



# A Joint Sequence Fusion Model for Video Question Answering and Retrieval

Youngjae Yu, Jongseok Kim, and Gunhee Kim<sup>(✉)</sup>

Department of Computer Science and Engineering,  
Seoul National University, Seoul, Korea  
{yj.yu, js.kim}@vision.snu.ac.kr, gunhee@snu.ac.kr  
<http://vision.snu.ac.kr/projects/jsfusion/>

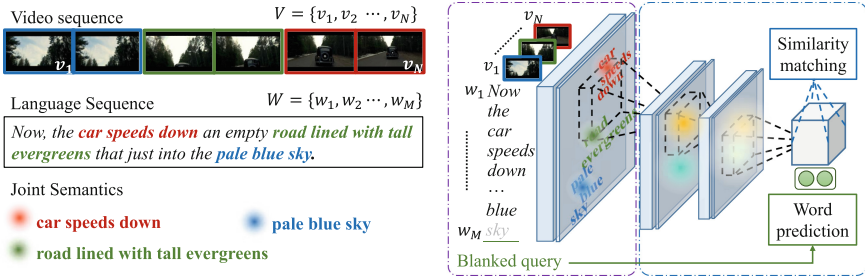
**Abstract.** We present an approach named JSFusion (Joint Sequence Fusion) that can measure semantic similarity between any pairs of multimodal sequence data (*e.g.* a video clip and a language sentence). Our multimodal matching network consists of two key components. First, the *Joint Semantic Tensor* composes a dense pairwise representation of two sequence data into a 3D tensor. Then, the *Convolutional Hierarchical Decoder* computes their similarity score by discovering hidden hierarchical matches between the two sequence modalities. Both modules leverage hierarchical attention mechanisms that learn to promote well-matched representation patterns while prune out misaligned ones in a bottom-up manner. Although the JSFusion is a universal model to be applicable to any multimodal sequence data, this work focuses on video-language tasks including multimodal retrieval and video QA. We evaluate the JSFusion model in three retrieval and VQA tasks in LSMDC, for which our model achieves the best performance reported so far. We also perform multiple-choice and movie retrieval tasks for the MSR-VTT dataset, on which our approach outperforms many state-of-the-art methods.

**Keywords:** Multimodal retrieval · Video question and answering

## 1 Introduction

Recently, various video-language tasks have drawn a lot of interests in computer vision research [1–3], including video captioning [4–9], video question answering (QA) [10, 11], and video retrieval for a natural language query [8, 12, 13]. To solve such challenging tasks, it is important to learn a hidden joint representation between word and frame sequences, for correctly measuring their semantic similarity. Video classification [14–18] can be a candidate solution, but tagging only a few labels to a video may be insufficient to fully relate multiple latent events in the video to a language description. Thanks to recent advance of deep representation learning, many methods for multimodal semantic embedding (*e.g.* [19–21]) have been proposed. However, most of existing methods embed each of visual and language information into a single vector, which is often insufficient especially for a video and a natural sentence. With single vectors for the two sequence

modalities, it is hard to directly compare multiple relations between subsets of sequence data (*i.e.* matchings between subevents in a video and short phrases in a sentence), for which hierarchical matching is more adequate. There have been some attempts to learn representation of hierarchical structure of natural sentences and visual scenes (*e.g.* [22, 23] using recursive neural networks), but they require groundtruth parse trees or segmentation labels (Fig. 1).



**Fig. 1.** The intuition of the Joint Sequence Fusion (JSFusion) model. Given a pair of a video clip and a language query, Joint Semantic Tensor (in purple) encodes a pairwise joint embedding between the two sequence data, and Convolutional Hierarchical Decoder (in blue) discovers hierarchical matching relations from JST. Our model is easily adaptable to many video QA and retrieval tasks. (Color figure online)

In this paper, we propose an approach that can measure semantic similarity between any pairs of multimodal sequence data, by learning bottom-up recursive matches via attention mechanisms. We apply our method to tackle several video question answering and retrieval tasks. Our approach, named as Joint Sequence Fusion (JSFusion) model, consists of two key components. First, the Joint Semantic Tensor (JST) performs dense Hadamard products between frames and words and encodes all pairwise embeddings between the two sequence data into a 3D tensor. JST further takes advantage of learned attentions to refine the 3D matching tensor. Second, the Convolutional Hierarchical Decoder (CHD) discovers local alignments on the tensor, by using a series of attention-based decoding modules, consisting of convolutional layers and gates. These two attention mechanisms promote well-matched representation patterns and prune out misaligned ones in a bottom-up manner. Finally, CHD obtains hierarchical composable representations of the two modalities, and computes a semantic matching score of the sequence pair.

We evaluate the performance of our JSFusion model on multiple video question answering and retrieval tasks on LSMDC [1] and MSR-VTT [2] datasets. First, we participate in three challenges of LSMDC: multiple-choice test, movie retrieval, and fill-in-the-blank, which require the model to correctly measure a semantic matching score between a descriptive sentence and a video clip, or to predict the most suitable word for a blank in a sentence for a query video. Our JSFusion model achieves the best accuracies reported so far with significant margins for the lsmdc tasks. Second, we newly create multiple-choice and

movie retrieval annotations for the MSR-VTT dataset, on which our approach also outperforms many state-of-the-art methods in diverse video topics (*e.g.* *TV shows*, *web videos*, and *cartoons*).

We summarize the contributions of this work as follows.

1. We propose the Joint Sequence Fusion (JSFusion) model, consisting of two key components: JST and CHD. To the best of our knowledge, it is a first attempt to leverage recursively learnable attention modules for measuring semantic matching scores between multimodal sequence data. Specifically, we propose two different attention models, including soft attention in JST and Conv-layers and Conv-gates in CHD.
2. To validate the applicability of our JSFusion model, especially on video question answering and retrieval, we participate in three tasks of LSMDC [1], and achieve the best performance reported so far. We newly create video retrieval and QA benchmarks based on MSR-VTT [2] dataset, on which our JSFusion outperforms many state-of-the-art VQA models. Our source code and benchmark annotations are publicly available in our project page.

