Graph-Structured Representations for Visual Question Answering

Damien Teney Lingqiao Liu Anton van den Hengel
Australian Centre for Visual Technologies
The University of Adelaide
{damien.teney,lingqiao.liu,anton.vandenhengel}@adelaide.edu.au

Abstract

This paper proposes to improve visual question answering (VQA) with structured representations of both scene contents and questions. A key challenge in VQA is to require joint reasoning over the visual and text domains. The predominant CNN/LSTM-based approach to VQA is limited by monolithic vector representations that largely ignore structure in the scene and in the question. CNN feature vectors cannot effectively capture situations as simple as multiple object instances, and LSTMs process questions as series of words, which do not reflect the true complexity of language structure. We instead propose to build graphs over the scene objects and over the question words, and we describe a deep neural network that exploits the structure in these representations. We show that this approach achieves significant improvements over the state-of-the-art, increasing accuracy from 71.2% to 74.4% on the "abstract scenes" multiple-choice benchmark, and from 34.7% to 39.1% for the more challenging "balanced" scenes, i.e. *image pairs with fine-grained differences and opposite yes/no answers to a same question.*

1. Introduction

The task of Visual Question Answering has received growing interest in the recent years (see [17, 4, 25] for example). One of the more interesting aspects of the problem is that it combines computer vision, natural language processing, and artificial intelligence. In its *open-ended* form, a question is provided as text in natural language together with an image, and a correct answer must be predicted, typically in the form of a single word or a short phrase. In the *multiple-choice* variant, an answer is selected from a provided set of candidates, alleviating evaluation issues related to synonyms and paraphrasing.

Multiple datasets for VQA have been introduced with either real [4, 14, 17, 21, 31] or synthetic images [4, 30]. Our experiments uses the latter, being based on clip art or "cartoon" images created by humans to depict realistic scenes (they are usually referred to as "abstract scenes", despite this being a misnomer). Our experiments focus on this dataset of clip art scenes as they allow to focus on semantic reasoning and vision-language interactions, in isolation from the performance of visual recognition (see examples in Fig. 5). They also allow the manipulation of the image data so as to better illuminate algorithm performance. A particularly attractive VQA dataset was introduced in [30] by selecting only the questions with binary answers (*e.g.* yes/no) and pairing each (synthetic) image with a minimally-different complementary version that elicits the opposite (no/yes) answer (see examples in Fig. 5, bottom rows). This strongly contrasts with other VQA datasets of real images, where a correct answer is often obvious without looking at the image, by relying on systematic regularities of frequent questions and answers [4, 30]. Performance improvements reported on such datasets are difficult to interpret as actual progress in scene understanding and reasoning as they might similarly be taken to represent a better modeling of the language prior of the dataset. This hampers, or at best obscures, progress toward the greater goal of general VQA. In our view, and despite obvious limitations of synthetic images, improvements on the aforementioned "balanced" dataset constitute an illuminating measure of progress in scene-understanding, because a language

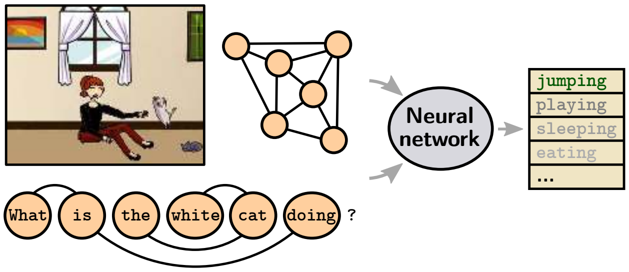

Figure 1. We encode the input scene as a graph representing the objects and their spatial arrangement, and the input question as a graph representing words and their syntactic dependencies. A neural network is trained to reason over these representations, and to produce a suitable answer as a prediction over an output vocabulary.

model alone cannot perform better than chance on this data.

Challenges The questions in the clip-art dataset vary greatly in their complexity. Some can be directly answered from observations of visual elements, *e.g. Is there a dog in the room ?*, or *Is the weather good ?*. Others require relating multiple facts or understanding complex actions, *e.g. Is the boy going to catch the ball?*, or *Is it winter?*. An additional challenge, which affects all VQA datasets, is the sparsity of the training data. Even a large number of training questions (almost 25,000 for the clip art scenes of [4]) cannot possibly cover the combinatorial diversity of possible objects and concepts. Adding to this challenge, most methods for VQA process the question through a recurrent neural network (such as an LSTM) trained from scratch solely on the training questions.

Language representation The above reasons motivate us to take advantage of the extensive existing work in the natural language community to aid processing the questions. First, we identify the syntactic structure of the question using a dependency parser [7]. This produces a graph representation of the question in which each node represents a word and each edge a particular type of dependency (*e.g. determiner*, *nominal subject*, *direct object*, *etc*.). Second, we associate each word (node) with a vector embedding pretrained on large corpora of text data [20]. This embedding maps the words to a space in which distances are semantically meaningful. Consequently, this essentially regularizes the remainder of the network to share learned concepts among related words and synonyms. This particularly helps in dealing with rare words, and also allows questions to include words absent from the training questions/answers. Note that this pretraining and *ad hoc* processing of the language part mimics a practice common for the image part, in which visual features are usually obtained from a fixed CNN, itself pretrained on a larger dataset and with a different (supervised classification) objective.

Scene representation Each object in the scene corresponds to a node in the scene graph, which has an associated feature vector describing its appearance. The graph is fully connected, with each edge representing the relative position of the objects in the image.

