Vol.2018-CVIM-212 No.38
2018/5/11

Linking videos and languages: Representations and Their Applications

Mayu Otani[5,a)] Yuta Nakashima[2,b)] Esa Rahtu[3,c)] Janne Heikkilä[4,d)] Naokazu Yokoya[1,e)]

Abstract: Mimicking the human ability to understand visual data (images or videos) is a long-standing goal of computer vision. To achieve visual content understanding in a computer, many recent works attempt to connect visual and natural language data including object labels and descriptions. This attempt is important not only for visual understanding but also for broad applications such as content-based visual data retrieval and automatic description generation to help visually impaired people. The goal of this paper is to develop cross-modal representations, which enable us to associate videos with natural language. We explorer two directions for constructing cross-modal representations: hand-crafted representations and data-driven representation learning. The experiments demonstrate the proposed representations can be applied to a wide range of practical applications including query-focused video summarization and content-based video retrieval with natural language queries.

1. Introduction

Once humans take a brief look at visual data (images or videos), they can easily and quickly list various concepts in the image and describe the visual content with natural language. Mimicking this human ability, *i.e.*, understanding and describing visual content, in a computer is a key technique for various applications such as content-based image or video retrieval and automatically describing visual content to help visually impaired people understand the visual content.

One approach for connecting visual and natural language data is to design a cross-modal embedding space. Figure 1 illustrates the idea of cross-modal embedding space. Both images and texts are represented as points in a common space so that those with similar semantics are located at close points. For example, an image of zebras in the field, as well as a sentence “a flock of zebras grazing” should be mapped to nearby points.

One straightforward approach to constructing cross-modal representation is to use visual concept recognition techniques. We can obtain a list of visual concepts from an image by visual concept classification or detection techniques. As visual concepts are often described with nouns or verbs in natural language, we can compute the semantic similarity between text and visual data by matching words in a text and detected concept labels. Based on this assumption, some work represents visual data by a set of concept labels and natural language data by a set of words. This cross-modal representation is often used as an intermediate representation in description generation methods [7], [12].

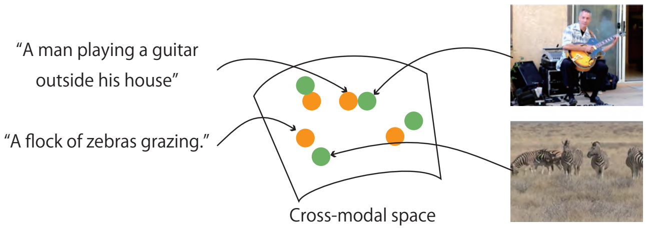

“A man playing a guitar outside his house”
“A flock of zebras grazing.”
Cross-modal space

Fig. 1 Illustration of cross-modal embedding space.

While extensive research efforts had been made so far for designing cross-modal representations, it was difficult to connect visual and natural language data because of the limited performance of visual concept recognition models. Visual concept recognition has been a challenging task, although it looks quite simple. Inferring visual concepts of an image involves finding patterns that might be relevant to the visual concepts. Since humans are not aware of how they find and generalize the visual patterns, implementing how to recognize visual concepts in a computer has been quite difficult.

The recent emergence of large-scale datasets and deep neural networks (DNNs) facilitate visual concept recognition. In computer vision, convolutional neural network-based classification models have shown substantial improvement in visual concept recognition tasks including object recognition [15], [26], [49] and action recognition [19], [21]. Now, the state-of-the-art models can even distinguish thousands of visual concepts [43].

An important advantage of the DNN-based approach is the integration of the whole process involved in a task in a deep model, which can be trained in an end-to-end manner. For example, previous object recognition involves several processes: low-

[1] Nara Institute of Science and Technology
[2] Osaka University
[3] Tampere University of Technology
[4] University of Oulu
[5] CyberAgent, Inc.
[a)] otani_mayu@cyberagent.co.jp
[b)] n-yuta@ids.osaka-u.ac.jp
[c)] esa.rahtu@tut.fi
[d)] janne.heikkila@ee.oulu.fi
[e)] yokoya@is.naist.jp

level feature extraction, feature transformation, and classification. They are separately designed, and tuning low-level processing (*e.g.*, low-level feature extraction) to maximize the performance of final output is hardly feasible. On the other hand, DNN-based approach integrates these process into one deep model, and the whole process can be optimized by fitting to large-scale data. Given pairs of input and correct output, deep models can learn useful features and how to use them.

This also accelerated the research for tasks that involve different modalities, such as image or video captioning [3], [63]. Recent works propose to associate embeddings of visual data and natural languages with DNNs. In end-to-end learning, how to extract features from different modalities, as well as how to fuse them can be learned seamlessly. This leads the improvement in learning cross-modal embedding spaces for visions and language [24], [64]. In DNN-based approach, one may not need to extract concepts from images or natural language explicitly. Instead, one will model how to map images and text to a common space.

The goal of this paper is to develop cross-modal representations for videos and natural languages. The representation should capture complex semantics, and their similarity should follow human intuition on semantic similarity. Most works in this direction have tried to connect static images and short phrases or sets of keywords [20], [24], and videos and natural language have still significant room to explore. Different from static images, videos have additional challenges to capture semantics because they have temporal changes. Due to the temporal changes, the semantics of videos are more complex. This complexity of content makes modeling video understanding more difficult. Similar challenge exists in natural language understanding. Since a sentence in natural language may include various words, and the semantics of each word highly depends on context, modeling the semantics of sentences is also difficult.

As we mentioned above, techniques to connect videos and text have some practical applications. To evaluate the performance of our cross-modal representations, we will apply them to several tasks, which involve videos and natural languages, such as query-focused video summarization, video captioning, and content-based video retrieval. By showing the results of these applications, we will investigate the capability of our cross-modal embedding space.

2. Related Work

The work in this paper is motivated by many previous works that address to link vision and language modalities. This section gives the overview of existing cross-modal representations and applications that involve visual and language data.

2.1 Cross-modal Representations for Videos and Languages

Some early works proposed to use a set of concept labels as cross-modal representations for static images and text [7], [33]. Farhadi *et al.* [7] introduced triplets of concept labels (object, action, and scene) as representations, which represent the abstract semantics of images and sentences. For videos, the approach by Lin *et al.* [33] associates a parsed semantic graph of a query sentence and visual cues based on object detection and tracking.

These works require explicit concept detection to construct representations. Therefore, they cannot handle images or text with unseen concepts. To achieve more flexible representations, some works propose to develop a common embedding space, in which visual and language data can be mapped [9], [22], [51]. This approach enables us to compute the semantic similarity between images and text based on the distance in the embedding space without explicit concept detectors. For example, Socher *et al.* [51] proposed to embed low-level image representations and word vectors of object labels into a common embedding space with neural network-based models. They demonstrated classification of unseen visual concepts in the embedding space, which is called as zero-shot learning.

The recent success of deep convolutional neural networks (CNNs) together with large-scale visual datasets has led to several powerful models for image understanding [15], [49], [52]. These models showed not only significant improvement in object classification, but also highly generalized visual representations obtained from hidden layers of the deep models [5]. Deep neural networks have also been used in the field of natural language processing [24], [29]. These works demonstrated that neural network-based models are capable of encoding semantics of text. For example, Kiros *et al.* [24] proposed sentence representation learning using recurrent neural networks (RNNs). They also demonstrated joint learning of image and sentence embedding models, which convert images and sentences to cross-modal representations.

Cross-modal representation learning using deep neural networks is explored in many tasks [9], [22], [34], [64], [70]. Frome *et al.* [9] proposed image classification by computing similarity between joint representations of images and labels, and Zhu *et al.* [70] addressed alignment of movie scenes with sentences in a book using joint representations for video clips and sentences. Their approach also computes the similarity between sentences and subtitles of video clips to improve the performance of video-sentence alignment.

2.2 Applications

Video Summarization

We develop video summarization methods as an applications of the proposed representations in Section 3. Video summarization is a technique to generate a compact representation of videos, which help users quickly understand their content.

To automatically select video excerpts from input videos, various ideas to asses the importance of video clips have been proposed. Attractiveness is a widely employed selection criterion, representing how well a clip attracts the attention of the audience [6], [13], [27], [36].

Another criterion for video clip selection is representativeness; video clips in a summary should be less redundant but cover most of the original content, [11], [14], [64], [69]. One major approach to retrieving representative clips is video-clip clustering. Gygli *et al.* [14] cast the selection of representative clips as a *k*-medoids problem, which can be efficiently optimized due to its sub-modularity. Zhao *et al.* [69] proposed an online video summarization method. Their method generates a video summary by picking out video clips that are able to reconstruct the remaining clips.

Another interesting research direction in video summarization is text-focused summarization, which controls the content of video summaries using textual cues. Sharghi *et al.* [48] proposed to extract video clips based on the relevance to keywords. Research in this line attempts to associate video content and text, such as scripts and query words, to generate a video summary based on the input text. In Section 3, we propose an object-based representation for videos and text for text-focused video summarization.

Content-based Video and Language Retrieval

Due to the explosive growth of images and videos on the web, visual retrieval has become a hot topic in computer vision and machine learning [30], [31], [37]. Early work addressed content-based video retrieval by detecting predefined concepts in videos, such as objects, actions, and events [50], [60]. A single visual concept may not be enough to spot the desired video, as users are more likely to query with their combinations. Video retrieval by natural language queries provides an intuitive way to make a combination of concepts in a specific context represented in a query. One possible approach is to detect visual concepts and match them to keywords in a query [28], [33], [59], [61], but as they require pre-trained concept detectors, unseen concepts are missed.

To overcome such limitations, Socher *et al.* [51] proposed to learn to embed images and concept labels into a common space, which can handle unseen concepts. Several approaches in this direction have been proposed on both image retrieval [9], [24] and video retrieval [40], [64], [70]. Xu *et al.* [64] proposed a deep neural network for video retrieval by sentence queries and vise versa. They embed a video clip and a sentence into a common space to compute the similarity between them. Yu *et al.*'s approach [67] learns a similarity metric between a whole video content and a query sentence. In contrast to these methods, we address to estimate the relevance that may vary within a video in Section 6.

3. Object-based Representations for Summarizing Personal Videos Using Blog Text

This section proposes object-based representations to capture the semantics of videos and text. We assume that objects in a video clip provide rich cues to understand events in a video, and nouns in text also tell key concepts of text's content as well. Based on this assumption, we construct an object-based representation that encodes objects in videos and nouns extracted from text. We also define a similarity metric with this object-based representation, which enables us to compute the semantic similarity between a video and text.

In this work, as an application of our object-based representations, we develop a video summarization method that edits a long video according to scripts written by a user. Specifically, our video summarization system takes a text written for a video blog post and unedited videos as input and produces a video summary that has semantically relevant content to the blog post. During video clip selection, we optimize the content similarity between an output video summary and the blog post, which can be computed with our object-based representation.

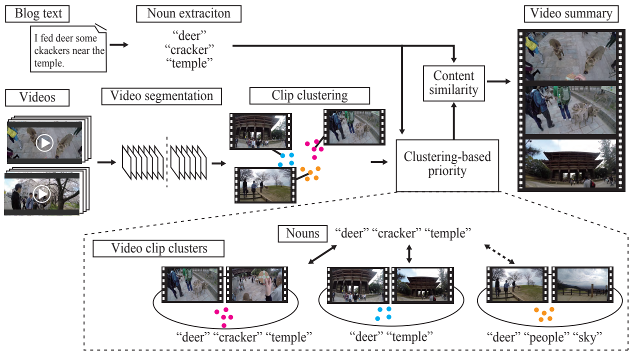


Fig. 2 Overview of the proposed text-based video summarization method. Given text written by a user, our method selects video clips based on the content of the text, such that the video summary reflects the user's intentions.

