

Unsupervised Keyword Extraction for Full-Sentence VQA

Kohei Uehara

The University of Tokyo
uehara@mi.t.u-tokyo.ac.jp

Tatsuya Harada

The University of Tokyo
RIKEN
harada@mi.t.u-tokyo.ac.jp

Abstract

In the majority of the existing Visual Question Answering (VQA) research, the answers consist of short, often single words, as per instructions given to the annotators during dataset construction. This study envisions a VQA task for natural situations, where the answers are more likely to be sentences rather than single words. To bridge the gap between this natural VQA and existing VQA approaches, a novel unsupervised keyword extraction method is proposed. The method is based on the principle that the full-sentence answers can be decomposed into two parts: one that contains new information answering the question (i.e., keywords), and one that contains information already included in the question. Discriminative decoders were designed to achieve such decomposition, and the method was experimentally implemented on VQA datasets containing full-sentence answers. The results show that the proposed model can accurately extract the keywords without being given explicit annotations describing them.

1 Introduction

Visual recognition is one of the most actively researched fields; this research is expected to be applied to real-world systems such as robots. Since innumerable object classes exist in the real world, training all of them in advance is impossible. Thus, to train image recognition models, it is important for real-world intelligent systems to actively acquire information. One promising approach to acquire information on the fly is *learning by asking*, i.e., generating questions to humans about unknown objects, and consequently learning new knowledge from the human response (Misra et al., 2018; Uehara et al., 2018; Shen et al., 2019). This implies that if we can build a Visual Question Answering (VQA) system (Antol et al., 2015) that functions in the real

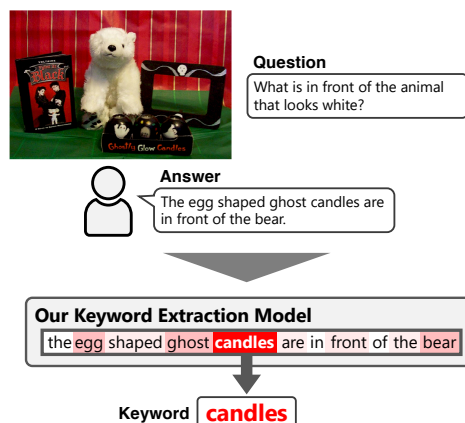


Figure 1: Example of the proposed task – keyword extraction from full-sentence VQA. Given an image, the question, and the full-sentence answer, the keyword extraction model extracts a keyword from the full-sentence answer. In this example, the word “candles” is the most important part, answering the question “What is in front of the animal that looks white?”. Therefore, “candles” is considered as the keyword of the answer.

world and extracts knowledge from human responses, we can realize an intelligent system that can learn autonomously.

VQA is a well-known vision and language task which aims to develop a system that can answer a question about an image. One typical dataset used in VQA is the VQA v2 dataset (Goyal et al., 2017). The answers in the VQA v2 dataset are essentially single words. This is because the annotators are instructed to keep the answer as short as possible when constructing the dataset.

The ultimate goal of the present work is to gain knowledge through VQA that can be easily transferred to other tasks, such as object class recognition and object detection. Therefore, the knowledge (VQA answers) should be represented by a single word, such as a class label. However, in real-world dialog, answers are rarely ex-

pressed by single words; rather, they are often expressed as complete sentences. In fact, in VisDial v1.0 (Das et al., 2017), a dataset of natural conversations about images that does not have a word limit for answers, the average length of answers is 6.5 words. This is significantly longer than the average length of the answers in the VQA v2 dataset (1.2 words).

To bridge the gap between existing VQA research and real-world VQA, a challenging problem must be solved: identifying the word in the sentence that corresponds to the answer to the question. It must also be considered that full-sentence answers provided by humans are likely to follow a variety of sentence structures. Thus, the traditional approaches, such as rule-based approaches based on Part-of-Speech tagging or shallow parsing, require a great deal of work on defining rules in order to extract the keywords. Our key challenge is to propose a novel keyword extraction method that leverages information from images and questions as clues, without the heavy work of annotating keywords or defining the rules.