Applying Neural Networks to graphs The two graph representations feed into a deep neural network that we will describe in Section 4. The advantage of this approach with text- and scene-graphs, rather than more typical representations, is that the graphs can capture relationships between words and between objects which are of semantic significance. This enables the GNN to exploit (1) the *unordered* nature of scene elements (the objects in particular) and (2) the *semantic relationships* between elements (and the grammatical relationships between words in particular). This contrasts with the typical approach of representing the image with CNN activations (which are sensitive to individual object locations but less so to relative position) and the processing words of the question serially with an RNN (despite the fact that grammatical structure is very non-linear). The graph representation ignores the order in which elements are processed, but instead represents the relationships between different elements using different edge types. Our network uses multiple layers that iterate over the features associated with every node, then ultimately identifies a soft matching between nodes from the two graphs. This matching reflects the correspondences between the words in the question and the objects in the image. The features of the matched nodes then feed into a classifier to infer the answer to the question (Fig. 1).

The main contributions of this paper are four-fold.

1) We describe how to use graph representations of scene and question for VQA, and a neural network capable of processing these representations to infer an answer.
2) We show how to make use of an off-the-shelf language parsing tool by generating a graph representation of text that captures grammatical relationships, and by making this information accessible to the VQA model. This representation uses a pre-trained word embedding to form node features, and encodes syntactic dependencies between words as edge features.
3) We train the proposed model on the VQA "abstract scenes" benchmark [4] and demonstrate its efficacy by raising the state-of-the-art accuracy from 71.2% to 74.4% in the multiple-choice setting. On the "balanced" version of the dataset, we raise the accuracy from 34.7% to 39.1% in the hardest setting (requiring a correct answer over *pairs* of scenes).
4) We evaluate the uncertainty in the model by presenting – for the first time on the task of VQA – precision/recall curves of predicted answers. Those curves provide more insight than the single accuracy metric and show that the uncertainty estimated by the model about its predictions correlates with the ambiguity of the human-provided ground truth.

2. Related work

The task of visual question answering has received increasing interest since the seminal paper of Antol *et al*. [4]. Most recent methods are based on the idea of a **joint embedding** of the image and the question using a deep neural network. The image is passed through a convolutional neural network (CNN) pretrained for image classification, from which intermediate features are extracted to describe the image. The question is typically passed through a re-

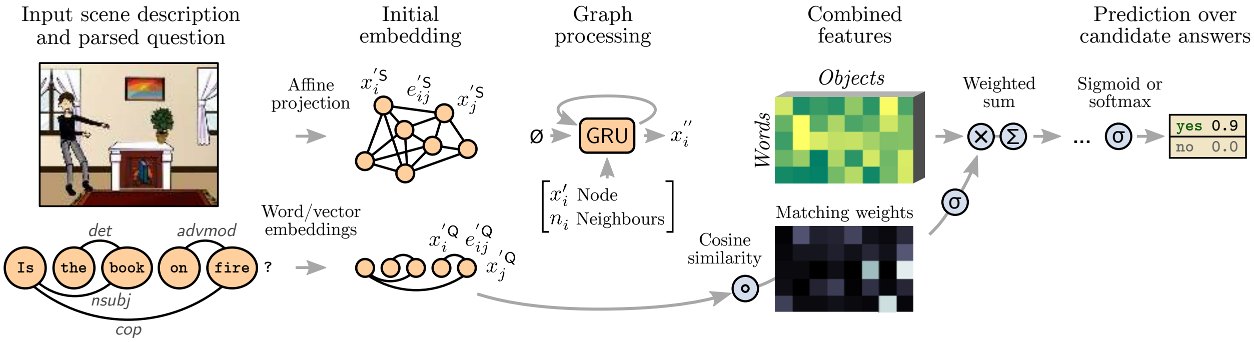

Figure 2. Architecture of the proposed neural network. The input is provided as a description of the scene (a list of objects with their visual characteristics) and a parsed question (words with their syntactic relations). The scene-graph contains a node with a feature vector for each object, and edge features that represent their spatial relationships. The question-graph reflects the parse tree of the question, with a word embedding for each node, and a vector embedding of types of syntactic dependencies for edges. A recurrent unit (GRU) is associated with each node of both graphs. Over multiple iterations, the GRU updates a representation of each node that integrates context from its neighbours within the graph. Features of all objects and all words are combined (concatenated) pairwise, and they are weighted with a form of attention. That effectively matches elements between the question and the scene. The weighted sum of features is passed through a final classifier that predicts scores over a fixed set of candidate answers.

current neural network (RNN) such as an LSTM, which produces a fixed-size vector representing the sequence of words. These two representations are mapped to a joint space by one or several non-linear layers. They can then be fed into a classifier over an output vocabulary, predicting the final answer. Most recent papers on VQA propose improvements and variations on this basic idea. Consult [25] for a survey.

A major improvement to the basic method is to use an **attention mechanism** [31, 27, 5, 12, 3, 28]. It models interactions between specific parts of the inputs (image and question) depending on their actual contents. The visual input is then typically represented a spatial feature map, instead of holistic, image-wide features. The feature map is used with the question to determine spatial weights that reflect the most relevant regions of the image. Our approach uses a similar weighting operation, which, with our graph representation, we equate to a subgraph matching. Graph nodes representing question words are associated with graph nodes representing scene objects and vice versa. Similarly, the co-attention model of Lu *et al*. [16] determines attention weights on both image regions and question words. Their best-performing approach proceeds in a sequential manner, starting with question-guided visual attention followed by image-guided question attention. In our case, we found that a joint, one-pass version performs better.

