

# Segment-then-Rank: Non-factoid Question Answering on Instructional Videos

Kyungjae Lee<sup>1\*</sup>, Nan Duan<sup>2</sup>, Lei Ji<sup>2,3</sup>, Jason Li<sup>4</sup>, Seung-won Hwang<sup>1†</sup>

<sup>1</sup>Department of Computer Science, Yonsei University, Seoul, South Korea

<sup>2</sup>Microsoft Research Asia, Beijing, China

<sup>3</sup>University of Chinese Academy of Science, Beijing, China

<sup>4</sup>STCA Multimedia Group, Microsoft, Beijing, China

{lkj0509, seungwonh}@yonsei.ac.kr, {nanduan, lei ji, jasonli}@microsoft.com

## Abstract

We study the problem of non-factoid QA on instructional videos. Existing work focuses either on visual or textual modality of video content, to find matching answers to the question. However, neither is flexible enough for our problem setting of non-factoid answers with varying lengths. Motivated by this, we propose a two-stage model: (a) multimodal segmentation of video into span candidates and (b) length-adaptive ranking of the candidates to the question. First, for segmentation, we propose *Segmenter* for generating span candidates of diverse length, considering both textual and visual modality. Second, for ranking, we propose *Ranker* to score the candidates, dynamically combining the two models with complementary strength for both short and long spans respectively. Experimental result demonstrates that our model achieves state-of-the-art performance.

## 1 Introduction

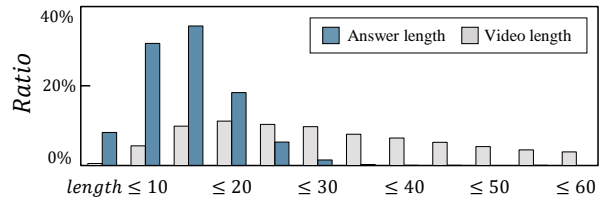
Question Answering (QA) over videos is one of important problems in both NLP and computer vision fields. Recently, with the creation of various datasets, such as TVQA (Lei et al. 2018), TGIF-QA (Jang et al. 2017), and MovieQA (Tapaswi et al. 2016), techniques for video-based QA have been advanced rapidly. However, these datasets commonly assume that the answer is a short text based on concise facts (*e.g.*, question “What is the color of the bird?” can be answered by “White”), while users may want to find longer non-factoid answers to questions such as “how” and “why” types. However, non-factoid QA on video contents, has been relatively under-studied.

For non-factoid questions, existing works focus on document (Yin 2006) or video retrieval (Li et al. 2009; Nie et al. 2011). Given a question, they aim to rank a relevant list of documents or videos as results. However, returning the entire document is too coarse-grained and requires users to read the entire document. Instead, WikiPassageQA task (Cohen, Yang, and Croft 2018) uses pre-segmented paragraphs

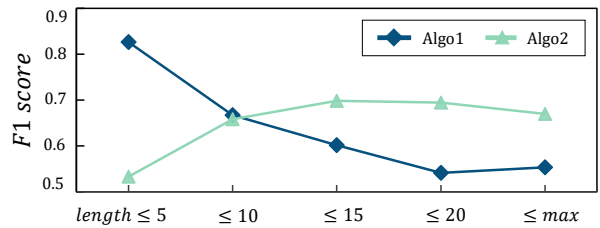
\*This work was partially done during the first author’s internship in MSR Asia and supported by MSR Asia grant

†Corresponding Author

Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.



(a) Distribution of answer/video lengths in our dataset



(b) Performance of two SOTA models over answer lengths (details in Section 4)

Figure 1: Two challenges of our research.

in the text, to return a finer-granular answer as the highest-scoring paragraph. However, this approach cannot apply to videos without clear semantic segmentations such as paragraphs in documents.

Inspired, we aim to extract a fine-grained answer to non-factoid questions, especially, “how-to” questions over instructional videos. Figure 2 illustrates an example, when user poses a question “How to replace a battery from galaxy?”; a desirable answer is annotated as a fine-granular span from 00:41 to 02:50 seconds.

For videos without pre-segmentation, a naive solution to generate span candidates is “split-then-rank” (Cheng et al. 2016; Gao et al. 2017; Liu et al. 2018), by splitting the video into **fixed-size** clips, then ranking each clip by their relevance to the question. However, there are two challenges left unaddressed in this solution. First, it cannot support diverse answer lengths, commonly observed for non-factoid QA, as well as in our dataset. In Figure 1(a), we can observe that the answer length varies in the range of 1 to 30 sentences. Second, when ranking the candidates after segmentation, an

**Question:** How to replace a battery from Galaxy








		Answer: Start 00:41, End 02:50					
Clips							
Transcript	I'm going to show you how to replace the battery in Galaxy.	Take caution and be patient as it's easy to break components.	We're going to try to remove the back panel.	You should be able to lift the back panel from the phone.	To remove the battery, we need to pry a bit of adhesive to get it out.	Now, you should be able to lift the display from the phone.	Don't forget to subscribe to our YouTube channel
Objects	woman, watermark, wall, television	woman, wall	phone, finger, thumb	phone, finger, hand, case	phone, finger, panel, case, buttons	phone, finger	women, wall
	00:04-00:09	00:31-00:35	00:41-00:45	01:49-01:56	02:32-02:50	05:01-05:08	05:14-05:20

Figure 2: An example of our task. While the video describes *how to replace a battery and screen from smart phone*, the answer corresponding to the given question is located between 00:41 and 02:50 seconds.

accurate scoring model for a short answer is not accurate for long (and vice versa). Figure 1(b) contrasts relevance scoring accuracy of two SOTA algorithms (details in Section 4).