This work handles the task of extracting a keyword when a full-sentence answer is obtained from VQA (**Full-sentence VQA**). The simplest approach to this task is to construct a dataset containing full-sentence answers and keyword annotations, and then train a model based on this dataset in a supervised manner. However, the cost of constructing a VQA dataset with full-sentence answers and keyword annotations is very high. If a keyword extraction model can be trained on a dataset without keyword annotations, we can eliminate the high cost of collecting keyword annotations.

We propose an unsupervised keyword extraction model using a full-sentence VQA dataset which contains no keyword annotations. Here, the principle is based on the intuition that the keyword is the most informative word in the full-sentence answer, and contains the information that is not included in the question (i.e., the concise answer). Essentially, the full-sentence answer can be decomposed into two types of words: (1) the keyword information that is not included in the question, and (2) the information that is already included in the question. For example, in the answer “The egg shaped ghost candles are in front of the bear.” to the question “What is in front of the animal that looks white?”, the word “candles”

is the keyword, while the remaining part “The egg shaped ghost *something* is in front of the bear” is either information already included in the question or additional information about the keyword. In this case, words like “egg,” “ghost,” and “bear” are also not in the question, making it difficult to find the keyword via naive methods, e.g., rule-based keyword extraction. Our proposed model utilizes image features and question features to calculate the importance score for each word in the full-sentence answer. Therefore, based on the contents of the image and the question, the model can accurately estimate which words in the full-sentence answer are important. To the best of our knowledge, this is the first attempt at extracting a keyword from full-sentence VQA in an unsupervised manner. The main contributions of this work are as follows: (1) We propose a novel task of extracting keywords from full-sentence VQA with no keyword annotations. (2) We designed a novel, unsupervised keyword extraction model by decomposing the full-sentence answer. (3) We conducted experiments on two VQA datasets, and provided both qualitative and quantitative results that show the effectiveness of our model.

2 Related Work

2.1 Unsupervised Keyword Extraction for Text

Unsupervised keyword extraction methods can be broadly classified into two categories: graph-based methods and statistical methods.

Graph-based methods construct graphs from target documents by using co-occurrence between words (Mihalcea and Tarau, 2004; Wan and Xiao). These methods are only applicable to documents with a certain length, as they require the words in the document to co-occur multiple times. The target document in this work is a full-sentence answer of VQA, whose average length is about 10 words. Therefore, graph-based methods are not suitable here.

Statistical methods rely on statistics obtained from a document. The most basic statistical method is TF-IDF (Ramos, 2003), which calculates the term frequency and inverse document frequency and scores each word in the target document. Recent work such as EmbedRank (Bennani-Smires et al., 2018) have utilized word embeddings for the unsupervised keyword extraction. EmbedRank calculates the cosine similarity

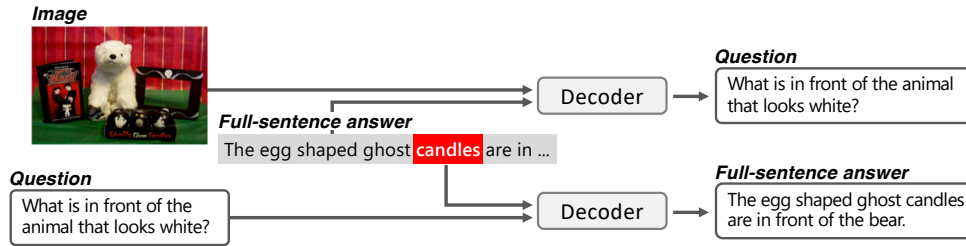


Figure 2: Illustration of the key concept. In this example, the word “candles” is the keyword for the full-sentence answer, “The egg shaped ghost candles are in front of the bear.” We consider the keyword extraction task as the decomposition of the full-sentence answer into answer information and question information. Therefore, if the keyword (i.e., the most informative word in the full-sentence answer) can be accurately extracted, the original full-sentence answer can be reconstructed from it. Additionally, the question can be reconstructed from the decomposed question information in the full-sentence answer.

between the candidate word (or phrase) embeddings and the sentence embeddings to retrieve the most representative word of the text.

2.2 Visual Question Answering

VQA is a well-known task that involves learning from image-related questions and answers. The most popular VQA dataset is VQA v2 (Goyal et al., 2017), and much research has used this dataset for performance evaluations. In VQA v2, the average number of words in an answer is only 1.2, and the variety of answers is relatively limited.