## 2 Related Work

Our work can be uniquely positioned in the context of two recent research directions: video retrieval and video question answering.

**Video Retrieval with Natural Language Sentences.** Visual information retrieval with natural language queries has long been tackled via joint visual-language embedding models [12, 19, 24–28]. In the video domain, it is more difficult to learn latent relations between a sequence of frames and a sequence of descriptive words, given that a video is not simply a multiple of images. Recently, there has been much progress in this line of research. Several deep video-language embedding methods [8, 12, 13] has been developed by extending image-language embeddings [20, 21]. Other recent successful methods benefit from incorporating concept words as semantic priors [9, 29], or relying on strong representation of videos like RNN-FV [30]. Another dominant approach may be leveraging RNNs or their variants like LSTM to encode the whole multimodal sequences (*e.g.* [9, 12, 29, 30]).

Compared to these existing methods, our model first finds dense pairwise embeddings between the two sequences, and then composes higher-level similarity matches from fine-grained ones in a bottom-up manner, leveraging hierarchical attention mechanisms. This idea improves our model’s robustness especially for local subset matching (*e.g.* at the activity-phrase level), which places our work in a unique position with respect to previous works.

**Video Question Answering.** VQA is a relatively new problem at the intersection of computer vision and natural language research [31–33]. Video-based VQA is often recognized as a more difficult challenge than image-based one, because video VQA models must learn spatio-temporal reasoning to answer problems, which requires large-scale annotated data. Fortunately, large-scale video

QA datasets have been recently emerged from the community using crowdsourcing on various sources of data (*e.g.* movies for MovieQA [10] and animated GIFs for TGIF-QA [11]). Rohrbach *et al.* [1] extend the LSMDC movie description dataset to the VQA domain, introducing several new tasks such as multiple-choice [12] and fill-in-the-blank [34].

The multiple-choice problem is, given a video query and five descriptive sentences, to choose a single best answer in the candidates. To tackle this problem, ranking losses on deep representation [9, 11, 12] or nearest neighbor search on the joint space [30] are exploited. Torabi *et al.* [12] use the temporal attention on the joint representation between the query videos and answer choice sentences. Yu *et al.* [9] use LSTMs to sequentially feed the query and the answer embedding conditioned on detected concept words. The fill-in-the-blank task is, given a video and a sentence with a single blank, to select a suitable word for the blank. To encode the sentential query sentence on the video context, MergingLSTMs [35] and LR/RL LSTMs [36] are proposed. Yu *et al.* [9, 29] attempt to detect semantic concept words from videos and integrate them with Bidirectional LSTM that encodes the language query. However, most previous approaches tend to focus too much on the sentence information and easily ignore visual cues. On the other hand, our model focuses on learning multi-level semantic similarity between videos and sentences, and consequently achieves the best results reported so far in these two QA tasks, as will be presented in section 4.

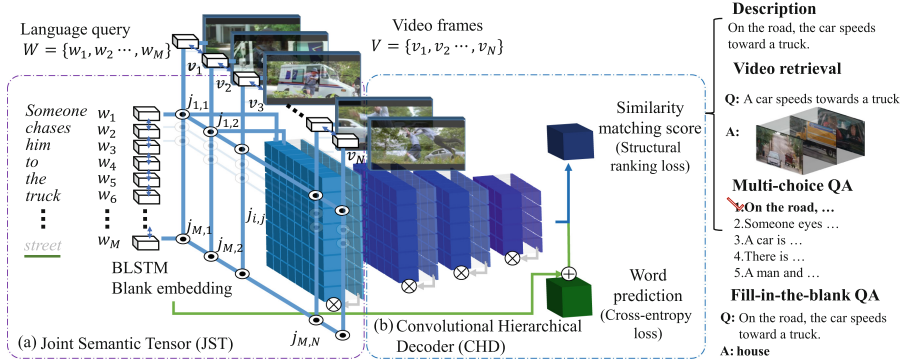
### 3 The Joint Sequence Fusion Model

We first explain the preprocessing steps for describing videos and sentences in Sect. 3.1, and then discuss the two key components of our JSFusion model in Sects. 3.2–3.4, respectively. We present the training procedure of our model in Sect. 3.5, and its applications to three video-language tasks in Sect. 3.6.

#### 3.1 Preprocessing

**Sentence Representation.** We encode each sentence in a word level. We first define a vocabulary dictionary  $\mathcal{V}$  by collecting the words that occur more than three times in the dataset. (*e.g.* the dictionary size is  $|\mathcal{V}| = 16,824$  for LSMDC). We ignore the words that are not in the dictionary. Next we use the pretrained glove.42B.300d [37] to obtain the word embedding matrix  $\mathbf{E} \in \mathcal{R}^{d \times |\mathcal{V}|}$  where  $d = 300$  is the word embedding dimension. We denote the description of each sentence by  $\{\mathbf{w}_m\}_{m=1}^M$  where  $M$  is the number of words in the sentence. We limit the maximum number of words per sentence to be  $M_{max} = 40$ . If a sentence is too long, we discard the remaining excess words, because only 0.07% of training sentences exceed this limit, and no performance gain is observed for larger  $M_{max}$ . Throughout this paper, we use  $m$  for denoting the word index.

**Video Representation.** We sample a video at five fps, to reduce the frame redundancy while minimizing information loss. We employ CNNs to encode both visual and audial information in videos. For visual description, we extract the



**Fig. 2.** The architecture of Joint Sequence Fusion (JSFusion) model. **Blue paths** indicate the information flows for multimodal similarity matching tasks, while **green paths** for the fill-in-the-blank task. (a) JST composes pairwise joint representation of language and video sequences into a 3D tensor, using a soft-attention mechanism. (b) CHD learns hierarchical relation patterns between the sequences, using a series of convolutional decoding module which shares parameters for each stage.  $\odot$  is Hadamard product,  $\oplus$  is addition, and  $\otimes$  is multiplication between representation and attentions described in Eqs. (2)–(4). We omit some fully-connected layers for visualization purpose. (Color figure online)

feature map of each frame from the pool5 layer ( $\mathbb{R}^{2,048}$ ) of ResNet-152 [38] pretrained on ImageNet. For audial information, we extract the feature map using the VGGish [39] followed by PCA for dimensionality reduction ( $\mathbb{R}^{128}$ ). We then concatenate both features as the video descriptor  $\{\mathbf{v}_n\}_{n=1}^N \in \mathbb{R}^{2,156 \times N}$  where  $N$  is the number of frames in the video. We limit the maximum number of frames to be  $N_{max} = 40$ . If a video is too long, we select  $N_{max}$  equidistant frames. We observe no performance gain for larger  $N_{max}$ . We use  $n$  for denoting the video frame index.