To address these challenges, we propose a two-stage approach, **Segmenter-Ranker**, by using a coarse-to-fine schema. The first step is a multimodal **Segmenter** to output span candidates with varying lengths. The goal of our Segmenter is to predict semantic boundaries dependent on the given question/video, considering both transcripts and visual contents in the video. To motivate why we need to consider multimodality, we revisit Figure 2: In the example, considering visual objects *phone* and *finger* extracted from a detector is critical, to exclude the greeting part in the first two clips not showing relevant visual contents. Using both transcripts and visual features, Segmenter suggests several begin- and end-clips as semantic boundaries, then top-k spans are identified to pass on to the next phase.

The second step is to design length-adaptive **Ranker** to ensure precision by re-ranking the answer candidates of varying sizes identified from Segmenter. As above mentioned, there is no single winner in our setting with varying lengths. More formally, according to Guo et al. (2019), existing work is categorized into representation- and interaction-based approaches with such complementary strength, shown as Algo1 and 2 respectively in Figure 1(b). Our contribution, to cover diverse length scenarios, is to adaptively combine representation- and interaction-based approaches with length-adaptive gating.

To benchmark whether our model can effectively answer “how-to” questions, we collect a novel QA dataset over videos and transcripts. Our results show that the proposed model achieves F1 score of 0.703, which leads to F1 score gains of 14.2% relative to the best baseline model. Our experiments demonstrate the impact of the proposed model. Furthermore, comparisons with variants of the proposed model demonstrate the impact of our *Segmenter* and *Ranker*.

In summary, we make the following contributions:

- While most existing QA models assume that the answer is inferred on short spans, we introduce a new video-based task to find diverse-sized spans corresponding to

non-factoid questions.

- We propose a *Segmenter-Ranker* (SR) model for the problem of non-factoid QA. To the best of our knowledge, our approach is the first attempt for non-factoid QA task over video contents.
- For such purpose, we contribute a labeled dataset of 37K QA pairs on instructional videos for benchmarking. Our experiments on the benchmark show that the proposed model achieves state-of-the-art results.

## 2 Task Description

Inspired by extractive QA tasks, predicting a consecutive span of words, we abstract our task as predicting a consecutive span of sentences, to support multi-sentence answers. We redefine **clip** as a sentence unit of video, and our problem as predicting a consecutive span clips.

In our task, given a question, the goal is to find a consecutive answer span, on a pair of video and transcript. Let a given question be  $Q = \{w_1^Q, w_2^Q, \dots, w_n^Q\}$ . Let a given video be  $T$ , and transcript with  $m$  sentences be  $S = \{S_1, S_2, \dots, S_m\}$ . We divide the video  $T$  into  $m$  clips, where each clip corresponds to a single sentence, i.e.,  $T = \{T_1, T_2, \dots, T_m\}$ . The  $k$ -th clip  $T_k$  contains the sentence  $S_k = \{w_1^{S_k}, \dots, w_n^{S_k}\}$ , and sampled image frames  $F_k = \{v_1^{F_k}, \dots, v_l^{F_k}\}$ . Then, given a pair of video  $T$  and question  $Q$ , the objective of our task is to estimate a consecutive answer span  $A$ , i.e.,  $A = \{T_i, T_{i+1}, \dots, T_j\}$  where  $A \subseteq T$ . For task evaluation, the predicted answer span  $A$  is compared with the ground-truth answer span  $A^*$ , in terms of F1 score, precision, and recall in experiment section.

For training and evaluating this task, we collect labelled resources of 37K QA pairs and 21K video (total 1,662 hours). From a commercial search engine, we sampled user queries as questions, and retrieved top1 Youtube video that have English subtitles. Through user interface, we provided a pair of the query and video, and crowd workers were asked to select the answer clips (i.e., sub-sequence of the video) corresponding to the given question. The annotators can return “unanswerable” if there is no answer on the video, and

