Reasoning Visual Dialogs with Structural and Partial Observations

Zilong Zheng*¹, Wenguan Wang*², Siyuan Qi*¹,³, Song-Chun Zhu¹,³
¹University of California, Los Angeles, USA  ²Inception Institute of Artificial Intelligence, UAE  ³International Center for AI and Robot Autonomy (CARA)
zilongzheng0318@ucla.edu, wenguanwang.ai@gmail.com, syqi@cs.ucla.edu, sczhu@stat.ucla.edu
https://github.com/zilongzheng/visdial-gnn

Abstract

We propose a novel model to address the task of Visual Dialog which exhibits complex dialog structures. To obtain a reasonable answer based on the current question and the dialog history, the underlying semantic dependencies between dialog entities are essential. In this paper, we explicitly formalize this task as inference in a graphical model with partially observed nodes and unknown graph structures (relations in dialog). The given dialog entities are viewed as the observed nodes. The answer to a given question is represented by a node with missing value. We first introduce an Expectation Maximization algorithm to infer both the underlying dialog structures and the missing node values (desired answers). Based on this, we proceed to propose a differentiable graph neural network (GNN) solution that approximates this process. Experiment results on the VisDial and VisDial-Q datasets show that our model outperforms comparative methods. It is also observed that our method can infer the underlying dialog structure for better dialog reasoning.

1. Introduction

Visual Dialog has drawn increasing research interests at the intersection of computer vision and natural language processing. In such tasks, an image is given as context input, associated with a summarizing caption and a dialog history of question-answer pairs. The goal is to answer questions posed in natural language about images [9], or recover a follow-up question based on the dialog history [22]. Despite its significance to artificial intelligence and human-computer interaction, it poses a richer set of challenges (see an example in Fig. 1) – requiring representing/understanding a series of multi-modal entities, and reasoning the rich semantic relations/structures among them. An ideal inference algorithm should be able to find out the underlying semantic structure and give a reasonable answer based on this structure.

Previous studies have explored this task through embedding rich features from image representation learned from convolutional neural networks and language (i.e., question-answer pairs, caption) representations learned from recurrent sequential models. Their impressive results well demonstrate the importance of mining and fusing multi-modal information in this area. However, they largely neglect the key role of the rich relational information in dialog. Although a few [67, 62] leveraged co-attention mechanisms to capture cross-modal correlations, their reasoning ability is still quite limited. They typically concatenate the multi-modal features together and directly project the concatenated feature into the answer feature space by a neural network. On one hand, their reasoning process does not fully utilize the rich relational information in this task due to their monolithic vector representations of dialog. On the other hand, their feed-forward network designs fail to deeply and iteratively mine and reason the information from different dialog entities over the inherent dialog structures.

In this work, we consider the problem of recovering the underlying semantic relations/structures among them.

Figure 1. An illustration of the visual dialog task. Left: context image. Middle: image caption, dialog history, current query question, and the predicted answer. Right: the underlying semantic dependencies between nodes in the dialog (darker green links indicate higher dependencies).

*Equal contribution.
dialog structure and reasoning about the question/answer simultaneously. We represent the dialog as a graph, where the nodes are dialog entities and the edges are semantic dependencies between nodes. Given the dialog history as input, we have a partial observation of the graph. Then we formulate the problem as inferring about the values of unobserved nodes (e.g., the queried answer) and the graph structure.

The challenge of the problem is that there is no label for dialog structures. For each individual dialog, we need to recover the underlying structure in an unsupervised manner. The node values could then be inferred iteratively with the graph structure: we can reason about the nodes based on the graph structure, and further improve the structure based on the node values. To tackle this challenge, the insight is that a graph structure essentially specifies a joint probability distribution for all the nodes in the graph. Therefore we can view the queried dialog entities as missing values in the data, the dialog structure as unknown parameters of the distribution. Specifically, we encode the dialog as a Markov Random Field (MRF) where some nodes are observed, and the goal is to infer the edge weights between nodes as well as the value of unobserved nodes. We formulate a solution based on the Expectation-Maximization (EM) algorithm, and provide a graph neural network (GNN) approach to approximate this inference.

Our model provides a unified framework which is applicable to diverse dialog settings (detailed in §4). Besides, it provides extra post-hoc interpretability to show the dialog structures through an implicit learning manner. We evaluate the performance of our method on VisDial v0.9 [9], VisDial v1.0 [9] and VisDial-Q [22] datasets. The experimental results prove that our model is able to automatically parse the dialog structure and infer reasonable answer, and further achieves promising performance.