3.1 Text-based Video Summarization

Our video summarization method takes videos with timestamps and the text written by the blog author as input and generates a video summary. The problem of video summarization can be cast as a problem of selecting the optimal subset of video clips. In this study, we design an objective function based on the content similarity between a set of clips and text. By selecting clips that have high content similarity to the input text, our method produces a video summary reflecting the blog author's intentions. Figure 2 illustrates an overview of our method. Our method first extracts nouns from the input text. The videos are then segmented and clustered into groups, each of which corresponds to an event. Based on these clusters, we compute the priority of clips; highly prioritized clips are more likely to be included in the video summary. After computing the priority, a video summary is produced by selecting the optimal subset of clips.

3.1.1 Object-based Representations for Videos and Text

Encoding Text

Since objects in videos are often described with nouns, we extract nouns from the input text. The input text is represented by an N-dimensional vector $\mathbf{y}$, where N is a vocabulary size, and assume that noun n corresponds to object n. We set $y_n = 1$ if noun n is included in the input text and 0 otherwise. For noun extraction, we apply parts-of-speech tagging to the input text [57]. We also remove predefined stop words because common words are hardly informative.

Encoding video clips

We first perform video segmentation on lengthy input videos. Because our method selects clips based on their objects, we set clip boundaries where objects appear or disappear. To find these clip boundaries, we employ the method by Huang *et al.* [17]. Their method tracks the number of keypoint matches and identifies local minima. These local minima often correspond to frames around which objects appear or disappear. Thus, we divide the video at such frames.

Each video clip after video segmentation is represented by object labels and their importance. Object-detection methods, such as [10], can automatically find objects in the clips; however, to

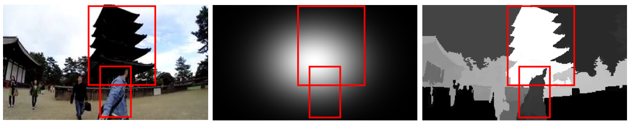


Fig. 3 Maps of location-based object importance (a) and saliency-based object importance (b).

focus on our clip selection performance without focusing on the performance of the object-detection method, we manually annotate object labels in this study, rather than detecting them automatically. To do so, we extracted the middle frame of each clip as a keyframe and annotated the object labels.

In this paper, we test two types of object importance: location-based object importance, and saliency-based object importance. Location-based object importance is simply based on the location and the size of the bounding box of each object. The computation of location-based importance relies on some heuristics: *viz.*, (i) an important object is more likely to be located near the center of a frame, and (ii) it occupies a large area. Based on these heuristics, the importance $x_{m,n}$ of object n in a clip m is defined as

$$x_{m,n} = \int_{\omega \in \Omega_n} \mathcal{N}(\omega|\mu, \Sigma) \mathrm{d}\omega, \tag{1}$$

where Ω_n is object n's bounding box, and $\mathcal{N}$ is the normal distribution whose mean μ is the frame's center position and whose variance Σ is a predefined parameter.

The other type of object importance incorporates saliency maps. Because salient objects are likely to be visually important, we employ the average of saliency values over a bounding box as the saliency-based object importance. In this thesis, we use saliency maps based on Yan *et al.*'s method [65]. Saliency maps are computed based on local contrast values and center bias, *i.e.*, areas near the center of an image are more likely to be important. To get stable results, their method generates multiple image layers, which are coarse representations at different levels, and computes a saliency map for each layer. Saliency maps in different scales are fused to produce a final output. Note that the proposed method can use any other method to obtain saliency maps, such as [1], [39], [41], without significant modification. Figure 3 shows the maps of location-based object importance and saliency-based object importance, where brighter areas are regarded as more important.

After computing the importance, the input videos are represented by a set of clips $X = \{\mathbf{x}_m \in \mathrm{R}^N \mid m = 1, \ldots, M\}$, where $\mathbf{x}_m$ is a vector representation of the clip m. N is the number of object categories, and each element $x_{m,n}$ denotes the importance of object n in that clip.

3.1.2 Text-based Clip Selection

Let $\boldsymbol{\psi}(S)$ be a function that gives an N-dimensional vector representation of a subset of clips $S \subseteq X$, given by

$$\boldsymbol{\psi}(S) = \sum_{x_m \in S} p_m(\mathbf{y}) \mathbf{x}_m, \tag{2}$$

where $p_m(\mathbf{y})$ and $\mathbf{x}_m$ denote a priority value of the clip m conditioned on the input text and an N-dimensional vector representation for clip m, respectively. The priority value represents how relevant the clip is to the input text, which is computed with cluster-based content similarity.

With the video summary representation, we formulate the problem of selecting a subset of clips $S^* \subseteq X$ as:

$$S^* = \operatorname*{argmax}_{S \subseteq X} O(\boldsymbol{\psi}(X), \mathbf{y}), \tag{3}$$

$$\text{s.t.} \sum_{\mathbf{x}_m \in S} l_m \leq L. \tag{4}$$

Here, L is the length of the resulting summary, which is given by the user, and l_m is the length of clip m. The objective function to be maximized in video summarization is a linear combination of two terms as follows:

$$O(\boldsymbol{\psi}(S), \mathbf{y}) = o_{\text{sim}}(\boldsymbol{\psi}(S), \mathbf{y}) + \alpha o_{\text{cov}}(\boldsymbol{\psi}(S)), \tag{5}$$

where o_{sim} is the content similarity between S and the input text $\mathbf{y}$, and o_{cov} is the content coverage. Moreover, α is a parameter that balances these two terms. Selecting a subset with high content similarity reflects the blog author's intentions in the resulting summary, and the content-coverage term encourages the summary to include various content, provided that it is relevant to the input text. We obtain a subset of video clips that maximize the objective function by solving a knapsack problem with dynamic programming algorithm.

The following sections detail the clip priority, the content-similarity term, and the content-coverage term.

Clip Priority

Our method uses content similarity based on the objects in each video clip and the nouns in the input text. However, this can be unreliable, because the clip usually contains a subset of objects that appear in the event. For example, suppose an input text writes about a certain event in which a certain object is involved. If this object is not very specific to the event, even though it appears throughout the input video, content similarity based solely on objects and nouns can pick out all clips that come with the object.

To find clips that are more relevant to the input text, we introduce clustering-based clip priority. First, we assume that an event is temporally concentrated, *i.e.*, clips capturing the same event have similar timestamps. Under this assumption, we can cluster clips based on their timestamps and the objects in them. For clustering, we use affinity propagation [8]. The similarity between two clips $\mathbf{x}_i$ and $\mathbf{x}_j$ is defined as

$$A(\mathbf{x}_i, \mathbf{x}_j) = \exp\left[-\frac{\lambda \min(|\tau_i - \tau_j|, \theta)}{M}\right] + \gamma J(\mathbf{x}_i, \mathbf{x}_j), \tag{6}$$

where τ_i is the temporal frame index of the middle frame in clip i, and M denotes the total number of frames in the input videos. Here, $J(\cdot, \cdot)$ gives the weighted Jaccard similarity, defined as

$$J(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_n \min(x_{i,n}, x_{j,n})}{\sum_n \max(x_{i,n}, x_{j,n})}. \tag{7}$$

In Eq. (6), λ controls the reduction in temporal similarity, and θ is a threshold for the temporal distance $|\tau_i - \tau_j|$. We suppose that clips extracted from different videos are temporally distinct. Thus, the temporal distance $|\tau_i - \tau_j|$ of such clips is set to a threshold θ. Moreover, γ is a parameter to balance the temporal similarity with the object based similarity. The number of clusters is automatically determined from data and self-similarity $A(\mathbf{x}_i, \mathbf{x}_j)$. Low self-similarity values result in a small number of clusters. We set the self-similarity values to the median of the pair-wise similarities as suggested in [8].

We assume that a cluster is relevant to the input text when the nouns corresponding to the objects in the cluster are included in the input text. Thus, we again use the weighted Jaccard similarity between a cluster and the input text to determine the priority of all clips in the cluster. Let $\mathbf{c}_i$ be a representation of the cluster that includes clip i, each element of which represents whether the corresponding object appears in the cluster. More specifically, we set $c_{i,n} = 1$ if any clip in the cluster has $x_{m,n} > 0$, and $c_{i,n} = 0$ otherwise. Using this, the priority value of clip i is computed as

$$p_i(\mathbf{y}) = J(\mathbf{c}_i, \mathbf{y}). \tag{8}$$

Content-similarity Term

We quantify the content similarity between the set S of clips and the input text y using the weighted Jaccard similarity in Eq. (7). This computes the similarity between object labels in selected videos and nouns in the input text as follows:

$$o_{\mathrm{sim}}(\mathbf{x}, \mathbf{y}) = J(\boldsymbol{\psi}(S), \mathbf{y}). \tag{9}$$

This similarity indirectly relies on priority through $\boldsymbol{\psi}(S)$. The value increases when S includes clips with high priority that have objects in common with the input text.

Content-coverage Term

If content coverage is not considered, some relevant clips can be rejected when their objects do not appear explicitly in the input text. This can result in a summary that is entirely composed of clips with similar content. To avoid this, our method encourages the inclusion of relevant clips that cover diverse content. Coverage of the original content is a criterion that is widely used in summarization tasks [53], [55], [58]. Insofar as our goal is to generate a summary that reflects the blog author's intentions, we list objects annotated to highly prioritized video clips. The coverage of the set of objects is rewarded during clip selection.

Let $\boldsymbol{\Gamma} = (\gamma_1, \dots, \gamma_N)$ represent a set of objects in highly prioritized clips, where $\gamma_n = 1$ if clip $\mathbf{x}_i \in X$ whose $p_i > \rho$ has $x_{i,n} > 0$ and $\gamma_n = 0$ otherwise. We define the coverage $o_{\mathrm{cov}}(\boldsymbol{\psi}(S))$ using the weighted Jaccard similarity in Eq. (7) to compute similarity between sets of objects in selected clips and prioritized ones as follows:

$$o_{\mathrm{cov}}(\boldsymbol{\psi}(S)) = J(\boldsymbol{\psi}(S), \boldsymbol{\Gamma}). \tag{10}$$

This term represents how well S covers the content of highly prioritized clips.

3.2 Evaluation and Discussion

Assessing the quality of video summaries is a challenging problem itself. Most previous methods are evaluated based on user studies [23], [35] or by comparing the resulting summaries with manually created reference summaries [13], [32], [42]. Since our task (*i.e.*, video summarization for video blogs) is a novel video summarization task, there is no established way to evaluate the performance of our method. Therefore, we opt to conduct a user study.

T1	On a warm day in March, we went to Nara Park. Before getting to Nara Park, we went to Saho river. There were cherry trees along the river. The river is well known for cherry blossom, and many people visit during the season of blossom. I took many videos of other students. One of the students, Nakashima used a special camera for his study. He took some videos, carrying the camera along the river. It was a beautiful place and I want to visit there next spring again.
T2	We went to Nara Park. A lot of deer were around the Nandaimon. There were also a few cracker shops, and many tourists enjoyed feeding deer. I bought some crackers and deer immediately gathered around me.
T3	Nandaimon is a famous gate in the Nara Park. I saw a statue of Nandaimon. There were many people.

Fig. 4 Original texts used in the experiment.

Table 1 Input and methods evaluated.

	Input	Method
(a)	Videos	Uniform sampling
(b)	Videos	Cluster-based
(c)	Videos and text	Proposed method
(d)	Videos and text	Description-based w/o content coverage
(e)	Videos and text	Description-based w/o content coverage and preference

The user study consists of two parts. First, a participant is asked to score multiple video summaries for a given blog post regarding their suitability to the blog post. To investigate our video summaries in detail, we administer an additional questionnaire regarding other properties, including redundancy, content coverage, and the relevance of the summary to the input text. This first part evaluates the video summaries from the perspective of the video blog viewers. The second part of the user study involves collecting blocks of text written by the participants and generating video summaries using the text. The participants are asked to score the video summaries based on their text. Consequently, this part evaluates the video summaries from the perspective of the blog authors.