A major contribution of our model is to use **structured representations** of the input scene and the question. This contrasts with typical CNN and RNN models which are limited to spatial feature maps and sequences of words respectively. The dynamic memory networks (DMN), applied to VQA in [26] also maintain a set-like representation of the input. As in our model, the DMN models interactions between different parts of the input. Our method can additionally take, as input, features characterizing arbitrary relations between parts of the input (the edge features in our graphs). This specifically allows making use of syntactic dependencies between words after pre-parsing the question.

Most VQA systems are trained end-to-end from questions and images to answers, with the exception of the visual feature extractor, which is typically a CNN pretrained for image classification. For the **language processing** part, some methods address the the semantic aspect with word embeddings pretrained on a language modeling task (*e.g*. [23, 9]). The syntactic relationships between the words in the question are typically overlooked, however. In [30], hand-designed rules serve to identify primary and secondary objects of the questions. In the Neural Module Networks [3, 2], the question is processed by a dependency parser, and fragments of the parse, selected with *ad hoc* fixed rules are associated with modules, are assembled into a full neural network. In contrast, our method is trained to make direct use of the output of a syntactic parser.

Neural networks on graphs have received significant attention recently [8, 11, 15]. The approach most similar to ours is the Gated Graph Sequence Neural Network [15], which associate a gated recurrent unit (GRU [6]) to each node, and updates the feature vector of each node by iteratively passing messages between neighbours. Also related is the work of Vinyals *et al*. [24] for embedding a set into fixed-size vector, invariant to the order of its elements. They do so by feeding the entire set through a recurrent unit multiple times. Each iteration uses an attention mechanism to focus on different parts of the set. Our formulation similarly incorporates information from neighbours into each node feature over multiple iterations, but we did not find any advantage in using an attention mechanism within the recurrent unit.

3. Graph representation of scenes and questions

The input data for each training or test instance is a question, and a parameterized description of contents of the scene. The question is processed with the Stanford dependency parser [7], which outputs the following.

- A set of N^{Q} words that constitute the nodes of the question graph. Each word is represented by its index in the input vocabulary, a token $x_i^{\mathsf{Q}} \in \mathbb{Z}$ ($i \in 1..N^{\mathsf{Q}}$).
- A set of pairwise relations between words, which constitute the edges of our graph. An edge between words i and j is represented by $e_{ij}^{\mathsf{Q}} \in \mathbb{Z}$, an index among the possible types of dependencies.

The dataset provides the following information about the image

- A set of N^{S} objects that constitute the nodes of the scene graph. Each node is represented by a vector $x_i^{\mathsf{S}} \in \mathbb{R}^C$ of visual features ($i \in 1..N^{\mathsf{S}}$). Please refer to the supplementary material for implementation details.
- A set of pairwise relations between all objects. They form the edges of a fully-connected graph of the scene. The edge between objects i and j is represented by a vector $e_{ij}^{\mathsf{S}} \in \mathbb{R}^D$ that encodes relative spatial relationships (see supp. mat.).

Our experiments are carried out on datasets of clip art scenes, in which descriptions of the scenes are provided in the form of lists of objects with their visual features. The method is equally applicable to real images, with the object list replaced by candidate object detections. Our experiments on clip art allows the effect of the proposed method to be isolated from the performance of the object detector. Please refer to the supplementary material for implementation details.

The features of all nodes and edges are projected to a vector space $\mathbb{R}^H$ of common dimension (typically H=300). The question nodes and edges use vector embeddings implemented as look-up tables, and the scene nodes and edges use affine projections:

$$x_i'^{\mathsf{Q}} = W_1 \big[x_i^{\mathsf{Q}} \big] \qquad e_{ij}'^{\mathsf{Q}} = W_2 \big[e_{ij}^{\mathsf{Q}} \big] \tag{1}$$

$$x_i'^{\mathsf{S}} = W_3 x_i^{\mathsf{S}} + b_3 \qquad e_{ij}'^{\mathsf{S}} = W_4 e_{ij}^{\mathsf{S}} + b_4 \tag{2}$$

with W_1 the word embedding (usually pretrained, see supplementary material), W_2 the embedding of dependencies, $W_3 \in \mathbb{R}^{h \times c}$ and $W_4 \in \mathbb{R}^{h \times d}$ weight matrices, and $b_3 \in \mathbb{R}^c$ and $b_4 \in \mathbb{R}^d$ biases.

4. Processing graphs with neural networks

We now describe a deep neural network suitable for processing the question and scene graphs to infer an answer. See Fig. 2 for an overview.