2. Related Work

Image Captioning aims to annotate images with natural language at the scene level automatically, which has been a long-term active research area in computer vision community. Early work [46, 18] typically tackled this task as a retrieval problem, i.e., finding the best fitting caption from a set of predetermined caption templates. Modern methods [40, 25, 59] were mainly based on a CNN-RNN framework, where the RNN leverages the CNN-representation of an input image to output a word sequence as the caption. In this way, they were freed from dependence of the predefined, expression-limited caption candidate pool. After that, some methods [63, 1, 35] tried to integrate the vanilla CNN-RNN architecture with neural attention mechanisms, like semantic attention [35], and bottom-up/top-down attention [1], to name a few representative ones. Another popular trend [15, 47, 24, 5, 42, 37, 6] in this area focuses on improving the discriminability of caption generations, such as stylized image captioning [15, 6], personalized image captioning [47], and context-aware image captioning [24, 5].

Visual Question Answering focuses on providing a natural language answer given an image and a free-form, open-ended question. It is a more recent (dated back to [39, 21]) and challenging task (need to access information from both the question and image). With the availability of large-scale datasets [49, 2, 16, 20, 23], numerous VQA models were proposed to build multimodal representations upon the CNN-RNN architecture [16, 49], and recently extended with differentiable attentions [63, 36, 64, 66, 1, 38]. Rather than above classification-based VQA models, there were some other work [52, 21, 56, 3] leveraged answer representations into the VQA reasoning, i.e., predicting whether or not an image-question-answer triplet is correct. Teney et al. [57] proposed to solve VQA with graph-structured representations of both visual content and questions, showing the advantages of graph neural network in such structure-rich problems. Narasimhan et al. [44] applied graph convolution networks for factual VQA. However, there are some notable differences between our model and [57, 44] in the fundamental idea and theoretical basis, besides the specific tasks. First, we model the visual dialog task as a problem of inference over a graph with partially observed data and unknown dialog structures. This is one step further than propagating information over a fixed graph structure. Second, we emphasize both graph structure inference (in a unsupervised manner) and unobserved node reasoning. Last, the proposed model provides an end-to-end network architecture to approximate the EM solution and offers a new glimpse into the visual dialog task.

Visual Dialog refers to the task of answering a sequence of questions about an input image [9, 11]. It is the latest vision-and-language problem, after the popularity of image captioning and visual question answering. It requires to reason about the image, the on-going question, as well as the past dialog history. [9] and [11] represented two early attempts towards this direction, but with different dialog settings. In [9], a VisDial dataset is proposed and the questions in this dataset are free-form and may concern arbitrary content of the images. Differently, in [11], a ‘Guess-What’ game is designed to identify a secret object through a series of yes/no questions. Following [9], Lu et al. [34] introduced a generator-discriminator architecture, where the generator are improved using a perceptual loss from the pre-trained discriminator. In [51], a neural attention mechanism, called Attention Memory, is proposed to resolve the current reference in the dialog. Das et al. [10] then extended [9] with an ‘image guessing’ game, i.e., finding a desired image from a lineup of images through multi-round dialog. Reinforcement Learning (RL) was used to tackle this task. Later methods to visual dialog include applying Parallel Attention to discover the object through dialog [67], learning a
Graph Neural Networks [19, 50] draw a growing interest in the machine learning and computer vision communities, with the goal of combining structural representation of graph/graphical models with neural networks. There are two main stream of approaches. One stream is to design neural network operations to directly operate on graph-structured data [13, 45, 43, 53, 12, 27]. Another stream is to build graphically structured neural networks to approximate the learning/inference process of graphical models [30, 55, 29, 14, 4, 17, 60, 8]. Our method falls into this category. Some of these methods [30, 55, 4, 17, 26, 48] implement every graph node as a small neural network and formulate the interactions between nodes as a message propagation process, which is designed to be end-to-end trainable. Some others [65, 30, 33, 31, 8] tried to integrate CRFs and neural networks in a fully differentiable framework, which is quite meaningful for semantic segmentation.

In this work, for the first time, we generalize the task of visual dialog to such a setting that we have partial observation of the nodes (i.e., image, caption and dialog history), and the graph structure (relations in dialog) needs to be automatically inferred. In this setting, the answer is the essentially unobserved node needs to be inferred based on the dialog graph, where the graph structure describes the dependencies among given dialog entities. We propose an essential neural network approach as an approximation to the EM solution of this problem. The proposed GNN is significantly different from most previous GNNs, which consider a graph structure is given.

3. Our Approach

We begin by describing the visual dialog task setup as introduced by Das et al. [9]. Formally, a visual dialog agent is given a dialog tuple \( D = \{(I, C, H_t, Q_t)\} \) as input, including an image \( I \), a caption \( C \), a dialog history till round \( t-1 \), \( H_t = \{(Q_k, A_k), k = 1, \cdots, t-1\} \), and the current question \( Q_t \), being asked at round \( t \). The visual dialog agent is required to return a response \( A_t \) to the question \( Q_t \), by ranking a list of 100 candidate answers.