3.3 Evaluation from the Viewers' Perspective

Because this is the first attempt to use video summarization for video blogs, we investigate whether blog viewers believe that the video summaries generated by our method are suitable for a given blog post. We also evaluate video summaries in terms of several properties, such as redundancy and content coverage, which are widely used criteria in the domain of video summarization.

To compile a dataset, we recorded multiple videos of a short trip, totaling 80 min. As input text, we used the three blocks of text shown in Figure 4, each of which describes different scenes from the input videos. We compared our method to multiple baseline methods (see Table 1). Methods (a) and (b) generate video summaries without text. Uniform sampling (a) is a simple yet effective way to produce video summaries, and this method is widely employed as a baseline. We sampled 2-sec. clips with uniform intervals. The clustering-based method (b) utilizes the clustering results described in Section 3.1.2. With this method, clips are selected from cluster representatives, such that they include as many objects as possible. We also compared some variants of our method. Method (c) is our full method. Method (d)

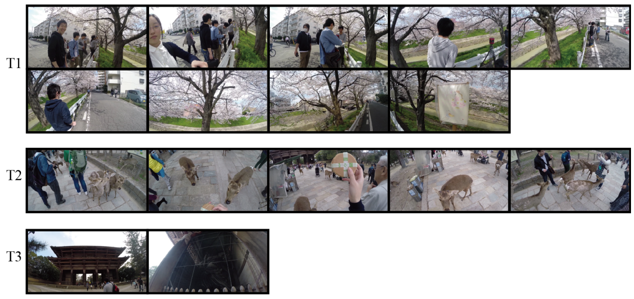


Fig. 5 Keyframes of our video summaries for each input text.

Table 2 Average scores regarding suitability to a video blog post. Bold values indicate the highest scores for each group.

method	Input text	Group		
		G1	G2	G3
(a) Uniform sampling	None	3.38	1.67	2.67
(b) Cluster-based	None	**4.38**	1.83	2.00
(c) Proposed method	T1	3.38	1.00	1.83
	T2	2.25	**4.33**	2.67
	T3	1.38	1.67	3.67
(d) Description-based w/o content coverage	T1	3.25	1.00	1.83
	T2	2.13	4.17	2.67
	T3	1.13	2.17	**4.00**
(e) Description-based w/o content coverage and preference	T1	2.25	3.00	2.50
	T2	2.00	3.17	3.00
	T3	2.13	2.67	3.17

is basically our text-based method, but with the content coverage term o_{cov} excluded (*i.e.*, $\alpha = 0$). In addition to the exclusion of the coverage term o_{cov}, method (e) also excludes clip priority by setting the priority values of all clips to 1. All of these variants used location-based object importance.

For location-based object importance, the parameters were set to $\Sigma = \text{diag}(8w, 8h)$, where w and h are the width and the height of the frame, respectively. Other parameters were heuristically determined as follows: $\alpha = 0.25$, $\lambda = 5$, $\theta = 3600$, $\gamma = 0.25$, $\rho = 0.1$, and $L = 20$. Here, θ corresponds to 60 seconds, because our input videos were 60 fps. We generated video summaries using methods (c)–(e) for each input text. In total, we generated 11 videos. Keyframes of the clips selected with our full method are shown in Figure 5. These resulting summaries show that our method selects clips from different scenes based on the content of the input texts.

We recruited 20 participants from both genders; all participants were in their 20s or 30s. They reviewed a video blog post and were asked to score each video in terms of how well the video suited the blog post. The scores ranged from 1 to 5, where 1 means that the video definitely does not suit the blog post, and 5 means that it suits the post very well. The participants were divided into three groups. Group 1 (G1), Group 2 (G2), and Group 3 (G3) had eight, six, and six people, respectively. The blog post T1 was displayed for subjects in G1, blog post T2 for G2, and blog post T3 for G3. After reviewing a blog post, subjects rated baseline video summaries and description-based video summaries. Subjects also scored video summaries generated using blog posts for other groups.

Table 2 shows the scores for each group. For all groups, our full method (c) was scored as either the first or second best. Variant (d) was also rated highly. Interestingly, the participants in G1 chose clustering-based video summary (b) as most suitable for text T1. In fact, the clustering-based method (b) only accidentally included many clips relevant to T1, which contributed to the high score. Furthermore, we found that only the summary generated by the clustering-based method (b) included scenes just before the events described in T1. Although the inclusion of such clips was not part of the design of the clustering-based method, such clips can lead to a better comprehension of the events by providing context. The effect of such connecting video clips on video summaries is discussed in [35].

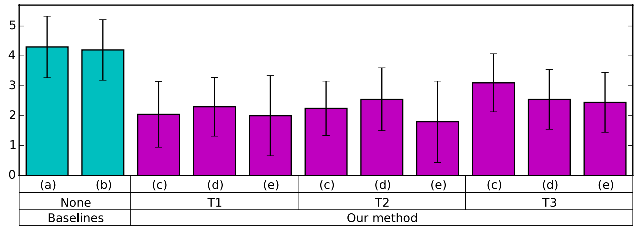


Fig. 6 Averages and standard deviations of the scores for Q2.

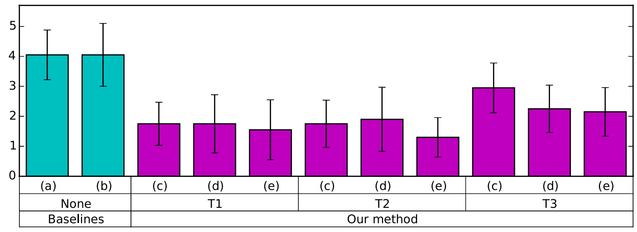


Fig. 7 Averages and standard deviations of the scores for Q3.

The results from comparing the scores among variants of our methods (c)–(e) imply that the content coverage term o_{cov} did not significantly affect the score. On the other hand, the use of clip priority resulted in a significant improvement in the suitability for the video blog. From these results, we conclude that the participants generally preferred our method over other methods. These results also suggest that the inclusion of clips that introduce scenes of interest can further improve the suitability for a blog post. The participants were also asked to score videos in terms of the following three aspects, to investigate the perception of our video summary compared to that of the baselines.

Q1 How well the video matches the input text (relevance to the input text).

Q2 How redundant the video is.

Q3 How well the summarized video covers the content of the entire video.

The scores ranged from 1 to 5. For Q1, a score of 1 means that the video does not represent the text at all, whereas 5 means that it represents the text very well. For Q2, scores 1 and 5 mean “significantly redundant” and “hardly redundant,” respectively. For Q3, score 1 means significant content is missing, whereas 5 means that most content is covered. The relevance to the input text is an important property for video summarization designed for video blogs. Because redundancy and content coverage is widely used in evaluations of video summaries, we also investigated these properties.

Table 3, Figure 6, and Figure 7 show the results for Q1, Q2,

Table 3 Average scores of similarity to the input text (Q1). Bold values are the highest scores for each input text.

	Baselines		Our method								
	(a)	(b)	(c)			(d)			(e)		
Text	None		T1	T2	T3	T1	T2	T3	T1	T2	T3
T1	3.45	3.65	**3.90**	1.60	1.30	3.8	1.65	1.15	2.75	1.55	1.65
T2	1.70	1.85	1.10	**4.50**	1.75	1.20	4.45	1.70	2.35	3.70	3.40
T3	1.35	1.20	1.10	1.65	4.70	1.10	1.30	**4.75**	1.55	2.80	3.10

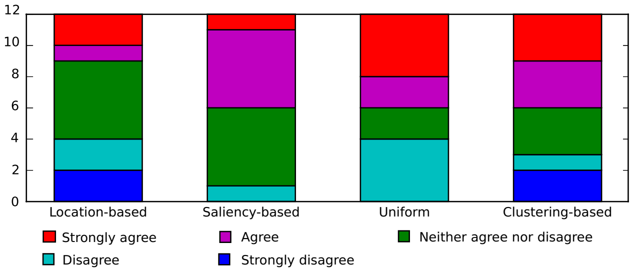


Fig. 8 Answers for Q4.

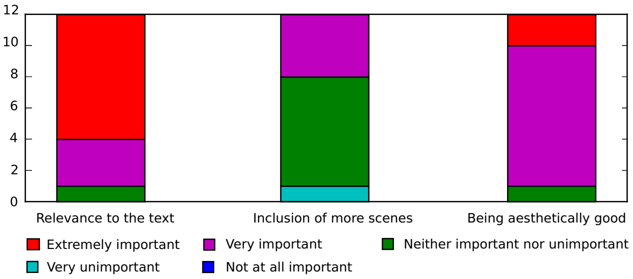


Fig. 9 Answers regarding the importance of video-summary properties.

and Q3, respectively. Regarding Q1, our summaries received the highest scores in terms of the relevance to the input text.

This means that our method was able to select clips appropriate to the input text. On the other hand, in terms of redundancy (Q2) and content coverage (Q3), our method received lower scores than uniform sampling (a) and the clustering-based method (b). Because our video summaries have multiple clips relevant to the input texts, the clips can have similar content. This resulted in lower scores for Q2. The score for Q3 was also expected because our method restricts those clips that are included in the summary based on the input text. Although our method was not rated highly for Q2 (redundancy) and Q3 (content coverage), the participants preferred our video summaries for the video blog posts, according to Table 2. This indicates that, for blog viewers, the relevance to the input text is more important for video blogs than redundancy or content coverage.

3.3.1 Evaluation from the Blog Author's Perspective

We also collected texts written by 12 participants, all male and all in their 20s. We asked them to score the video summaries that were generated based on their texts. The participants reviewed all unedited videos in our dataset and wrote a short description of what interested them. By comparing participants ’responses, we investigated how their intention was reflected in the video summaries.

The video dataset was the same as that of the previous section. In this evaluation, we compared four methods. Two were the same as the baselines (a) and (b) in the previous section. The other two methods were our proposed method, with location-based object importance and saliency-based object importance. The parameter ρ was set to the minimum of the priority for the top-90% of the clips.

For the first question, participants were asked to rate whether they would want to use the video summary generated by each method for their video blog post (Q4). Scores 1 and 5 indicate "strongly disagree" and "strongly agree," respectively. Table 4 shows the average and the standard deviation of the scores. Whereas our method with saliency-based object importance and uniform sampling received the same average score, the standard deviation of our method was smaller. Figure 8 shows details of the results. This reveals that uniform sampling received both positive and negative responses, whereas only a few participants negatively rated our method with saliency-based object importance.

To identify the factors that affect these scores, we asked the participants to answer an additional questionnaire regarding their assessment of the following properties of video summaries:

- Relevance to the text in the blog post
- Inclusion of more scenes than the text
- Aesthetic quality (composition, camera motion, etc.).

The participants were asked whether these respective properties were important. Figure 9 shows the results. The results show that many participants thought that the relevance to the blog post and the aesthetic quality were important for video summaries. We believe that this is the main reason why the participants preferred saliency-based object importance to location-based object importance.

3.4 Summary

We proposed object-based cross-modal representations for videos and text and introduced the semantic similarity using the representations. We demonstrated a text-based video summarization method for video blog authoring as an application of our representation. The user study showed that our video summarization method benefits video blog authoring. The results also suggest that considering the aesthetic quality in addition to relevance to a blog post can further improve video summaries.

4. Learning Semantic Representations by Linking Videos and Sentences

This section describes a representation learning method to associate videos and sentences. Different from the object-based representations described in Section 3, we address to incorporate various concepts including actions, scenes and attributes to compute cross-modal representations. Specifically, we construct deep

Table 4 Averages and standard deviations of scores for Q4.

