Stacked Attention Networks for Image Question Answering

Zichao Yang[1], Xiaodong He[2], Jianfeng Gao[2], Li Deng[2], Alex Smola[1]
[1]Carnegie Mellon University, [2]Microsoft Research, Redmond, WA 98052, USA
`zichaoy@cs.cmu.edu, {xiaohe, jfgao, deng}@microsoft.com, alex@smola.org`

Abstract

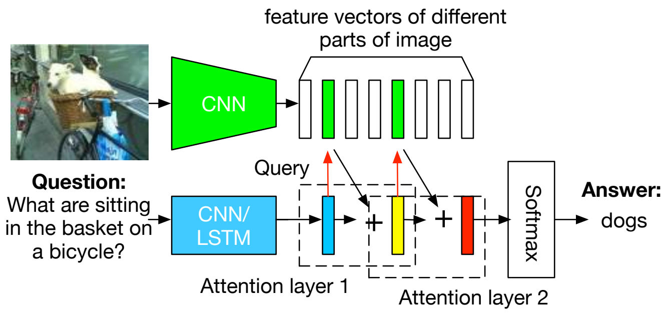
This paper presents stacked attention networks (SANs) that learn to answer natural language questions from images. SANs use semantic representation of a question as query to search for the regions in an image that are related to the answer. We argue that image question answering (QA) often requires multiple steps of reasoning. Thus, we develop a multiple-layer SAN in which we query an image multiple times to infer the answer progressively. Experiments conducted on four image QA data sets demonstrate that the proposed SANs significantly outperform previous state-of-the-art approaches. The visualization of the attention layers illustrates the progress that the SAN locates the relevant visual clues that lead to the answer of the question layer-by-layer.

1. Introduction

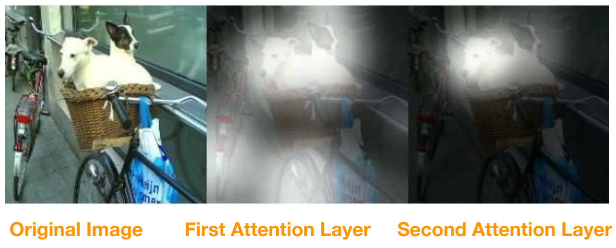
With the recent advancement in computer vision and in natural language processing (NLP), image question answering (QA) becomes one of the most active research areas [7, 21, 18, 1, 19]. Unlike pure language based QA systems that have been studied extensively in the NLP community [28, 14, 4, 31, 3, 32], image QA systems are designed to automatically answer natural language questions according to the content of a reference image.

Most of the recently proposed image QA models are based on neural networks [7, 21, 18, 1, 19]. A commonly used approach was to extract a global image feature vector using a convolution neural network (CNN) [15] and encode the corresponding question as a feature vector using a long short-term memory network (LSTM) [9] and then combine them to infer the answer. Though impressive results have been reported, these models often fail to give precise answers when such answers are related to a set of *fine-grained* regions in an image.

By examining the image QA data sets, we find that it is often that case that answering a question from an image requires multi-step reasoning. Take the question and image in Fig. 1 as an example. There are several objects in the image: `bicycles, window, street, baskets` and


feature vectors of different parts of image
CNN
Query
Question:
What are sitting in the basket on a bicycle?
CNN/ LSTM
Attention layer 1
Attention layer 2
Softmax
Answer:
dogs

(a) Stacked Attention Network for Image QA



(b) Visualization of the learned multiple attention layers. The stacked attention network first focuses on all referred concepts, e.g., `bicycle, basket` and objects in the basket (`dogs`) in the first attention layer and then further narrows down the focus in the second layer and finds out the answer `dog`.

Figure 1: Model architecture and visualization

`dogs`. To answer the question `what are sitting in the basket on a bicycle`, we need to first locate those objects (e.g. `basket, bicycle`) and concepts (e.g., `sitting in`) referred in the question, then gradually rule out irrelevant objects, and finally pinpoint to the region that are most indicative to infer the answer (i.e., `dogs` in the example).