In our approach, we represent the entire dialog by a graph, and we solve for the optimal queried answer by a GNN as an approximate inference (see Fig. 2). In this graph, the dialog entities \( H_t = \{(Q_k, A_k), k = 1, \cdots, t-1\} \), \( Q_t \), and \( A_t \) are represented as nodes. The graph structure (i.e., edges) represents the semantic dependencies between those nodes. The joint distribution of all the question and answer nodes are described by a Markov Random Field, where the values for some nodes are observed (i.e., the history questions & answers, the current question). The node value for the current answer is unknown, and the model needs to infer its value as well as the graph structure encoded by the edge weights in this MRF.

The joint distribution in this MRF over all the nodes is specified by its potential functions and the graph structure. The potential functions can be learned in the training phase to maximize the likelihood of the training data, and used for inference in the testing phase. However, we cannot learn a fixed graph structure for all dialogs since they are different from dialog to dialog. For dialogs in both training and testing, we need to automatically infer the semantic structures.

In addition, because there is no label (also is hard to obtain) for such graph structures, our model needs to infer them in an unsupervised manner. Viewing the input nodes (i.e., the history questions & answers, the current question) as observed data, the queried answer node as missing data, we adopt an EM algorithm to recover both the distribution parameter (the edge weights) and the missing data (the current answer). In this algorithm, the edge weights and the queried answer node are inferred to maximize the expected log likelihood. Finally, we resemble this inference process by a GNN approach, in which the node values and edge weights are updated in an iterative fashion.

3.1. Dialog as Markov Random Field

We model a dialog as an MRF, in which the nodes represent questions and answers and the edges encode semantic dependencies. Specifically, in a fully connected MRF model, the joint probability of all the nodes \( v \) is:

\[
p(v) = \frac{1}{Z} \exp \left\{ - \sum_i \phi_u(v_i) - \sum_{(i,j) \in E} \phi_p(v_i, v_j) \right\},
\]  

where \( Z \) is a normalizing constant, \( \phi_u(v_i) \) is the unary potential function, and \( \phi_p(v_i, v_j) \) is the pairwise potential function.

In our task, we want to learn a general potential function for all dialogs. We also want to maintain soft relations between nodes (i.e., a connectivity between 0 and 1) instead of just binary relations. Hence we generalize the above form to an MRF with \( 0 \sim 1 \) weighted edges:

\[
p(v|W) = \frac{1}{Z} \exp \left\{ - \sum_i w_i \phi_u(v_i) - \sum_{i,j} w_{i,j} \phi_p(v_i, v_j) \right\},
\]

where \( w_i \) and \( w_{i,j} \) are the weights that compose the edge weight matrix \( W \). Note that we write \( \Phi(v) \) the potential matrix as a compact form of all the potentials between nodes, where \( \Phi_{i,i} = \phi_u(v_i) \) and \( \Phi_{i,j} = \phi_p(v_i, v_j) \).
mizes the expected log likelihood:

\[ \theta \] parameter \( \theta \) parametrized by \( \theta \). The goal is to infer the most likely parameter \( \theta \) and random variable \( z \). The EM algorithm optimizes the expected log likelihood:

\[
Q(\theta, \theta^{old}) = \int p(z|x, \theta^{old}) \log p(x, z|\theta) dz. \tag{3}
\]

An EM algorithm is an iterative process of two steps: expectation (E-step) and maximization (M-step). In the E-step, the above expected likelihood is computed. In the M-step, the parameter \( \theta \) is optimized to maximize this objective:

\[
\theta = \arg \max_{\theta} Q(\theta, \theta^{old}). \tag{4}
\]

The EM iteration always increases the observed data likelihood and terminates when a local minimum is found. However, the expected log likelihood Eq. 3 is often intractable. In the visual dialog task, to compute this quantity we need to enumerate all possible answers to the current question in the entire language space. In practice, we can use a surrogate objective in the E-step, in which we compute the plug-in approximation [58] by a maximum a posteriori (MAP) estimate:

\[
\hat{Q}(\theta, \theta^{old}) = \max_z p(z|x, \theta^{old}) \log p(x, z|\theta). \tag{5}
\]

Then in the M-step we update the \( \theta \) according to this surrogate objective.

3.2. Inference with Partial Observation

Next we briefly review EM as a typical approach to do inference with missing data. Suppose we have observed data \( x \) and unobserved data \( z \), whose joint distribution is parametrized by \( \theta \). The edges represent semantic dependencies between nodes. Some nodes’ values are observed (i.e., nodes that represent the dialog history), and we need to infer the missing values for the unobserved node (i.e., the queried answer) based on the underlying dialog structure. The forward pass of the network emulates an EM algorithm, in which the M-step estimates the edge weights and E-step updates all hidden node states (embeddings) by neural message passing. After a few iterations, the hidden state for the unobserved node (answer) contains the inferred embedding for the missing value.