As stated in Section 1, in natural question answering by humans, the answers will be expressed as a sentence rather than a single word. Some datasets that have both full-sentence answers and keyword annotations exist.

FSVQA (Shin et al., 2016) is a VQA dataset with answers in the form of full sentences. In it, full-sentence answers are automatically generated by applying the numerous rule-based natural language processing patterns to the questions and single-word answers in the VQA v1 dataset (Antol et al., 2015).

The recently proposed dataset, named GQA (Hudson and Manning, 2019), also contains automatically generated full-sentence answers. This dataset is constructed on the Visual Genome (Krishna et al., 2017), which has rich and complex annotations about images, including dense captions, questions, and scene graphs. The questions and answers (both single-word and full-sentence) in the GQA dataset are created from scene graph annotations of the images.

The full-sentence answers in both datasets described above are annotated automatically, i.e.,

not by humans. Therefore, neither dataset has both full-sentence answers and manually annotated keywords.

2.3 Attention

The attention mechanism is a technique originally proposed in machine translation (Bahdanau et al., 2015), aimed at focusing on the most important part of the input sequences for a task. Since the method proposed herein utilizes an attention mechanism to calculate the importance score of the word in the full-sentence answer, some prior works on attention mechanisms are discussed.

In general, an attention mechanism essentially learns the mapping between a query and key-value pairs. Transformer (Vaswani et al., 2017) is one of the most popular attention mechanisms for machine translation. It enables machine translation without using recurrent neural networks, using a self-attention mechanism and feed-forward networks instead.

Another study uses an attention mechanism for weakly supervised keyword extraction (Wu et al., 2018). They first trained a model for document classification and extracted the word to which the model pays “attention” to perform the classification. This system requires additional annotations of document class labels to train the model, whereas we aim to extract keywords without any additional annotations.

3 Model

This section describes the proposed method in detail.

First, the principal concept of the model is shown in Figure 2. To extract the keyword, we intend to obtain two features from the full-sentence

answer, each representing the keyword information and the information derived from the question, respectively. To ensure that these two features discriminatively include keyword information and question information, we intend to reconstruct the original questions and answers from the question features and keyword features, respectively. Thus, if we successfully extract the keyword and the question information from the full-sentence answer, we can reconstruct original full-sentence answer and the question. Essentially, given an image, its corresponding question, and full-sentence answer, our proposed model extracts the keyword of the answer by decomposing the keyword information and the question information in the answer.

3.1 Overview

An overview of the model is shown in Figure 3. To realize decomposition-based keyword extraction, we designed a model which consists of the encoder \mathbf{E} , the attention scoring modules \mathbf{S}_a and \mathbf{S}_q , and the decoder modules \mathbf{D}_{all} , \mathbf{D}_a , and \mathbf{D}_q .

An image I and the corresponding question Q and full-sentence answer $A = \{w_1^{(a)}, w_2^{(a)}, \dots, w_n^{(a)}\}$ are considered as the model input. Here, $w_i^{(a)}$ represents the i -th word in the full-sentence answer.

Given I and Q , \mathbf{E} extracts image and question features and integrates them into joint features f_j , i.e., $\mathbf{E}(I, Q) = f_j$.

Next, \mathbf{S}_a and \mathbf{S}_q use f_j and A as input and output the weight vectors \mathbf{a}_k and \mathbf{a}_q . Here, $\mathbf{a}_k = \{a_1^{(k)}, a_2^{(k)}, \dots, a_n^{(k)}\}$ and $\mathbf{a}_q = \{a_1^{(q)}, a_2^{(q)}, \dots, a_n^{(q)}\}$ for each word in A . We denote $a_i \in (0, 1)$ as the weight score of the i -th word in A .

Then, we consider the keyword vector f_k as the embedding vector of the word with the highest weight score in \mathbf{a}_k . Meanwhile, the question information vector f_q is considered as the weighted sum of the embedding vectors of A corresponding to the weight score \mathbf{a}_q .

Following this, \mathbf{D}_{all} uses LSTM to reconstruct the original full-sentence answer using f_q and f_k . f_q and f_k are intended to represent the question information and the keyword vector of the full-sentence answer, respectively. However, \mathbf{D}_{all} only ensures that both features have the information of the full-sentence answer. To separate them, we designed the additional decoders, \mathbf{D}_a and \mathbf{D}_q . The

former reconstructs the BoW features of the answer using f_k , while the latter reconstructs those of the question using f_q with auxiliary vectors. The objective of this operation is to make f_k and f_q representative features for the full-sentence answer and the question, respectively.

