Efficient Attention Mechanism for Handling All the Interactions between Many Inputs with Application to Visual Dialog

Van-Quang Nguyen\textsuperscript{1} Masanori Suganuma\textsuperscript{1,2} Takayuki Okatani\textsuperscript{1,2}
\textsuperscript{1}Graduate School of Information Sciences, Tohoku University \textsuperscript{2}RIKEN Center for AIP
Sendai, 980-8579, Japan
{quang, suganuma, okatani}@vision.is.tohoku.ac.jp

Abstract

It has been a primary concern in recent studies of vision and language tasks to design an effective attention mechanism dealing with interactions between the two modalities. The Transformer has recently been extended and applied to several bi-modal tasks, yielding promising results. For visual dialog, it becomes necessary to consider interactions between three or more inputs, i.e., an image, a question, and a dialog history, or even its individual dialog components. In this paper, we present a neural architecture that can efficiently deal with all the interactions between many such inputs. It has a block structure similar to the Transformer and employs the same design of attention computation, whereas it has only a small number of parameters, yet has sufficient representational power for the purpose. Assuming a standard setting of visual dialog, a network built upon the proposed attention block has less than one-tenth of parameters as compared with its counterpart, a natural Transformer extension. We present its application to the visual dialog task. The experimental results validate the effectiveness of the proposed approach, showing improvements of the best NDCG score on the VisDial v1.0 dataset from 57.59 to 60.92 with a single model, from 64.47 to 66.53 with ensemble models, and even to 74.88 with additional finetuning.

1. Introduction

Recently, an increasing amount of attention has been paid to problems lying at the intersection of the vision and language domains. Many pilot tasks in this intersecting region have been designed and introduced to the research community, together with datasets. Visual dialog, which is arguably the most advanced one among them, has been developed aiming at an even higher level of vision-language interactions \cite{8}. It is an extended version of VQA (visual question answering) \cite{2}, which is a single round of question-answering about the image of a scene, to multiple rounds; given an image and a history of question-answer pairs about the image, an agent is required to answer a new question. For example, to answer the question ‘What color are they?’, the agent needs to understand the context from a dialog history to know what ‘they’ refers to and look at the relevant image region to find out a color.

In recent studies of vision-language tasks, a primary concern has been to design an attention mechanism that can effectively deal with interactions between the two modalities. In the case of visual dialog, it becomes further necessary to consider interactions between an image, a question, and a dialog history (three vs. three) or additionally multiple question-answer pairs in the history (many vs. many). Thus, the key to success will be how to deal with such interactions between three and more entities. Following a recent study \cite{34}, we will use the term utility to represent each of these entities (e.g., a collection of image features, etc.) for clarity, since the term modality is inconvenient to distinguish between the question and the dialog history.

Existing studies have considered attention from one utility to another based on different hypotheses, such as “question → history → image” path in \cite{19, 27}, and “question → image → history → question” path in \cite{13, 40}, etc. These methods cannot take all the interactions between utilities into account, although the missing interactions could be crucial. Motivated by this, a recent study tries to capture all the possible interactions by using a factor graph \cite{34}. However, building the factor graph is computationally inefficient, which seemingly hinders the method from unleashing the full potential of modeling all the interactions, especially when the dialog history grows long.

The Transformer \cite{39} has become a standard neural architecture for various tasks in the field of natural language processing, especially since the huge success of its pre-trained model, BERT \cite{12}. Its basic mechanism has recently been extended to the bi-modal problems of vision and language, yielding promising results \cite{7, 14, 25, 26, 44}. Then, it appears to be natural to extend it further to deal with many-to-many utility interactions. However, it is not easy due to several reasons. As its basic structure is designed to be deal with self-attention, even in the simplest case of
The contribution of this study is stated as follows. i) A novel attention mechanism that can deal with all interactions between many utilities is proposed. It is lightweight and yet has a sufficient representational power. ii) Its application to visual dialog is presented, achieving the new state-of-the-art results on the VisDial datasets, e.g., high NDCG scores on the VisDial v1.0 dataset. These validate the effectiveness of the proposed method. iii) Detailed analyses of the experimental results on VisDial v1.0 are given, providing interpretation of how the proposed mechanism works.

2. Related Work

2.1. Attention Mechanisms for Vision-Language Tasks

Attention mechanisms have become an indispensable component to build neural architectures for vision-language tasks, such as VQA [5, 17, 20, 28, 29, 42, 45, 46] and visual grounding [11, 43, 49], etc. Inspired by the recent success of the Transformer for uni-modal language tasks [12, 39], several studies have proposed its extensions to bi-modal vision-language tasks [7, 14, 25, 26, 38, 44]. Specifically, for VQA, it is proposed to use intra-modal and inter-modal attention blocks and stack them alternately to fuse question and image features [14]; it is also proposed to use a cascade of modular co-attention layers that compute the self-attention and guided-attention of question and image features [44]. The method of pretraining a Transformer model used in BERT [12] is employed along with Transformer extension to bi-modal tasks for several vision-language tasks [7, 25, 26]. They first pretrain their models on external datasets, such as COCO Captions [6] or Conceptual Captions dataset [36], and then transfer the models to several target tasks by finetuning them.

2.2. Visual Dialog

The task of visual dialog has recently been proposed by two groups of researchers concurrently [8, 10]. De Vries et al. introduced the GuessWhat?! dataset, which is built upon goal-oriented dialogs held by two agents to identify unknown objects in an image through a set of yes/no questions [10]. Das et al. released the VisDial dataset, which is built upon dialogs consisting of pairs of a question and an answer about an image that are provided in the form of natural language texts [8].

Most of the existing approaches to the task employ an encoder-decoder architecture [37]. They can be categorized into the following three groups by the design of the encoder: i) fusion-based methods, e.g., LF [8] and HRE [8], which fuses the inputs by their concatenation followed by the application of a feed-forward or recurrent network, and Synergistic [15], which fuses the inputs at multiple stages; ii) attention-based methods that compute attended features of the input image, question, and history utilities, e.g., MN [8], CoAtt [40], HIAE [27], Synergistic [15], HACAN [41], ReDAN [13], and FGA [34]; HACAN and ReDAN compute the attention over several reasoning steps, and FGA models all the interactions over many utilities via a factor graph; iii) methods that attempt to resolve visual co-reference, e.g., RvA [30] and CorefNMN [22], which use neural modules to form an attention mechanism, DAN [19], which employs a network having two attention modules, and AMEM [35], which utilizes a memory mechanism for attention.
There are several different designs of the decoder, which can be categorized into the following two groups: i) discriminative decoders that rank the candidate answers using the cross-entropy loss [8] or the n-pair loss [27]; and ii) generative decoders that yield an answer by using a MLE loss [8], weighted likelihood estimation [47], or a combination above categorizations, including GNN [48], which models it to the generator with auxiliary adversarial learning.