The two graphs representing the question and the scene are processed independently in a recurrent architecture. We drop the exponents S and Q for this paragraph as the same procedure applies to both graphs. Each node x_i is associated with a gated recurrent unit (GRU [6]) and processed over a fixed number T of iterations (typically T=4):

$$h_i^0 = 0 \tag{3}$$

$$n_i = \mathrm{pool}_j\big(e_{ij}' \circ x_j' \big) \tag{4}$$

$$h_i^t = \mathrm{GRU}\big(h_i^{t-1},\ [x_i' \,;\, n_i] \big) \qquad t \in [1, T]. \tag{5}$$

Square brackets with a semicolon represent a concatenation of vectors, and $\circ$ the Hadamard (element-wise) product. The final state of the GRU is used as the new representation of the nodes: $x_i'' = h_i^T$. The pool operation transforms features from a variable number of neighbours (*i.e.* connected nodes) to a fixed-size representation. Any commutative operation can be used (*e.g.* sum, maximum). In our implementation, we found the best performance with the average function, taking care of averaging over the variable number of connected neighbours. An intuitive interpretation of the recurrent processing is to progressively integrate context information from connected neighbours into each node's own representation. A node corresponding to the word 'ball', for instance, might thus incorporate the fact that the associated adjective is 'red'. Our formulation is similar but slightly different from the gated graph networks [15], as the propagation of information in our model is limited to the first order. Note that our graphs are typically densely connected.

We now introduce a form of attention into the model, which constitutes an essential part of the model. The motivation is two-fold: (1) to identify parts of the input data most relevant to produce the answer and (2) to align specific words in the question with particular elements of the scene. Practically, we estimate the relevance of each possible pairwise combination of words and objects. More precisely, we compute scalar "matching weights" between node sets $\{x_i'^{\mathsf{Q}}\}$ and $\{x_i'^{\mathsf{S}}\}$. These weights are comparable to the "attention weights" in other models (*e.g.* [16]). Therefore, $\forall\ i \in 1..N^{\mathsf{Q}}, j \in 1..N^{\mathsf{S}}$:

$$a_{ij} = \sigma\left(W_5 \Big(\frac{x_i'^{\mathsf{Q}}}{\|x_i'^{\mathsf{Q}}\|} \circ \frac{x_j'^{\mathsf{S}}}{\|x_j'^{\mathsf{S}}\|} \Big) + b_5 \right) \tag{6}$$

where $W_5 \in \mathbb{R}^{1 \times h}$ and $b_5 \in \mathbb{R}$ are learned weights and biases, and σ the logistic function that introduces a non-linearity and bounds the weights to $(0, 1)$. The formulation is similar to a cosine similarity with learned weights on the feature dimensions. Note that the weights are computed using the initial embedding of the node features (pre-GRU). We apply the scalar weights a_{ij} to the corresponding pairwise combinations of question and scene features, thereby focusing and giving more importance to the matched pairs (Eq. 7). We sum the weighted features over the scene elements (Eq. 8) then over the question elements (Eq. 9), interleaving the sums with affine projections and non-linearities to obtain a final prediction:

$$
\begin{aligned}
y_{ij} &= a_{ij} \, . \, [x_i''^{\mathsf{Q}} \, ; \, x_j''^{\mathsf{S}}] && (7)\\
y_i' &= f\big(W_6 \textstyle\sum_j^{N^{\mathsf{S}}} y_{ij} + b_6\big) && (8)\\
y'' &= f'\big(W_7 \textstyle\sum_i^{N^{\mathsf{Q}}} y_i' + b_7\big) && (9)
\end{aligned}
$$

with W_6, W_7, b_6, b_7 learned weights and biases, f a ReLU, and f' a softmax or a logistic function (see experiments, Section 5.1). The summations over the scene elements and question elements is a form of pooling that brings the variable number of features (due to the variable number of words and objects in the input) to a fixed-size output. The final output vector $y'' \in \mathbb{R}^T$ contains scores for the possible answers, and has a number of dimensions equal to 2 for the binary questions of the "balanced" dataset, or to the number of all candidate answers in the "abstract scenes" dataset. The candidate answers are those appearing at least 5 times in the training set (see supplementary material for details).

5. Evaluation

Datasets Our evaluation uses two datasets: the original "abstract scenes" from Antol *et al.* [4] and its "balanced" extension from [30]. They both contain scenes created by humans in a drag-and-drop interface for arranging clip art objects and figures. The original dataset contains $20k/10k/20k$ scenes (for training/validation/test respectively) and $60k/30k/60k$ questions, each with 10 human-provided ground-truth answers. Questions are categorized based on the type of the correct answer into *yes/no*, *number*, and *other*, but the same method is used for all categories, the type of the test questions being unknown. The "balanced" version of the dataset contains only the subset of questions which have binary (yes/no) answers and, in addition, complementary scenes created to elicit the opposite answer to each question. This is significant because guessing the modal answer from the training set will the succeed only half of the time (slightly more than 50% in practice because of disagreement between annotators) and give 0% accuracy over complementary pairs. This contrasts with other VQA datasets where blind guessing can be very effective. The pairs of complementary scenes also typically differ by only one or two objects being displaced, removed, or slightly modified (see examples in Fig. 5, bottom rows). This makes the questions very challenging by requiring to take into account subtle details of the scenes.

Metrics The main metric is the average "VQA score" [4], which is a soft accuracy that takes into account variability of ground truth answers from multiple human annotators. Let us refer to a test question by an index $q = 1..M$, and to each possible answer in the output vocabulary by an index a. The **ground truth score** $s(q, a) = 1.0$ if the answer a was provided by $m{\geq}3$ annotators. Otherwise, $s(q, a) = m/3$[1]. Our method outputs a **predicted score** $\hat{s}(q, a)$ for each question and answer (y'' in Eq. 9) and the overall accuracy is the average ground truth score of the highest prediction per question, *i.e.* $\frac{1}{M} \sum_q^M s(q, \arg\max_a \hat{s}(q, a))$.