The entire model is trained to minimize the disparity between the reconstructed sentences A_{recon} and the original full-sentence answers, as well as that between the BoW features of the full-sentence answers and the questions.

3.2 Encoder

The module \mathbf{E} encodes the image I and the question Q and obtains the image feature f_I , the question feature f_Q , and the joint feature f_j . To generate f_I , we use the image feature extracted from a deep CNN, which is pre-trained on a large-scale image recognition dataset. For f_Q , each word token was converted into a word embeddings and averaged. Following this, l_2 normalization was performed on both features. Finally, those features were concatenated to the joint feature $f_j \in \mathbb{R}^{d_j}$, i.e., $\mathbf{E}(I, Q) = f_j = [f_I; f_Q]$, where d_j is the dimension of the joint feature and $[\cdot]$ indicates concatenation. Note that we did not update the model parameters of \mathbf{E} during training.

3.3 Attention Scoring Module

This module takes f_j as input and weights each words in the full-sentence answer. We used two of these modules, \mathbf{S}_a and \mathbf{S}_q . \mathbf{S}_a and \mathbf{S}_q compute the weights based on the importance of a word for the full-sentence answer and that for the question, respectively. \mathbf{S}_a and \mathbf{S}_q have a nearly identical structure. Therefore, the details of \mathbf{S}_a are presented first, following which the difference between \mathbf{S}_a and \mathbf{S}_q is described.

The weight scoring in these modules is based on the attention mechanism used in Transformer (Vaswani et al., 2017). First, each word in the full-sentence answer was encoded, and the full-sentence answer vector $f_A = \{w_1^{(a)}, w_2^{(a)}, \dots, w_n^{(a)}\} \in \mathbb{R}^{d_e \times n}$ was created. Here, $w_i^{(a)}$ denotes the embedding vector of the i -th word, n is the length of the full-sentence answer, and d_e is the dimension of the word embedding vector. To represent the word order, positional encoding was applied to f_A . Specifically, before feeding f_A into scoring modules, we add positional embedding vectors to f_A , similar to

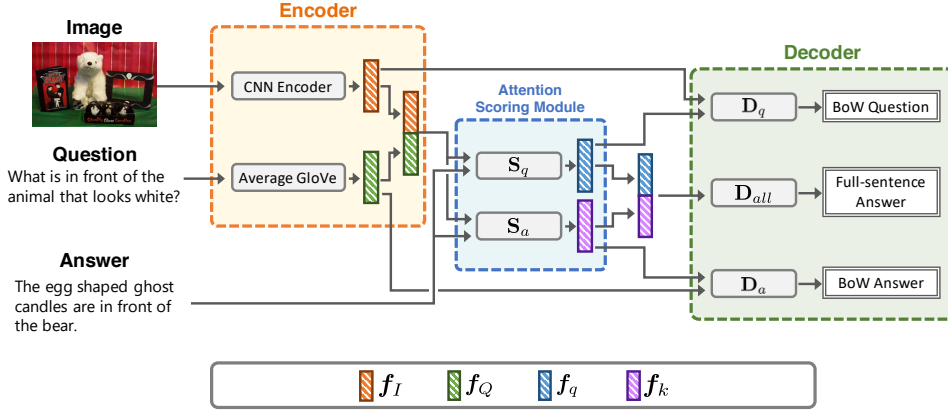


Figure 3: Overall pipeline of the model. First, the Encoder Module extracts the image features f_I and the question features f_Q and integrates them into a joint feature f_j . Then, the Attention Scoring Modules S_a and S_q compute the attention weight and calculate the weighted sum of the word-embedding vectors of the full-sentence answer. The output of S_a i.e., f_k , is the keyword-aware feature of the full-sentence answer, and the output of S_q i.e., f_q , is the question-aware feature. D_{all} reconstructs the full-sentence answer from both f_k and f_q . D_a estimates the Bag-of-Words(BoW) feature of the full-sentence answer from f_k and f_q . Additionally, D_q estimates the BoW feature of the question from f_q and f_I .

those introduced in BERT (Devlin et al., 2019).