There are also other approaches that are not fit for the above categorizations, including GNN [48], which models relations in a dialog by an unknown graph structure to resolve via an EM algorithm and [4, 9], which use reinforcement learning to train two agents to play a guessing game.

3. Efficient Attention Mechanism for Many Utilities

3.1. Attention Mechanism of Transformer

As mentioned earlier, the Transformer has been applied to several bi-modal vision-language tasks, yielding promising results. The Transformer computes and uses attention to several bi-modal vision-language tasks, yielding promising results. The Transformer computes and uses attention to several bi-modal vision-language tasks, yielding promising results.

The above computation is usually multi-plexed in the way called multi-head attention. It enables to use a number of attention distributions in parallel, aiming at an increase in representational power. The outputs of ‘heads’ are concatenated, followed by linear transformation with learnable weights \( W^O \in \mathbb{R}^{d \times d} \) as

\[
A^M(Q, K, V) = \left[ \text{head}_1, \ldots, \text{head}_H \right] W^O. \tag{2}
\]

Each head is computed as follows:

\[
\text{head}_h = A(QW^Q_h, KW^K_h, VW^V_h), \quad h = 1, \ldots, H, \tag{3}
\]

where \( W^Q_h, W^K_h, W^V_h \in \mathbb{R}^{d \times d_H} \) each are learnable weights inducing a linear projection from the feature space of \( d \)-dimensions to a lower space of \( d_H (= d/H) \)-dimensions.

Thus, one attentional block \( A^M(Q, K, V) \) has the following learnable weights:

\[
(W^Q_1, W^K_1, W^V_1), \ldots, (W^Q_H, W^K_H, W^V_H) \text{ and } W^O. \tag{4}
\]

Figure 1: (a) Source-to-target attention for bi-modal problems implemented by the standard Transformer block; the source \( Y \) is attended by weights computed from the similarity between the target \( X \) and \( Y \). (b) The proposed block that can deal with many utilities; the source features \( \{Y_1, \ldots, Y_{U-1}\} \) are attended by weights computed between them and the target \( X \). Shaded boxes have learnable weights.

3.2. Application to Bi-Modal Tasks

While \( Q, K, \) and \( V \) in NLP tasks are of the same modality (i.e., language), the above mechanism has been extended to bi-modality and applied to vision-language tasks in recent studies [7, 14, 25, 26, 38, 44]. They follow the original idea of the Transformer, considering attention from source features \( Y \) to target features \( X \) as

\[
A_Y(X) = A^M(X, Y, Y). \tag{5}
\]

In MCAN [44], language feature is treated as the source and visual feature is as the target.

In [25] and others [7, 14, 26, 38], co-attention, i.e., attention in the both directions, is considered. Self-attention, i.e., the attention from features to themselves, is given as a special case by

\[
A_X(X) = A^M(X, X, X). \tag{6}
\]

In the above studies, the Transformer block with the source-to-target attention and that with the self-attention are independently treated and are stacked, e.g., alternately or sequentially.

3.3. Many-Source-One-Target Attention

Now suppose we wish to extend the above attention mechanism to a greater number of utilities\(^1\); we denote the

\(^1\) As we stated in Introduction, we use the term utility here to mean a collection of features, e.g., either the visual feature, the question feature, or the dialog feature(s) in the case of visual dialog.
number by \( U \). If we consider every possible source-target pairs, there are \( U(U-1) \) cases in total, as there are \( U \) targets, for each of which \( U-1 \) sources exist. Thus, we need to consider attention computation \( \mathcal{A}_Y(X) \) over \( U-1 \) sources \( Y \)’s for each target \( X \). Thus, the straightforward extension of the above attention mechanism to \( U \) utilities needs \( U(U-1) \) times the number of parameters listed in Eq.(4). If we stack the block, the total number of parameters further increases proportionally.

To cope with this, we remove all the weights from Eq.(5). To be specific, for each head \( h(i=1,\ldots,H) \), we choose and freeze \( (W^Q_h, W^K_h, W^V_h) \) as

\[
W^Q_h = W^K_h = W^V_h = [O_{dh}, \ldots, O_{dh}, I_{dh}, O_{dh}, \ldots, O_{dh}]^\top, \tag{7}
\]

where \( O_{dh} \) is a \( d_H \times d_H \) zero matrix and \( I_{dh} \) is a \( d_H \times d_H \) identity matrix. In short, the subspace for each head is determined to be one of \( H \) subspaces obtained by splitting the \( d \)-dimensional feature space with its axis indexes. Besides, we set \( W^{O_h} = I \), which is the linear mapping applied to the concatenation of the heads’ outputs. Let \( \mathcal{A}_Y(X) \) denote this simplified attention mechanism.

Now let the utilities be denoted by \( \{X, Y_1, \ldots, Y_{U-1}\} \), where \( X \in \mathbb{R}^{M \times d} \) is the chosen target and others \( Y_i \in \mathbb{R}^{N_i \times d} \) are the sources. Then, we compute all the source-to-target attention as \( \mathcal{A}_{Y_1}(X), \ldots, \mathcal{A}_{Y_{U-1}}(X) \). In the standard Transformer block (or rigorously its natural extensions to bi-modal problems), the attended features are simply added to the target as \( X + \mathcal{A}_Y(X) \), followed by normalization and subsequent computations. To recover some of the loss in representational power due to the simplification yielding \( \mathcal{A}_Y(X) \), we propose a different approach to aggregate \( \mathcal{A}_{Y_1}(X), \ldots, \mathcal{A}_{Y_{U-1}}(X) \) and \( X \). Specifically, we concatenate all the source-to-target attention plus the self-attention \( \mathcal{A}_X(X) \) from \( X \) to \( X \) as

\[
X_{\text{concat}} = [\mathcal{A}_X(X), \mathcal{A}_{Y_1}(X), \ldots, \mathcal{A}_{Y_{U-1}}(X)], \tag{8}
\]

where \( X_{\text{concat}} \in \mathbb{R}^{M \times d} \). We then apply linear transformation to it given by \( W \in \mathbb{R}^{U \times d \times d} \) and \( b \in \mathbb{R}^d \) with a single fully-connected layer, followed by the addition of the original \( X \) and layer normalization as:

\[
\tilde{X} = \text{LayerNorm}(\text{ReLU}(X_{\text{concat}}W + 1_M^d b^\top + X)), \tag{9}
\]

where \( 1_M^d \) is \( M \)-vector with all ones. With this method, we aim at recovery of representational power as well as the effective aggregation of information from all the utilities.