It has been argued that the "balanced" dataset can better evaluate a method's level of visual understanding than other datasets, because it is less susceptible to the use of language priors and dataset regularities (*i.e.* guessing from the question[30]). Our initial experiments confirmed that the performances of various algorithms on the balanced dataset were indeed better separated, and we used it for our ablative analysis. We also focus on the hardest evaluation setting [30], which measures the accuracy over *pairs* of complementary scenes. This is the only metric in which blind models (guessing from the question) obtain null accuracy. This setting also does not consider pairs of test scenes deemed ambiguous because of disagreement between annotators. Each test scene is still evaluated independently however, so the model is unable to increase performance by forcing opposite answers to pairs of questions. The metric is then a standard "hard" accuracy, *i.e.* all ground truth scores $s(i, j) \in \{0, 1\}$. Please refer to the supplementary material for additional details.

5.1. Evaluation on the "balanced" dataset

We compare our method against the three models proposed in [30]. They all use an ensemble of models exploiting either an LSTM for processing the question, or an elaborate set of hand-designed rules to identify two objects as the focus of the question. The visual features in the three models are respectively empty (blind model), global (scene-wide), or focused on the two objects identified from the question. These models are specifically designed for binary questions, whereas ours is generally applicable. Nevertheless, we obtain significantly better accuracy than all three (Table 1). Differences in performance are mostly visible in the "pairs" setting, which we believe is more reliable as it discards ambiguous test questions on which human annotators disagreed.

During training, we take care to keep pairs of complementary scenes together when forming mini-batches. This has a significant positive effect on the stability of the optimization. Interestingly, we did not notice any tendency toward overfitting when training on balanced scenes. We hypothesize that the pairs of complementary scenes have a strong regularizing effect that force the learned model to focus on relevant details of the scenes. In Fig. 5 (and in the supplementary material), we visualize the matching weights between question words and scene objects (Eq. 6). As expected, these tend to be larger between semantically related

[1] Ground truth scores are also averaged in a 10–*choose*–9 manner [4].

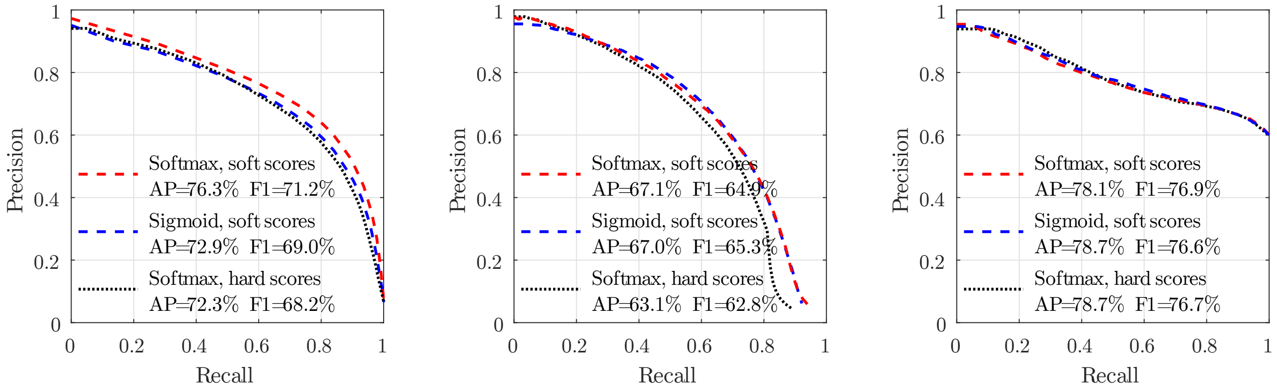

Figure 3. Precision/recall on the “abstract scenes” (**left**: multiple choice, **middle**: open-ended) and “balanced” datasets (**right**). The scores assigned by the model to predicted answers is a reliable measure of its certainty: a strict threshold (low recall) filters out incorrect answers and produces a very high precision. On the “abstract scenes” dataset (left and middle), a slight advantage is brought by training for soft target scores that capture ambiguities in the human-provided ground truth.

elements (*e.g.* daytime↔sun, dog↔puppy, boy↔human) although some are more difficult to interpret.

Our best performance of about 39% is still low in absolute terms, which is understandable from the wide range of concepts involved in the questions (see examples in Fig. 5 and in the supplementary material). It seems unlikely that these concepts could be learned from training question/answers alone, and we suggest that any further significant improvement in performance will require external sources of information at training and/or test time.

Ablative evaluation We evaluated variants of our model to measure the impact of various design choices (see numbered rows in Table 1). On the **question side**, we evaluate (row 1) our graph approach without syntactic parsing, building question graphs with only two types of edges, *previous/next* and linking consecutive nodes. This shows the advantage of using the graph method together with syntactic parsing. Optimizing the word embeddings from scratch (row 2) rather than from pretrained Glove vectors [20] produces a significant drop in performance. On the **scene side**, we removed the edge features (row 3) by setting $e^{\mathsf{S}}_{ij} = 1$. It confirms that the model makes use of the spatial relations between objects encoded by the edges of the graph. In rows 4–6, we disabled the recurrent graph processing ($x''_i = x'_i$) for the either the question, the scene, or both. We finally tested the model with uniform matching weights ($a_{ij} = 1$, row 10). As expected, it performed poorly. Our weights act similarly to the attention mechanisms in other models (*e.g.* [31, 27, 5, 12, 28]) and our observations confirm that such mechanisms are crucial for good performance.