	Our Methods		Baseline Methods	
	Location-based (c)	Saliency-based	Uniform (a)	Clustering-based (b)
Avg.	2.92	**3.50**	3.50	3.33
Std.	1.31	0.80	1.31	1.44

models that convert a video clip and a sentence to vector representations in a common embedding space, where their semantic similarity correlates to their negative distance.

Our cross-modal representations are validated on the task of content-based video retrieval. We demonstrate content-based video and sentence retrieval between video clips and sentences using our cross-modal representations. Our embedding models are further extended with web image search to disambiguate the semantics of sentences, which can be helpful for content-based video and sentence retrieval applications.

4.1 Cross-modal Representation Learning for Videos and Sentences

We propose a neural network-based embedding model to extract cross-modal representations for videos and sentences. Our embedding model consists of two sub-networks, each of which encodes videos and sentences, respectively. Moreover, we also propose to enhance sentence embedding using web images.

4.2 Video Embedding

In our approach, we employ two CNN architectures: 19-layer VGG [49] and GoogLeNet [52], both of which are pre-trained on ImageNet [46]. We replace the classifier layer in each model with two fully-connected layers. Specifically, we compute activations of the VGG's `fc7` layer or the GoogLeNet's `inception 5b` layer and feed them to additional fully-connected layers.

We extract frames from a video at 1 fps as in [64]. Let $V = \{v_1, \dots v_N\}$ be a set of frames v_i, where $v_n \in \mathrm{R}^{d_\mathrm{v}}$ is a visual feature extracted from n-th frame (d_v=4,096 for VGG, and d_v=1,024 for GoogLeNet). The video embedding $x \in \mathrm{R}^{d_\mathrm{e}}$ is computed by:

$$x = \operatorname*{mean}_{v \in V}[\tanh(W_{\mathrm{v}_2} \tanh(W_{\mathrm{v}_1} v + b_{\mathrm{v}_1}) + b_{\mathrm{v}_2})]. \qquad (11)$$

Here, $W_{\mathrm{v}_1} \in \mathrm{R}^{d_\mathrm{h} \times d_\mathrm{v}}$, $b_{\mathrm{v}_1} \in \mathrm{R}^{d_\mathrm{h}}$, $W_{\mathrm{v}_2} \in \mathrm{R}^{d_\mathrm{e} \times d_\mathrm{h}}$, and $b_{\mathrm{v}_2} \in \mathrm{R}^{d_\mathrm{e}}$ are the learnable parameters of the fully-connected layers. mean[·] denotes a mean pooling, which take the average of input vectors.

4.3 Sentence Embedding

For the sentence sub-network, we use skip-thought vector model by Kiros *et al.* [24], which encodes a sentence into 4800-dimensional vectors with an RNN. Similarly to the video sub-network, we introduce two fully-connected layers with hyperbolic tangent nonlinearity (but without a mean pooling layer). We encode sentences into vector representations using skip-thought that is an RNN pre-trained with a large-scale book corpus [24].

We use combine-skip in [24], which is a concatenation of outputs from two separate RNNs trained with different datasets. We denote the output of combine-skip by $t_\mathrm{cs} \in \mathrm{R}^{d_\mathrm{c}}$, where d_c=4,800.

We then transform the skip-thought vectors t_cs into a sentence embedding y with two fully-connected layers as:

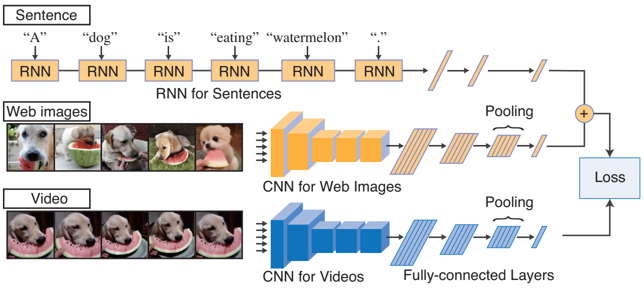


Fig. 10 Illustration of our video and sentence embedding with web images. The orange component is the sentence embedding model that takes a sentence and corresponding web images as input. Video embedding model is denoted by the blue component.

$$y = \tanh(W_{\mathrm{s}_2} \tanh(W_{\mathrm{s}_1} t_\mathrm{cs} + b_{\mathrm{s}_1}) + b_{\mathrm{s}_2}), \qquad (12)$$

where $W_{\mathrm{s}_1} \in \mathrm{R}^{d_\mathrm{h} \times d_\mathrm{c}}$, $b_{\mathrm{s}_1} \in \mathrm{R}^{d_\mathrm{h}}$, $W_{\mathrm{s}_2} \in \mathrm{R}^{d_\mathrm{e} \times d_\mathrm{h}}$, and $b_{\mathrm{s}_2} \in \mathrm{R}^{d_\mathrm{e}}$ are the learnable parameters of sentence embedding.

4.4 Sentence Embedding with Web Images

In addition to the sentence embedding model, we propose to extend the sentence embedding with web images. To enhance the sentence embedding, we retrieve relevant web images that are expected to disambiguate semantics of the sentence. For example, the word "keyboard" can be interpreted as a musical instrument or an input device for computers. If the word comes with "play," the meaning of "keyboard" narrows down to a musical instrument. This means that a specific combination of words can reduce the possible visual concepts relevant to the sentence, which may not be fully encoded even with the state-of-the-art RNN-based approach like [24].

We propose to take this into account by using web image search results. Since most image search engines use surrounding text to retrieve images, we can expect that they are responsive to such word combinations. Consequently, we retrieve web images using the input sentence as a query and download the results. The web images are fused with the input sentence by applying a two-branch neural network as shown in Figure 10.

This sentence embedding model consists of two branches that merge the outputs of a CNN-based network for web images and an RNN-based network for a sentence described in Section 4.3. Before computing the sentence embedding, we download top-K results of web image search with the input sentence as a query. Let $Z = \{z_1 \dots z_K\}$ be a set of web images. We utilize the same architecture as the video embedding and compute an intermediate representation $e_z \in \mathrm{R}^{d_\mathrm{e}}$ that integrates the web images as:

$$e_\mathrm{z} = \operatorname*{mean}_{z \in Z}[\tanh(W_{\mathrm{z}_2} \tanh(W_{\mathrm{z}_1} z + b_{\mathrm{z}_1}) + b_{\mathrm{z}_2})], \qquad (13)$$

where $W_{\mathrm{z}_1} \in \mathrm{R}^{d_\mathrm{h} \times d_\mathrm{v}}$, $b_{\mathrm{z}_1} \in \mathrm{R}^{d_\mathrm{h}}$, $W_{\mathrm{z}_2} \in \mathrm{R}^{d_\mathrm{e} \times d_\mathrm{h}}$, and $b_{\mathrm{z}_2} \in \mathrm{R}^{d_\mathrm{e}}$ are the leanable parameters of the two fully-connected layers.

Once the outputs e_z are computed, the sentence embedding using web images y_z is computed as:

$$y_z = \frac{1}{2}(y + e_z). \tag{14}$$

By this simple mixture of y and e_z, the sentence and web images directly influence the sentence embedding.

4.5 Joint Learning of Embedding Models

We jointly train both embedding models for videos and sentences using pairs of videos and associated sentences in a training set by minimizing the contrastive loss [4]. In our approach, the contrastive loss decreases when embeddings of videos and sentences with similar semantics get closer to each other in the embedding space, and those with dissimilar semantics get farther apart.

The training process requires a set of positive and negative video-sentence pairs. A positive pair contains a video and a sentence that are semantically relevant, and a negative pair contains irrelevant ones. During training, we get a positive pair by sampling a video and its description. Let $\{(x_n, y_n) \mid n = 1, \dots, N\}$ be the set of positive pairs. Given a positive pair (x_n, y_n), we sample irrelevant sentences and compute their embeddings $\mathcal{Y}' = \{y'_1 \dots y'_{N_c}\}$, as well as, videos $\mathcal{X}' = \{x'_1 \dots x'_{N_c}\}$ from the training set, which are used to build two sets of negative pairs $\{(x_n, y') \mid y' \in \mathcal{Y}'\}$ and $\{(x', y_n) \mid x' \in \mathcal{X}'\}$. Our embedding models for sentences and videos are jointly optimized by minimizing the contrastive loss defined as:

$$\begin{aligned} Loss(x_n, x_n) = \frac{1}{1 + 2N_c} \Big\{ & d(x_n, y_n) \\ & + \sum_{y' \in \mathcal{Y}'} \max(0, \alpha - d(x_n, y')) \\ & + \sum_{x' \in \mathcal{X}'} \max(0, \alpha - d(x', y_n)) \Big\}, \end{aligned} \tag{15}$$

where $d(\cdot, \cdot)$ denotes euclidean distance between embeddings. The hyperparameter α is a margin. Negative pairs with smaller distances than α are penalized. Margin α is set to the largest distance of positive pairs before training so that most negative pairs influence the model parameters at the beginning of training.

4.6 Experiments

Dataset: We used the YouTube dataset [2] consisting of 80K English descriptions for 1,970 videos. We first divided the dataset into 1,200, 100, and 670 videos for training, validation, and test, respectively, as in [66]. Then, we extracted five-second clips from each original video in a sliding-window manner. As a result, we obtained 8,001, 628, and 4,499 clips for the training, validation, and test sets, respectively. For each clip, we picked five ground truth descriptions out of those associated with its original video.

We collected top-5 image search results for each sentence using the Bing image search engine. We used a sentence modified by lowercasing and punctuation removal as a query. In order to eliminate cartoons and clip art, the image type was limited to photos using Bing API.

Video Retrieval: Given a video and a query sentence, we extracted five-second video clips from the video and computed Euclidean distances from the query to the clips. We used their median as the distance of the original video and the query. We ranked the videos based on the distance to each query and recorded the rank of the ground truth video.

Sentence Retrieval: For the sentence retrieval task, we ranked sentences for each query video. We computed the distances between a sentence and a query video in the same way as the video retrieval task. Note that each video has five ground truth sentences; thus, we recorded the highest rank among them.

Evaluation Metrics: We report recall rates at top-1, -5, and -10, the average and median rank, which are standard metrics employed in the retrieval evaluation. We found that some videos in the dataset had sentences whose semantics were almost the same (*e.g.*, "A group of women is dancing" and "Women are dancing"). For the video that is annotated with one of such sentences, the other sentence is treated as incorrect with the recall rates, which does not agree with human judges. Therefore, we employed additional evaluation metrics widely used in the description generation task, *i.e.*, CIDEr, BLUE@4, and METEOR [3]. They compute agreement scores in different ways using a retrieved sentence and a set of ground truth ones associated with a query video. Thus, these metrics give high scores for semantically relevant sentences even if they are not annotated to a query video. We computed the scores of the top-ranked sentence for each video using the evaluation script provided in the Microsoft COCO Evaluation Server [3]. In our experiments, all ground truth descriptions for each original video are used to compute these scores.

4.6.1 Effects of Each Component of Our Approach

In order to investigate the influence of each component of our approach, we tested some variations of our full model. The scores of the models on the video and sentence retrieval tasks are shown in Table 5. Our full model that computes sentence embedding using web images is denoted by ALL_2. ALL_1 is a variation of ALL_2 that computes embeddings with one fully-connected layer with the unit size of d_e. Comparison between ALL_1 and ALL_2 indicates that the number of fully-connected layers in embedding is not essential.

Our model which does not use web images to compute sentence embeddings is denoted by VS. The comparison between our full model ALL and VS reveals the contributions of web images. VGG+ALL_2 had better average rank (aR) than VGG+VS on both video and sentence retrieval, and comparison between GoogLeNet+ALL_2 and GoogLeNet+VS also shows a clear advantage of incorporating web images.