### 3.4. Interactions between All Utilities

We have designed a basic block (Fig. 1(b)) that deals with attention from many sources to a single target. We wish to consider all possible interactions between all the utilities, not a single utility being the only target. To do this, we use \( U \) basic blocks to consider all the source-to-target attention. Using the basic block as a building block, we show how an architecture is designed for visual dialog having three utilities, visual features \( V \), question features \( Q \), and dialog history features \( R \), in Fig. 2.

### 4. Application to Visual Dialog

We apply the proposed architecture to visual dialog. In this section, we describe the detailed implementation.

#### 4.1. Problem Definition

The problem is stated as follows. An agent is given the image of a scene and a dialog history containing \( T \) entities, which consists of a caption and question-answer pairs at \( T-1 \) rounds. Then, the agent is further given a new question at round \( T \) along with 100 candidate answers for it and requested to answer the question by choosing one or scoring each of the candidate answers.

#### 4.2. Representation of Utilities

We first extract features from an input image, a dialog history, and a new question at round \( T \) to obtain their representations. For this, we follow the standard method employed in many recent studies. For the image utility, we use the bottom-up mechanism [1], which extracts region-level image features using the Faster-RCNN [33] pre-trained on the Visual Genome dataset [23]. For each region (i.e., a bounding box = an object), we combine its CNN feature and geometry to get a \( d \)-dimensional vector \( v_i \) \((i = 1, \ldots, K)\), where \( K \) is the predefined number of regions. We then define \( V = [v_1, v_2, \ldots, v_K]^\top \in \mathbb{R}^{K \times d} \). For the question utility, after embedding each word using an embedding layer initialized by pretrained GloVe vectors, we use two-layer Bi-LSTM to transform them to \( q_i \) \((i = 1, \ldots, N)\), where \( N \) is the number of words in the question. We optionally use the positional embedding widely used in NLP studies. We examine its effects in an ablation test. We then define \( Q = [q_1, \ldots, q_N]^\top \in \mathbb{R}^{N \times d} \). For the dialog history utility,
we choose to represent it as a single utility here. Thus, each of its entities represents the initial caption or the question-answer pair at one round. As with the question utility, we use the same embedding layer and a two-layer Bi-LSTM together with the positional embeddings for the order of dialog rounds to encode them with a slight difference in formation of an entity vector \( r_i \) \((i = 1, \ldots ,T)\), where \( T \) is the number of Q&A plus the caption. We then define \( R = [r_1, \ldots ,r_T]^{\top} \in \mathbb{R}^{T \times d} \). More details are provided in the supplementary material.

4.3. Overall Network Design

The entire network consists of an encoder and a decoder. The encoder consists of \( L \) stacks of the proposed attention blocks; a single stack has \( U \) blocks in parallel, as shown in Fig. 2(b). We set \( V_0 = V, Q_0 = Q, \) and \( R_0 = R \) as the inputs of the first stack. After the \( l \)-th stack, the representations of the image, question, and dialog history utilities are updated as \( V_l, Q_l, \) and \( R_l \), respectively. In the experiments, we apply dropout with the rate of 0.1 to the linear layer inside every block. There is a decoder(s) on top of the encoder. We consider a discriminative decoder and a generative decoder, as in previous studies. Their design is explained below.

4.4. Design of Decoders

Decoders receive the updated utility representations, \( V_L, Q_L, \) and \( R_L \) at their inputs. Following [29], we convert them independently into \( d \)-dimensional vectors \( c_V, c_Q, \) and \( c_R \), respectively. This conversion is performed by a simple self-attention computation. We take \( c_V \) as an example here. First, attention weights over the entities of \( V_L \) are computed by a two layer network as

\[
a_V = \text{softmax}(\text{ReLU}(V_L W_1 + 1_K b_1)W_2 + 1_K b_2),
\]

where \( W_1 \in \mathbb{R}^{d \times d}, W_2 \in \mathbb{R}^{d \times 1}, b_1 \in \mathbb{R}^d, b_2 \in \mathbb{R}^1, \) and \( 1_K \) is \( K \)-vector with all ones. Then, \( c_V \) is given by

\[
c_V = \sum_{i=1}^{K} v_{L,i}^\top a_{V,i},
\]

where \( v_{L,i} \) is the \( i \)-th row vector of \( V_L \) and \( a_{V,i} \) is the \( i \)-th attention weight (a scalar). The others, i.e., \( c_Q \) and \( c_R \), can be obtained similarly.

These vectors are integrated and used by the decoders. In our implementation for visual dialog, we found that \( c_R \) does not contribute to better results; thus we use only \( c_V \) and \( c_Q \). Note that this does not mean the dialog utility \( R \) is not necessary; it is interacted with other utilities inside the attention computation, contributing to the final prediction. The two \( d \)-vectors \( c_V \) and \( c_Q \) are concatenated as \( [c_V^\top, c_Q^\top]^\top \), and this is projected to \( d \)-dimensional space, yielding a context vector \( c \in \mathbb{R}^d \).

We design the discriminative and generative decoders following the previous studies. Receiving \( c \) and the candidate answers, the two decoders compute the score of each candidate answer in different ways. See details for the supplementary material.

4.5. Multi-Task Learning

We observed in our experiments that accuracy is improved by training the entire network using the two decoders simultaneously. This is simply done by minimizing the sum of the losses, \( \mathcal{L}_D \) for the discriminative one and \( \mathcal{L}_G \) for the generative one (we do not use weights on the losses):

\[
\mathcal{L} = \mathcal{L}_D + \mathcal{L}_G.
\]

The increase in performance may be attributable to the synergy of learning two tasks while sharing the same encoder. Details will be given in Sec. 5.3.

5. Experimental Results

This section reports the results of the experiments we conducted to evaluate the proposed method on the VisDial datasets.

5.1. Experimental Setup

Datasets We use the VisDial v0.9 and v1.0 datasets [8]. The v0.9 dataset consists of the train split (82,783 images) and the val split (40,504 images); they are combined to form the trainval v1.0 split in the v1.0 dataset. The v1.0 dataset additionally contains the val 1.0 split (2,064 images) and test v1.0 split (8,000 images). Each image has a dialog composed of 10 question-answer pairs along with a caption. For each question-answer pair, 100 candidate answers are given. The val v1.0 split and 2,000 images of the trainval v1.0 are provided with dense annotations (i.e., relevance scores) for all candidate answers. Although the test v1.0 split was also densely annotated, the information about the ground truth answers and the dense annotations are not publicly available.