### 3.2 The Joint Semantic Tensor

The Joint Semantic Tensor (JST) first composes pairwise representations between two multimodal sequences into a 3D tensor. Next, JST applies a self-gating mechanism to the 3D tensor to refine it as an attention map that discovers fine-grained matching between all pairwise embeddings of the two sequences while pruning out unmatched joint representations

**Sequence Encoders.** Give a pair of multimodal sequences, we first represent them using encoders. We use *bidirectional LSTM networks* (BLSTM) encoder [40, 41] for word sequence and CNN encoder for video frames. It is often advantageous to consider both future and past contexts to represent each element in a sequence, which motivates the use of BLSTM encoders.  $\{\mathbf{h}_t^f\}_{t=1}^T$  and  $\{\mathbf{h}_t^b\}_{t=1}^T$  are the forward and backward hidden states of the BLSTM, respectively:

$$\mathbf{h}_t^f = \text{LSTM}(\mathbf{x}_t, \mathbf{h}_{t-1}^f), \quad \mathbf{h}_t^b = \text{LSTM}(\mathbf{x}_t, \mathbf{h}_{t+1}^b), \quad (1)$$

where we set  $\mathbf{h}_t^b, \mathbf{h}_t^f \in \mathbb{R}^{512}$ , with initializing them as zeros:  $\mathbf{h}_{T+1}^b = \mathbf{h}_0^f = \mathbf{0}$ . Finally, we obtain the representation of each modality at each step by concatenating the forward/backward hidden states and the input features:  $\mathbf{x}_{w,t} = [\mathbf{h}_{w,t}^f, \mathbf{h}_{w,t}^b, \mathbf{w}_t]$  for words. For visual domain, we use 1-d CNN encoder representation for  $v_t$ ,  $h^{cnn} \in \mathbb{R}^{2,048}$  instead,  $\mathbf{x}_{v,t} = [\mathbf{h}_{v,t}^{cnn}, \mathbf{v}_t]$ .

**Table 1.** The detailed setting of layers in the JSFusion model. No padding is used for each layer. *Dk* means a fully-connected dense layer, and *Convk* and *ConGk* indicate convolutional and convolutional-gating layer, respectively.

FC layers	size	Conv layer	kernel/stride	channel
D1 <sup>v</sup> , D1 <sup>w</sup>	512	Conv1	3 × 3 / 1	256
D2	512	ConvG1	3 × 3 / 1	1
D3, D4	512	Conv2	3 × 3 / 1	256
D5	256	ConvG2	3 × 3 / 1	1
D6	256	Conv3	3 × 3 / 2	256
D7	128	ConvG3	3 × 3 / 2	1
D8	1	MeanPool	17 × 17 / 17	256

**Attention-Based Joint Embedding.** We then feed the output of the sequence encoder into fully-connected (dense) layer [D1] for each modality separately, which results in  $D1^v(\mathbf{x}_v), D1^w(\mathbf{x}_w) \in \mathbb{R}^{d_{D1}}$ , where  $d_{D1}$  is a hidden dimension of [D1]. We summarize the details of all the layers in our JSFusion model in Table 1. Throughout the paper, we denote fully-connected layers as *Dk* and convolutional layers as *Convk*.

Next, we compute attention weights  $\alpha$  and representation  $\gamma$ , from which we obtain the JST as a joint embedding between every pair of sequential features:

$$\mathbf{j}_{nm} = \alpha_{nm} \gamma_{nm}, \text{ where } \alpha_{nm} = \sigma(\mathbf{w}^T D2(\mathbf{t}_{nm})), \gamma_{nm} = D4(D3(\mathbf{t}_{nm})), \quad (2)$$

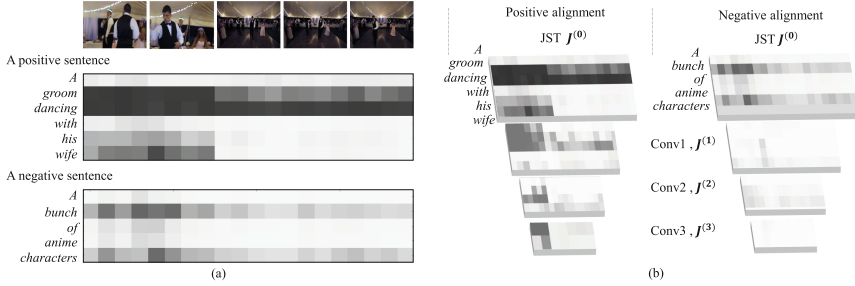
$$\mathbf{t}_{nm} = D1^v(\mathbf{x}_{v,n}) \odot D1^w(\mathbf{x}_{w,m}). \quad (3)$$

$\odot$  is a hadamard product,  $\sigma$  is a sigmoid function, and  $\mathbf{w} \in \mathbb{R}^{d_{D2}}$  is a learnable parameter. Since the output of the sequence encoder represents each frame conditioned on the neighboring video (or each word conditioned on the whole sentence), the attention  $\alpha$  is expected to figure out which pairs should be more weighted for the joint embedding among all possible pairs. For example of Fig. 2, expectedly,  $\alpha_{3,6}(v_3, w_6) > \alpha_{8,6}(v_8, w_6)$ , if  $w_6$  is *truck*, and the third video frame contains the *truck* while the eighth frame does not.

From Eqs. (2)–(3), we obtain JST in a form of 3D tensor:  $\mathbf{J} = [\mathbf{j}_{n,m}]_{m=1:M_{max}}^{n=1:N_{max}}$  and  $\mathbf{J} \in \mathbb{R}^{N_{max} \times M_{max} \times d_{D4}}$ .