We also tested a model without sentences, which is denoted by VI. In VI, the sentence embeddings are computed only from web images. We investigated the effect of using both sentences and web images by comparing VI to our full model ALL_2. The results show that sentences are necessary. The comparison between VI and VS also indicates that sentences provide main cues for the retrieval task.

The scores of retrieved sentences computed by CIDEr, BLEU@4, and METEOR are shown in Table 6. In all metrics, our full model using both sentences and web images (ALL_1 and ALL_2) outperformed to other models (VS and VI). In summary, contributions by sentences and web images were non-trivial, and the best performance was achieved by using both of them.

Table 5 Video and sentence retrieval results. R@K is recall at top K results (higher values are better). aR and mR are the average and median of rank (lower values are better). Bold values denote best scores of each metric.

	Video retrieval					Sentence retrieval				
Models	**R@1**	**R@5**	**R@10**	**aR**	**mR**	**R@1**	**R@5**	**R@10**	**aR**	**mR**
Random Ranking	0.15	0.79	1.48	335.92	333	0.22	0.69	1.32	561.32	439
VGG+VS	6.12	21.88	33.22	58.98	24	7.01	18.66	27.16	131.33	35
VGG+VI	4.03	13.70	21.40	94.62	48	5.67	17.91	28.21	116.86	38
VGG+ALL$_1$	6.48	20.15	30.51	59.53	26	**10.60**	25.22	36.42	85.90	21
VGG+ALL$_2$	5.97	21.31	32.54	56.01	24	8.66	22.84	33.13	100.14	29
GoogLeNet+VS	7.49	22.84	33.10	54.14	22	8.51	21.34	30.45	114.66	33
GoogLeNet+VI	4.24	16.42	24.96	84.48	41	6.87	17.31	30.00	96.78	30
GoogLeNet+ALL$_1$	5.52	18.93	28.90	60.38	28	9.85	**27.01**	**38.36**	**75.23**	**19**
GoogLeNet+ALL$_2$	**7.67**	**23.40**	**34.99**	**49.08**	**21**	9.85	24.18	33.73	85.16	22
ST [24]	2.63	11.55	19.34	106.00	51	2.99	10.90	17.46	241.00	77
DVCT [64]	-	-	-	224.10	-	-	-	-	236.27	-

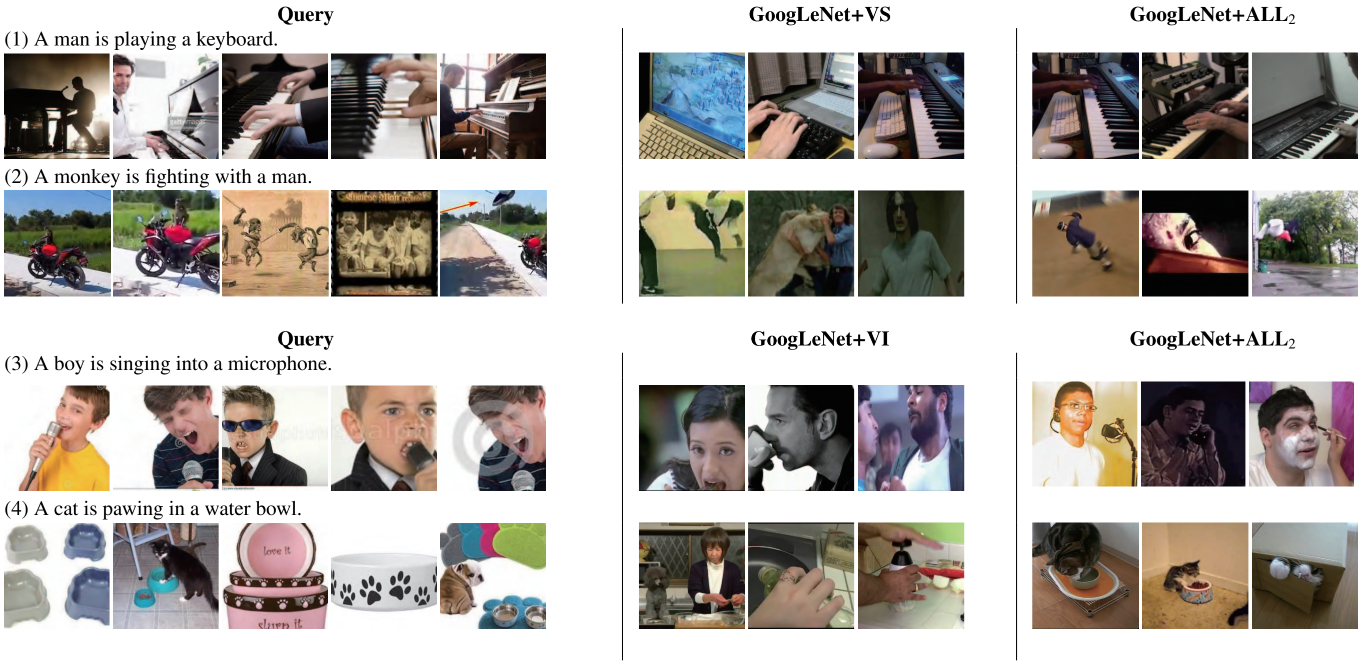

Query
(1) A man is playing a keyboard.
(2) A monkey is fighting with a man.
GoogLeNet+VS
GoogLeNet+ALL$_2$
Query
(3) A boy is singing into a microphone.
(4) A cat is pawing in a water bowl.
GoogLeNet+VI
GoogLeNet+ALL$_2$

Fig. 11 Examples of video retrieval results. Left: Query sentences and web images. Center: Top-3 retrieved videos by GoogLeNet+VS and VI. Right: Top-3 retrieved videos by GoogLeNet+ALL$_2$.

Table 6 Evaluated scores of retrieved sentences. All values are reported in percentage (%). Higher scores are better.

Models	**CIDEr**	**BLEU**	**METEOR**
VGG+VS	30.44	27.16	25.74
VGG+VI	29.00	22.42	22.99
VGG+ALL$_1$	42.52	**30.81**	**27.77**
VGG+ALL$_2$	32.56	27.39	26.58
GoogLeNet+VS	33.82	26.97	25.99
GoogLeNet+VI	35.08	24.56	24.16
GoogLeNet+ALL$_1$	**43.52**	29.99	27.48
GoogLeNet+ALL$_2$	38.08	29.28	26.50

Some examples of retrieved videos by GoogLeNet+VS, GoogLeNet+VI, and GoogLeNet+ALL$_2$ are shown in Figure 11. These results suggest that web images reduced the ambiguity of queries' semantics by providing hints on their visual concepts. For example, with the sentence (1) "A man is playing a keyboard," retrieval results of GoogleNet+VS includes two videos of a keyboard on a laptop as well as one on a musical instrument. On the other hand, all top-3 results by GoogleNet+ALL$_2$ are about musical instruments. We observed that web images retrieved by the query (1) included several images of musical instruments, which looked to be helpful to clarify the semantics of the query. However, irrelevant image search results can harm the video retrieval performance. A query "A monkey is fighting with a man" in (2) resulted in irrelevant web images, and our full model failed to get correct videos.

Compared to GoogLeNet+VI, our full model obtained more videos with relevant content for other queries. These results suggest that both sentence and web images are important for the performance of content-based video retrieval. The example in (4) also got irrelevant images as in (2), but this result indicates that our model may recover from irrelevant image search results by combining a query sentence.

4.6.2 Comparison to Prior Work

The approach for image and sentence retrieval by Kiros *et al.* [24] applies linear transformations to CNN-based image and RNN-based sentence representations to embed them into a common space. Note that their model was designed for the image and sentence retrieval tasks; thus, we extracted the middle frame as a keyframe and trained the model with pairs of a keyframe and a sentence. Xu *et al.* [64] introduced neural network-based embedding models for videos and sentences. Their approach embeds videos and SVO triplets extracted from sentences into an embed-

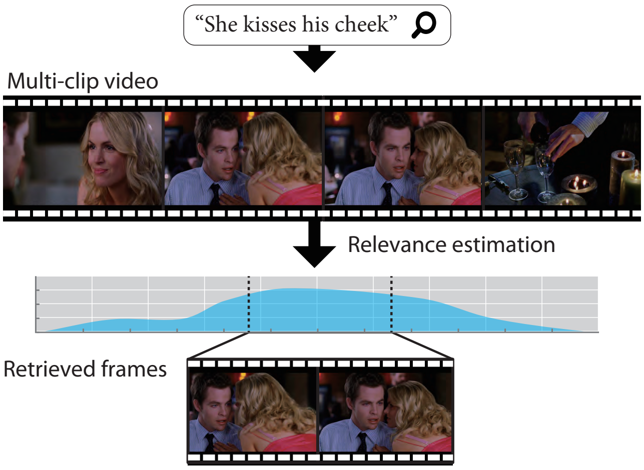


Fig. 12 Given a natural language query, fine-grained video retrieval finds video frames which the query describes. An input video consists of multiple video clips.

ding space. Kiros *et al.*'s and Xu *et al.*'s approaches are denoted by ST and DVCT, respectively.

Scores in Table 5 indicates that our model clearly outperformed prior work in both video and sentence retrieval tasks. There is a significant difference in performance of DVCT and others. ST and ours encode all words in a sentence, while DVCT only encodes its SVO triplets. This suggests that using all words in a sentence together with an RNN is necessary to get good embeddings.

5. Summary

We developed neural network-based embedding models for videos and sentences. Our embedding models are tested on the task of content-based video and sentence retrieval. The experimental results demonstrated that our embedding model extended with web images can disambiguate the semantics of sentences and benefits the retrieval performance. The future work includes the development of a video embedding that considers temporal structures of videos.

6. Representation Learning for Fine-grained Video Retrieval by Sentence Queries

In this section, we address to learn time-varying representations for content-based video retrieval (CBVR). The embedding models in Section 4 encode a video clip into one feature vector. However, that approach cannot capture the change of semantics along time. In order to represent the dynamic change of content within a video clip, we propose to produce a sequence of feature vectors as a video representation. We expect that this time-varying representation is helpful to model real-world videos such as movies or YouTube videos, which are long and consist of multiple video clips.

As in Section 4, we try to map both sentences and videos into a common embedding space, where a video is represented by a sequence of feature vectors. One interesting application of this representation is localizing content in a multi-clip video with a natural language query. Given a description, *e.g.* "She kisses his cheek", we would like to find corresponding short video clips from a long video (Figure 12). We call this task as fine-grained video retrieval (FGVR). In contrast to existing CBVR tasks, FGVR aims to handle more complex videos which may have multiple clips and varying content within a video. Thus, we expect that FGVR techniques contribute to many applications for real-world videos, for example, scene search from a lengthy video, and alignment of roughly annotated metadata and videos.

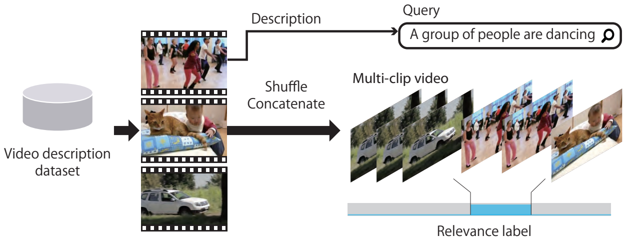


Fig. 13 FGVR examples are generated from video-description datasets. A video clip associated with a description is combined with randomly sampled videos. This results in a multi-clip video and a sentence which describes only a part of the video.

In this section, we describe representation learning by solving the FGVR task. We construct FGVR models that encode videos and sentences into cross-modal representations as in Section 4. The models are trained to localize video content which is semantically relevant to a sentence query. The task of FGVR works as strong supervision that makes a model encode time-varying semantics into a sequential representation, as well as, map videos and sentences into a cross-modal embedding space.