Evaluation metrics From the visual dialog challenge 2018, normalized discounted cumulative gain (NDCG) has been used as the principal metric to evaluate methods on the VisDial v1.0 dataset. Unlike other classical retrieval metrics such as R@1, R@5, R@10, mean reciprocal rank (MRR), and mean rank, which are only based on a single ground truth answer, NDCG is computed based on the relevance scores of all candidate answers for each question, which can properly handle the case where each question has more than one correct answer, such as ‘yes it is’ and ‘yes’; such cases do occur frequently.
Table 1: Comparison of the performances of different methods on the validation set of VisDial v1.0 with discriminative and generative decoders.

| Model   | NDCG ↑ | MRR ↑ | R@1 ↑ | R@5 ↑ | R@10 ↑ | Mean ↓ |
|---------|--------|-------|-------|-------|--------|--------|
| MN      | 55.13/66.99 | 60.42/67.83 | 46.09/38.01 | 78.14/57.49 | 88.05/64.08 | 4.63/18.76 |
| CoAtt   | 57.72/59.24 | 62.91/69.09 | 48.86/40.09 | 80.41/59.37 | 89.83/65.92 | 4.21/17.86 |
| HClIAE | 57.75/59.70 | 62.96/69.09 | 48.94/39.72 | 80.50/58.23 | 89.66/64.73 | 4.24/18.43 |
| ReDAN   | 59.32/60.47 | 64.21/50.02 | 50.60/40.27 | 81.39/59.93 | 90.26/66.78 | 4.05/17.40 |
| Ours    | 62.72/63.58 | 62.32/50.74 | 48.94/40.44 | 78.65/61.61 | 87.88/69.71 | 4.80/14.93 |

Other configurations. We employ the standard method used by many recent studies for the determination of hyperparameters etc. For the visual features, we detect $K = 100$ objects from each image. For the question and history features, we first build the vocabulary composed of 11,322 words that appear at least five times in the training split. The captions, questions, and answers are truncated or padded to 40, 20, and 20 words, respectively. Thus, $N = 20$ for the question utility $Q$, $T$ for the history utilities varies depending on the number of dialogs. We use pre-trained 300-dimensional GloVe vectors [32] to initialize the embedding layer, which is shared for all the captions, questions, and answers.

For the attention blocks, we set the dimension of the feature space to $d = 512$ and the number of heads $H$ in each attention block to 4. We mainly use a two-layer model having two stacks of the proposed attention block. We train our models on VisDial v0.9 and VisDial v1.0 dataset using the Adam optimizer [21] with 5 epochs and 15 epochs respectively. The learning rate is warmed up from $1 \times 10^{-5}$ to $1 \times 10^{-3}$ in the first epoch, then halved every 2 epochs. The batch size is set to 32 for the both datasets.

5.2. Comparison with State-of-the-art Methods

Compared methods. We compare our method with a number of previously published methods on the VisDial v0.9 and VisDial v1.0 datasets, which include LF, HRE, MN [8], LF-Att, MN-Att (with attention) [8], SAN [42], AMEM [35], SF [18], HClIAE [27] and Sequential CoAttention model (CoAtt) [40], Synergistic [15], FGA [34], GNN [48], RvA [30], CorefNMN [22], HCAN [41], DAN [19], and ReDAN [13]. Their comparison in terms of several accuracy metrics on different data splits is summarized in Table 1, 2, 3, and 4, in which ↑ indicates “the higher the better” while ↓ indicates “the lower the better”. Unless noted otherwise, the results of our models are obtained from the output of discriminative decoders.

Results on the val v1.0 split. We first compare single-model performance on the val v1.0 split. We select here MN, CoAtt, HClIAE, and ReDAN for comparison, as their performances from the both decoders in all metrics are available in the literature. To be specific, we use the accuracy values reported in [13] for a fair comparison, in which these methods are reimplemented using the bottom-up-attention features. Similar to ours, all these methods employ the standard design of discriminative and generative decoders as in [8]. Table 1 shows the results. It is seen that our method outperforms all the compared methods on the NDCG metric with large margins regardless of the decoder type. Specifically, as compared with ReDAN, the current state-of-the-art on VisDial v1.0, our model has improved NDCG from 59.32 to 62.72 and from 60.47 to 63.58 with discriminative and generative decoders, respectively.

Results on the test-standard v1.0 split. We next consider performance on the test-standard v1.0 split. In our experiments, we encountered a phenomenon that accuracy values measured by NDCG and other metrics show a trade-off relation, depending much on the choice of metrics (i.e., NDCG or others) for judging convergence at the training time. This is observed in the results reported in [13] and is attributable to the inconsistency between the two types of metrics. Thus, we show two results here, the one obtained using NDCG for judging convergence and the one using MRR for it; the latter is equivalent to performing early stopping.

Table 2 shows single-model performances on the blind
test-standard v1.0 split. With the outputs from the discriminative decoder, our model gains improvement of 3.33 points in NDCG from the best model. When employing the aforementioned early stopping, our model achieves at least comparable or better performance not only in NDCG but also in other metrics.

Many previous studies report improved performance obtained by an ensemble of multiple models. To make a comparison, we create an ensemble with some differences, from initialization with different random seeds to whether to use sharing weights across attention blocks or not, the number of attention blocks (i.e., \( L = 2, 3 \)), and the number of objects in the image (i.e., \( K = 50, 100 \)). Aiming at achieving the best performance, we also enrich the image features by incorporating the class label and attributes of each object in an image, which are also obtained from the pretrained Faster-RCNN model. Details are given in the supplementary material. We take the average of the outputs (probability distributions) from the discriminative decoders of the 16 models to rank the candidate answers. Furthermore, we also test fine-tuning each model with its discriminative decoder on the available dense annotations from the trainval v1.0 and val v1.0, where the cross-entropy loss with soft labels (i.e., relevance scores) is minimized for two epochs.

Table 3 shows the results. It is observed that our ensemble model (w/o the fine-tuning) achieves the best NDCG = 66.53 in all the ensembles of other state-of-the-art models. It yields reasonable performance in other metrics. The fine-tuning further gains a large improvement in NDCG, setting a new state-of-the-art NDCG = 74.88 on the test-standard split of VisDial v1.0 dataset.

**Results on the VisDial v0.9 dataset** Following the previous studies, we report the performance of our method (specifically, the discriminative decoder) on the VisDial dataset v0.9. Table 4 shows the results along with performances of other methods. It shows that our model consistently outperforms all the methods across all metrics: MRR, R@1, R@5, R@10 and Mean.