We describe our attention mechanism as a mapping between Query and Key-Value pairs. First, we calculate Query vector $Q \in \mathbb{R}^h$, Key vector $K \in \mathbb{R}^{h \times n}$, and Value vector $V \in \mathbb{R}^{h \times n}$.

$$Q = \text{FFN}_q(f_j) \quad (1)$$

$$K = \text{FFN}_k(f_A) \quad (2)$$

$$V = \text{FFN}_v(f_A) = \{v_1^{(a)}, v_2^{(a)}, \dots, v_n^{(a)}\} \quad (3)$$

where FFN_q , FFN_k , FFN_v are single-layer feed-forward neural networks. Then, the attention weight vector $a_k = \{a_1^{(k)}, a_2^{(k)}, \dots, a_n^{(k)}\} \in \mathbb{R}^n$, where $a_i^{(k)}$ is the weighted score of the i -th word, is computed as the product of Q and K , as shown below.

$$a_k = K^T Q \quad (4)$$

Then, the word with the highest weighted score is chosen as the keyword of the full-sentence answer:

$$i^{(k)} = \underset{i}{\text{argmax}}(a_i^{(k)}) \quad (5)$$

$$f_k = v_{i^{(k)}}^{(a)} \quad (6)$$

However, the argmax operation is non-differentiable. Therefore, we use an approximation of this operation by softmax with temperature.

$$f_k = V \text{softmax}\left(\frac{a_k}{\tau}\right) \quad (7)$$

where τ is a temperature parameter, and as τ approaches 0, the output of the softmax function becomes a one-hot distribution.

S_q has the same structure as S_a up to the point of computing the attention weight vector a_q . For the keyword vector, we have the intention to focus on the specific word in the full-sentence answer. Therefore, we use the softmax with temperature. However, for the question vector, there is no need to focus on one word. Therefore, the question vector is calculated as the weighted sum of the attention score:

$$f_q = V \text{softmax}(a_q) \quad (8)$$

Then, we applied single-layer feed-forward neural network, followed by layer normalization (Ba et al., 2016) to the output of this module f_k, f_q .

3.4 Decoder

Entire Decoder In the entire decoder D_{all} , the full-sentence is reconstructed from the output of the attention scoring modules f_k and f_q , i.e., $A_{recon} = D_{all}(f_k, f_q)$, where A_{recon} denotes the reconstructed full-sentence answer. We use an LSTM as the sentence generator. As the input to

the LSTM at each step (x_t), \mathbf{f}_k and \mathbf{f}_q are concatenated to the output of the previous step as follows:

$$x_0 = \mathbf{W}_{x_0}[\mathbf{f}_k; \mathbf{f}_q] \quad (9)$$

$$x_t = \mathbf{W}_x[\mathbf{f}_k; \mathbf{f}_q; \hat{s}_{t-1}] \quad (10)$$

where \hat{s}_{t-1} is the output of the LSTM at the $t - 1$ step, and \mathbf{W}_{x_0} and \mathbf{W}_x are the learned parameters.

The objective of \mathbf{D}_{all} is defined by the cross-entropy loss:

$$L_{all} = - \sum_{t=1}^n \log(p(\hat{s}_t = s_t^{(ans)} | s_{1:t-1}^{(ans)})) \quad (11)$$

where $s^{(ans)}$ is the ground-truth full-sentence answer.

Further, word dropout (Bowman et al., 2016), a method of masking input words with a specific probability, is applied. This forces the decoder to generate sentences based on the \mathbf{f}_k and \mathbf{f}_q rather than relying on the previous word.

Discriminative Decoders \mathbf{D}_{all} attempts to reconstruct the full-sentence answer from \mathbf{f}_k and \mathbf{f}_q . Thus, \mathbf{D}_{all} allows the feature vectors to contain the answer information. However, the keyword and question information are intended to be represented by \mathbf{f}_k and \mathbf{f}_q , respectively. Therefore, we designed the discriminative decoders, \mathbf{D}_a and \mathbf{D}_q , to generate \mathbf{f}_k and \mathbf{f}_q , respectively, thus capturing the desired information separately.