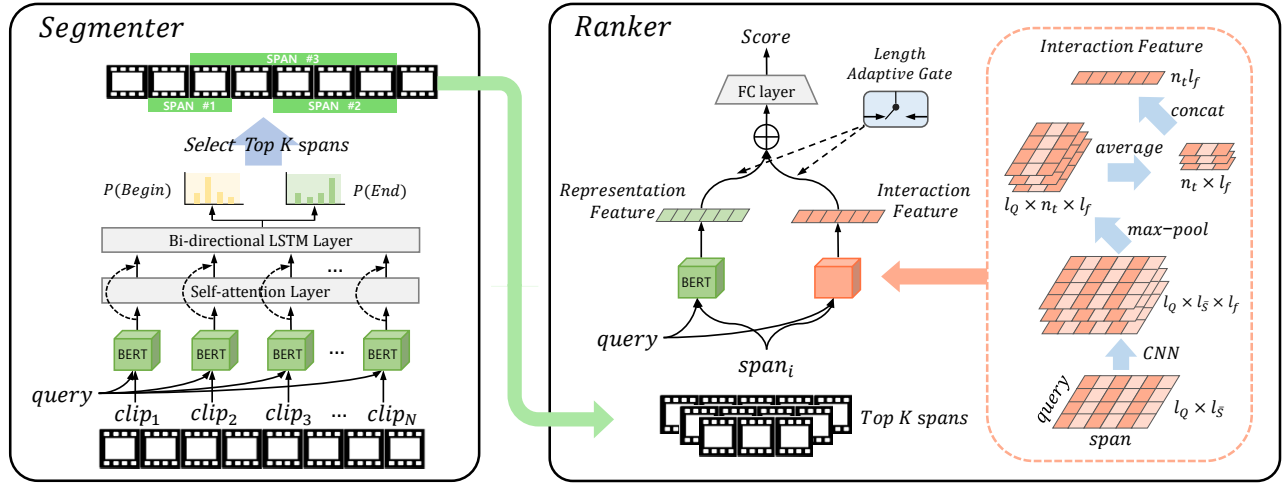


Figure 3: Architecture of our two-stage model

we filter the unanswerable cases. Each query has an answer span label on the video, where the average length of the answers and videos are 15.3 sentences (86 sec) and 60.2 sentences (285 sec), respectively. On a single video, multiple queries can be occurred (average 1.76 in our dataset). We divide the dataset into 29K/4k/4k as training/dev/test set respectively, where the videos do not overlap in each set. We summarize the statistics of the dataset in Table 1 and distribution of video lengths in Figure 1.

We stress again that this dataset presents a new challenge of non-factoid answers of varying length, while in existing span-based QA tasks such as SQuAD, most answers (99.83%) is inferred in a single factoid sentence (Du, Shao, and Cardie 2017), and in action localization task (Regneri et al. 2013), text description consists of a short sentence (9 words on average).

Table 1: Statistics of the datasets.  $|V|$  and  $|A|$  indicate the number of clips in the video and answer, respectively

	# of QA-pairs	# of Videos	Average $ V $	Average $ A $
Training	29K	17K	60.1	15.2
Dev	4K	2K	60.4	15.4
Test	4K	2K	61.2	15.7

### 3 Model

Our goal is to find answer parts of varying lengths over video, using both video and textual modality. We propose a *Segmenter-Ranker* (SR) model as a two-stage approach in Figure 3. To keep multimodal representation light-weight, we propose the use of visual concepts obtained from object detection as an auxiliary features. Since fast R-CNN based object detector (Anderson et al. 2018) provides meaningful features, such information has been found to be useful for various multimodal tasks (Yin and Ordonez 2017; Lei et al. 2018; Kim et al. 2019). While another feature from

ImageNet-based model (e.g., ResNet) is also available, we do not use this for the following reason: since image features from ResNet-50 or 101 have a large dimension (2048d), which is computationally expensive, its adoption was possible on factoid QA with short clips (TGIF-QA (Jang et al. 2017): 3.1s, MovieFIB (Maharaj et al. 2017): 4.1s, on average). However, it may be overkill, for realistic scenario with average length 285s in our dataset. In contrast, visual concepts from object detector do not incur memory overhead, which can be represented as 1600 categories. Using open-source detector<sup>1</sup>, we extract object categories  $c$  from images in clip  $T_k$ , and append all categories in the frames to a set. We redefine frames  $F_k$  as follow:  $F_k = \{c_1^{F_k}, c_2^{F_k}, \dots, c_p^{F_k}\}$ , where  $p$  is the number of objects in  $F_k$ .

#### 3.1 Segmenter Network

The goal of *Segmenter* is to extract several likely candidates as an answer span. For such goal, we train this module to identify whether the clip is begin (or end) position of the answer, or not. As a naive approach, one may consider ranking model to score clips in a fixed-size window individually, which ignores temporal property and global context of videos. To consider such context, we design a hierarchical structure to first encode each clip with a given query, then insert the encoded vectors of all clips into sequential model.

For encoding each clip with a query, we build a multi-modal encoder to deal with textual and visual contents. Formally, a clip consists of a sentence and image objects, which are denoted as two tuples: (query  $Q$ , sentence  $S_k$ ) and (query  $Q$ , object set  $F_k$ ) per each clip  $T_k$ . To encode these tuples jointly, we extend the pre-trained BERT model (Devlin et al. 2018), shown to be effective in a wide variety of language tasks, especially both representing words and matching a pair of two texts.

To encode  $(Q, S)$ , we append words in  $Q$  and  $S$ , with starter symbol  $[CLS]$  and separator symbol  $[SEP]$ , and feed