Precision/recall We are interested in assessing the confidence of our model in its predicted answers. Most existing VQA methods treat the answering as a hard classification over candidate answers, and almost all reported results consist of a single accuracy metric. To provide more insight, we produce precision/recall curves for predicted answers. A precision/recall point (p, r) is obtained by setting a thresh-

Method	Avg. score over scenes	Avg. accuracy over pairs
Zhang *et al.* [30] blind	63.33	0.00
with global image features	71.03	23.13
with attention-based image features	74.65	34.73
Graph VQA (full model)	**74.94**	**39.1**
(1) Question: no parsing (graph with previous/next edges)		37.9
(2) Question: word embedding not pretrained		33.8
(3) Scene: no edge features (e'^{S}_{ij}=1)		36.8
(4) Graph processing: disabled for question (x''^{Q}_i=x'^{S}_i)		37.1
(5) Graph processing: disabled for scene (x''^{S}_i=x'^{Q}_i)		37.0
(6) Graph processing: disabled for question/scene		35.7
(7) Graph processing: only 1 iteration for question (T^{Q}=1)		39.0
(8) Graph processing: only 1 iteration for scene (T^{S}=1)		37.9
(9) Graph processing: only 1 iteration for question/scene		39.1
(10) Uniform matching weights (a_{ij}=1)		24.4

Table 1. Results on the test set of the “balanced” dataset [30] (in percents , using balanced versions of both training and test sets). Numbered rows report accuracy over pairs of complementary scenes for ablated versions of our method.

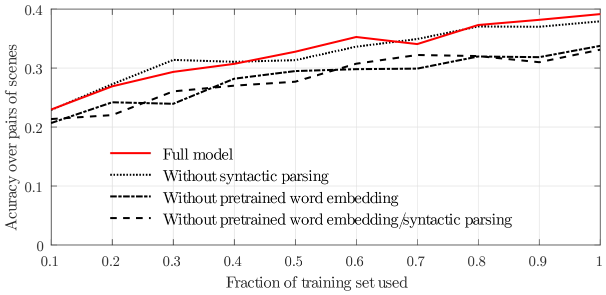

Figure 4. Impact of the amount of training data on performance (accuracy over pairs on the "balanced" dataset). Language preprocessing always improve generalization: pre-parsing and pretrained word embeddings both have a positive impact individually, and their effects are complementary to each other.

old t on predicted scores such that

$$p = \frac{\sum_{i,j} \mathbb{1}\big(\hat{s}(i,j) > t\big)\; s(i,j)}{\sum_{i,j} \mathbb{1}\big(\hat{s}(i,j) > t\big)} \tag{10}$$

$$r = \frac{\sum_{i,j} \mathbb{1}\big(\hat{s}(i,j) > t\big)\; s(i,j)}{\sum_{i,j} s(i,j)} \tag{11}$$

where $\mathbb{1}(\cdot)$ is the $0/1$ indicator function. We plot precision/recall curves in Fig. 3 for both datasets[2]. The predicted score proves to be a reliable indicator of the model confidence, as a low threshold can achieve near-perfect accuracy (Fig. 3, left and middle) by filtering out harder and/or ambiguous test cases.

We compare models trained with either a softmax or a sigmoid as the final non-linearity (Eq. 9). The common practice is to train the softmax for a hard classification objective, using a cross-entropy loss and the answer of highest ground truth score as the target. In an attempt to make better use of the multiple human-provided answers, we propose to use the *soft* ground truth scores as the target with a logarithmic loss. This shows an advantage on the "abstract scenes" dataset (Fig. 3, left and middle). In that dataset, the soft target scores reflect frequent ambiguities in the questions and the scenes, and when synonyms constitute multiple acceptable answers. In those cases, we can avoid the potential confusion induced by a hard classification for one specific answer. The "balanced" dataset, by nature, contains almost no such ambiguities, and there is no significant difference between the different training objectives (Fig. 3, right).

Effect of training set size Our motivation for introducing language parsing and pretrained word embeddings is to better generalize the concepts learned from the limited training examples. Words representing semantically close concepts ideally get assigned close word embeddings. Similarly, paraphrases of similar questions should produce parse graphs with more similarities than a simple concatenation of words would reveal (as in the input to traditional LSTMs).

We trained our model with limited subsets of the training data (see Fig. 4). Unsurprisingly, the performance grows steadily with the amount of training data, which suggests that larger datasets would improve performance. In our opinion however, it seems unlikely that sufficient data, covering all possible concepts, could be collected in the form of question/answer examples. More data can however be brought in with other sources of information and supervision. Our use of parsing and word embeddings is a small step in that direction. Both techniques clearly improve generalization (Fig. 4). The effect may be particularly visible in our case because of the relatively small number of training examples (about 20k questions in the "balanced" dataset). It is unclear whether huge VQA datasets could ultimately negate this advantage. Future experiments on larger datasets (*e.g.* [14]) may answer this question.

5.2. Evaluation on the "abstract scenes" dataset

We report our results on the original "abstract scenes" dataset in Table 2. The evaluation is performed on an automated server that does not allow for an extensive ablative analysis. Anecdotally, performance on the validation set corroborates all findings presented above, in particular the strong benefit of pre-parsing, pretrained word embeddings, and graph processing with a GRU. At the time of our submission, our method occupies the top place on the leader board in both the open-ended and multiple choice settings. The advantage over existing method is most pronounced on the binary and the counting questions. Refer to Fig. 5 and to the supplementary for visualizations of the results.