### 3.3 The Convolutional Hierarchical Decoder

The Convolutional Hierarchical Decoder (CHD) computes a compatibility score for a pair of multimodal sequences by exploiting the compositionality in the joint



**Fig. 3.** Attention examples for (a) Joint Semantic Tensor (JST) and (b) Convolutional Hierarchical Decoder (CHD). Higher values are shown in darker. (a) JST assigns high weights on positively aligned joint semantics in the two sequence data. Attentions are highlighted darker where words coincide well with frames. (b) Each layer in CHD assigns high weights to where structure patterns are well matched between the two sequence data. For a wrong pair of sequences, a series of Conv-gating ( $ConvG2$ ) prune out misaligned patterns with low weights.

vector space of JST. We pass the JST tensor through a series of a convolutional (Conv) layer and a Conv-gating block, whose learnable kernels progressively find matched embeddings from those of each previous layer. That is, starting from the JST tensor, the CHD recursively activates the weights of positively aligned pairs than negatively aligned ones.

Specifically, we apply three sets of Conv layer and Conv-gating to the JST:

$$\mathbf{J}^{(k)} = Convk(\mathbf{J}^{(k-1)}) \cdot \sigma(ConvGk(\mathbf{J}^{(k-1)})) \quad (4)$$

for  $k = 1, 2, 3$ . We initialize  $\mathbf{J}^{(0)} = \mathbf{J}$  from the JST, and  $[Convk]$  is the  $k$ -th Conv layer for joint representation,  $[ConvGk]$  is the  $k$ -th Conv-gating layer for matching filters, whose details are summarized in Table 1.

We apply mean pooling to  $\mathbf{J}^{(3)}$  to obtain a single video-sentence vector representation  $\mathbf{J}_{out}$  (e.g.  $\mathbb{R}^{17 \times 17 \times 256} \rightarrow \mathbb{R}^{1 \times 1 \times 256}$ ). Finally, we compute similarity matching score by feeding  $\mathbf{J}_{out}$  into four dense layers  $[D5, D6, D7, D8]$ :

$$\begin{aligned} \text{score} &= \mathbf{W}_{D8}(D7(D6(D5(\mathbf{J}_{out})))) + \mathbf{b}_{D8} \\ \text{where } Dk(\mathbf{x}) &= \tanh(\mathbf{W}_{Dk}\mathbf{x} + \mathbf{b}_{Dk}), \quad k = 5, 6, 7. \end{aligned} \quad (5)$$

We use the tanh activation for all dense layers except  $[D8]$ .

### 3.4 An Illustrative Example of How the JSFusion Model Works

Figure 3 illustrates with actual examples how the attentions of JST and CHD work.

Figure 3(a) visualizes the learned attention weights  $\alpha_{nm}$  in Eq. (2) of all pairs between frames in a video and words in a positive and a negative sentence. The attentions are highlighted with higher values (shown in darker) when the words coincide better with the content in the frames, dominantly in a positive pair.

Figure 3(b) shows the output  $\mathbf{J}^{(k)}$  of each Conv layer and Conv-gating block in Eq. (4) for the same example. During training, each Conv layer learns to compose joint embedding from the ones in the lower layer, while the Conv-gating layer learns frequent matching patterns in the training pairs of videos and sentences. At test time, when it comes to compute a similarity score, the Conv-gating layers prune out misaligned patterns; if the pair is negative where there is no common aligned structure in the two sequences, as shown in the right of Fig. 3(b), most elements of  $\mathbf{J}^{(k)}$  have very low values. As a result, the CHD can selectively filter lower-layer information that needs to be propagated to the final-layer representation, and the final layer of CHD assigns a high score only if the jointly aligned patterns are significant between the sequence data.

The motivation behind the JSFusion model is that long sequence data like videos and sentences are too complicated to compare them in a single vector space, although most previous approaches depend on single LSTM embedding such as neural visual semantic embedding [19] and previous LSMDC winners [9, 30]. Instead, in our approach, JST first composes a dense pairwise 3D tensor representation between multimodal sequence data, from which CHD then exploits convolutional gated layers to learn multi-stage similarity matching. Therefore, our JST model can be more robust for detecting partial matching between short phrases and subshots.

### 3.5 Training

We train our JSFusion model using the ranking loss. Each training batch consists of  $L$  video-sentence pairs, including a single positive pair and  $L - 1$  randomly sampled negative pairs. We use batch shuffling in every training epoch. Finally, we train the model using a max-margin structured loss objective as follows:

$$\mathcal{L} = \sum_k \sum_{l=1}^L \max(0, S_{k,l} - S_{k,l^*} + \Delta) + \lambda \|\theta\|^2 \quad (6)$$

where  $l^*$  is the answer pair among  $L$  candidates,  $\lambda$  is a hyperparameter and  $\theta$  denotes weight parameters. This objective encourages a positive video-sentence pair to have a higher score than a misaligned negative pair by a margin  $\Delta$ . We use  $\lambda = 0.0005$ ,  $\Delta = 10$  in our experiments. We train all of our models using the Adam optimizer [42], with an initial learning rate in the range of  $10^{-4}$ . For regularization, we apply batch normalization [43] to every dense layer.

### 3.6 Implementation of Video-Language Models

We below discuss how the JSFusion model is implemented for three video-language tasks, video retrieval, multiple-choice test, and fill-in-the-blank. We apply the same JSFusion model to both video retrieval and multiple-choice test with slightly different hyperparameter settings. For the fill-in-the-blank, we make a minor modification in our model to predict a word for a blank in the middle of the sentence.



**For Retrieval.** The retrieval model takes a query sentence and ranks 1,000 test videos according to the relevance between the query and videos. For training, we set  $L = 10$  as the size of each training batch. At test, for each query sentence  $k$ , we compute scores  $\{S_{k,l}\}_l$  for all videos  $l$  in the test set. From the score matrix, we can rank the videos for the query. As will be presented in Sects. 4.3 and 4.4, our method successfully finds hierarchical matching patterns between complex natural language query and video frames with sounds.

