recog.*, 1711–1719.

Shi, L., Zhang, Y., Cheng, J., & Lu, H. (2019). Skeleton-based action recognition with directed graph neural networks. *IEEE Conf. Comput. Vis. Pattern Recog.*, 7912–7921.

Shi, L., Zhang, Y., Cheng, J., & Lu, H. (2019). Two-stream adaptive graph convolutional networks for skeleton-based action recognition. *IEEE Conf. Comput. Vis. Pattern Recog.*, 2026–12035.

Song, Y., Vallmitjana, J., Stent, A., & Jaimes, A. (2015). Tvsum: Summarizing web videos using titles. *IEEE Conf. Comput. Vis. Pattern Recog.*, 5179–5187.

Sun, M., Farhadi, A., Taskar, B., & Seitz, S. (2014). Salient montages from unconstrained videos. *Eur. Conf. Comput. Vis.*, 472–488.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *IEEE Conf. Comput. Vis. Pattern Recog*.

Teng, Y., Wang, L., Li, Z., & Wu, G. (2021). Target adaptive context aggregation for video scene graph generation. *IEEE Int. Conf. Comput. Vis.*, 13688–13697.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Adv. Neural Inform. Process. Syst.*, 5998–6008.

Wang, X., & Gupta, A. (2018). Videos as space-time region graphs. *Eur. Conf. Comput. Vis.*, 413–431.

Wang, S., Gao, L., Lyu, X., Guo, Y., Zeng, P., & Song, J. (2022). Dynamic scene graph generation via temporal prior inference. *ACM Int. Conf. Multimedia*, 5793–5801.

Wang, X., Girshick, R., Gupta, A., & He, K. (2018). Non-local neural networks. *IEEE Conf. Comput. Vis. Pattern Recog.*, 7794–7803.

Wu, G., Lin, J., & Silva, C. T. (2021). Era: Entity-relationship aware video summarization with wasserstein gan. *Brit. Mach. Vis. Conf.*, 1–14.

Wu, G., Lin, J., & Silva, C. T. (2022). Intentvizor: Towards generic query guided interactive video summarization. *IEEE Conf. Comput. Vis. Pattern Recog.*, 10503–10512.

Wu, L., Wang, Y., Shao, L., & Wang, M. (2019). 3-d personvlad: Learning deep global representations for video-based person reidentification. *IEEE Trans. Neural Net. and Learn. Syst., 30*(11), 3347–3359.

Xu, H., Ghosh, G., Huang, P.-Y., Okhonko, D., Aghajanyan, A., Metze, F., Zettlemoyer, L., & Feichtenhofer, C. (2021). Videoclip: Contrastive pre-training for zero-shot video-text understanding. *ACL Conf. Emp. Meth. Natural Lang. Process.*, 6787–6800.

Yan, S., Xiong, Y., & Lin, D. (2018). Spatial temporal graph convolutional networks for skeleton-based action recognition. *AAAI Conf. Art. Intell.*, 7444–7452.

Yang, J., Zheng, W.-S., Yang, Q., Chen, Y., & Tian, Q. (2020). Spatial-temporal graph convolutional network for video-based person re-identification. *IEEE Conf. Comput. Vis. Pattern Recog.*, 3289–3299.

Ye, F., Pu, S., Zhong, Q., Li, C., Xie, D., & Tang, H. (2020). Dynamic gcn: Context-enriched topology learning for skeleton-based action recognition. *ACM Int. Conf. Multimedia*, 55–63.

Yeung, S., Fathi, A., & Fei-Fei, L. (2014). Videoset: Video summary evaluation through text. arXiv preprint arXiv:1406.5824.

Yu, B., Yin, H., & Zhu, Z. (2018). Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *Int. Joint. Conf. Art. Intell.*, 3634–3640.

Yuan, Y., Liang, X., Wang, X., Yeung, D.-Y., & Gupta, A. (2017). Temporal dynamic graph lstm for action-driven video object detection. *IEEE Int. Conf. Comput. Vis.*, 1801–1810.

Yuan, L., Tay, F. E., Li, P., Zhou, L., & Feng, J. (2019). Cycle-sum: Cycle-consistent adversarial lstm networks for unsupervised video summarization. *AAAI Conf. Art. Intell.*, 9143–9150.

Zeng, R., Huang, W., Tan, M., Rong, Y., Zhao, P., Huang, J., & Gan, C. (2019). Graph convolutional networks for temporal action localization. *IEEE Int. Conf. Comput. Vis.*, 7094–7103.

Zeng, R., Huang, W., Tan, M., Rong, Y., Zhao, P., Huang, J., & Gan, C. (2021). Graph convolutional module for temporal action localization in videos. *IEEE Trans. Pattern Anal. Mach. Intell., 44*(10), 6209–6223.

Zhang, K., Chao, W.-L., Sha, F., & Grauman, K. (2016). Summary transfer: Exemplar-based subset selection for video summarization. *IEEE Conf. Comput. Vis. Pattern Recog.*, 1059–1067.

Zhang, K., Chao, W.-L., Sha, F., & Grauman, K. (2016). Video summarization with long short-term memory. *Eur. Conf. Comput. Vis*.

Zhang, K., Grauman, K., & Sha, F. (2018). Retrospective encoders for video summarization. *Eur. Conf. Comput. Vis.*, 391–408.

Zhang, Z., Shi, Y., Yuan, C., Li, B., Wang, P., Hu, W., & Zha, Z. (2020). Object relational graph with teacher-recommended learning for video captioning. *IEEE Conf. Comput. Vis. Pattern Recog.*, 13278–13288.

Zhang, Z., Zhao, Z., Zhao, Y., Wang, Q., Liu, H., & Gao, L. (2020). Where does it exist: Spatio-temporal video grounding for multi-form sentences. *IEEE Conf. Comput. Vis. Pattern Recog.*, 10668–10677.

Zhang, X., Zhu, Z., Zhao, Y., Chang, D., & Liu, J. (2019). Seeing all from a few: ℓ_1-norm-induced discriminative prototype selection. *IEEE Trans. Neural Net. and Learn. Syst., 30*(7), 1954–1966.

Zhao, B., & Xing, E.P. (2014). Quasi real-time summarization for consumer videos. *IEEE Conf. Comput. Vis. Pattern Recog*.

Zhao, B., Li, X., & Lu, X. (2017). Hierarchical recurrent neural network for video summarization. *ACM Int. Conf. Multimedia*, 863–871.

Zhao, B., Li, X., & Lu, X. (2020). Hsa-rnn: Hierarchical structure-adaptive rnn for video summarization. *IEEE Conf. Comput. Vis. Pattern Recog.*, 7405–7414.

Zhao, B., Li, H., Lu, X., & Li, X. (2021). Reconstructive sequence-graph network for video summarization. *IEEE Trans. Pattern Anal. Mach. Intell., 44*(5), 2793–2801.

Zhou, K., Qiao, Y., & Xiang, T. (2018). Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. *AAAI Conf. Art. Intell.*, 7582–7589.

Zhu, Z., Tang, W., Wang, L., Zheng, N., & Hua, G. (2021). Enriching local and global contexts for temporal action localization. *IEEE Int. Conf. Comput. Vis.*, 13516–13525.

Zhu, W., Lu, J., Li, J., & Zhou, J. (2020). Dsnet: A flexible detect-to-summarize network for video summarization. *IEEE Trans. Image Process., 30*, 948–962.

Zwillinger, D., & Kokoska, S. (1999). *Crc standard probability and statistics tables and formulae*. Boca Raton: CRC Press.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.