<sup>1</sup><https://github.com/peteanderson80/bottom-up-attention>

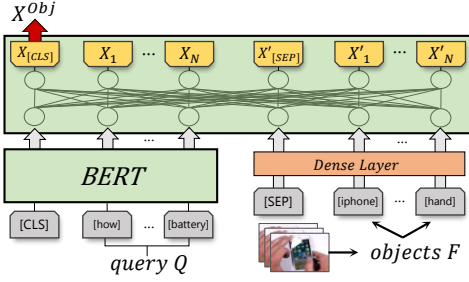


Figure 4: BERT adoption for multimodal encoder

them into BERT model. BERT generates  $L$  layers of hidden states for all BPE tokens in a input sequence. For extracting a vector to represent  $(Q, S_k)$ , we use the output in the position of  $[CLS]$  token, which is denoted as:

$$X_k^{Sent} = BERT_{CLS}(Q, S_k), \quad k \in [1, m] \quad (1)$$

To encode  $(Q, F)$  with multimodality, we extend BERT, as shown in Figure 4. First, for natural language query, we use BERT to encode word representations in the query. Second, for a set of object categories, we encode each category of objects as word embedding, which is passed through a dense layer. We use positional embedding for representing query words, but not for a set of object. Third, for matching between query and objects, we stack two self-attention layers (Vaswani et al. 2017), which aggregates all words in the query and objects. During training, BERT in our *Segmenter* is fine-tuned, jointly learning how to match the query with visual objects through self-attention layers.

Figure 4 summarizes such distinction:  $Q$  and  $F$  are embedded, with  $[CLS]$  and  $[SEP]$ , and inserted into BERT model. For extracting features of  $(Q, F_k)$ , we use the output of  $[CLS]$  token. After obtaining the feature of  $(Q, F_k)$ , we combine two features from the tuples in  $T_k$ , as follows:

$$\begin{aligned} X_k^{Obj} &= h_{CLS}(BERT(Q), NN(F_k)), \quad k \in [1, m] \\ \hat{X}_k &= NN(concat(X_k^{Obj}, X_k^{Sent})), \quad k \in [1, m] \end{aligned} \quad (2)$$

where  $h_{CLS}$  indicates the output of self-attention in positions of  $[CLS]$ , and  $NN$  is a single-layer neural network using tanh function.

After encoding each clip  $(Q, T_k)$  to  $\hat{X}_k$ , we consider the features  $\hat{X}$  as a sequence, by stacking a hierarchical structure. First, we leverage two sequential models: (1) self-attention for global context, and (2) bi-directional LSTM for temporal context. Self-attention of transformer (Vaswani et al. 2017) can capture global dependencies, by attending to all positions. As shown in Figure 3, the features of all clips are aggregated by self-attention, and then are passed through bi-LSTM layers, as the following:

$$\begin{aligned} Y &= \text{Multihead-Attention}(\hat{X}) \\ \hat{Y}_k &= \text{concat}(\hat{X}_k, Y_k), \quad k \in [1, m] \\ M_k^{beg} &= \overrightarrow{LSTM}_1(\hat{Y}_k), \quad k \in [1, m] \\ M_k^{end} &= \overleftarrow{LSTM}_2([\hat{Y}_k; M_k^{beg}]), \quad k \in [1, m] \end{aligned} \quad (3)$$

Second, for prediction, we represent the answer as a sequence of indexes indicating the positions of the begin and end clip in the answer, and train two LSTM networks to generate the probabilities of begin and end at each position. The mathematical formulation is shown below:

$$\begin{aligned} a_k^{beg} &= w_b \cdot \sigma(W_1 M_k^{beg} + b_1), \quad k \in [1, m] \\ a_k^{end} &= w_e \cdot \sigma(W_2 M_k^{end} + b_2), \quad k \in [1, m] \\ p^{beg} &= \text{softmax}(a^{beg}) \\ p^{end} &= \text{softmax}(a^{end}) \end{aligned} \quad (4)$$

where  $W_1, W_2 \in R^{d \times d}$ , and  $w_b, w_e, b_1, b_2 \in R^d$  are trainable parameters, and  $\sigma$  is tanh function. The  $p^{beg}$  and  $p^{end}$  are the normalized probabilities to present begin and end clips. During training, we use cross-entropy loss with labels where the ground-truth in begin and end position is equal to 1, otherwise 0. At inference time, we extract top  $N$  spans, where the top  $N$  values of  $p_i^{beg} \times p_j^{end}$  ( $i \leq j$ ) are selected.

Meanwhile, top 1 span among the candidates could be viewed as a final answer, which means that *Segmenter* has a capability to predict answer spans. To ensure precision, we design a multigranular *Ranker* to re-rank the candidates, adapting to the answer of varying length.

### 3.2 Ranker Network

After span candidates are extracted from *Segmenter*, the goal of *Ranker* is to re-rank the spans, so as to retrieve a final answer corresponding to a given query. The two main model families for ranking are: First, representation-based approach (Wang and Nyberg 2015; Yin et al. 2016) that aggregates all context into a single vector has shown effectiveness in sentence-level prediction tasks, such as answer selection, classification, and paraphrase detection, but less in tasks to deal with long text (Guo et al. 2016; Cohen, Yang, and Croft 2018). Second, for matching long texts, other approaches (Guo et al. 2016; Hui et al. 2017; 2018; Ruckle, Moosavi, and Gurevych 2019) leveraging local interactions to capture relevance between a query and long texts, has been more successful.

However, as shown in Figure 1(a), answer spans for non-factoid setting have diverse length, ranging from a single clip to 30 clips. Therefore, the capability of adapting to short and long answers is crucial for non-factoid QA. To be robust on both long and short answers, we aim to selectively leverage the above two approaches (**representation- and interaction-based** approaches).