\mathbf{D}_a and \mathbf{D}_q reconstruct the full-sentence answer and the question, respectively. This reconstruction is performed with the target of the BoW features of the sentence, rather than the sentence itself. This is because we intend to focus on the content of the sentence and not its sequential information. Sentence reconstruction was also considered as an alternative, but this is difficult to train using LSTM. The BoW feature $\mathbf{b} \in \mathbb{R}^{n_s}$ is represented as a vector whose i -th elements is N_i/L_s , where n_s is the vocabulary size, N_i is the number of occurrences of the i -th word, and L_s is the number of the words in the sentence.

The input to these discriminative decoders consists not only of feature vectors, but also auxiliary vectors, the additional features that assist in reconstruction. Specifically, the auxiliary vector for \mathbf{D}_a is the average of the word embedding vectors in the question, \mathbf{f}_Q , and, for \mathbf{D}_q , the auxiliary vector is the image feature \mathbf{f}_I .

We build the decoder as the following fully-connected layers:

$$\mathbf{y}_a = \mathbf{W}_A[\mathbf{f}_k; \mathbf{f}_Q] + \mathbf{B}_A \quad (12)$$

$$\mathbf{y}_q = \mathbf{W}_Q[\mathbf{f}_q; \mathbf{f}_I] + \mathbf{B}_Q \quad (13)$$

The loss function for the discriminative decoder is the cross-entropy loss between the ground-truth BoW features and the predicted BoW features:

$$L_a = - \sum_{i=1}^{n_a} \mathbf{b}_a[i] \log(\text{softmax}(\mathbf{y}_a[i])) \quad (14)$$

$$L_q = - \sum_{i=1}^{n_q} \mathbf{b}_q[i] \log(\text{softmax}(\mathbf{y}_q[i])) \quad (15)$$

where \mathbf{b} denotes the ground-truth of the BoW features, and n_a and n_q are the vocabulary sizes of the answer and the question, respectively.

3.5 Full Objectives

Finally, the overall objective function for the proposed model is written as

$$L = \lambda_{all}L_{all} + \lambda_aL_a + \lambda_qL_q, \quad (16)$$

where λ_{all} , λ_a , and λ_q are hyper-parameters that balance each loss function.

3.6 Implementation Details

In the encoder \mathbf{E} , image features of size $2048 \times 14 \times 14$ were extracted from the pool-5 layer of the ResNet152 (He et al., 2016). These were pre-trained on ImageNet, and global pooling was applied to obtain 2048-dimensional features. To encode the question words, we used 300-dimensional GloVe embeddings (Pennington et al., 2014). These were pre-trained on the Wikipedia / Gigaword corpus¹.

To convert each word in the full-sentence answer into \mathbf{f}_A , the embedding matrix in the attention scoring module was initialized with the pre-trained GloVe embeddings. The temperature parameter τ is gradually annealed using the schedule $\tau_i = \max(\tau_0 e^{-ri}, \tau_{min})$, where i is the overall training iteration, and other parameters are set as $\tau_0 = 0.5$, $r = 3.0 \times 10^{-5}$, $\tau_{min} = 0.1$. The LSTM in the \mathbf{D}_{all} has a hidden state of 1024 dimensions. The word dropout rate was set to 0.25.

We used the Adam (Kingma and Ba, 2015) optimizer to train the model, which has an initial learning rate of 1.0×10^{-3} .

¹<http://nlp.stanford.edu/projects/glove/>

Dataset		Size	Answer length
GQA	train	943,000	6.69
	val	132,062	6.70
FSVQA	train	139,038	6.11
	val	68,265	6.07
VQA v2	train	443,757	1.16
	val	214,354	1.16

Table 1: Basic statistics of the dataset we used. Note that these FSVQA dataset values were taken after pre-processing. Although VQA v2 was not used in this work, it is included in this table for reference.

4 Experimental Setup

4.1 Dataset

We conducted experiments on two datasets: GQA and FSVQA. In Table 1, we present the basic statistics of both datasets.

GQA GQA (Hudson and Manning, 2019) contains 22M questions and answers. The questions and answers are automatically generated from image scene graphs, and the answers include both the single-word answers and the full-sentence answers. The questions and answers in GQA have unbalanced answer distributions. Therefore, we used a balanced version of this dataset, which is down-sampled from the original dataset and contains 1.7M questions. As pre-processing, we removed the periods, commas, and question marks.