Most methods that temporally associate video content with languages, such as action localization, use frame-level labels indicating the start and the end point of the desired content. However, there do not exist many datasets that have multi-clip videos and sentences with temporal annotation. Making a dataset that is large enough to develop recent deep neural network models will require an immense amount of human intervention, thus we facilitate representation learning by synthesizing examples using existing datasets. While we do not have video-sentence datasets with temporal annotation, there are several large-scale datasets that provide videos and their descriptions only [2], [44], [47], [62]. We propose to compile a query sentence and a multi-clip video with temporal annotation from video-description datasets and a training scheme using the synthesized video-query pairs. As our data generation scheme can be applied to any video-description datasets, we can scale training datasets. Importantly, the experimental results demonstrate that our training scheme enables FGVR models to localize query-relevant content in real-world videos, while the models are trained on synthesized videos.

6.1 Fine-grained Video Retrieval

In the FGVR task, the input is a video consisting of multiple clips and a natural language query. The goal is to retrieve a subset of frames whose content is semantically relevant to the query (Figure 12). Specifically, given a sentence and video frames $V = \{v_1, \dots v_T\}$, where v_t is a visual feature extracted from the t-th frame, FGVR estimates relevance scores $R = \{r_1, \dots, r_T\}$ at each time step to retrieve frames. This task is similar to the video retrieval task for finding videos in a database which are relevant to a query. However, video retrieval tasks often implicitly assume that each video in the dataset is short and can be represented by a single query sentence. This assumption is not valid for most videos, *e.g.*, broadcast programs, movies, and even YouTube videos. A majority of these videos are lengthy and come with multiple concepts or scenes. The FGVR task relaxes this assumption; that is only a small part of the target video is relevant to a sentence query.

6.2 Data Generation

Training deep models usually requires large-scale datasets. Since there are no existing datasets for FGVR, we compile training examples for FGVR by extending the existing CBVR datasets. For FGVR examples, videos must 1) consist of multiple clips, 2) have corresponding query sentence related to a part of the video, and 3) be annotated with frame-level relevance labels. Since there is no dataset tailored for this task, we make video and query pairs from a large-scale video-description dataset, such as [44], [62].

The data generation using a video-description dataset is illustrated in Figure 13. To get a video consisting of multiple clips, we sample several video clips and their corresponding descriptions. We then choose one of the descriptions as a query sentence and concatenate the video clips in random order. Concatenation of multiple videos results in shot boundaries like most edited videos. The frames in a video clip corresponding to the selected query sentence are labeled as relevant frames, and other frames as irrelevant ones. By doing this, we can generate a number of videos where only a small part of it is relevant to a query sentence. Our data generation scheme can be applied to any dataset which provides videos and descriptions. This enables us to train FGVR methods on diverse videos provided by existing datasets.

6.3 Models for FVGR

We introduce several video embedding models that read video frames and produce a sequence of feature vectors. In order to cover possible models to capture content dynamics, we develop models with clip-level and frame-level video encoding. Each video clip or frame and a query sentence are mapped to a common feature space, and we estimate the relevance between them by computing the similarity of their representations in the feature space. In all methods, we employ the `pool5` layer of ResNet-50 [15] to extract visual features V from video frames.

6.3.1 Text Embedding Models

For text encoding, we employ two models that encode a sequence of words $\{w_1, \ldots, w_N\}$ into a vector representation t, where w_n is a word embedding vector. One is the word pooling-based model (**W-Pool**). With W-Pool model, input word embeddings are averaged to be transformed with a fully-connected layer.

The other is the word LSTM model (**W-LSTM**) that encodes a sequence of word embeddings with an LSTM layer, *i.e.*,

$$h_n, c_n = \mathrm{LSTM}(w_n, h_{n-1}, c_{n-1}), \tag{16}$$

where h_n and c_n are a hidden state and a memory cell of the LSTM layer, respectively. We employ the last hidden state h_N as a representation of the sentence in the common feature space.

6.3.2 Dynamic Video Embedding Models

Clip-level Video Embedding

One possible approach is to divide an input video into short video clips and outputs a feature vector for each video clip as illustrated in Figure 14 (left). We call this approach as a clip-level approach. We test two temporal video segmentation for this approach: Ground truth video segmentation uses clip boundaries in synthesized videos, and uniform segmentation divides videos with a uniform interval. Similarly to [56], we implement two neural network models that take a sequence of frames $\{v_{t_s}, \ldots, v_{t_e}\}$ in a video clip as input and produce a vector representation x that summarizes the frames.

Frame pooling (F-Pool): This model summarizes the frames $\{v_{t_s}, \ldots, v_{t_e}\}$ in a video clip by average pooling. The averaged feature vectors are fed to a fully-connected layer with the hyperbolic tangent non-linearity. In this work, we set the unit size of the fully-connected layer to 256.

Weighted average (WA): This model incorporates the soft-attention mechanism [66] in frame pooling. The weights a_i of the frame v_i is computed based on the frame feature and a query sentence by

$$e_i = w_{\mathrm{a}}^{\mathrm{T}} \tanh(W_{\mathrm{a}}[y, v_i] + b_{\mathrm{a}}), \tag{17}$$

$$a_i = \exp(e_i) / \sum_{j=t_s}^{t_n} \exp(e_j), \tag{18}$$

where w_{a}, W_{a}, and b_{a} are learnable parameters, and $[\cdot,\cdot]$ denotes the concatenation of vectors. The vector y is a text embedding computed with a text encoding model described in Section 6.3.1. Using the weights, we obtain a weighted sum of frames and feed it to a fully-connected layer to get a clip representation x as:

$$\tilde{v}_{\mathrm{wa}} = \sum_{i=t_s}^{t_e} a_i v_i, \tag{19}$$

$$x = \tanh(W_{\mathrm{wa}} \tilde{v}_{\mathrm{wa}} + b_{\mathrm{wa}}), \tag{20}$$

where W_{wa} and b_{wa} are parameters of the fully-connected layer.

Frame-level Video Encoding

In the clip-level approach, an input video needs to be segmented beforehand; however, segment boundaries are not always available, and temporal video segmentation itself is still a challenging task. Another direction for this task is to read frames and produce a feature vector at each time step as in Figure 14 (right). For this approach, we implemented three models that encodes video frames to a sequence of vector representations $\{x_1, \ldots, x_T\}$.

Sliding window (SW): This model reads an input frame sequence in the sliding window fashion. At each time step, we perform average pooling over frames within a temporal window and feed its output to a fully-connected layer in the same way as the F-Pool model. We set the temporal window size to 5 and the model reads frames with a stride of 1.

Bidirectional-LSTM (biLSTM): The biLSTM model utilizes a two-layer LSTM network that reads frames in forward and backward directions. This bidirectional LSTM is employed in several recent works to model video frames [16], [68]. Hidden states at each time step are concatenated and transformed with a

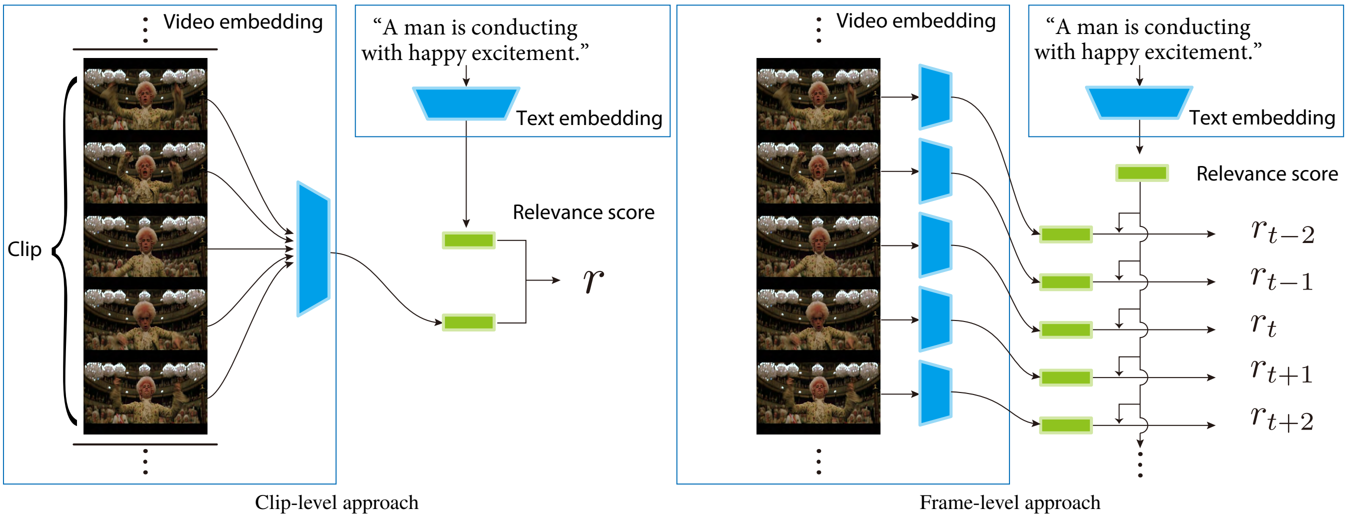


Fig. 14 Illustration of clip-level (left) and frame-level (right) embedding models. Clip-level model summarizes frames in a clip and produces a feature vector. On the other hand, frame-level approach outputs a feature vector for every frame. These models are trained to localize video parts, which are semantically relevant to a query sentence.

fully-connected layer as:

$$x_t = \tanh(W[h_t^{\text{forward}}, h_t^{\text{backward}}] + b), \qquad (21)$$

where h_t^{forward} and h_t^{backward} are hidden states of the forward-LSTM and the backward-LSTM layers for the input frame v_t, respectively.

Fully-connected (FC): This model is a variation of the biLSTM model. We remove the temporal connection by replacing the bidirectional LSTM layers with two fully-connected layers. This model estimates relevance scores in a frame-by-frame fashion. Therefore, this model is equivalent to frame-level CBVR.

6.4 Similarity Metrics for Relevance Score

After a vector representations for clips or frames are obtained, relevance scores $R = \{r_1, \ldots, r_T\}$ are computed. In this study, we test cosine similarity and partial order similarity [56]. Partial order similarity between two vectors is computed as:

$$r_t = -\|\max(y - x_t, 0)\|^2, \qquad (22)$$

where y and x_t are non-negative vectors. Therefore, we compute the absolute values of the outputs of models and apply L2-normalization before computing the partial order similarity. Note that partial order similarity is not order-invariant.