**For Multiple-Choice Test.** The multiple-choice model takes a video and five choice sentences among which only one is the correct answer. Since our model can calculate the compatibility score between the query video and each sentence choice, we use the same model as the retrieval task. We simply select the choice with the highest score as an answer. For training, we set  $L = 10$  so that each training batch contains 10 pairs of videos and sentences, which include only a single correct sentence, four wrong choices, and 5 randomly selected sentences from other training data.

**For Fill-in-the-Blank.** The fill-in-the-blank model takes a video and a sentence with one blank, and predict a correct word for the blank. Since this task requires more difficult inference (*i.e.* selecting a word out of vocabulary  $\mathcal{V}$ , instead of computing a similarity score), we make two modifications as follows. First, we use deeper dimensions for layers:  $d_{D1} = d_{D5} = d_{D6} = d_{D7} = 1,024$ ,  $d_{D2} = d_{D3} = d_{D4} = 2,048$ ,  $d_{D8} = |\mathcal{V}|$ ,  $d_{Conv1.1} = d_{Conv2.1} = d_{Conv3.1} = 1,024$ , instead of the numbers in Table 1.

Second, we add a skip-connection part to our model, which is illustrated as the green paths of Fig. 2. Letting  $b$  as the blank position in the query sentence, we use the BLSTM output from the blank word token BLANK as a sentential context of the blank position:  $\mathbf{t}_b = D1^w(\mathbf{w}_b)$ . We make a summation between the output of [D7]  $\in \mathbb{R}^{1,024}$  and the sentential context  $\mathbf{t}_b \in \mathbb{R}^{1,024}$ , and then feed it into [D8] to predict a word.

For training, we set the batch size as  $L = 32$ . We use the different objective, the cross-entropy loss, because this task is classification rather than ranking:

$$\mathcal{L} = -\log p(\mathbf{y}) + \lambda \|\theta\|^2 \quad (7)$$

where  $\theta$  denotes weight parameters and  $\lambda = 0.0005$ . We use dropout with a rate of 0.2.

## 4 Experiments

We report the experimental results of JSFusion models for the three tasks of LSMDC [1] and two tasks of MSR-VTT [2].

### 4.1 LSMDC Dataset and Tasks

The LSMDC 2017 consists of four video-language tasks for movie understanding and captioning, among which we focus on the three tasks in our experiments:

**Table 2.** Performance comparison for the movie retrieval task using Recall@k (R@k, higher is better) and Median Rank (MedR, lower is better). We report the results on the two datasets of LSMDC [1] (L) and MSR-VTT [2] (M).

Tasks	Movie retrieval							
	R@1		R@5		R@10		MedR	
Dataset	L	M	L	M	L	M	L	M
LSTM-fusion	3.0	3.0	8.9	9.6	15.9	17.1	95	67
SA-G+SA-FC7 [12]	3.0	3.1	8.8	9.0	13.2	13.4	114	91
LSTM+SA-FC7 [12]	3.3	3.2	10.2	11.1	15.6	15.7	88	69
C+LSTM+SA-FC7 [12]	4.3	4.2	12.6	12.9	18.9	19.9	98	55
VSE-LSTM [19]	3.1	3.8	10.4	12.7	16.5	17.1	79	66
EITanque [30]	4.7	4.7	15.9	16.6	23.4	24.1	64	41
SNUVL [29]	3.6	3.5	14.7	15.9	23.9	23.8	50	44
CT-SAN [9]	4.5	4.4	14.1	16.6	20.9	22.3	67	35
Miech <i>et al.</i> [44]	7.3	–	19.2	–	27.1	–	52	–
JSTfc	4.7	5.1	17.2	21.1	25.2	29.1	52	30
JSTlstm	7.6	9.2	19.2	28.2	27.1	41.1	36	18
JSTmax	6.7	8.8	18.0	29.8	27.2	41.0	39	17
JSTmean	7.5	9.0	20.9	27.2	28.2	40.9	36	18
JSFusion-noattention	6.4	8.7	18.4	27.4	28.4	39.5	41	19
JSFusion-noaudio	9.0	9.2	20.9	28.3	32.1	41.3	39	17
JSFusion	<b>9.1</b>	<b>10.2</b>	<b>21.2</b>	<b>31.2</b>	<b>34.1</b>	<b>43.2</b>	<b>36</b>	<b>13</b>

movie retrieval, multiple-choice test, and fill-in-the-blank. The challenge provides a subset of the LSMDC dataset, which contains a parallel corpus of 118,114 sentences and 118,081 video clips of about 4–5 s long sampled from 202 movies. We strictly follow the evaluation protocols of the challenge. We defer more details of the dataset and challenge rules to [1] and the homepage<sup>1</sup>.

**Multiple-Choice Test.** Given a video query and five candidate captions, the goal is to find the correct one for the query out of five possible choices. The correct answer is the groundtruth (GT) caption and four other distractors are randomly chosen from other captions that have different activity-phrases from the correct answer. The evaluation metric is the percentage of correctly answered test questions out of 10,053 public-test data.

**Movie Retrieval.** The test set consists of 1,000 video/activity phrase pairs sampled from the LSMDC17 public-test data. Then, the objective is, given a short query activity-phrase (*e.g. answering phone*), to find its corresponding video out of 1,000 test videos. The evaluation metrics include Recall@1, Recall@5, Recall@10, and Median Rank (MedR). The Recall@*k* means the percentage of

<sup>1</sup> <https://sites.google.com/site/describingmovies/lsmdc-2017>.

**Table 3. Left:** Performance comparison for the multiple-choice test using the accuracy in percentage. We report the results on the two datasets of LSMDC (L) and MSR-VTT (M). **Right:** Accuracy comparison (in percentage) for the movie fill-in-the-blank task.