3.3. MRF with Partial Observations

In the visual dialog task, the question & answer history and the current question is given, hence we know the values for those nodes in the MRF. The task is to find out the missing value of the current answer node and the underlying semantic structure. Suppose in an MRF, we observe some nodes in the graph and we do not know the edge weights \( W \). Denote the observed nodes as \( x \) and the unobserved nodes as \( z \), where \( v = x \cup z \) and \( x \cap z = \emptyset \). Here the weight matrix \( W \) parametrizes the joint distribution of \( x \) and \( z \), hence it can be viewed as the \( \theta \) in the previous section. To jointly infer \( W \) the graph structure (e.g., the semantic dependencies) and \( z \) the missing values (e.g., the queried answer), we run an EM algorithm:

**E-step:** We compute \( z^* = \arg\max_{z} p(z|x, W^{old}) \) to obtain \( \hat{Q}(\theta, \theta^{old}) \) in Eq. 5. This is achieved by a max-product loopy belief propagation [61]. At every iteration, each node sends a (different) message to each of its neighbors and receives a message from each neighbor. After receiving message from neighbors, the belief \( b(v_i) \) for each node \( v_i \) is updated by the max-product update rule:

\[
b(v_i) = \alpha \phi_u(v_i) \prod_{v_j \in N(v_i)} m_{j|i}(v_i), \tag{6}
\]

where \( \alpha \) is a normalizing constant, \( N(v_i) \) denotes the neighbor nodes of \( v_i \), and \( m_{j|i}(v_i) \) is the message from \( v_j \) to \( v_i \). The message is given by:

\[
m_{j|i}(v_i) = \max_{v_j} w_{ij} \phi_p(v_i, v_j) \prod_{v_k \in N(v_j) \setminus v_i} m_{k|j}(v_j). \tag{7}
\]

where \( N(v_j) \setminus v_i \) indicates the all the neighboring nodes of \( v_j \) except \( v_i \).

**M-step:** Based on the estimated \( z^* \) in the E-step, we want to find the edge weights that maximizes the objective Eq. 5:

\[
W = \arg \max_{W} \hat{Q}(W, W^{old}) \tag{8}
\]

\[
= \arg \max_{W} p(z^*|x, W^{old}) \log p(x, z^*|W)
\]

\[
= \arg \max_{W} \log p(x, z^*|W). \tag{8}
\]

The M-step together with E-step forms a coordinate descent algorithm in the objective function \( \hat{Q}(W, W^{old}) \). This algorithm contains two loops: an outer loop of inferring \( z \) and \( \theta \) alternatively, and an inner loop of inferring the missing values \( z \) by iterative belief propagation.
Note that in the partial observed case, for the E-step we fix the observed nodes \( v_e \in x \) and only update the unobserved nodes \( v_u \in z \). Hence we also only need to compute messages from observed nodes to unobserved nodes. The message passing and belief update process iterate until convergence. When the iteration terminates, we obtain an MAP estimate \( z^* \) for the missing values, conditioned on the observed nodes \( x \) and current estimated edge weights \( W \).

### 3.4. GNN with Partial Observations

We design a GNN for the visual dialog task guided by the above formulations. The network is structured as an MRF, in which the caption and each question/answer pair is represented as a node embedding, and the semantic relations are represented by edges. The model contains three different neural modules: message functions, update functions, and link functions. These modules are called iteratively to emulate the above EM algorithm.

**E-step:** We perform a neural message passing/belief propagation [17] for an approximate inference of missing values \( z^* \). This process emulates the belief propagation in the E-step. For each node, we use an hidden state/embedding to represent its value. During belief propagation, the observed variables \( x \) and the edge weights \( W \) are fixed. The hidden states of the unobserved nodes are iteratively updated by communicating with other nodes. Specially, we use message functions \( M(\cdot) \) to summarize messages to nodes coming from other nodes, and update functions \( U(\cdot) \) to update the hidden node states according to the incoming messages.

At each iteration step \( s \), the update function computes a new hidden state for a node after receiving incoming messages:

\[
h^s_{v_i} = U(h^s_{v_i}^{-1}, m^s_{v_i}),
\]

where \( h^s_{v_i} \) is the hidden state/embedding for node \( v \). \( m^s_{v_i} \) is the summarized incoming message for node \( v \) at \( s \)-th iteration. The messages are given by:

\[
m^s_{v_i} = \sum_{v_j \in \mathcal{N}(v_i)} w_{ij} M(h^s_{v_i}^{-1}, h^s_{v_j}^{-1}),(10)
\]

The message passing phase runs for \( S \) iterations towards convergence. At the first iteration, the node hidden states \( h^0_{v_i} \) are initialized by node features \( F_{v_i} \).