6.5 Training

We train the models described in Section 6.3 using video-sentence pairs synthesized as in Section 6.2. The models for videos and sentences are jointly trained so that the query relevance scores of relevant frames are larger than those of others. We compute an averaged score of relevant and irrelevant frames and update the model to make the difference between the scores larger. During the training, a model is trained by minimizing the loss computed from predicted relevance score $R = \{r_1, \ldots, r_T\}$ and ground truth label $L = \{l_1, \ldots, l_T\}$ as:

$$\text{Loss}(R, L) = \max(-R_{\text{pos}} + R_{\text{neg}} + \mu, 0), \qquad (23)$$

$$R_{\text{pos}} = \frac{1}{N_{\text{pos}}} \sum_{t=1}^{T} l_t r_t, \qquad (24)$$

$$R_{\text{neg}} = \frac{1}{N_{\text{neg}}} \sum_{t=1}^{T} (1 - l_t) r_t, \qquad (25)$$

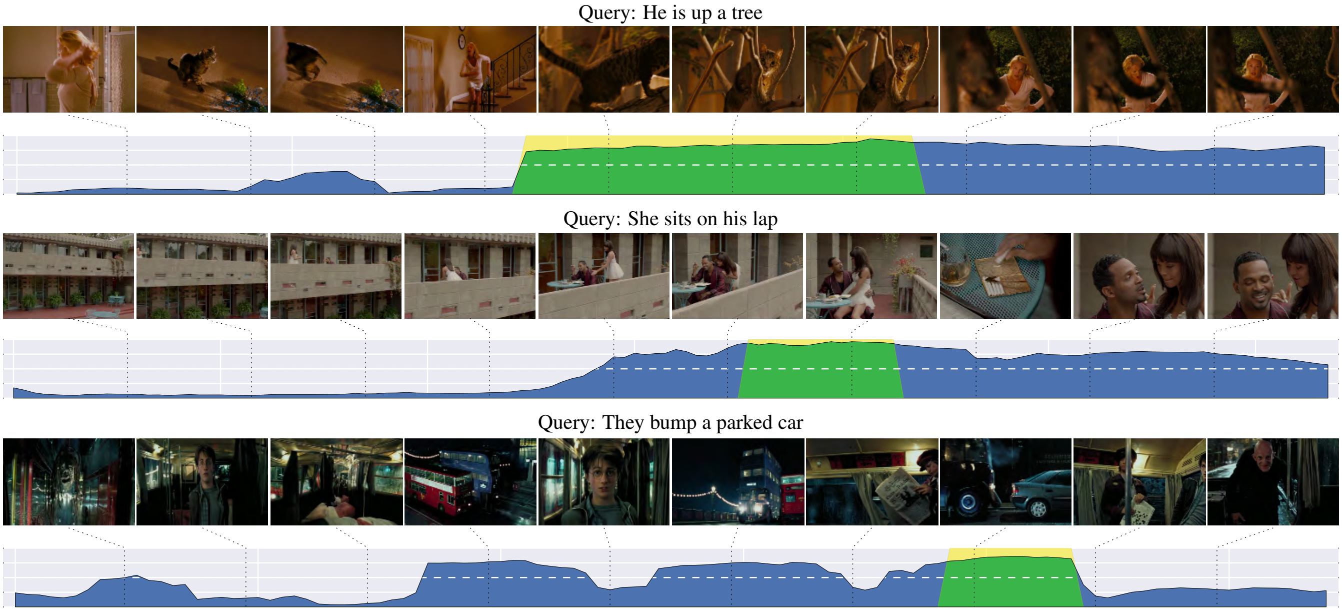
where N_{pos} and N_{neg} are the number of relevant and irrelevant frames in a video, respectively. l_t is a label representing the frame's relevance/irrelevance to a query sentence and r_t is relevance score, which is computed as in Section 6.4. We set $l_t = 1$ if the frame is relevant, and otherwise 0. The parameter μ is a predefined margin to penalize the smaller difference between the averaged score of relevant and irrelevant frames than the margin. Models of the clip-level approach do not produce frame-level scores, thus we spread a clip-level score to all frames in the clip.

6.6 Experiments

We evaluated learned representations on the task of FGVR. We generated FGVR examples from two datasets, MSR Video to Text (MSR-VTT) [62] and the MPII Movie Description dataset (MPII-MD) [45], and investigate the performance of each model.

6.6.1 Datasets

We tested video and sentence encoding models on the MSR-VTT and the MPII-MD datasets. The MSR-VTT dataset includes 10,000 YouTube video clips, and 20 descriptions are annotated for each video clip. MPII Movie Description dataset has 118,507 video clips from movies, and each video clip is annotated with one description. For the MSR-VTT dataset, we used training and test splits provided by the MSR-VTT official web page. For the MPII-MD dataset, we used splits for the LSMDC'16 movie annotation and retrieval task [56]. Word vocabulary is collected from descriptions in the training split. The descriptions were normalized by punctuation removal and lowercasing, then we compiled a vocabulary dictionary by sampling words occurring more than three times in training queries, which results in 8,935 words for the YouTube dataset and 10,066 words for the movie



Query: He is up a tree

Query: She sits on his lap

Query: They bump a parked car

Fig. 15 Examples of relevance score estimation by the biLSTM model on movie videos.

dataset. The videos were down-sampled at 5 fps and rescaled to 244 × 244. During training, we sampled two video clips for each video-description pair to create FGVR examples as in Figure 13. Since most video clips in the datasets have similar durations, which can be a strong prior, the first and the last few seconds of video clips are randomly trimmed so that the video clips have 20-100% of their original length. The average durations of videos compiled from the MSR-VTT dataset is 32 seconds and those from the MPII-MD dataset is 8.6 seconds.

6.6.2 Qualitative Evaluation

We show examples of the biLSTM model on short movie excerpts to demonstrate that the model can be used for real-world videos (see Figure 15). The ground truth parts are indicated by yellow areas in the figure. Note that the ground truth labels are based on where the query sentence is originally annotated in a video captioning dataset, and other frames can also be relevant to the query. Moreover, the start and end points of a specific event are ambiguous, especially in movies. The examples of top two rows show that the biLSTM can roughly localize content relevant to the queries. However, the biLSTM model failed by giving high scores for irrelevant frames in the bottom row. The input video of this example shows a dark scene, and the scene changes rapidly. Moreover, the video has unusual events since it is a fantasy movie. We assume that these characteristics of the input video made it difficult to capture the video content.

6.6.3 Quantitative Evaluation

We conducted a quantitative evaluation of predicting relevant frames from multi-clipped videos on the MSR-VTT and the MPII-MD datasets. We generated test videos in the same way as in Section 6.2 from test splits of the datasets. For each test sample, we computed frame-level relevance scores of a video to a query sentence, and then evaluated the performance with average precision (AP). We report the mean and the standard deviation (the values in parenthesis) of the AP scores over all test samples in Table 7. To compute AP, the clip-level scores were transformed to frame-level scores by simply spreading the clip-level score to all frames in the clip. The scores obtained by random score prediction are reported in the bottom row.

Overall, cosine similarity performs better than partial order similarity in this task. For clip-level approaches, there are no significant differences between models. Note that these scores with ground truth clip boundaries (GT) can be regarded as a sort of upper bounds of the clip-level approaches. We also report scores obtained by uniformly dividing an input video into three clips (UNI). These results suggest that the performance of clip-level FGVR methods highly relies on temporal video segmentation.

We can also observe that the frame-level approach (FC, SW, and biLSTM models), which does not require temporal video segmentation, achieves good retrieval performance on the MSR-VTT dataset. This suggests that video segmentation is not necessary for FGVR. From the comparison between models for the frame-level approach, we can see that incorporating nearby frames improves the performance. This might be because context obtained from other frames is helpful to understand a video content.

For the MPII-MD dataset, all baselines resulted in lower scores since videos and sentences in the dataset are more challenging. Many of the sentences often describe complex scenes, of which LSTM may have difficulties in encoding the semantics. Moreover, movies often have dark and low-contrast scenes, which may cause failures in understanding the visual content.

7. Summary

In this work, we propose to learn sequential vector representation for videos to encode dynamics of content within a video. Our video embedding model and sentence embedding model are jointly trained by solving the FGVR task to localize video content according to a query sentence. The FGVR results on some videos from movies suggest that our approach can retrieve video parts from real-world videos although our models are trained on generated video-query pairs. We expect that text embedding methods

Table 7 Mean average precision (AP) scores (%) of FGVR. GT denotes ground truth clip boundaries, and UNI denotes uniform segmentation.

video model /	clip	MSR-VTT		MPII-MD	
sentence model	boundaries	cosine	p-order	cosine	p-order
F-Pool / W-Pool	GT	**86.5 (27.9)**	80.9 (31.5)	**77.7 (33.2)**	73.6 (34.8)
	UNI	81.1 (22.5)	76.0 (25.2)	74.4 (26.6)	70.7 (27.4)
F-Pool / W-LSTM	GT	85.4 (28.7)	79.2 (32.3)	74.8 (34.3)	69.0 (35.8)
	UNI	80.1 (23.1)	75.9 (25.3)	72.5 (27.6)	68.2 (28.4)
WA / W-LSTM	GT	86.4 (28.0)	75.9 (33.7)	75.8 (34.0)	69.0 (35.9)
	UNI	79.7 (23.2)	71.0 (26.7)	72.6 (27.4)	67.4 (28.5)
FC / W-LSTM	—	80.9 (23.7)	75.7 (25.2)	73.1 (27.7)	63.3 (27.6)
SW / W-LSTM	—	83.3 (22.9)	76.3 (25.7)	73.5 (27.9)	69.8 (28.8)
biLSTM / W-LSTM	—	**83.8 (22.7)**	72.5 (25.7)	**76.1 (28.9)**	61.7 (26.5)
by chance	—	47.0 (12.2)		49.4 (17.6)	

that can handle long sentences, which have complex semantics, will be a key component for further improvement.

8. Conclusion

This paper has proposed several cross-modal representations for videos and languages. We have explored two approaches for developing cross-modal representations. One is manually designing a cross-modal representation for videos and languages. The other approach is data-driven representation learning. Evaluating the performance of representation is unclear; thus we investigate how our cross-modal representations work in practical tasks, such as video summarization and content-based video retrieval. Experimental results demonstrated that our cross-modal representations well capture the semantics of videos and languages, and benefit some applications described above.

One significant criticism for recent video understanding research is that settings of many video understanding tasks including content-based video retrieval and captioning do not involve temporal reasoning. Many of these tasks can be often solved by focusing on a single keyframe. For further improvement of cross-modal representations, it is insightful to explore novel tasks or applications that require temporal reasoning. There are several emerging applications, such as video question answering [18], [38], [54], and dense video captioning [25]. These are challenging tasks, and addressing these tasks will lead to a new framework to associate videos and languages.

References

[1] Alexe, B., Deselaers, T. and Ferrari, V.: What is an object?, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 73–80 (2010).

[2] Chen, D. L. and Dolan, W. B.: Collecting highly parallel data for paraphrase evaluation, *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 190–200 (2011).

[3] Chen, X., Fang, H., Lin, T., Vedantam, R., Gupta, S., Dollr, P. and Zitnick, C. L.: Microsoft COCO Captions: Data Collection and Evaluation Server, *arXiv preprint arXiv:1504.00325* (7 pages, 2015).

[4] Chopra, S., Hadsell, R. and LeCun, Y.: Learning a Similarity Metric Discriminatively, with Application to Face Verification, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1, pp. 539–546 (2005).

[5] Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E. and Darrell, T.: DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition, *IEEE International Conference on Machine Learning*, pp. 647–655 (2014).

[6] Evangelopoulos, G., Zlatintsi, A., Potamianos, A., Maragos, P., Rapantzikos, K., Skoumas, G. and Avrithis, Y.: Multimodal saliency and fusion for movie summarization based on aural, visual, and textual attention, *IEEE Transactions on Multimedia*, Vol. 15, No. 7, pp. 1553–1568 (2013).

[7] Farhadi, A., Hejrati, M., Sadeghi, M. A., Young, P., Rashtchian, C., Hockenmaier, J. and Forsyth, D.: Every picture tells a story: Generating sentences from images, *European Conference on Computer Vision*, pp. 15–29 (2010).

[8] Frey, B. J. and Delbert, D.: Clustering by passing messages between data points, *Science*, Vol. 315, pp. 972–976 (2007).

[9] Frome, A., Corrado, G. S., Shlens, J., Bengio, S., Dean, J., Ranzato, M. A. and Mikolov, T.: DeViSE: A Deep Visual-Semantic Embedding Model, *Advances in Neural Information Processing Systems*, pp. 2121–2129 (2013).

[10] Girshick, R., Donahue, J., Darrell, T. and Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587 (2014).

[11] Gong, B., Chao, W.-L., Grauman, K. and Sha, F.: Diverse sequential subset selection for supervised video summarization, *Advances in Neural Information Processing Systems*, pp. 2069–2077 (2014).