Specifically, let top  $N$  candidate spans from *Segmenter* be  $C_i$  where  $i = 1, \dots, N$ , and  $C_i \subseteq T$ . Suppose that the span  $C_i$  has a word sequence  $\bar{S}_i$  and object set  $\bar{F}_i$ , where  $\bar{S}_i = \{w_1^{\bar{S}_i}, w_2^{\bar{S}_i}, \dots, w_n^{\bar{S}_i}\}$  and  $\bar{F}_i = \{c_1^{\bar{F}_i}, c_2^{\bar{F}_i}, \dots, c_n^{\bar{F}_i}\}$ .

**Representation-based feature:** First, for representation feature, we use BERT encoder, which is the same structure with *Segmenter*. While *Segmenter* takes all clips as input, *Ranker* treats a span as a single clip, without a hierarchical structure. The representation feature  $V_{rep}$  is computed by using Eq. (1) and (2). We argue that this encoder supports matching between a query and short answer.



**Interaction-based feature:** Second, to encode interaction feature, we use CNN-based approach, inspired by (Hui et al. 2017; 2018). To encode  $(Q, \bar{S})$ , we calculate the cosine similarity matrix  $M_{l_Q \times l_{\bar{S}}}$  to capture matching signals between words in  $Q$  and  $\bar{S}$ . From the matrix, we extract local interactions between n-by-n grams, through CNN layers. For such n-gram matching, we use multiple ( $l_f$ ) CNN layers with  $(2 \times 2, 3 \times 3, 4 \times 4)$  filters. After passing through CNN, we extract top  $n_t$  strongest signals, by max-pooling over sequence  $\bar{S}$ . The above processes are calculated as follows:

$$\begin{aligned} M(i, j) &= \cos(w_i^Q, w_j^{\bar{S}}) \\ P_{l_Q \times l_{\bar{S}} \times l_f} &= \text{Conv}(M_{l_Q \times l_{\bar{S}}}) \\ \hat{P}_{l_Q \times n_t \times l_f} &= \max(P_{l_Q \times l_{\bar{S}} \times l_f}) \end{aligned} \quad (5)$$

where  $M(i, j)$  indicates  $(i, j)$ -th element in matrix  $M_{l_Q \times l_{\bar{S}}}$ . To make  $(Q, \bar{S})$  as a single vector, we average the  $l_Q$  matrices in terms of each query term, and the averaged matrix  $(n_t \times l_f)$  is concatenated into a  $(n_t \cdot l_f)$ -dimensional vector.

For encoding  $(Q, \bar{F})$ , we use the same structure with CNN layers and max-pooling. But, since visual objects are not sequential, we treat  $\bar{F}$  as bag-of-words, by changing the filter size into  $(2 \times 1, 3 \times 1, 4 \times 1)$ . After concatenating the outputs of  $(Q, \bar{S})$  and  $(Q, \bar{F})$ , the final vector of this interaction feature is passed through a dense layer, as follows:

$$\begin{aligned} H' &= \text{concat}(\text{CNN}(Q, \bar{S}), \text{CNN}(Q, \bar{F})) \\ V_{int} &= NN(H') \end{aligned} \quad (6)$$

where  $NN$  indicates a single-layer neural network using tanh function.

**Our combination approach:** To combine the above two complementary advantages, we propose a length-adaptive gate  $g$  to control their importance as span lengths. We expect that representation feature is effective for short sequence, while interaction feature is effective for long sequence, which will be validated empirically in experiment section as well. For such controller, it needs a monotonic function to give a correlation between the gate’s weight and span length, and have trainable parameters for fitting data. As such function, we use Weibull Distribution (Hazewinkel 2001) to combine  $V_{rep}$  and  $V_{int}$ , and the final score of span  $C_i$  is combined as follows:

$$\begin{aligned} g(x) &= 1 - e^{-(x/\alpha)^\beta} \\ V_{final} &= (1 - g(s_i)) \cdot V_{rep} + g(s_i) \cdot V_{int} \\ S(Q, C_i) &= \sigma(W_5 V_{final} + b_5) \end{aligned} \quad (7)$$

where  $\alpha, \beta, b_5 \in R^1$ ,  $W_5 \in R^{1 \times d}$  are trainable parameters, and  $s_i$  indicates the number of clips in span  $C_i$ . Since  $g(x)$  is a monotonic function, the gate  $g$  weighs on interaction feature as increasing the number of clips. Through a dense layer with sigmoid function, we obtain a score  $S(Q, C_i)$  from the feature  $V_{final}$ . During training, we consider ground-truth span as positive, and randomly select spans as negative.

$$\mathcal{L}(Q, C; \Theta) = \sum_{C^+ \in A} \log \frac{\exp(S(Q, C^+))}{\sum_{k=1}^n \exp(S(Q, C_k^-))} \quad (8)$$

where  $A$  is a set of ground-truth spans,  $C^+$  and  $C^-$  are positive and negative spans, respectively. At inference time, we insert span candidates obtained from *Segmenter*, and select span with highest score as a final answer.

## 4 Experiment

In this section, we formulate our research questions in Section 4.1 to guide our experiments. Then we describe our implementation and evaluation results in Section 4.2 and 4.3, respectively.