### 5.3. Ablation Study

To evaluate the effect of each of the components of our method, we perform the ablation study on the val v1.0 split of VisDial dataset. We evaluate here the accuracy of the discriminative decoder and the generative decoder separately. We denote the former by D-NDCG and the latter by G-NDCG, and the accuracy of their averaged model by A-NDCG (i.e., averaging the probability distributions over the candidate answers obtained by the discriminative and generative decoders). The results are shown in Table 5.

The first block of the table shows the effect of the number of stacks of the proposed attention blocks. We observe that the use of two to three stacks achieves good performance on all three measures. More stacks did not bring further improvement, and thus their results are omitted in the table. The best results are obtained in the case of two stacks.

The second block shows the effect of self-attention, which computes the interaction within a utility (i.e., \( \bar{A}_X(X) \)). We examine this, because it can be removed from the attention computation. It is seen that self-attention does contribute to good performance.

The third block shows the impact of sharing the weights across the stacks of the attention blocks. If the weights can be shared as in [24], it contributes a further decrease in the number of parameters. We observe that the performance does drop if weight sharing is employed, but the drop is not very large.

---

**Table 3:** Ablation study on the components of our method on the validation set of VisDial dataset v1.0. ↑ indicates the higher the better.

| Component | Details | A-NDCG ↑ | D-NDCG ↑ | G-NDCG ↑ |
|-----------|---------|----------|----------|----------|
| Number of attention blocks | 1 | 65.37 | 62.06 | 62.95 |
| | 2 | 65.75 | 62.72 | 63.58 |
| | 3 | 65.42 | 62.48 | 63.22 |
| Self-Attention | No | 65.38 | 61.76 | 63.31 |
| | Yes | 65.75 | 62.72 | 63.58 |
| Shared Attention weights | No | 65.75 | 62.72 | 63.58 |
| | Yes | 65.57 | 62.50 | 63.24 |
| Context feature aggregation | [Q] | 65.12 | 61.50 | 63.19 |
| | [Q, V] | 65.75 | 62.72 | 63.58 |
| | [Q, V, R] | 65.53 | 62.37 | 63.38 |
| Decoder Type | Gen | - | - | 62.35 |
| | Disc | - | 61.80 | - |
| | Both | 65.75 | 62.72 | 63.58 |
| The number of objects in an image | 36 | 65.25 | 62.40 | 63.08 |
| | 50 | 65.24 | 62.29 | 63.12 |
| | 100 | 65.75 | 62.72 | 63.58 |
| Positional and spatial embeddings | No | 65.18 | 61.84 | 62.96 |
| | Yes | 65.75 | 62.72 | 63.58 |
The young boy is playing tennis at the court.

Q3: What color is his hair? [It's black]

Q4: Does he wear shorts? [Yes]

Figure 3: Examples of attention weights generated in our model at two Q&A rounds on two images. For each, two types of attention weights are displayed. One is the self-attention weights to compute the context vectors $c_V$ and $c_Q$ from the final utility representations $V_L$ and $Q_L$, which are displayed by the brightness of image bounding boxes and the darkness of question words, respectively. The other is the image-to-question attention $A_V(Q)$ and the history-to-question attention $A_R(Q)$, which are displayed by red boxes connected between an image region (w/ the largest weight) and a question word (w/ the largest self-attention weight) and by the blue highlights on Q&As connecting to a question word in a blue box.

The forth block shows the effect of how to aggregate the context features $c_V$, $c_Q$, and $c_R$ in the decoder(s), which are obtained from the outputs of our encoder. As mentioned above, the context vector $c_R$ of the dialog history does not contribute to the performance. However, the context vector $c_R$ of the image is important for achieving the best performance.

The fifth block shows the effects of simultaneously training the both decoders (with the entire model). It is seen that this contributes greatly to the performance; this indicates the synergy of learning two tasks while sharing the encoder, resulting better generalization as compared with those trained with a single decoder.

We have also confirmed that the use of fewer objects leads to worse results. Besides, the positional embedding for representing the question and history utilities as well as the spatial embedding (i.e., the bounding box geometry of objects) for image utility representation have a certain amount of contribution.

5.4. Visualization of Generated Attention

Figure 3 shows attention weights generated in our model on two rounds of Q&A on two images. We show here two types of attention. One is the self-attention weights used to compute the context vectors $c_V$ and $c_Q$. For $c_V$, the attention weights $a_V$ are generated over image regions (i.e., bounding boxes), as in Eq.(10). Similarly, for $c_Q$, the attention weights are generated over question words. These two sets of attention weights are displayed by brightness of the image bounding-boxes and darkness of question words, respectively, in the center and the rightmost columns. It can be observed from these that the relevant regions and words are properly highlighted at each Q&A round.

The other attention we visualize is the source-to-target attention computed inside the proposed block. We choose here the image-to-question attention $A_V(Q)$ and the history-to-question attention $A_R(Q)$. For each, we compute the average of the attention weights over all the heads computed inside the block belonging to the upper stack. In Fig. 3, the former is displayed by the red boxes connected between an image region and a question word; only the region with the largest weight is shown for the target word; the word with the largest self-attention weight is chosen for the target. The history-to-question attention is displayed by the Q&As highlighted in blue color connected to a selected question word that is semantically ambiguous, e.g., ‘its’, ‘he’, and ‘his’. It is seen that the model performs proper visual grounding for the important words, ‘hair’, ‘shorts’, and ‘tusks’. It is also observed that the model properly resolves the co-reference for the words, ‘he’ and ‘its’.

6. Summary and Conclusion

In this paper, we have proposed an attention mechanism that is specifically designed to solve tasks with many input utilities, such as an image, a question, and a dialog history or instead individual Q&As in it, as in visual dialog. The proposed mechanism can efficiently deal with all the interactions between many input utilities. It has a block structure similar to the Transformer block and is stackable to form a deep network. We have also shown its application to visual dialog. The experimental results validate the effectiveness of the proposed approach.
References

[1] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6077–6086, 2018. 4, 11

[2] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In Proceedings of the IEEE International Conference on Computer Vision, pages 2425–2433, 2015. 1

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. arXiv preprint arXiv:1607.06450, 2016. 11

[4] Prithvi Chattopadhyay, Deshraj Yadav, Viraj Prabhu, Arjun Chandrasekaran, Abhishek Das, Stefan Lee, Dhruv Batra, and Devi Parikh. Evaluating conversational agents via cooperative human-ai games. In Proceedings of AAAI Conference on Human Computation and Crowdsourcing, 2017. 3