**M-step:** Based on the updated hidden states of all the nodes in the E-step, we update the edge weights \( W \) by link functions. A link function \( L(\cdot) \) estimates the connectivity \( w_{ij} \) between two nodes \( v_i \) and \( v_j \) based on their current hidden states:

\[
w_{ij} = L(h^s_{v_i}, h^s_{v_j}).
\]

### 3.5. Network Architecture

At each round of the dialog, we aim to answer the query question based on the image, caption, and the question & answer (QA) history. For dialog round \( t \), we construct \( t+1 \) nodes in which one node represents the caption, \( t-1 \) nodes represent the history of \( t-1 \) rounds of QAs, and one last node represents the answer to the current query question. The embedding for each node is initialized by fusing the image feature and the language embedding of the corresponding sentence(s). As shown in Fig. 3, for the caption node we extract the language embedding of the caption, and fuse it with the image feature as an initialization. For the last node representing the queried answer, we use the corresponding question embedding fused with the image feature to initialize the hidden state. For the rest nodes, the hidden states are initialized by fusing the QA embedding and the image feature. The fusing of language embeddings and image features are achieved by co-attention techniques [36], and more details are introduced in §4. The goal of our approach is to infer the hidden state of the queried answer by the emulated EM algorithm.

After initializing the node hidden states with feature embeddings, we start the iterative inference by first estimating the edge weights. The edge weights are estimated by Eq. 11, where the link function is given by a dot product between transformed hidden states:

\[
w_{ij} = L(h^s_{v_i}, h^s_{v_j}) = \langle fc(h^s_{v_i}), fc(h^s_{v_j}) \rangle
\]

where \( \langle \cdot, \cdot \rangle \) denotes dot product, and \( fc(\cdot) \) denotes multiple
unobserved nodes

\begin{algorithm}
\caption{EM for Graph Neural Network}
\begin{algorithmic}
\State \textbf{Input:} Extracted features $F_{v_x}$ for observed nodes $v_x \in \mathcal{X}$
\State \textbf{Output:} Graph structure $W$, node embeddings $h_{v_x}$ for unobserved nodes $v_x \in \mathcal{Z}$
1: \textit{\textit{Initialization}} */
2: for each observed node $v_x \in \mathcal{X}$ do
3: \quad Initialize $h_{v_x}$ to be $F_{v_x}$
4: end for
5: for each unobserved node $v_x \in \mathcal{Z}$ do
6: \quad Initialize $h_{v_x}$ to be the question embedding
7: end for
8: for $t^\text{th}$ \textit{Expectation-Maximization: outer loop} */
9: while not converged do
10: \quad \textit{\textit{M-step}} */
11: \quad \hspace{1em} for each node pair $(v_i, v_j)$ do
12: \quad \hspace{2em} $w_{ij} = L(h_{v_i}, h_{v_j}) = \langle fc(h_{v_i}), fc(h_{v_j}) \rangle$
13: \quad \hspace{1em} end for
14: \quad \textit{\textit{E-step: inner loop for message passing}} */
15: \quad \hspace{1em} for step $s$ from 1 to $S$ do
16: \quad \hspace{2em} for each $v_x \in \mathcal{Z}$ do
17: \quad \hspace{3em} Compute incoming message for observed $v_x$ */
18: \quad \hspace{3em} $m^{v_x} = \sum_{v_j \in N(v_x)} w_{ij} h^{v_j-1}$
19: \quad \hspace{3em} Update embedding for unobserved $v_x$ */
20: \quad \hspace{3em} $h^{v_x} = U(h^{v_x-1}, m^{v_x}) = \text{GRU}(h^{v_x-1}, m^{v_x})$
21: \quad \hspace{1em} end for
22: \quad \hspace{1em} end for
23: \quad \hspace{1em} end while
\end{algorithmic}
\end{algorithm}

fully connected layers with Rectified Linear Units (ReLU) in between the layers.

Using $M(h^{v_i-1}, h^{v_j-1}) = h^{v_j-1}$ as the message function, the summarized message from all neighbor nodes is computed as $m^{v_x} = \sum_{v_j \in N(v_x)} w_{ij} h^{v_j-1}$. To stabilize the training of the update function, we normalize the sum of weights for edges coming into one node to 1 by a softmax function. Then the node hidden state is update by a Gated Recurrent Unit (GRU) [7]:

$$h^{v_i}_v = U(h^{v_i-1}, m^{v_x}) = \text{GRU}(h^{v_i-1}, m^{v_x}). \quad (13)$$

Here the GRU is chosen for two reasons. First, Eq. 13 has a natural recurrent form. GRU is one type of Recurrent Neural Networks (RNN) that known to be more computationally efficient than Long short-term memory (LSTM). Second, Li et al. [29] has shown that GRU performs well in GNNs as update functions.