[12] Guadarrama, S., Venugopalan, S., Austin, U. T., Krishnamoorthy, N., Mooney, R., Malkarnenkar, G., Darrell, T. and Berkeley, U. C.: YouTube2Text: Recognizing and Describing Arbitrary Activities Using Semantic Hierarchies and Zero-shot Recognition, *International Conference on Computer Vision*, pp. 2712–2719 (2013).

[13] Gygli, M., Grabner, H., Riemenschneider, H. and van Gool, L.: Creating summaries from user videos, *European Conference on Computer Vision*, pp. 505–520 (2014).

[14] Gygli, M., Grabner, H. and Van Gool, L.: Video summarization by learning submodular mixtures of objectives, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3090–3098 (2015).

[15] He, K., Zhang, X., Ren, S. and Sun, J.: Deep Residual Learning for Image Recognition, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778 (2016).

[16] Hori, C., Hori, T., Lee, T.-Y., Sumi, K., Hershey, J. R. and Marks, T. K.: Attention-Based Multimodal Fusion for Video Description, *International Conference on Computer Vision*, pp. 4193–4202 (2017).

[17] Huang, C. R., Lee, H. P. and Chen, C. S.: Shot change detection via local keypoint matching, *IEEE Transactions on Multimedia*, Vol. 10, No. 6, pp. 1097–1108 (2008).

[18] Jang, Y., Song, Y., Yu, Y., Kim, Y. and Kim, G.: TGIF-QA: Toward Spatio-Temporal Reasoning in Visual Question Answering, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2758–2766 (2017).

[19] Ji, S., Xu, W., Yang, M. and Yu, K.: 3D Convolutional Neural Networks for Human Action Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 35, No. 1, pp. 221–231 (online), available from ⟨http://ieeexplore.ieee.org/document/6165309/⟩ (2013).

[20] Johnson, J., Karpathy, A. and Fei-Fei, L.: DenseCap: Fully Convolutional Localization Networks for Dense Captioning, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4565–4574 (2016).

[21] Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R. and Fei-Fei, L.: Large-Scale Video Classification with Convolutional Neural Networks, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1725–1732 (online), DOI: 10.1109/CVPR.2014.223 (2014).

[22] Karpathy, A., Joulin, A. and Fei-Fei, L.: Deep Fragment Embeddings for Bidirectional Image Sentence Mapping, *Advances in Neural Information Processing Systems*, pp. 1889–1897 (2014).

[23] Khosla, A., Hamid, R., Lin, C.-J. and Sundaresan, N.: Large-scale video summarization using web-image priors, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2698–2705 (2013).

[24] Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A. and Fidler, S.: Skip-Thought Vectors, *Advances in Neural Information Processing Systems*, pp. 3276–3284 (2015).

[25] Krishna, R., Hata, K., Ren, F., Fei-Fei, L. and Niebles, J. C.: Dense-Captioning Events in Videos, *IEEE International Conference on Computer Vision*, pp. 706–715 (2017).

[26] Krizhevsky, A., Sutskever, I. and Hinton, G.: ImageNet classification with deep convolutional neural networks, *Advances in Neural Information Processing Systems*, pp. 1097–1105 (2012).

[27] Laganière, R., Bacco, R., Hocevar, A., Lambert, P., Païs, G. and Ionescu, B. E.: Video summarization from spatio-temporal features, *ACM TRECVid Video Summarization Workshop*, pp. 144–148 (2008).

[28] Le, D.-D., Phan, S., Vinh-Tiep, N., Benjamin, R., Nguyen, T. A., Hoang, V.-N., Ngo, T. D., Tran, M.-T., Watanabe, Y., Klinkigt, M., Hiroike, A., Duong, D. A., Miyao, Y. and Satoh, S.: NII-HITACHI-UIT at TRECVID 2016, *TRECVID Workshops* (25 pages, 2016).

[29] Le, Q. V. and Mikolov, T.: Distributed Representations of Sentences and Documents, *International Conference on Machine Learning*, pp. 1188–1196 (2014).

[30] Lee, J. and Abu-El-Haija, S.: Large-Scale Content-Only Video Recommendation, *IEEE International Conference on Computer Vision Workshops*, pp. 987 —- 995 (online), available from ⟨https://research.google.com/pubs/pub46446.html⟩ (2017).

[31] Li, S., Xiao, T., Li, H., Zhou, B., Yue, D. and Wang, X.: Person Search with Natural Language Description, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1970–1979 (online), available from ⟨http://arxiv.org/abs/1702.05729⟩ (2017).

[32] Li, Y., Merialdo, B. and Antipolis, S.: VERT: Automatic evaluation of video summaries, *ACM International Conference on Multimedia*, pp. 851–854 (2010).

[33] Lin, D., Fidler, S., Kong, C. and Urtasun, R.: Visual Semantic Search: Retrieving Videos via Complex Textual Queries, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2657–2664 (2014).

[34] Lin, T.-Y., Belongie, S. and Hays, J.: Learning deep representations for ground-to-aerial geolocalization, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5007–5015 (2015).

[35] Lu, Z. and Grauman, K.: Story-driven summarization for egocentric video, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2714–2721 (2013).

[36] Ma, Y., Lu, L., Zhang, H. and Li, M.: A user attention model for video summarization, *ACM International Conference on Multimedia*, pp. 533–542 (2002).

[37] Maybank, S.: A Survey on Visual Content-Based Video Indexing and Retrieval, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 41, No. 6, pp. 797–819 (2011).

[38] Mun, J., Seo, P. H., Jung, I. and Han, B.: MarioQA: Answering Questions by Watching Gameplay Videos, *IEEE International Conference on Computer Vision*, pp. 2867–2875 (online), available from ⟨http://arxiv.org/abs/1612.01669⟩ (2017).

[39] Nakashima, Y. and Yokoya, N.: Inferring what the videographer wanted to capture, *IEEE International Conference on Image Processing*, pp. 191–195 (2013).

[40] Otani, M., Nakashima, Y., Rahtu, E., Heikkilä, J. and Yokoya, N.: Learning Joint Representations of Videos and Sentences with Web Image Search, *European Conference on Computer Vision Workshops*, pp. 651–667 (2016).

[41] Perazzi, F., Krahenbuhl, P., Pritch, Y. and Hornung, A.: Saliency filters: Contrast based filtering for salient region detection, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 733–740 (2012).

[42] Potapov, D., Douze, M., Harchaoui, Z. and Schmid, C.: Category-specific video summarization, *European Conference on Computer Vision*, pp. 540–555 (2014).

[43] Redmon, J. and Farhadi, A.: YOLO9000: Better, Faster, Stronger, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7263–7271 (2017).

[44] Rohrbach, A., Rohrbach, M., Tandon, N. and Schiele, B.: A Dataset for Movie Description, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3202–3212 (2015).

[45] Rohrbach, A., Torabi, A., Rohrbach, M., Tandon, N., Pal, C., Larochelle, H., Courville, A. and Schiele, B.: Movie Description, *International Journal of Computer Vision*, Vol. 123, No. 1, pp. 94–120 (2017).

[46] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C. and Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge, *International Journal of Computer Vision*, Vol. 115, No. 3, pp. 211–252 (2015).

[47] Senina, A., Rohrbach, M., Qiu, W., Friedrich, A., Amin, S., Andriluka, M., Pinkal, M. and Schiele, B.: Coherent Multi-Sentence Video Description with Variable Level of Detail, *German Conference on Pattern Recognition*, pp. 184–195 (2014).

[48] Sharghi, A., Gong, B. and Shah, M.: Query-Focused Extractive Video Summarization, *European Conference on Computer Vision*, pp. 3–19 (2016).

[49] Simonyan, K. and Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recoginition, *International Conference on Learning Representation* (14 pages, 2015).

[50] Snoek, C. G. M. and Worring, M.: Concept-Based Video Retrieval, *Foundations and Trends in Information Retrieval*, Vol. 2, No. 4, pp. 215–322 (2009).

[51] Socher, R., Ganjoo, M., Manning, C. D. and Ng, A. Y.: Zero-Shot Learning Through Cross-Modal Transfer, *Advances in Neural Information Processing Systems*, pp. 935–943 (2013).

[52] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A.: Going Deeper with Convolutions, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9 (2015).

[53] Takamura, H. and Okumura, M.: Text summarization model based on maximum coverage problem and its variant, *Conference of the European Chapter of the Association for Computational Linguistics*, pp. 781–789 (2009).

[54] Tapaswi, M., Zhu, Y., Stiefelhagen, R., Torralba, A., Urtasun, R. and Fidler, S.: MovieQA: Understanding Stories in Movies through Question-Answering, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4631–4640 (2016).

[55] Taskiran, C. M., Pizlo, Z., Amir, A., Ponceleon, D. and Delp, E. E. J.: Automated video program summarization using speech transcripts, *IEEE Transactions on Multimedia*, Vol. 8, No. 4, pp. 775–790 (2006).

[56] Torabi, A., Tandon, N. and Sigal, L.: Learning Language-Visual Embedding for Movie Understanding with Natural-Language, *European Conference on Computer Vision Workshops* (13 pages, 2016).

[57] Toutanova, K., Klein, D., Manning, C. D. and Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network, *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 173–180 (2003).

[58] Tschiatschek, S., Iyer, R. K., Wei, H. and Bilmes, J. A.: Learning mixtures of submodular functions for image collection summarization, *Advances in Neural Information Processing Systems*, pp. 1413–1421 (2014).

[59] Ueki, K., Kikuchi, K., Saito, S. and Kobayashi, T.: Waseda at TRECVID 2016: Ad-hoc Video Search, *TRECVID Workshops* (5 pages, 2016).

[60] Wang, M., Hong, R., Li, G., Zha, Z. J., Yan, S. and Chua, T. S.: Event driven web video summarization by tag localization and key-shot identification, *IEEE Transactions on Multimedia*, Vol. 14, No. 4, pp. 975–985 (2012).

[61] Xiong, B., Kim, G. and Sigal, L.: Storyline Representation of Egocentric Videos With an Applications to Story-Based Search, *IEEE International Conference on Computer Vision*, pp. 4525–4533 (2015).

[62] Xu, J., Mei, T., Yao, T. and Rui, Y.: MSR-VTT: A Large Video Description Dataset for Bridging Video and Language, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5288–5296 (2016).

[63] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R. and Bengio, Y.: Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, *International Conference on Machine Learning*, pp. 2048–2057 (2015).

[64] Xu, R., Xiong, C., Chen, W. and Corso, J.: Jointly modeling deep video and compositional text to bridge vision and language in a unified framework, *Association for the Advancement of Artificial Intelligence*, pp. 2346–2352 (2015).

[65] Yan, Q., Xu, L., Shi, J. and Jia, J.: Hierarchical saliency detection, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1155–1162 (2013).

[66] Yao, L., Ballas, N., Larochelle, H. and Courville, A.: Describing Videos by Exploiting Temporal Structure, *IEEE International Conference on Computer Vision*, pp. 4507 – 4515 (2015).

[67] Yu, Y., Ko, H., Choi, J. and Kim, G.: End-to-end Concept Word Detection for Video Captioning, Retrieval, and Question Answering, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3165–3173 (online), available from ⟨http://arxiv.org/abs/1610.02947⟩ (2017).

[68] Zeng, K.-H., Chen, T.-H., Niebles, J. C. and Sun, M.: Title Generation for User Generated Videos, *European Conference on Computer Vision*, pp. 609–625 (2016).

[69] Zhao, B. and Xing, E. P.: Quasi real-time summarization for consumer videos, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2513–2520 (2014).

[70] Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A. and Fidler, S.: Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books, *IEEE International Conference on Computer Vision*, pp. 19–27 (2015).