[5] Kan Chen, Jiang Wang, Liang-Chieh Chen, Hao yuan Gao, Wei Xu, and Ram Nevatia. Abc-cnn: An attention based convolutional neural network for visual question answering. arXiv preprint arXiv:1511.05960, 2015. 2

[6] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. arXiv preprint arXiv:1504.00325, 2015. 2

[7] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Learning universal image-text representations. arXiv preprint arXiv:1909.11740, 2019. 1, 2, 3

[8] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. Visual dialog. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 326–335, 2017. 1, 2, 3, 5, 6, 7, 12

[9] Abhishek Das, Satwik Kottur, José MF Moura, Stefan Lee, and Dhruv Batra. Learning cooperative visual dialog agents with deep reinforcement learning. In Proceedings of the IEEE International Conference on Computer Vision, pages 2951–2960, 2017. 3

[10] Harm De Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron Courville. Guesswhat?! visual object discovery through multi-modal dialogue. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5503–5512, 2017. 2

[11] Chaorui Deng, Qi Wu, Qingyao Wu, Fuyuan Hu, Fan Lyu, and Mingkui Tan. Visual grounding via accumulated attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 7746–7755, 2018. 2

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018. 1, 2

[13] Zhe Gan, Yu Cheng, Ahmed El Kholy, Linjie Li, Jingjing Liu, and Jianfeng Gao. Multi-step reasoning via recurrent dual attention for visual dialog. In Proceedings of the Conference of the Association for Computational Linguistics, pages 6463–6474, 2019. 1, 2, 6

[14] Peng Gao, Zhengkai Jiang, Haoxuan You, Pan Lu, Steven CH Hoi, Xiaogang Wang, and Hongsheng Li. Dynamic fusion with intra-and inter-modality attention flow for visual question answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6639–6648, 2019. 1, 2, 3

[15] Dalu Guo, Chang Xu, and Dacheng Tao. Image-question-answer synergistic network for visual dialog. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 10434–10443, 2019. 2, 6

[16] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. In Proceedings of the IEEE International Conference on Computer Vision, pages 804–813, 2017. 7

[17] Ilija Ilievski, Shuicheng Yan, and Jiashi Feng. A focused dynamic attention model for visual question answering. arXiv preprint arXiv:1604.01485, 2016. 2

[18] Unnat Jain, Svetlana Lazebnik, and Alexander G Schwing. Two can play this game: visual dialog with discriminative question generation and answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5754–5763, 2018. 6, 7

[19] Gi-Cheon Kang, Jaeseo Lim, and Byoung-Tak Zhang. Dual attention networks for visual reference resolution in visual dialog. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 2024–2033, 2019. 1, 2, 6, 7

[20] Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. Bilinear attention networks. In Advances in Neural Information Processing Systems, pages 1564–1574, 2018. 2

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014. 6

[22] Satwik Kottur, José MF Moura, Devi Parikh, Dhruv Batra, and Marcus Rohrbach. Visual coreference resolution in visual dialog using neural module networks. In Proceedings of the European Conference on Computer Vision, pages 153–169, 2018. 2, 6, 7

[23] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. International Journal of Computer Vision, 123(1):32–73, 2017. 4, 11

[24] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942, 2019. 7

[25] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. arXiv preprint arXiv:1908.03557, 2019. 1, 2, 3
[26] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. arXiv preprint arXiv:1908.02265, 2019. 1, 2, 3

[27] Jiasen Lu, Anitha Kannan, Jianwei Yang, Devi Parikh, and Dhruv Batra. Best of both worlds: Transferring knowledge from discriminative learning to a generative visual dialog model. In Advances in Neural Information Processing Systems, pages 314–324, 2017. 1, 2, 3, 6, 7

[28] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. In Advances in Neural Information Processing Systems, pages 289–297, 2016. 2

[29] Duy-Kien Nguyen and Takayuki Okatani. Improved fusion of visual and language representations by dense symmetric co-attention for visual question answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6087–6096, 2018. 2, 5, 13

[30] Yulei Niu, Hanwang Zhang, Manli Zhang, Jianhong Zhang, Zhiwu Lu, and Ji-Rong Wen. Recursive visual attention in visual dialog. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6679–6688, 2019. 2, 6, 7

[31] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 13

[32] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 1532–1543, 2014. 6, 11

[33] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in Neural Information Processing Systems, pages 91–99, 2015. 4

[34] Idan Schwartz, Seunghak Yu, Tamir Hazan, and Alexander G Schwing. Factor graph attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2039–2048, 2019. 1, 2, 6, 7

[35] Paul Hongsuck Seo, Andreas Lehrmann, Bohyung Han, and Leonid Sigal. Visual reference resolution using attention memory for visual dialog. In Advances in Neural Information Processing Systems, pages 3719–3729, 2017. 2, 6, 7

[36] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, pages 2556–2565, 2018. 2

[37] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Advances in Neural Information Processing Systems, pages 3104–3112, 2014. 2

[38] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2019. 2, 3

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in Neural Information Processing Systems, pages 5998–6008, 2017. 1, 2, 12

[40] Qi Wu, Peng Wang, Chunhua Shen, Ian Reid, and Anton van den Hengel. Are you talking to me? reasoned visual dialog generation through adversarial learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6106–6115, 2018. 1, 2, 3, 6, 7

[41] Tianhao Yang, Zheng-Jun Zha, and Hanwang Zhang. Making history matter: History-advantage sequence training for visual dialog. In Proceedings of the IEEE International Conference on Computer Vision, pages 2561–2569, 2019. 2, 6, 7

[42] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 21–29, 2016. 2, 6, 7

[43] Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. Mattnet: Modular attention network for referring expression comprehension. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1307–1315, 2018. 2

[44] Zhou Yu, Jun Yu, Yuhao Cui, Dacheng Tao, and Qi Tian. Deep modular co-attention networks for visual question answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6281–6290, 2019. 1, 2, 3

[45] Zhou Yu, Jun Yu, Jianping Fan, and Dacheng Tao. Multimodal factorized bilinear pooling with co-attention learning for visual question answering. In Proceedings of the IEEE International Conference on Computer Vision, pages 1821–1830, 2017. 2

[46] Zhou Yu, Jun Yu, Chenchao Xiang, Jianping Fan, and Dacheng Tao. Beyond bilinear: Generalized multimodal factorized high-order pooling for visual question answering. IEEE Transactions on Neural Networks and Learning Systems, 29(12):5947–5959, 2018. 2

[47] Heming Zhang, Shalini Ghosh, Larry Heck, Stephen Walsh, Junting Zhang, Jie Zhang, and C-C Jay Kuo. Generative visual dialogue system via weighted likelihood estimation. In Proceedings of the International Joint Conference on Artificial Intelligence, pages 1025–1031, 2019. 3