### 4.1 Research Questions

To evaluate the effectiveness of our Segmenter-Ranker method, we address the following research questions:

- **RQ1:** How effective is our proposed method for non-factoid QA on videos? Does it outperform state-of-the-art baselines?
- **RQ2:** What is the impact of multimodal encoder?
- **RQ3:** Is our model robust to answers with varying lengths?

### 4.2 Implementation

We use a base version of BERT (Devlin et al. 2018) with 12 layers as our encoder, following its default setting. We train our model on BERT until 3 epochs, and use the Adam optimizer with a learning rate of 0.00005. In *Segmenter*, we extract  $N = 9$  span candidates, from the output probabilities. In *Ranker*, training data has 1:9 positive and negative ratio, then this module ranks top 9 candidates at inference time. For CNN layer, the number  $l_f$  of layers is 30, and top  $n_t = 7$  elements in max-pooling are extracted, which are optimized on dev set. For object features, we utilize a bottom-up-attention model (Anderson et al. 2018), which can detect objects in 1600 categories. For detecting image objects, we sample frames as 1 fps in videos.

### 4.3 Evaluation

For task evaluation, we treat the predicted span and ground-truth as bags of clips, and compute three metrics: (1) Precision, (2) Recall, (3) F1 score. For overall scores, we average each score over all of the instances. We compare our proposed model with the following baselines:

- **Base-I:** Based on only text, BERT for machine reading comprehension (MRC) task (Devlin et al. 2018) can be applied as a baseline. As MRC models are commonly designed to predict word-level span, we change clip-level span to word-level for training, then convert the word-level prediction back into clip-level span at test time.
- **Base-II:** Another text-based model is QANet (Yu et al. 2018). When considering only text, the commonality of our task and QANet is to require the ability of fusing question and passage in encoder. To compare such ability, we replace BERT of our Segmenter with the encoder in the QANet. The encoder of QANet consists of convolutional, self-attention, and query-context attention layers.

Table 2: The comparison of the proposed models on our test set. Compared to version w/o object features, our models (F1 score) outperform with statistical significance (\* indicates  $p < 0.05$ ).

Model	Description	Precision	Metric Recall	F1 score
Base-I	MRC-BERT (Devlin et al. 2018)	50.63	46.14	48.28
Base-II	(Lei et al. 2018) + our decoder	53.43	64.01	58.24
Base-III	(Yu et al. 2018) + our decoder	57.03	66.89	61.57
R-I	Ranker (split-then-rank)	55.24	52.76	53.97
S-I	Segmenter (w/o self-att, LSTM)	50.54	52.11	51.31
S-II	Segmenter (w/o self-att)	63.02	65.17	64.08
S-III	Segmenter (full w/o object feature)	64.01	64.45	64.23
S-IV	Segmenter (full)	64.29	67.21*	65.71*
SR-I	Segmenter + Ranker (representation-only)	64.76	70.65	67.57
SR-II	Segmenter + Ranker (interaction-only)	65.93	71.44	68.57
SR-III	Segmenter + Ranker (concatenation)	65.12	72.78	68.74
SR-IV	Segmenter + Ranker (full w/o object feature)	65.95	71.33	68.53
SR-V	Segmenter + Ranker (full)	<b>66.37</b>	<b>74.75*</b>	<b>70.31*</b>

- **Base-III:** Similarly to our task, a baseline in (Lei et al. 2018) requires to fuse question and video (transcript and visual concepts), using object detection. Since this task is proposed for multiple choice questions, we replace BERT of our Segmenter with their multimodal encoder.
- **Our models:** Our proposed model consists of *Segmenter* and *Ranker*, which is implemented as follows. First, among candidates split by a fixed window (we used the average of answer lengths as the window size ( $=15$ )), *Ranker* selects the most relevant candidate as an answer, which denoted as **R**. Second, we consider Top 1 candidate from *Segmenter* as an answer, which denoted as **S**. Third, *Ranker* selects the most likely answer among Top  $N$  candidates from *Segmenter*, denoted as **SR**. For an ablation study, we remove each individual component in *Segmenter* and *Ranker*.

Table 2 shows the results of different models on test set. As we can see, our full model (SR-V) obtains the best results by achieving F1 score of 70.31, outperforming all baselines. A two-stage method using both *Segmenter* and *Ranker* contributes to higher accuracy, when compared to the version using only *Segmenter*, which leads to 4.6 F1 score gains.

For an ablation study, we remove each component in *Segmenter* and *Ranker*. In *Segmenter*, full S-IV model using self-attention and LSTM outperforms S-I and S-II without hierarchical modeling. This supports the effectiveness of hierarchical modeling considering temporal and global context. To validate the effectiveness of our multimodality, we compare S-III vs S-IV and SR-IV vs SR-V, removing object features in our encoder. In the comparison, the encoder using object features improves performance, especially leading to 2.76 and 3.42 recall gains, on S and SR respectively.

In SR (combining *Segmenter* and *Ranker*), we perform an ablation study for representation- and interaction-based features. When comparing between SR-I and SR-II, interaction-based model outperforms representation-based model on all metrics. Meanwhile, SR-III concatenating two features im-

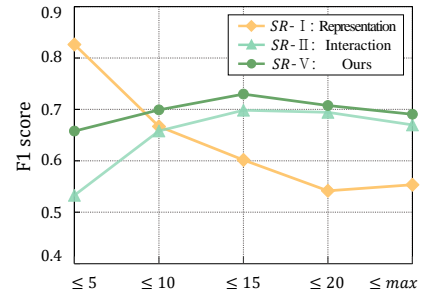


Figure 5: F1 score of SR models over answer lengths. In this figure, the y- and x-axis indicate F1 score and the lengths of the ground-truth span, respectively

proves each model (SR-I,II) slightly. By combining their strengths using our length-adaptive gate (SR-IV), we improve F1 score, compared with SR-I,II,III.