In this paper, we propose stacked attention networks (SANs) that allow multi-step reasoning for image QA. SANs can be viewed as an extension of the attention mechanism that has been successfully applied in image captioning [30] and machine translation [2]. The overall architecture of SAN is illustrated in Fig. 1a. The SAN consists of three major components: (1) the image model, which uses a CNN to extract high level image representations, e.g. one vector for each region of the image; (2) the question model, which uses a CNN or a LSTM to extract a semantic vector of the question and (3) the stacked attention model, which locates, via multi-step reasoning, the image regions that are relevant to the question for answer prediction. As illustrated in Fig. 1a, the SAN first uses the question vector to query the image vectors in the first visual attention layer, then combine the question vector and the retrieved image vectors to form a refined query vector to query the image vectors again in the second attention layer. The higher-level attention layer gives a sharper attention distribution focusing on the regions that are more relevant to the answer. Finally, we combine the image features from the highest attention layer with the last query vector to predict the answer.

The main contributions of our work are three-fold. First, we propose a stacked attention network for image QA tasks. Second, we perform comprehensive evaluations on four image QA benchmarks, demonstrating that the proposed multiple-layer SAN outperforms previous state-of-the-art approaches by a substantial margin. Third, we perform a detailed analysis where we visualize the outputs of different attention layers of the SAN and demonstrate the process that the SAN takes multiple steps to progressively focus the attention on the relevant visual clues that lead to the answer.

2. Related Work

Image QA is closely related to image captioning [5, 30, 6, 27, 12, 10, 20]. In [27], the system first extracted a high level image feature vector from GoogleNet and then fed it into a LSTM to generate captions. The method proposed in [30] went one step further to use an attention mechanism in the caption generation process. Different from [30, 27], the approach proposed in [6] first used a CNN to detect words given the images, then used a maximum entropy language model to generate a list of caption candidates, and finally used a deep multimodal similarity model (DMSM) to re-rank the candidates. Instead of using a RNN or a LSTM, the DMSM uses a CNN to model the semantics of captions.

Unlike image captioning, in image QA, the question is given and the task is to learn the relevant visual and text representation to infer the answer. In order to facilitate the research of image QA, several data sets have been constructed in [19, 21, 7, 1] either through automatic generation based on image caption data or by human labeling of questions and answers given images. Among them, the image QA data set in [21] is generated based on the COCO caption data set. Given a sentence that describes an image, the authors first used a parser to parse the sentence, then replaced the key word in the sentence using question words and the key word became the answer. [7] created an image QA data set through human labeling. The initial version was in Chinese and then was translated to English. [1] also created an image QA data set through human labeling. They collected questions and answers not only for real images, but also for abstract scenes.

Several image QA models were proposed in the literature. [18] used semantic parsers and image segmentation methods to predict answers based on images and questions. [19, 7] both used encoder-decoder framework to generate answers given images and questions. They first used a LSTM to encoder the images and questions and then used another LSTM to decode the answers. They both fed the image feature to every LSTM cell. [21] proposed several neural network based models, including the encoder-decoder based models that use single direction LSTMs and bi-direction LSTMs, respectively. However, the authors found the concatenation of image features and bag of words features worked the best. [1] first encoded questions with LSTMs and then combined question vectors with image vectors by element wise multiplication. [17] used a CNN for question modeling and used convolution operations to combine question vectors and image feature vectors. We compare the SAN with these models in Sec. 4.

To the best of our knowledge, the attention mechanism, which has been proved very successful in image captioning, has not been explored for image QA. The SAN adapt the attention mechanism to image QA, and can be viewed as a significant extension to previous models [30] in that multiple attention layers are used to support multi-step reasoning for the image QA task.

3. Stacked Attention Networks (SANs)

The overall architecture of the SAN is shown in Fig. 1a. We describe the three major components of SAN in this section: the image model, the question model, and the stacked attention model.