[48] Zilong Zheng, Wenguang Wang, Siyuan Qi, and Song-Chun Zhu. Reasoning visual dialogs with structural and partial observations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6669–6678, 2019. 3, 6, 7

[49] Bohan Zhuang, Qi Wu, Chunhua Shen, Ian Reid, and Anton van den Hengel. Parallel attention: A unified framework for visual object discovery through dialogs and queries. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4252–4261, 2018. 2
A. Representations of Utilities

A.1. Image Utility

The image utility is represented by the standard method employed in many recent studies. It is based on the bottom-up mechanism [1], which extracts region-level image features using the Faster-RCNN pre-trained on the Visual Genome dataset [23]. For each input image, we select the top K objects, and represent each of them by a visual feature \( v_i^f \in \mathbb{R}^{2048} \) and a bounding box expressed by \((x_{i1}, x_{i2})\) and \((x_{i3}, x_{i4})\) (the coordinates of the upper-left and lower-right corners.)

The feature vector \( v_i^f \) is then converted into another vector \( v_i^b \in \mathbb{R}^d \) as follows. We introduce the following notation to express a single FC layer with ReLU, to which dropout regularization is applied:

\[
MLP(x) \equiv \text{Dropout(ReLU}(W^T x + b)),
\]

where \( x \in \mathbb{R}^k \) is the input and \( W \in \mathbb{R}^{k \times d} \) and \( b \in \mathbb{R}^d \) are the weights and biases. Then, \( v_i^f \) is obtained by

\[
v_i^f = \text{LayerNorm}(MLP(v_i^f)),
\]

where LayerNorm is the layer normalization [3] applied to the output.

The bounding box geometry is converted into \( v_i^b \in \mathbb{R}^d \) in the following way. First, the image is resized to \( 600 \times 600 \) pixels and the bounding box geometry is transformed accordingly. Then, representing each of the four coordinates by a one-hot vector of size 600, we convert them into the embedding vectors \( \hat{x}_{i1}, \ldots, \hat{x}_{i4} \in \mathbb{R}^d \) using four different embedding layers. Then, we obtain \( v_i^b \) as below

\[
v_i^b = \sum_{j=1}^{4} \text{LayerNorm}(\text{MLP}(\hat{x}_{i,j})).
\]

Finally, \( v_i^f \) encoding the visual feature and \( v_i^b \) encoding the spatial feature are aggregated by adding and normalizing as

\[
v_i = \text{LayerNorm}(v_i^f + v_i^b).
\]

The resulting \( v_i \)'s for the \( K \) objects (\( i = 1, \ldots, K \)) comprise a matrix \( V = [v_1, v_2, \cdots, v_K]^T \in \mathbb{R}^{K \times d} \), which gives the representation of the visual utility.

Optional Image Feature Enrichment In the experiment of comparing ensembles on the test split of Visdial v1.0, we enrich the image features for further improvement. To be specific, for each object, we also obtain a class label with highest probability (e.g. 'cat', 'hair', and 'car') and the top 20 attributes for each class label (e.g., 'curly', 'blond', 'long', and so on, for the label 'hair'). These can be extracted from the Faster-RCNN along with the above CNN features and bounding box geometry. We incorporate these into the image utility representation in the following way.

The class label for the \( i \)-th object is first encoded into an embedding vector \( e_i^c \in \mathbb{R}^{300} \) using the same embedding layer as the question. Then, we convert \( e_i^c \) into a \( d \)-dimensional vector \( v_i^c \) by

\[
v_i^c = \text{LayerNorm}(\text{MLP}(e_i^c)).
\]

Similarly, for the top 20 attributes of each object \( i \), we encode them into embedding vectors of size 300, i.e. \( e_{i,1}^a, \ldots, e_{i,20}^a \), and then convert them further into \( v_i^a \in \mathbb{R}^d \) as

\[
v_i^a = \sum_{j=1}^{20} \text{LayerNorm}(\text{MLP}(e_{i,j}^a)w_{i,j}^a),
\]

where \( w_{i,j}^a \) is the confidence score extracted from the Faster-RCNN for attribute \( j \) of the \( i \)-th object. Then, the visual feature \( v_i^f \), the spatial feature \( v_i^b \), the class feature \( v_i^c \), and the attribute feature \( v_i^a \) are aggregated by their addition followed by normalization as

\[
v_i = \text{LayerNorm}(v_i^f + v_i^b + v_i^c + v_i^a).
\]

We then use these vectors to form the matrix \( V \) instead of Eq.(16).

A.2. Question Utility

The question utility is also obtained by the standard method but with one exception, the employment of positional embedding used in NLP studies. Note that we examine its effects in an ablation test shown in the main paper. A given question sentence is first fit into a sequence of \( N \) words; zero-padding is applied if necessary. Each word \( w_i \ (i = 1, \ldots, N) \) is embedded into a vector \( e_i \) of a fixed size using an embedding layer initialized with pretrained GloVe vectors [32]. They are then inputted into two-layer Bi-LSTM, obtaining two \( d \)-dimensional vectors \( \overrightarrow{h}_i \) and \( \overleftarrow{h}_i \) as their higher-layer hidden state:

\[
\overrightarrow{h}_i = \text{LSTM}(e_i, \overrightarrow{h}_{i-1}),
\overleftarrow{h}_i = \text{LSTM}(e_i, \overleftarrow{h}_{i+1}).
\]
Their concatenation, $h_i = [\tilde{h}_i^T, \hat{h}_i^T]^T$, is then projected back to a $d$-dimensional space using a linear transformation, yielding a vector $q_i$. Positional embedding $q_i^p$ from the paper [39] is added to get the final representation $q_i \in \mathbb{R}^d$ of $w_i$ as

$$q_i = \text{LayerNorm}(q_i^f + q_i^p). \quad (21)$$

The representation of the question utility is given as $Q = [q_1, \ldots, q_N]^T \in \mathbb{R}^{N \times d}$.

### A.3. Dialog History Utility

In this study, we choose to represent the dialog history as a single utility. Each of its entities represents the question-answer pair at one round. As with previous studies, the caption is treated as the first round of $2N$-word which is padded or truncated if necessary. For each round $t > 1$, the word sequences of the question and the answer at the round is concatenated into $2N$-word sequence with zero padding if necessary. As with the question utility, after embedding each word into a GloVe vector, the resulting sequence of $2N$ embedded vectors is inputted to two-layer Bi-LSTM, from which only their last (higher-layer) hidden states are extracted to construct a one-hot vector $c_{\text{d}}$.