In order to validate our robustness over the answers of varying length, we split test set by answer lengths (per 5 clips), as shown in Figure 5. In SR-I model (representation-based), F1 score tends to decrease as the answer length increases. In contrast, we observe that SR-II model (interaction-based) performs worse, especially for shorter answers. Our SR-V model shows the robustness of maintaining F1 score over 65, in all lengths. This demonstrates that our length-adaptive gate contributes to robustness, by controlling the weights of the two features as lengths. That is, the gate on longer answers promotes the usage of interaction feature, while that on shorter answers weighs on representation feature. By leveraging these two features adaptively to varying lengths, SR-V obtains the best scores over all lengths except over 0-5 clips.

#### 4.4 Qualitative Results

Figure 6 shows the results of the answer predicted by our model, given the questions about *how to make Korean beef*

**Question:** How to cook Korean beef bulgogi



(a) Example 1 describes how to cook Korean beef.

**Question:** How to make a pizza cake



(b) Example 2 describes how to make a pizza cake

Figure 6: The examples of the results on our full model (SR-IV and V). The red boxes indicate the begin and end of ground-truth answer, while the yellow box is wrong prediction by model.

and a pizza cake. In Figure 6(a), our full model (SR-V) successfully predicts the correct answer span, where the red boxes indicate the begin and end clips of ground-truth. In Figure 6(b), SR-IV (text-only) selects the first clip (yellow box) as the begin point, which is a wrong answer. Meanwhile, SR-V considering multimodality predicts the correct answer (the third clip) in the red box. In the first clip, the narration is related about a pizza, not the given question (pizza cake). Human performing for a pizza cake actually starts at the third clip, along with visual contents of the food. In the clip, the visual objects can be a hint, sharing some words with that in the question. In this case, our full model (SR-V) successfully segmented the begin clip of the answer, while the text-only model failed.

## 5 Related Work

### 5.1 Instructional Video Datasets

Existing datasets on instructional videos include *How2* (Sanabria et al. 2018) and *YouCook2* (Zhou, Xu, and Corso 2018). *How2* dataset collects instructional videos totalling 2,000 hours. While *How2* set contains a large-scale collection of videos with English/Portuguese subtitles, this is not designed for QA task, such that no answer label is provided. *YouCook2* dataset contains 2000 cooking videos from YouTube, with human annotations segmenting cooking procedures, each of which is a single sentence. Our work is closely related, by segmenting videos into answer spans, but their work is limited to the span corresponding to a single sentence.

### 5.2 Text-based Extractive QA

Examples of extractive QA are SQuAD (Rajpurkar et al. 2016), NewsQA (Trischler et al. 2016), and TriviaQA (Joshi et al. 2017), based on web documents. In these tasks, given a pair of passage  $P$  and question  $Q$ , the objective is to estimate an answer span  $A$ , where  $A \subseteq P$ . The majority of extractive QA models (Seo et al. 2016; Wang et al. 2017; Yu et al. 2018) predict the probability of each position in the passage being the begin or end of an answer span. Meanwhile, extractive QA model based on video transcript has been proposed (Gupta, Mehrotra, and Gupta 2018). They similarly tackle the challenge of unsegmented text, by dividing transcripts into meaningful chunks first, and extracting answer spans for factoid questions. However, such approach also targets on short answers, not with varying length. Beyond factoid questions, retrieving a paragraph of answering why- or how-questions has been studied (Ruckle, Moosavi, and Gurevych 2019; Han et al. 2019; Tan et al. 2015). While these approaches can deal with longer answers, they assume that pre-segmented paragraphs are available, which is not available in our problem setting.

## 6 Conclusion

In this paper, we propose a new approach for video-based non-factoid QA, with additional challenges of multimodal segmentation and multigranular matching. In the proposed model, *Segmenter* identifies candidate spans, and *Ranker* computes a multigranular matching score between question-answer pair of diverse lengths. Our experimental results show that we outperform state-of-the-art baselines.