3.1. Image Model

The image model uses a CNN [13, 23, 26] to get the representation of images. Specifically, the VGGNet [23] is used to extract the image feature map f_I from a raw image I:

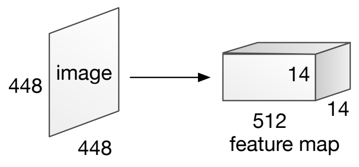


Figure 2: CNN based image model

$$f_I = \text{CNN}_{vgg}(I). \quad (1)$$

Unlike previous studies [21, 17, 7] that use features from the last inner product layer, we choose the features f_I from the last pooling layer, which retains spatial information of the original images. We first rescale the images to be 448×448

pixels, and then take the features from the last pooling layer, which therefore have a dimension of $512 \times 14 \times 14$, as shown in Fig. 2. 14×14 is the number of regions in the image and 512 is the dimension of the feature vector for each region. Accordingly, each feature vector in f_I corresponds to a 32×32 pixel region of the input images. We denote by $f_i, i \in [0, 195]$ the feature vector of each image region.

Then for modeling convenience, we use a single layer perceptron to transform each feature vector to a new vector that has the same dimension as the question vector (described in Sec. 3.2):

$$v_I = \tanh(W_I f_I + b_I), \tag{2}$$

where v_I is a matrix and its i-th column v_i is the visual feature vector for the region indexed by i.

3.2. Question Model

As [25, 22, 6] show that LSTMs and CNNs are powerful to capture the semantic meaning of texts, we explore both models for question representations in this study.

3.2.1 LSTM based question model

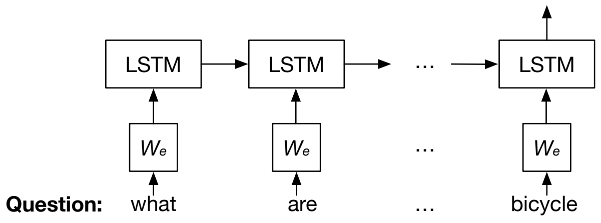


Figure 3: LSTM based question model

The essential structure of a LSTM unit is a memory cell c_t which reserves the state of a sequence. At each step, the LSTM unit takes one input vector (word vector in our case) x_t and updates the memory cell c_t, then output a hidden state h_t. The update process uses the gate mechanism. A forget gate f_t controls how much information from past state c_{t-1} is preserved. An input gate i_t controls how much the current input x_t updates the memory cell. An output gate o_t controls how much information of the memory is fed to the output as hidden state. The detailed update process is as follows:

$$\begin{aligned}
i_t =& \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i), && (3)\\
f_t =& \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f), && (4)\\
o_t =& \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o), && (5)\\
c_t =& f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c), && (6)\\
h_t =& o_t \tanh(c_t), && (7)
\end{aligned}$$

where i, f, o, c are input gate, forget gate, output gate and memory cell, respectively. The weight matrix and bias are parameters of the LSTM and are learned on training data.

Given the question $q = [q_1, ...q_T]$, where q_t is the one hot vector representation of word at position t, we first embed the words to a vector space through an embedding matrix $x_t = W_e q_t$. Then for every time step, we feed the embedding vector of words in the question to LSTM:

$$\begin{aligned}
x_t =& W_e q_t, t \in \{1, 2, ...T\}, && (8)\\
h_t =& \text{LSTM}(x_t), t \in \{1, 2, ...T\}. && (9)
\end{aligned}$$

As shown in Fig. 3, the question `what are sitting in the basket on a bicycle` is fed into the LSTM. Then the final hidden layer is taken as the representation vector for the question, i.e., $v_Q = h_T$.