They are then projected with a linear transform to a $2$-dimensional vector $[\vec{h}_0^T, \vec{h}_{2N}^T]^T$. We then project it with a linear transform to a $d$-dimensional space, yielding $r_i^f \in \mathbb{R}^d$. For the linear projection, we use different learnable weights from the question utility. As in Eq.(21), we add positional embedding, which represents the order of rounds, and then apply layer normalization, yielding a feature vector of the round $t$ question-answer pair. The history utility is then given by $R = [r_1, \ldots, r_T]^T \in \mathbb{R}^{T \times d}$.

### B. Design of Decoders

#### B.1. Discriminative Decoder

A discriminative decoder outputs the likelihood score for each of 100 candidate answers for the current question at round $T$ in the following way. We use a similar architecture to the one used to extract question features in Sec. A.2 to convert each candidate answer (indexed by $i(=1, \ldots, 100))$ to a feature vector $a_i \in \mathbb{R}^d$. Specifically, it is two-layer Bi-LSTM receiving a candidate answer at its input, on top of which there is a linear projection layer followed by layer normalization. Using the resulting vectors, the score $p_i$ for $i$-th candidate answer is computed by $p_i = \text{softmax}(a_i^T c, \ldots, a_{100}^T c)$. In the test phase, we sort the candidate answers using these scores. In the training phase, the cross-entropy loss $L_D$ between $p = [p_1, \ldots, p_{100}]^T$ and the ground truth label encoded by a one-hot vector $y$ is minimized:

$$L_D = -\sum_{i=1}^{100} y_i p_i. \quad (22)$$

When relevance scores $s = [s_1, \ldots, s_{100}]^T$ over the answer candidates are available (called dense annotation in the VisDial dataset) rather than a single ground truth answer, we can use them by setting $y_i = s_i$ for all $i$’s and minimize the above loss. We employ dropout with rate of 0.1 for the LSTM.

#### B.2. Generative Decoder

Following [8], we also consider a generative decoder to score the candidate answers using the log-likelihood scores. The generative decoder consists of a two-layer LSTM to generate an answer using the context vector $c$ as the initial hidden state. In the training phase, we predict the next token based on the current token from the ground truth answer. In details, we first append the special token “SOS” at the beginning of the ground truth answer, then embedding all the sentence into the embedding vectors $a_{qt} = [w_0, w_1, \ldots, w_N]$ where $w_0$ is the embedding vector of “SOS” token. The hidden state $h_n \in \mathbb{R}^d$ at the $n$-th timestep (extracted from the higher-layer LSTM) is computed given $w_{n-1}$ and $h_{n-1}$ as follows:

$$h_n = \text{LSTM}(w_{n-1}, h_{n-1}), \quad (23)$$

where $h_0$ is initialized by $c$. Thus, we compute $p_n$, the log-likelihood of $n$-th word as

$$p_n = \text{logsoftmax}_p(W_n^T h_n + b_n), \quad (24)$$

where $W_n \in \mathbb{R}^{d \times |V|}$ and $p_n \in \mathbb{R}^{|V|}$, where $|V|$ is the vocabulary size; and $j$ is the index of $n$-th word in the vocabulary.

In the training phase, we minimize $L_G$, the summation of the negative log-likelihood defined by

$$L_G = -\sum_{n=1}^{N} p_n. \quad (25)$$

In the validation and test phase, for each candidate answer $A_{T,i}$, we compute $s_i = \sum_{n=1}^{N} p_n(A_{T,i})$ where $p_n(A_{T,i})$ is the log-likelihood of the $n$-th word in the candidate answer $A_{T,i}$ which is computed similarly as in Eq.(24). Then, the rankings of the candidate answers are derived as $\text{softmax}(s_1, \ldots, s_{100})$. We employ dropout with rate of 0.1 for the LSTM.

### C. Implementation Details

When computing $\bar{A}_Y(X)$, we perform the following form of computation

$$A(Q, K, V) = \text{softmax} \left( \frac{Q K^\top}{\sqrt{d}} \right) V,$$

where we compute a matrix product $Q K^\top$ as above. In the computation of $\bar{A}_X(Y)$, we need another matrix product,
Table 6: Hyperparameters used in the training procedure.

| Hyperparameter                        | Value  |
|---------------------------------------|--------|
| Warm-up learning rate                 | $1e^{-5}$ |
| Warm-up factor                        | 0.2    |
| Initial learning rate after the 1st epoch | $1e^{-3}$ |
| $\beta_1$ in Adam                    | 0.9    |
| $\beta_2$ in Adam                    | 0.997  |
| $\epsilon$ in Adam                   | $1e^{-9}$ |
| Weight decay                          | $1e^{-5}$ |
| Number of workers                     | 8      |
| Batch size                            | 32     |

but it is merely the transposed matrix $KQ^T$ due to the symmetry between $X$ and $Y$. For the computational efficiency, we perform computation of $\bar{A}_Y(X)$ and $\bar{A}_X(Y)$ simultaneously; see MultiHeadAttention($X, Y$) in our code. Further, following [29], we also pad $X$ and $Y$ with two $d$-dimensional vectors that are randomly initialized with He normal initialization. This implements “no-where-to-attend” features in the computation of $\bar{A}_Y(X)$ and $\bar{A}_X(Y)$.

Table 6 shows the hyperparameters used in our experiments, which are selected following the previous studies. We perform all the experiments on a GPU server that has four Tesla V100-SXM2 of 16GB memory with CUDA version 10.0 and Driver version 410.104. It has Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz of 80 cores with the RAM of 376GB memory. We use Pytorch version 1.2 [31] as the deep learning framework.

**D. Qualitative Results**

We provide additional examples of the results obtained by our method in Figs. 4-8. They are divided into two groups, results for which the top-1 prediction coincides with the ground truth answer (Figs. 4-6) and those for which they do not coincide (Figs. 7-8). For each result, we show the attention maps created on the input image and question, respectively.
Figure 4: Examples of results for which the top-1 prediction is the same as the ground truth answer on the validation split of Visdial v1.0. Each row shows selected two rounds of Q&A for one image.
Figure 5: Examples of results for which the top-1 prediction is the same as the ground truth answer on the validation split of Visdial v1.0. Each row shows selected two rounds of Q&A for one image.
Figure 6: Examples of results for which the top-1 prediction is the same as the ground truth answer on the validation split of Visdial v1.0. Each row shows selected two rounds of Q&A for one image.
Figure 7: Examples of results for which the top-1 prediction is different from the ground truth answer on the validation split of Visdial v1.0.
Figure 8: Examples of results for which the top-1 prediction is different from the ground truth answer on the validation split of Visdial v1.0.