## References

- Anderson, P.; He, X.; Buehler, C.; Teney, D.; Johnson, M.; Gould, S.; and Zhang, L. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, volume 3.
- Cheng, Z.; Li, X.; Shen, J.; and Hauptmann, A. G. 2016. Which information sources are more effective and reliable in video search. In *SIGIR*, 1069–1072. ACM.
- Cohen, D.; Yang, L.; and Croft, W. B. 2018. Wikipassageqa: A benchmark collection for research on non-factoid answer passage retrieval. In *SIGIR*. ACM.
- Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Du, X.; Shao, J.; and Cardie, C. 2017. Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106*.
- Gao, J.; Sun, C.; Yang, Z.; and Nevatia, R. 2017. Tall: Temporal activity localization via language query. *arXiv preprint arXiv:1705.02101*.
- Guo, J.; Fan, Y.; Ai, Q.; and Croft, W. B. 2016. A deep relevance matching model for ad-hoc retrieval. In *CIKM*.
- Guo, J.; Fan, Y.; Pang, L.; Yang, L.; Ai, Q.; Zamani, H.; Wu, C.; Croft, W. B.; and Cheng, X. 2019. A deep look into neural ranking models for information retrieval. *arXiv preprint arXiv:1903.06902*.
- Gupta, A.; Mehrotra, R.; and Gupta, M. 2018. Neural attention reader for video comprehension.
- Han, H.; Choi, S.; Park, H.; and Hwang, S.-w. 2019. Micron: Multigranular interaction for contextualizing representation in non-factoid question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 5892–5897.
- Hazewinkel, M. 2001. Weibull distribution. In *Encyclopedia of Mathematics*. Springer New York.
- Hui, K.; Yates, A.; Berberich, K.; and de Melo, G. 2017. Pacrr: A position-aware neural ir model for relevance matching. *arXiv preprint arXiv:1704.03940*.
- Hui, K.; Yates, A.; Berberich, K.; and de Melo, G. 2018. Copacrr: A context-aware neural ir model for ad-hoc retrieval. In *WSDM*, 279–287. ACM.
- Jang, Y.; Song, Y.; Yu, Y.; Kim, Y.; and Kim, G. 2017. Tgifqa: Toward spatio-temporal reasoning in visual question answering. In *CVPR*, 2758–2766.
- Joshi, M.; Choi, E.; Weld, D. S.; and Zettlemoyer, L. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.
- Kim, J.; Ma, M.; Kim, K.; Kim, S.; and Yoo, C. D. 2019. Gaining extra supervision via multi-task learning for multi-modal video question answering. *arXiv preprint arXiv:1905.13540*.
- Lei, J.; Yu, L.; Bansal, M.; and Berg, T. L. 2018. Tvqa: Localized, compositional video question answering. *arXiv preprint arXiv:1809.01696*.
- Li, G.; Ming, Z.; Li, H.; and Chua, T.-S. 2009. Video reference: question answering on youtube. In *Proceedings of the 17th ACM international conference on Multimedia*.
- Liu, M.; Wang, X.; Nie, L.; He, X.; Chen, B.; and Chua, T.-S. 2018. Attentive moment retrieval in videos. In *SIGIR*.
- Maharaj, T.; Ballas, N.; Rohrbach, A.; Courville, A.; and Pal, C. 2017. A dataset and exploration of models for understanding video data through fill-in-the-blank question-answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6884–6893.
- Nie, L.; Wang, M.; Zha, Z.; Li, G.; and Chua, T.-S. 2011. Multimedia answering: enriching text qa with media information. In *SIGIR*, 695–704. ACM.
- Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.
- Regneri, M.; Rohrbach, M.; Wetzels, D.; Thater, S.; Schiele, B.; and Pinkal, M. 2013. Grounding action descriptions in videos. *TACL* 1:25–36.
- Ruckle, A.; Moosavi, N.; and Gurevych, I. 2019. Coala: A neural coverage-based approach for long answer selection with small data. *AAAI*.
- Sanabria, R.; Caglayan, O.; Palaskar, S.; Elliott, D.; Bar-rault, L.; Specia, L.; and Metze, F. 2018. How2: a large-scale dataset for multimodal language understanding. *arXiv preprint arXiv:1811.00347*.
- Seo, M.; Kembhavi, A.; Farhadi, A.; and Hajishirzi, H. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.
- Tan, M.; Santos, C. d.; Xiang, B.; and Zhou, B. 2015. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*.
- Tapaswi, M.; Zhu, Y.; Stiefelhagen, R.; Torralba, A.; Urtasun, R.; and Fidler, S. 2016. Movieqa: Understanding stories in movies through question-answering. In *CVPR*.
- Trischler, A.; Wang, T.; Yuan, X.; Harris, J.; Sordani, A.; Bachman, P.; and Suleman, K. 2016. Newsqa: A machine comprehension dataset. *arXiv preprint arXiv:1611.09830*.
- Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *NeurIPS*, 5998–6008.
- Wang, D., and Nyberg, E. 2015. A long short-term memory model for answer sentence selection in question answering. In *ACL (Volume 2: Short Papers)*, volume 2, 707–712.
- Wang, W.; Yang, N.; Wei, F.; Chang, B.; and Zhou, M. 2017. Gated self-matching networks for reading comprehension and question answering. In *ACL*, volume 1, 189–198.
- Yin, X., and Ordonez, V. 2017. Obj2text: Generating visually descriptive language from object layouts. *arXiv preprint arXiv:1707.07102*.
- Yin, W.; Schütze, H.; Xiang, B.; and Zhou, B. 2016. Abcnn:



Attention-based convolutional neural network for modeling sentence pairs. *TACL* 4:259–272.

Yin, L. 2006. A two-stage approach to retrieving answers for how-to questions. In *EACL : Student Research Workshop*.

Yu, A. W.; Dohan, D.; Luong, M.-T.; Zhao, R.; Chen, K.; Norouzi, M.; and Le, Q. V. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*.

Zhou, L.; Xu, C.; and Corso, J. J. 2018. Towards automatic learning of procedures from web instructional videos. In *Thirty-Second AAAI Conference on Artificial Intelligence*.