The algorithm stops after several iterations of the outer loop for EM, in which the edge weights $W$ and the node hidden states $h_v$ are updated alternatively. Inside each iteration, an inner loop is performed to update the node hidden states. The inner loop emulates the E-step, where a belief propagation is performed. The algorithm is illustrated in Alg. 1. For the visual dialog task, the set of unobserved nodes include only the node that represents the current queried answer.

Finally, we regard the hidden state of the last node as the embedding of the queried answer. To choose one answer from the pre-defined options provided by the dataset, we compute $\langle h_v, h_o \rangle$ where $h_v$ is the node hidden state from the last node and $h_o$ is the language embedding for an option. A softmax activation function is applied to those dot products, and a multi-class cross entropy loss is computed to train the GNN.

4. Experiments

4.1. Performance on VisDial v0.9 [9]

\begin{itemize}
\item **Dataset:** We first evaluate the proposed approach on VisDial v0.9 [9], which was collected via two Amazon Mechanical Turk (AMT) subjects chatting about an image. The first person is allowed to see only the image caption, and instructed to ask questions about the hidden image to better understand the scene. The second worker has access to both image and caption, and is asked to answer the first person’s questions. Both are encouraged to talk in a natural manner. Their conversation is ended after 10 rounds of question answering. VisDial v0.9 contains a total of 1,232,870 dialog question-answer pairs on MSCOCO images [32]. It is split into 80K for \texttt{train}, 3K for \texttt{val} and 40K as the \texttt{test}, in a manner consistent with [9].
\item **Evaluation Protocol:** We follow [9] to evaluate individual responses at each round ($t = 1, 2, \cdots, 10$) in a retrieval setup. Specifically, at test time, every question is coupled with a list of 100 candidate answer options, which a VisDial model is asked to return a sorting of the candidate answers. The model is evaluated on standard retrieval metrics [9]: Recall@1, Recall@5, Recall@10, Mean Reciprocal Rank (MRR), and Mean Rank of human response. Lower value for MR and higher values for all the other metrics are desirable.
\item **Data Preparation:** To pre-process the data, we first resize each image into $224 \times 224$ resolution, and use the output of the last pooling layer (pool5) of VGG-19 [54] as the image feature ($512 \times 7 \times 7$). For the text data, i.e., caption, questions and answers, we convert words to numbers, and remove contractions, before tokenizing. The captions, questions, answers longer than 40, 20, 20 words respectively are truncated. All the texts in the experiment are lowercased. Each word is then turned into a vector representation with a look-up table, whose entries are 300-d vectors learned along other parameters during training. Thus for caption, question and answer, we have the sequences of word embedding with size of $40 \times 300$, $20 \times 300$, and $20 \times 300$, respectively. The embedding of the caption, question or answer, is passed through a two-layered LSTM with 512 hidden states and the output state is used as our final text embeddings. We use the same LSTM and word embedding matrix across question, history, caption and options.
\item **Implementation Details:** We use 2 layers of fully connected layer in Eq. 12. The update function $U(\cdot)$ in Eq. 13
\end{itemize}
Table 1. Quantitative evaluation of discriminative methods on val split of VisDial v0.9 [9]. Our model outperforms most competitors. See §4.1 for more details.

<table>
<thead>
<tr>
<th>Methods</th>
<th>MRR</th>
<th>R@1</th>
<th>R@5</th>
<th>R@10</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>LF [9]</td>
<td>0.5807</td>
<td>43.82</td>
<td>74.68</td>
<td>84.07</td>
<td>5.78</td>
</tr>
<tr>
<td>HRE [9]</td>
<td>0.5846</td>
<td>44.67</td>
<td>74.50</td>
<td>84.22</td>
<td>5.72</td>
</tr>
<tr>
<td>HREA [9]</td>
<td>0.5868</td>
<td>44.82</td>
<td>74.81</td>
<td>84.36</td>
<td>5.66</td>
</tr>
<tr>
<td>MN [9]</td>
<td>0.5965</td>
<td>45.55</td>
<td>76.22</td>
<td>85.37</td>
<td>5.46</td>
</tr>
<tr>
<td>SAN-QI [64]</td>
<td>0.5764</td>
<td>43.44</td>
<td>74.26</td>
<td>83.72</td>
<td>5.88</td>
</tr>
<tr>
<td>HeiCoAtt-Q [36]</td>
<td>0.5788</td>
<td>43.51</td>
<td>74.49</td>
<td>83.96</td>
<td>5.84</td>
</tr>
<tr>
<td>AMEM [51]</td>
<td>0.6160</td>
<td>47.34</td>
<td>78.04</td>
<td>86.84</td>
<td>4.99</td>
</tr>
<tr>
<td>HkieAE-NP-ATT [34]</td>
<td>0.6222</td>
<td>48.48</td>
<td>78.75</td>
<td>87.59</td>
<td>4.81</td>
</tr>
<tr>
<td>LF-Att [9]</td>
<td>0.6242</td>
<td>48.55</td>
<td>78.96</td>
<td>87.75</td>
<td>4.70</td>
</tr>
<tr>
<td>SF [22]</td>
<td>0.6398</td>
<td>50.29</td>
<td>80.71</td>
<td>88.81</td>
<td>4.47</td>
</tr>
<tr>
<td>Ours</td>
<td>0.6285</td>
<td>48.95</td>
<td>79.65</td>
<td>88.36</td>
<td>4.57</td>
</tr>
</tbody>
</table>