6. Conclusions

We presented a deep neural network for visual question answering that processes graph-structured representations of scenes and questions. This enables leveraging existing natural language processing tools, in particular pretrained word embeddings and syntactic parsing. The latter showed significant advantage over a traditional sequential processing of the questions, *e.g.* with LSTMs. In our opinion, VQA systems are unlikely to learn everything from question/answer examples alone. We believe that any significant improvement in performance will require additional sources of information and supervision. Our explicit processing of the language part is a small step in that direction. It has clearly shown to improve generalization without resting entirely on VQA-specific annotations. We have so far applied our method to datasets of clip art scenes. Its direct extension to real images will be addressed in future work, by replacing nodes in the input scene graph with proposals from pretrained object detectors.

[2] The "abstract scenes" test set is not available publicly, and precision/recall can only be provided on its validation set.

	Multiple choice				Open-ended			
Method	Overall	Yes/no	Other	Number	Overall	Yes/no	Other	Number
LSTM blind [4]	61.41	76.90	49.19	49.65	57.19	76.88	38.79	49.55
LSTM with global image features [4]	69.21	77.46	66.65	52.90	65.02	77.45	56.41	52.54
Zhang *et al*. [30] (yes/no only)	35.25	79.14	—	—	35.25	79.14	—	—
Multimodal residual learning [13]	67.99	79.08	61.99	52.57	62.56	79.10	48.90	51.60
U. Tokyo MIL (ensemble) [22, 1]	71.18	79.59	67.93	56.19	69.73	80.70	**62.08**	58.82
Graph VQA (full model)	**74.37**	**79.74**	**68.31**	**74.97**	**70.42**	**81.26**	56.28	**76.47**

Table 2. Results on the test set of the "abstract scenes" dataset (average scores in percents).