FSVQA FSVQA (Shin et al., 2016) contains 370K questions and full-sentence answers. This dataset was built by applying rule-based processing to the VQA v1 dataset (Antol et al., 2015), and captions in the MSCOCO dataset (Lin et al., 2014), to obtain the full-sentence answers. There are ten annotations (i.e., single-word answers) per question in the VQA v1 dataset. Of these, the annotations with the highest frequency is chosen to create full-sentence answers. If all the frequencies are equal, an annotation is chosen at random. Since the authors do not provide the mapping between single-word answers and full-sentence answers, we considered the annotations with the highest frequency as the single-word answers matching the full-sentence answers. Questions for which the highest frequency annotation cannot be determined were filtered out. Following this process, we obtained 139,038 questions for the training set, and 68,265 questions for the validation set.

4.2 Settings

The model performance was determined based on the keyword accuracy and the Mean Rank. Mean Rank is the average rank of the correct keyword when sorting each word in order of the importance score. Mean Rank is formulated as:

$$\text{Mean Rank} = \frac{1}{N} \sum_i \text{rank}_i. \quad (17)$$

Here, rank_i is the number representing the keyword rank when the words in the i -th answer sentence are arranged in order of the importance (TF-IDF score or attention score, i.e., $a_i^{(k)}$ in Eqn. 5), and N is the size of the overall samples.

We ran experiments with the various existing unsupervised keyword extraction methods for the comparison: (1) TF-IDF (Ramos, 2003), (2) YAKE (Campos et al., 2020), and (3) EmbedRank (Bennani-Smires et al., 2018). Since YAKE removes the words with less than three characters as preprocessing, the Mean Rank cannot be calculated under the same conditions as other methods. Therefore, the Mean Rank of YAKE is not shown. We also conducted an ablation study to show the importance of \mathbf{D}_a and \mathbf{D}_q . In addition, we changed the reconstruction method from BoW estimation to the original sentence generation using LSTM.

5 Experimental Results

The experimental results are shown in Table 2. Also, we provide the accuracy per question types in Appendix A for further analysis. The proposed model, which used BoW estimation in \mathbf{D}_a and \mathbf{D}_q , achieves superior performance on almost all metrics and datasets except for the Mean Rank of FSVQA. As can be seen in the results of the ablation study, this superior performance is achieved even without \mathbf{D}_a and \mathbf{D}_q , which demonstrates the effectiveness of the proposed reconstruction-based method. When using LSTM in \mathbf{D}_a and \mathbf{D}_q , the accuracy and mean rank worsens as compared to those of the proposed model, which reconstructs the BoW in those modules. This is considered to be because sentence reconstruction with LSTM requires management of the sequential information of the sentence, which is more complex than BoW estimation. Since we intended to focus on the contents of the sentence, the BoW is more suitable for these modules.

Model	GQA		FSVQA	
	Accuracy (\uparrow)	Mean Rank (\downarrow)	Accuracy (\uparrow)	Mean Rank (\downarrow)
TF-IDF	0.275	2.86	0.278	3.22
YAKE	0.269	–	0.107	–
EmbedRank	0.306	2.15	0.302	2.32
Ours	0.429 \pm 0.03	2.04 \pm 0.10	0.351 \pm 0.04	2.38 \pm 0.14
Ours w/o D_q	0.318 \pm 0.02	2.44 \pm 0.04	0.298 \pm 0.03	2.67 \pm 0.05
Ours w/o D_a, D_q	0.350 \pm 0.06	3.01 \pm 0.49	0.347 \pm 0.05	2.49 \pm 0.21
Ours (LSTM D_a, D_q)	0.329 \pm 0.01	2.36 \pm 0.04	0.347 \pm 0.01	3.35 \pm 0.11

Table 2: Keyword extraction performance on GQA and FSVQA. Higher accuracies and lower Mean Ranks are desirable. We conducted experiments three times using the proposed method. Note that comparison methods are deterministic algorithms, experiments with them were conducted only once.