Table 2. Quantitative evaluation of discriminative methods on test-standard split of VisDial v1.0 [9]. Our model outperforms all other models across all metrics. See §4.2 for more details.

<table>
<thead>
<tr>
<th>Methods</th>
<th>MRR</th>
<th>R@1</th>
<th>R@5</th>
<th>R@10</th>
<th>Mean</th>
<th>NDCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>LF [9]</td>
<td>0.5542</td>
<td>40.95</td>
<td>72.45</td>
<td>82.83</td>
<td>5.95</td>
<td>0.4531</td>
</tr>
<tr>
<td>HRE [9]</td>
<td>0.5416</td>
<td>39.93</td>
<td>70.45</td>
<td>81.50</td>
<td>6.41</td>
<td>0.4546</td>
</tr>
<tr>
<td>MN [9]</td>
<td>0.5549</td>
<td>40.98</td>
<td>72.30</td>
<td>83.30</td>
<td>5.92</td>
<td>0.4750</td>
</tr>
<tr>
<td>LF-Att [9]</td>
<td>0.5707</td>
<td>42.08</td>
<td>74.82</td>
<td>85.05</td>
<td>5.41</td>
<td>0.4976</td>
</tr>
<tr>
<td>MN-Att [9]</td>
<td>0.5690</td>
<td>42.42</td>
<td>74.00</td>
<td>84.35</td>
<td>5.59</td>
<td>0.4958</td>
</tr>
<tr>
<td>Ours</td>
<td>0.6137</td>
<td>47.33</td>
<td>77.98</td>
<td>87.83</td>
<td>4.57</td>
<td>0.5282</td>
</tr>
</tbody>
</table>

is implemented as a one-layer GRU with 512 hidden states. We use a single Titan Xp GPU to train the network with a batch size of 32. In the experiments, we use the Adam optimizer with a base learning rate of 1e-3 further decreasing to 5e-5. The training converges after ~5 epochs.

Quantitative Results: We compare our method with several state-of-the-art discriminative dialog models, i.e., LF [9], HRE [9], HREA [9], MN [9], SAN-QI [64], HeiCoAtt-Q [36], AMEM [51], HkieAE-NP-ATT [34], SF [22], and SCA [62]. Table 1 summarizes the quantitative results of above competitors and our model. Our model consistently outperforms most approaches, highlighting the importance of understanding the dependencies in visual dialog. Specifically, our R@k (k = 1, 5, 10) is at least 0.4 point higher than SF. Our method only performs slightly worse than SCA, which adopts adversarial learning techniques.

Qualitative Results: In Fig. 4, we present our inferred dialog structure according to the edge weight between each node. We show that the edge weight is relatively high when the correlation between the node pairs is strong. (ii) We present our inferred dialog structure according to the edge weight between each node. We show that the edge weight is relatively high when the correlation between the node pairs is strong. (iii) Table 1 and Fig. 4 illustrate the interpretable and grounded nature of our model. As seen, the suggested model successfully captures the relations in dialog and attend to dialog fragments which are relevant to current question.

4.2. Performance on VisDial v1.0 [9]

Dataset: Then we test our model on the newest version of VisDial v1.0, which is collected in a similar way of VisDial v0.9. For VisDial v1.0, all the VisDial v0.9 (i.e., 1,232,870 dialog question-answer pairs on MSCOCO images [32]) is used for training, extra 20,640 and 8,000 dialog question-answer pairs are used for validation and test, respectively.

Evaluation Protocol: In addition to the five evaluation metrics (i.e., Recall@1, Recall@5, Recall@10, MRR, and Mean Rank of human response) used in VisDial v0.9, an extra metric, Normalized Discounted Cumulative Gain (NDCG), is involved for a more comprehensive quantitative performance study. Higher value for NDCG is better.