Multiple-Choice Dataset	Accuracy		Fill-in-the-Blank	Accuracy
	L	M		
LSTM-fusion	52.8	38.3	Text-only BLSTM [34]	32.0
SA-G+SA-FC7 [12]	55.1	55.8	Text-only Human [34]	30.2
LSTM+SA-FC7 [12]	56.3	59.1	GoogleNet-2D + C3D [34]	35.7
C+LSTM+SA-FC7 [12]	58.1	60.2	Ask Your Neurons [46]	33.2
VSE-LSTM [19]	63.0	67.3	Merging-LSTM [35]	34.2
SNUVL [29]	63.1	65.4	SNUVL [29]	38.0
ST-VQA-Sp.Tp [11]	63.5	66.1	CT-SAN [9]	41.9
EITanque [30]	63.7	65.5	LR/RL LSTMs [36]	40.9
CT-SAN [9]	63.8	66.4	LR/RL LSTMs (Ensemble) [36]	43.5
MLB [45]	69.0	76.1	MLB [45]	41.6
JSTfc	64.7	68.7	JSTfc	42.9
JSTlstm	72.1	79.7	JSTlstm	43.7
JSTmax	68.3	74.4	JSTmax	41.3
JSTmean	70.2	80.0	JSTmean	44.2
JSFusion-noattention	69.4	79.2	JSFusion-noattention	44.5
JSFusion-VGG-noaudio	68.7	75.6	JSFusion-VGG-noaudio	44.2
JSFusion-noaudio	72.5	82.9	JSFusion-noaudio	45.26
JSFusion	<b>73.5</b>	<b>83.4</b>	JSFusion	<b>45.52</b>
			Human [34]	68.7

GT videos in the first  $k$  retrieved videos, and the MedR indicates the median rank of GT videos. The challenge winner is determined by the metric of Recall@10.

**Movie Fill-in-the-Blank.** This track is related to visual question answering. The task is, given a video clip and a sentence with a blank in it, to predict a single correct word for the blank. The test set includes 30,000 examples from 10,000 clips (*i.e.* about 3 blanks per sentence). The evaluation metric is the prediction accuracy (*i.e.* the percentage of predicted words that match with GTs).

## 4.2 MSR-VTT-(RET/MC) Dataset and Tasks

The MSR-VTT [2] is a large-scale video description dataset. It collects 118 videos per query of 257 popular queries, and filters manually to 7,180 videos. From the videos, it selects 10 K video clips with 41.2 h and 200 K clip-sentence pairs.

Based on the MSR-VTT dataset, we newly create two video-text matching tasks: (i) multiple-choice test and (ii) video retrieval. The task objectives for these tasks are identical to those of corresponding tasks in the LSMDC benchmark. To collect annotations for the two tasks, we exactly follow the protocols that are used in the LSMDC dataset, as described in [12].

**Multiple-Choice Test:** We generate 2,990 questions in total for the multiple-choice test, using all the test video clips of MSR-VTT. For each test video, we

use the associated GT caption for the correct answer, while randomly sampled descriptions from other test data for four negative choices.

**Video Retrieval:** For retrieval, we first sample 1,000 pairs of video clips and description queries from the test set of MSR-VTT. We use 1,000 as the size of the test set, following the LSMDC benchmark. As a result, the retrieval task is to find out the video that corresponds to the query caption out of 1000 candidates.

### 4.3 Quantitative Results

Tables 2 and 3 summarize the results of our experiments for the three video-language tasks. For LSMDC experiments, we report the results in the published papers and the official leaderboard of LSMDC 2017<sup>2</sup>. For MSR-VTT experiments, we run some participants of LSMDC, including SNUVL, EITanque, VSE-LSTM, ST-VQA-Sp.Tp and CT-SAN, using the source codes provided by the original authors. We implement the other baselines by ourselves, only except Miech *et al.* that require an additional person tracker, which is unavailable to use. Other variants of our method will be discussed in details below in the ablation study.

Tables 2 and 3 clearly show that our JSFusion achieves the best performance with significant margins from all the baselines over the three tasks on both datasets. That is, the two components of our approach, JST and CHD, indeed helps measure better the semantic similarity between multimodal sequences than a wide range of state-of-the-art models, such as a multimodal embedding method (VSE-LSTM), a spatio-temporal attention-based QA model (ST-VQA-Sp.Tp), and a language model based QA inference (Text-only BLSTM). Encouragingly, the JSFusion single model outperforms even the ensemble method of runner-up (LR/RL LSTMs) in the fill-in-the-blank task.

Among baselines, multimodal low-rank bilinear attention network (MLB) [45] is competitive. The main differences of our model from (MLB) are two-fold. First, JSFusion embeds both a video and a sentence to feature sequences, whereas (MLB) represents the sentence as a single feature. Second, JSFusion uses the self-gating to generate fine-grained matching between all pairwise embeddings of the two sequences, while (MLB) uses the attention to find a position in the visual feature space that best fits for the sentence vector. Moreover, JSFusion consistently shows better performance than (MLB) in all experiments.

**Ablation Study.** We conduct ablation experiments on different variants of our JSFusion model and present the results in Tables 2 and 3. As one naive variant of our model, we test a simple LSTM baseline (LSTM-fusion) that only carries out the Hadamard product on a pair of final states of video and language LSTM encoders. That is, (LSTM-fusion) is our JSFusion model that has neither JST nor CHD, which are the two main contributions of our model. We train (LSTM-fusion) in the same way as done for the JSFusion model in Sect. 3.5.

<sup>2</sup> FIB: <https://competitions.codalab.org/competitions/11691#results>.

Multichoice: <https://competitions.codalab.org/competitions/11491#results>.

As easily expected, the performance of (LSTM-fusion) is significantly worse than our JSFusion in all the tasks.

To further validate the contribution of each component, we remove or replace key components of our model with simpler ones. To understand the effectiveness of BLSTM encoding, we test two baselines: (JSTfc) that replaces BLSTM with fully-connected layers and (JSTlstm) that replaces BLSTM with LSTM. (JSTmax) and (JSTmean) denote our variants that use max pooling and mean pooling, instead of the *Convk* convolutional layers in CHD. That is, they use fixed max/mean pooling operations instead of convolutions with learnable kernels. These comparisons reveal that the proposed CHD is critical to improve the performance of JSFusion nontrivially on all the tasks on both datasets. We also compare our model with (JSFusion-noattention) that discards Conv-gating operations of CHD. (JSFusion-noattention) shows nontrivial performance drops as MC (acc): 4.1%p, 4.2%p, RET (R@10): 5.7%p, 3.7%p for LSMDC and MSR-VTT, respectively. Finally, we test our model with using no audio information denoted by (JSFusion-noaudio), which is also much better than other baselines but only slightly worse than our original model.