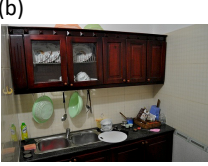
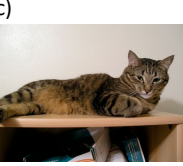
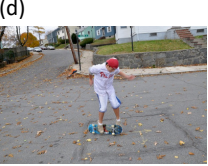
	(a)	(b)	(c)	(d)
Question	 Is the backpack on the right?	 Is the filled tray that is to the right of the bowl made of wicker or wood?	 What color is the wall behind the cat?	 What is the boy riding on?
Full-sentence Answer	No, the backpack is on the left of the image.	The tray is made of wicker.	The wall is behind the cat white.	The boy is riding on skateboard.
GT Keyword	no	wicker	white	skateboard
TF-IDF	backpack	wicker	behind	skateboard
YAKE	backpack	tray	wall	boy
EmbedRank	image	tray	cat	skateboard
Ours	no	wicker	white	skateboard

Figure 4: Examples of the keyword extraction results in the GQA dataset (a, b) and the FSVQA dataset (c, d).

We provide some examples in Figure 4. The examples on the left and right are from GQA and FSVQA, respectively. Since the statistical methods such as TF-IDF tend to choose rarer words as keywords, they are likely to fail if the keyword is a common word (Figure 4 (a), (c)). On the other hand, the model proposed herein can accurately extract keywords even in such cases.

6 Conclusion

In this paper, we proposed the novel task of unsupervised keyword extraction from full-sentence VQA. A novel model was designed to handle this task based on information decomposition of full-sentence answers and the reconstruction of questions and answers. Both qualitative and quantitative experiments show that our model successfully extracts the keyword of the full-sentence answer with no keyword supervision.

In future work, the extracted keywords will

be utilized in other tasks, such as VQA, object classification, or object detection. This work could also be combined with recent works on VQG (Uehara et al., 2018; Shen et al., 2019). In these works, the system generates questions to acquire information from humans. However, they assume that the answers are obtained as single words, which will pose a problem when applying it to the real-world question answering. By combining these studies with our research, an intelligent system can ask humans about unseen objects and learn new knowledge from the answer, even if the answer consists of more than a single word.

Acknowledgement This work was partially supported by JST CREST Grant Number JP-MJCR1403, and partially supported by JSPS KAKENHI Grant Number JP19H01115 and JP20H05556. We would like to thank Yang Li, Sho Maeoki, Sho Inayoshi, and Antonio Tejero-de-Pablos for helpful discussions.

References

- Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *ICCV*.
- Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.
- Kamil Bannani-Smires, Claudiu Musat, Andreea Hossmann, Michael Baeriswyl, and Martin Jaggi. 2018. Simple unsupervised keyphrase extraction using sentence embeddings. In *CoNLL*.
- Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *CoNLL*.
- Ricardo Campos, Vtor Mangaravite, Arian Pasquali, Alpio Jorge, Clia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*.
- Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, Jose M. F. Moura, Devi Parikh, and Dhruv Batra. 2017. Visual dialog. In *CVPR*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.
- Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *CVPR*.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.
- Drew A. Hudson and Christopher D. Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *CVPR*.
- Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.
- Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*.
- Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *ECCV*.
- Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *EMNLP*.
- Ishan Misra, Ross Girshick, Rob Fergus, Martial Hebert, Abhinav Gupta, and Laurens van der Maaten. 2018. Learning by asking questions. In *CVPR*.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.
- Juan Ramos. 2003. Using TF-IDF to Determine Word Relevance in Document Queries. In *the first instructional conference on machine learning*.
- Tingke Shen, Amlan Kar, and Sanja Fidler. 2019. Learning to caption images through a lifetime by asking questions. In *ICCV*.
- Andrew Shin, Yoshitaka Ushiku, and Tatsuya Harada. 2016. The color of the cat is gray: 1 million full-sentences visual question answering (fsvqa). *arXiv preprint arXiv:1609.06657*.
- Kohei Uehara, Antonio Tejero-De-Pablos, Yoshitaka Ushiku, and Tatsuya Harada. 2018. Visual question generation for class acquisition of unknown objects. In *ECCV*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.
- Xiaojun Wan and Jianguo Xiao. Single document keyphrase extraction using neighborhood knowledge.
- Xing Wu, Zhikang Du, and Yike Guo. 2018. A visual attention-based keyword extraction for document classification. *Multimedia Tools and Applications*, 77(19):25355–25367.