Quantitative Results: Five discriminative dialog models (i.e., LF [9], HRE [9], MN [9], LF-Att [9], MN-Att [9]) were included in our experiments. Table 2 presents the overall quantitative comparison results. As seen, the suggested model consistently gaining promising results.

4.3. Performance on VisDial-Q Dataset [9, 22]

Dataset: VisDial-Q dataset [9] provides a solid foundation for assessing the performance of a visual dialog system answering questions. To test the questioner side of visual dialog, Jain et al. [22] further propose a VisDial-Q dataset, which is built upon VisDial v0.9 [9]. The dataset splitting is the same as VisDial v0.9.

Evaluation Protocol: VisDial-Q dataset is accompanied with a retrieval based ‘VisDial-Q evaluation protocol’, analogous to the ‘VisDial evaluation protocol’ in VisDial dataset detailed before. A visual dialog system is required to choose one out of 100 next questions for a given question-answer pair. Similar methodology in [9] is adopted to collect the 100 follow-up question candidates. Therefore, the metrics described in §4.1: Recall@k, MRR, and Mean Rank, are also used for quantitative evaluation.

Data Preparation: We use the same text embedding techniques as we used for §4.1. Different from VisDial task, the first round of QA pair is given to predict next round of question. Thus the maximum round of dialog in the VisDial-Q task is set as 9. Similar as we illustrate in §3.5, we construct t + 1 node with caption and previous history as the first t nodes and the expected question as the last node. We initialize our question node with language embedding of the caption and set the language embedding of corresponding nodes and the expected question as the last node. We initialize our question node with language embedding of the caption and set the language embedding of corresponding nodes and the expected question as the last node.
3 EM iterations outperforms the comparative method in all evaluation metrics. Particularly, we can see that our model with constant graph has a similar performance to the comparative method. This shows the effectiveness of our EM-based inference process. Experiment results on this dataset also shows the generality of our approach: it can infer the underlying dialog structure and reason accordingly about unobserved nodes (next question or current answer).

### 4.4. Diagnostic Experiments

To assess the effect of some essential component of our model, we implement and test several variants: (i) constant graph that fixes edge weight between each pair of nodes to be 1; (ii) graph without EM iteration; and (iii) graph with n EM iterations. Table 4 shows the quantitative evaluations of these model variants on VisDial v0.9 [9]. We summarize our observations here: (a) model without EM iterations performs the worst among all variants. This shows the importance of iteratively updating the node embeddings. (b) In our experiments, message passing with 3 iterations shows the best performance of our proposed model. (c) model using constant graph (3 iterations) performs better than worse than the model without EM iterations, since it iteration-ative updates of node embeddings. However, it is outper-

<table>
<thead>
<tr>
<th>Methods</th>
<th>MRR</th>
<th>R@1</th>
<th>R@5</th>
<th>R@10</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>SF-QI [22]</td>
<td>0.3021</td>
<td>17.38</td>
<td>42.32</td>
<td>57.16</td>
<td>14.03</td>
</tr>
<tr>
<td>SF-QIH [22]</td>
<td>0.4060</td>
<td>26.76</td>
<td>55.17</td>
<td>70.39</td>
<td>9.32</td>
</tr>
<tr>
<td>Ours (w/o iter)</td>
<td>0.3977</td>
<td>25.69</td>
<td>54.52</td>
<td>70.33</td>
<td>9.38</td>
</tr>
<tr>
<td>Ours (const. graph)</td>
<td>0.4025</td>
<td>26.08</td>
<td>53.30</td>
<td>70.83</td>
<td>9.24</td>
</tr>
<tr>
<td>Ours (full, 3 iter)</td>
<td>0.4126</td>
<td>27.15</td>
<td>56.47</td>
<td>71.97</td>
<td>8.86</td>
</tr>
</tbody>
</table>

Table 3. Quantitative evaluation on VisDial-Q dataset [9, 22] with VisDial-Q evaluation protocol. See §4.3 for more details.

In this paper, we develop a novel model for the visual di-

alog task. The backbone of this model is a GNN, in which each node represents a dialog entity and the edge weights represent the semantic dependencies between nodes. An EM-style inference algorithm is proposed for this GNN to estimate the latent relations between nodes and the missing values of unobserved nodes. Experiments are performed on the VisDial and VisDial-Q dataset. Results show that our method is able to find and utilize underlying dialog structures for dialog inference in both tasks, demonstrating the generality and effectiveness of our method.

**Acknowledgements** We thank Prof. Ying Nian Wu from UCLA Statistics Department for helpful discussions. This work reported herein was supported by DARPA XAI grant N66001-17-2-4029, ONR MURI grant N00014-16-1-2007 and ARO grant W911NF-18-1-0296.
References