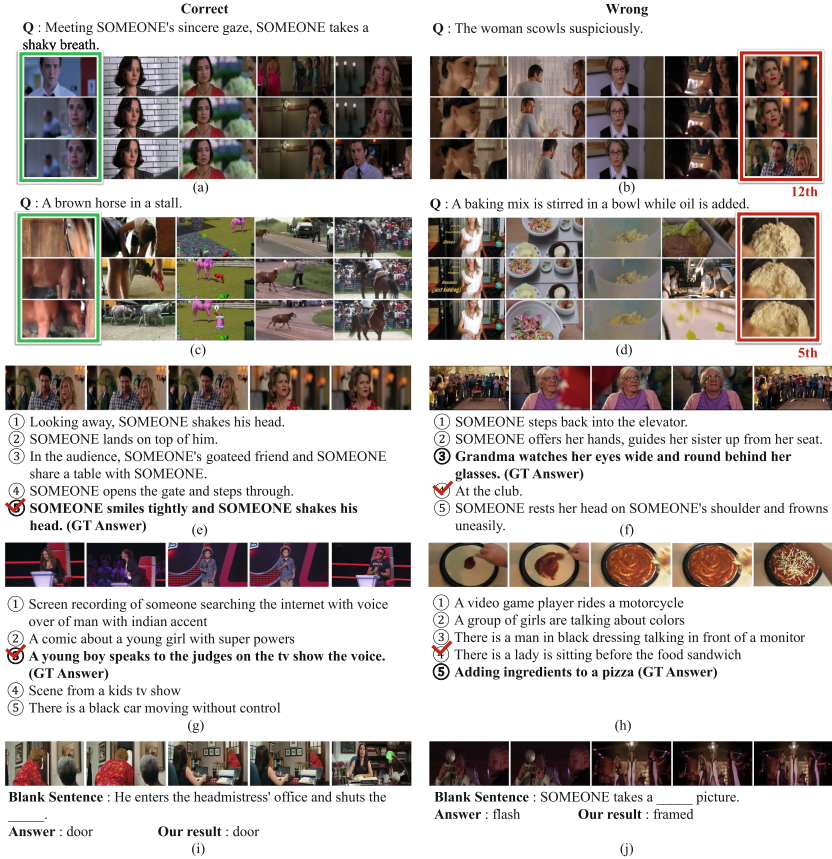
#### 4.4 Qualitative Results

Figure 4 illustrates qualitative results of our JSFusion algorithm with correct (left) and near-miss (right) examples for each task. In each set, we show natural language query and sampled frames of a video. We present both groundtruth (GT), our prediction (Ours).

**Movie Retrieval.** Figure 4(a) is an example that our model can understand human behaviors like *gaze*. Figure 4(b) shows the model’s failure to distinguish a small motion (*e.g.* facial expression), and simply retrieve the videos containing the face of a *woman*. Figure 4(c) shows that our model successfully catches the features of *horses* in both web videos and 3D animation, and correctly select the highest ranking video by focusing on the word *stall*. In Fig. 4(d), although the model can retrieve relevant videos of *cooking with bowl*, it fails to find out the answer video that contains the query description of *baking mix*.

**Movie Multiple-Choice Test.** Figure 4(e) delivers an evidence that our model uses the whole sentence for computing matching scores, because the model successfully chooses ⑤ instead of ① that shares the same phrases (*e.g.* *shakes his head*). Figure 4(f) is an example of focusing on a wrong video subsequence, where our model chooses the word *club* by looking at a subsequence with crowded people, but the answer is related to another subsequence with *grandmother*. Figure 4(g) is an example that the model learns words in a phrase. Choice ④ can be very tempting, since it contains the word *kids, tv* and *show*. But our model successfully choose the right answer by identifying that *kids tv show* and *kids in tv show* mean differently. Figure 4(h) shows that our model fails to distinguish the details.

**Movie Fill-in-the-Blank.** In Fig. 4(i), the model successfully finds the answer by using both structural information of a sentence and a video (*e.g.* *door* is a



**Fig. 4.** Qualitative examples of the three video-language tasks: movie retrieval on LSMDC (a)–(b) and MSR-VTT-RET (c)–(d), multiple-choice on LSMDC (e)–(f) and MSR-VTT-MC (g)–(h), and (i)–(j) fill-in-the-blank on LSMDC. The left column shows correct examples, while the right column shows near-miss examples. In (b),(d), we show our retrieval ranks of the GT clips (in the red box). (Color figure online)

likely word after *shuts the*). Figure 4(j) is an example that the model focuses too much on the word *picture* that follows the blank, instead of visual information, and thus choose a wrong answer *framed picture* rather than *flash picture*.

## 5 Conclusion

We proposed the Joint Sequence Fusion (JSFusion) model for measuring hierarchical semantic similarity between two multimodal sequence data. The two key components of the model, Joint Semantic Tensor (JST) and Convolutional Hierarchical Decoder (CHD), are easily adaptable in many video-and-language tasks, including multimodal matching or video question answering. We demonstrated

that our method significantly improved the performance of video understanding through natural language description. Our method achieved the best performance in challenge tracks of LSMDC, and outperformed many state-of-the-art models for VQA and retrieval tasks on the MSR-VTT dataset.

Moving forward, we plan to expand the applicability of JSFusion; since our model is usable to any multimodal sequence data, we can explore other retrieval tasks of different modalities, such as videos-to-voices or text-to-human motions.

**Acknowledgements.** We thank Jisung Kim and Antoine Miech for helpful comments about the model. This research was supported by Brain Research Program by National Research Foundation of Korea (NRF) (2017M3C7A1047860).