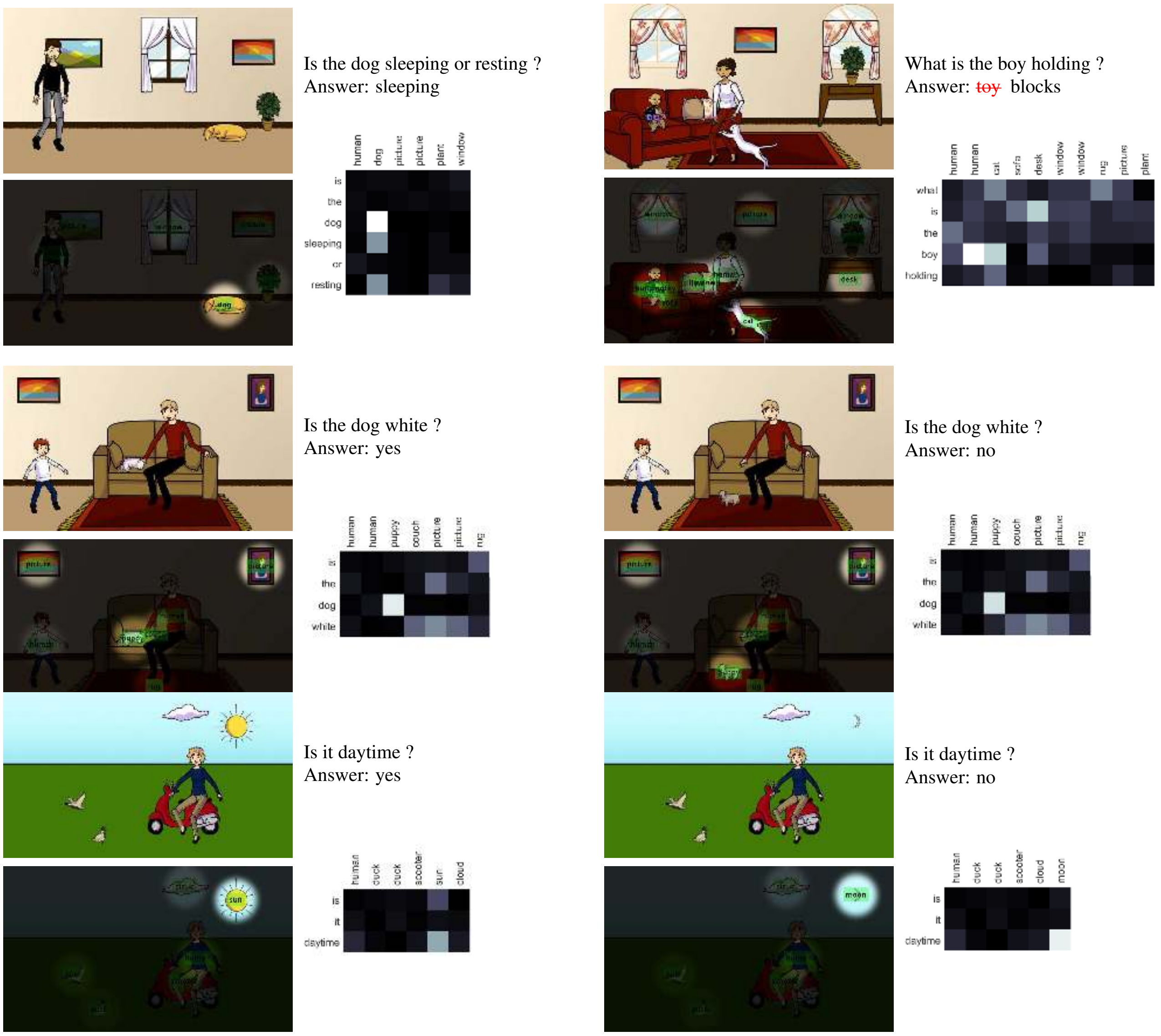

Figure 5. Qualitative results on the "abstract scenes" dataset (top row) and on "balanced" pairs (middle and bottom row). We show the input scene, the question, the predicted answer, and the correct answer when the prediction is erroneous. We also visualize the matrices of matching weights (Eq. 6, brighter correspond to higher values) between question words (vertically) and scene objects (horizontally). The matching weights are also visualized over objects in the scene, after summation over words, giving an indication of their estimated relevance. The ground truth object labels are for reference only, and not used for training or inference.

References

[1] VQA Challenge leaderboard. http://visualqa.org/challenge.html. 8

[2] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Learning to compose neural networks for question answering. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2016. 3

[3] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Neural Module Networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2016. 3

[4] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. VQA: Visual Question Answering. In *Proc. IEEE Int. Conf. Comp. Vis.*, 2015. 1, 2, 5, 8

[5] K. Chen, J. Wang, L.-C. Chen, H. Gao, W. Xu, and R. Nevatia. ABC-CNN: An Attention Based Convolutional Neural Network for Visual Question Answering. *arXiv preprint arXiv:1511.05960*, 2015. 3, 6

[6] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proc. Conf. Empirical Methods in Natural Language Processing*, 2014. 3, 4

[7] M.-C. de Marneffe and C. D. Manning. The stanford typed dependencies representation. In *COLING Workshop on Cross-framework and Cross-domain Parser Evaluation*, 2008. 2, 4, 10

[8] D. K. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams. Convolutional networks on graphs for learning molecular fingerprints. *arXiv preprint arXiv:1509.09292*, 2015. 3

[9] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847*, 2016. 3

[10] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proc. Int. Conf. Artificial Intell. & Stat.*, pages 249–256, 2010. 10

[11] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2016. 3

[12] A. Jiang, F. Wang, F. Porikli, and Y. Li. Compositional Memory for Visual Question Answering. *arXiv preprint arXiv:1511.05676*, 2015. 3, 6

[13] J.-H. Kim, S.-W. Lee, D.-H. Kwak, M.-O. Heo, J. Kim, J.-W. Ha, and B.-T. Zhang. Multimodal residual learning for visual qa. *arXiv preprint arXiv:1606.01455*, 2016. 8

[14] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. Bernstein, and L. Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *arXiv preprint arXiv:1602.07332*, 2016. 1, 7

[15] Y. Li, D. Tarlow, M. Brockschmidt, and R. S. Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015. 3, 4

[16] J. Lu, J. Yang, D. Batra, and D. Parikh. Hierarchical question-image co-attention for visual question answering. *arXiv preprint arXiv:1606.00061*, 2016. 3, 4

[17] M. Malinowski and M. Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Proc. Advances in Neural Inf. Process. Syst.*, pages 1682–1690, 2014. 1

[18] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. http://nlp.stanford.edu/projects/glove/. 10

[19] J. Pennington, R. Socher, and C. Manning. Stanford dependency parser website. http://nlp.stanford.edu/software/stanford-dependencies.shtml. 10

[20] J. Pennington, R. Socher, and C. Manning. Glove: Global Vectors for Word Representation. In *Conference on Empirical Methods in Natural Language Processing*, 2014. 2, 6, 10

[21] M. Ren, R. Kiros, and R. Zemel. Image Question Answering: A Visual Semantic Embedding Model and a New Dataset. In *Proc. Advances in Neural Inf. Process. Syst.*, 2015. 1

[22] K. Saito, A. Shin, Y. Ushiku, and T. Harada. Dualnet: Domain-invariant network for visual question answering. *arXiv preprint arXiv:1606.06108*, 2016. 8

[23] K. J. Shih, S. Singh, and D. Hoiem. Where to look: Focus regions for visual question answering. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2016. 3

[24] O. Vinyals, S. Bengio, and M. Kudlur. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391*, 2015. 3

[25] Q. Wu, D. Teney, P. Wang, C. Shen, A. Dick, and A. van den Hengel. Visual Question Answering: A Survey of Methods and Datasets. *arXiv preprint arXiv:1607.05910*, 2016. 1, 3

[26] C. Xiong, S. Merity, and R. Socher. Dynamic memory networks for visual and textual question answering. In *Proc. Int. Conf. Mach. Learn.*, 2016. 3

[27] H. Xu and K. Saenko. Ask, Attend and Answer: Exploring Question-Guided Spatial Attention for Visual Question Answering. *arXiv preprint arXiv:1511.05234*, 2015. 3, 6

[28] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola. Stacked Attention Networks for Image Question Answering. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2016. 3, 6

[29] M. D. Zeiler. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012. 10

[30] P. Zhang, Y. Goyal, D. Summers-Stay, D. Batra, and D. Parikh. Yin and yang: Balancing and answering binary visual questions. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2016. 1, 3, 5, 6, 8

[31] Y. Zhu, O. Groth, M. Bernstein, and L. Fei-Fei. Visual7W: Grounded Question Answering in Images. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2016. 1, 3, 6