## References

1. Rohrbach, A., et al.: Movie Description. [arXiv:1605.03705](https://arxiv.org/abs/1605.03705) (2016)
2. Xu, J., Mei, T., Yao, T., Rui, Y.: MSR-VTT: a large video description dataset for bridging video and language. In: CVPR (2016)
3. Chen, D.L., Dolan, W.B.: Collecting highly parallel data for paraphrase evaluation. In: ACL (2011)
4. Donahue, J., et al.: Long-term Recurrent Convolutional Networks for Visual Recognition and Description. In: CVPR (2015)
5. Guadarrama, S., et al.: YouTube2Text: recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In: ICCV (2013)
6. Rohrbach, A., Rohrbach, M., Schiele, B.: The long-short story of movie description. In: GCPR (2015)
7. Venugopalan, S., Marcus, R., Jeffrey, D., Raymond, M., Trevor, D., Kate, S.: Sequence to sequence - video to text. In: ICCV (2015)
8. Xu, R., Xiong, C., Chen, W., Corso, J.J.: Jointly modeling deep video and compositional text to bridge vision and language in a unified framework. In: AAAI (2015)
9. Yu, Y., Ko, H., Choi, J., Kim, G.: End-to-end concept word detection for video captioning, retrieval, and question answering. In: CVPR (2017)
10. Tapaswi, M., Zhu, Y., Stiefelhagen, R., Torralba, A., Urtasun, R., Fidler, S.: MovieQA: understanding stories in movies through question-answering. In: CVPR (2016)
11. Jang, Y., Song, Y., Yu, Y., Kim, Y., Kim, G.: TGIF-QA: toward spatio-temporal reasoning in visual question answering. In: CVPR (2017)
12. Torabi, A., Tandon, N., Sigal, L.: Learning language-visual embedding for movie understanding with natural-language. [arXiv:1609.08124](https://arxiv.org/abs/1609.08124) (2016)
13. Otani, M., Nakashima, Y., Rahtu, E., Heikkilä, J., Yokoya, N.: Learning joint representations of videos and sentences with web image search. [arXiv:1608.02367](https://arxiv.org/abs/1608.02367) (2016)
14. Laptev, I., Lindeberg, T.: Space-time Interest Points. In: ICCV (2003)
15. Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies. In: CVPR (2008)
16. Soomro, K., Zamir, A.R., Shah, M.: UCF101: A Dataset of 101 Human Actions Classes from Videos in the Wild (2012)
17. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: CVPR (2014)

18. Caba Heilbron, F., Escorcia, V., Ghanem, B., Carlos Niebles, J.: ActivityNet: a large-scale video benchmark for human activity understanding. In: CVPR (2015)
19. Kiros, R., Salakhutdinov, R., Zemel, R.S.: Unifying visual-semantic embeddings with multimodal neural language models. In: TACL (2014)
20. Frome, A., et al.: DeViSE: a deep visual-semantic embedding model. In: NIPS (2013)
21. Socher, R., Karpathy, A., Le, Q.V., Manning, C.D., Ng, A.Y.: Grounded compositional semantics for finding and describing images with sentences. TACL (2014)
22. Socher, R., Lin, C.C., Manning, C., Ng, A.Y.: Parsing natural scenes and natural language with recursive neural networks. In: ICML (2011)
23. Socher, R., Bauer, J., Manning, C.D., et al.: Parsing with compositional vector grammars. In: ACL (2013)
24. Hodosh, M., Young, P., Hockenmaier, J.: Framing image description as a ranking task: data, models and evaluation metrics. JAIR **47**, 853–899 (2013)
25. Lin, D., Fidler, S., Kong, C., Urtasun, R.: Visual semantic search: retrieving videos via complex textual queries. In: CVPR (2014)
26. Vendrov, I., Kiros, R., Fidler, S., Urtasun, R.: Order-embeddings of images and language. [arXiv:1511.06361](https://arxiv.org/abs/1511.06361) (2015)
27. Hu, R., Xu, H., Rohrbach, M., Feng, J., Saenko, K., Darrell, T.: Natural language object retrieval. In: CVPR (2016)
28. Mao, J., Huang, J., Toshev, A., Camburu, O., Yuille, A.L., Murphy, K.: Generation and comprehension of unambiguous object descriptions. In: CVPR (2016)
29. Yu, Y., Ko, H., Choi, J., Kim, G.: Video captioning and retrieval models with semantic attention. arXiv preprint [arXiv:1610.02947](https://arxiv.org/abs/1610.02947) (2016)
30. Kaufman, D., Levi, G., Hassner, T., Wolf, L.: Temporal tessellation for video annotation and summarization. In: ICCV (2017)
31. Malinowski, M., Fritz, M.: A multi-world approach to question answering about real-world scenes based on uncertain input. In: NIPS (2014)
32. Antol, S., et al.: VQA: visual question answering. In: ICCV (2015)
33. Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., Parikh, D.: Making the V in VQA matter: elevating the role of image understanding in visual question answering. In: CVPR (2017)
34. Maharaj, T., Ballas, N., Courville, A.C., Pal, C.J.: A dataset and exploration of models for understanding video data through fill-in-the-blank question-answering. [arXiv:1611.07810](https://arxiv.org/abs/1611.07810)
35. Mazaheri, A., Zhang, D., Shah, M.: Video fill in the blank with merging LSTMs. [arXiv:1610.04062](https://arxiv.org/abs/1610.04062) (2016)
36. Mazaheri, A., Zhang, D., Shah, M.: Video fill in the blank using LR/RL LSTMs with spatial-temporal attentions. In: ICCV (2017)
37. Pennington, J., Socher, R., Manning, C.D.: GloVe: global vectors for word representation. In: EMNLP (2014)
38. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
39. Hershey, S., et al.: CNN architectures for large-scale audio classification. In: ICASSP (2017)
40. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. In: IEEE TSP (1997)
41. Hochreiter, S., Schmidhuber, J.: Long short-term memory. In: IEEE (1997)
42. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (2015)
43. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: ICML (2015)



44. Miech, A., Alayrac, J.B., Bojanowski, P., Laptev, I., Sivic, J.: Learning from video and text via large-scale discriminative clustering. In: ICCV (2017)
45. Kim, J.H., On, K.W., Lim, W., Kim, J., Ha, J.W., Zhang, B.T.: Hadamard product for low-rank bilinear pooling. In: ICLR (2017)
46. Tzeng, E., Hoffman, J., Darrell, T., Saenko, K.: Simultaneous deep transfer across domains and tasks. In: ICCV (2015)