3.2.2 CNN based question model

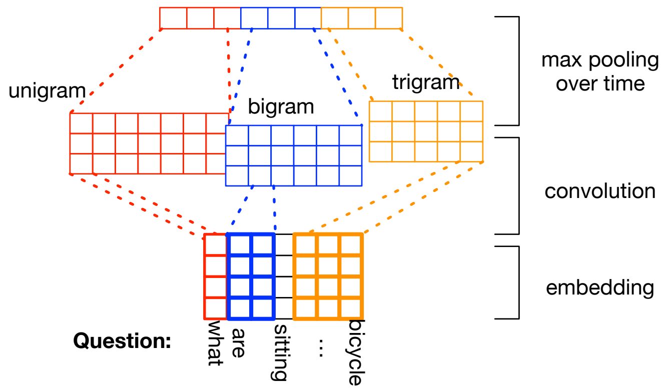


Figure 4: CNN based question model

In this study, we also explore to use a CNN similar to [11] for question representation. Similar to the LSTM-based question model, we first embed words to vectors $x_t = W_e q_t$ and get the question vector by concatenating the word vectors:

$$x_{1:T} = [x_1, x_2, ..., x_T]. \tag{10}$$

Then we apply convolution operation on the word embedding vectors. We use three convolution filters, which have the size of one (unigram), two (bigram) and three (trigram) respectively. The t-th convolution output using window size c is given by:

$$h_{c,t} = \tanh(W_c x_{t:t+c-1} + b_c). \tag{11}$$

The filter is applied only to window $t : t + c - 1$ of size c. W_c is the convolution weight and b_c is the bias. The feature map of the filter with convolution size c is given by:

$$h_c = [h_{c,1}, h_{c,2}, ..., h_{c,T-c+1}]. \tag{12}$$

Then we apply max-pooling over the feature maps of the convolution size c and denote it as

$$\tilde{h}_c = \max_t [h_{c,1}, h_{c,2}, ..., h_{c,T-c+1}]. \tag{13}$$

The max-pooling over these vectors is a coordinate-wise max operation. For convolution feature maps of different sizes $c = 1, 2, 3$, we concatenate them to form the feature representation vector of the whole question sentence:

$$h = [\tilde{h}_1, \tilde{h}_2, \tilde{h}_3], \tag{14}$$

hence $v_Q = h$ is the CNN based question vector.

The diagram of CNN model for question is shown in Fig. 4. The convolutional and pooling layers for unigrams, bigrams and trigrams are drawn in red, blue and orange, respectively.

3.3. Stacked Attention Networks

Given the image feature matrix v_I and the question feature vector v_Q, SAN predicts the answer via multi-step reasoning.

In many cases, an answer only related to a small region of an image. For example, in Fig. 1b, although there are multiple objects in the image: `bicycles, baskets, window, street` and `dogs` and the answer to the question only relates to `dogs`. Therefore, using the one global image feature vector to predict the answer could lead to sub-optimal results due to the noises introduced from regions that are irrelevant to the potential answer. Instead, reasoning via multiple attention layers progressively, the SAN are able to gradually filter out noises and pinpoint the regions that are highly relevant to the answer.

Given the image feature matrix v_I and the question vector v_Q, we first feed them through a single layer neural network and then a softmax function to generate the attention distribution over the regions of the image:

$$h_A = \tanh(W_{I,A} v_I \oplus (W_{Q,A} v_Q + b_A)), \tag{15}$$
$$p_I = \text{softmax}(W_P h_A + b_P), \tag{16}$$

where $v_I \in \mathbf{R}^{d \times m}$, d is the image representation dimension and m is the number of image regions, $v_Q \in \mathbf{R}^d$ is a d dimensional vector. Suppose $W_{I,A}, W_{Q,A} \in \mathbf{R}^{k \times d}$ and $W_P \in \mathbf{R}^{1 \times k}$, then $p_I \in \mathbf{R}^m$ is an m dimensional vector, which corresponds to the attention probability of each image region given v_Q. Note that we denote by $\oplus$ the addition of a matrix and a vector. Since $W_{I,A} v_I \in \mathbf{R}^{k \times m}$ and both $W_{Q,A} v_Q, b_A \in \mathbf{R}^k$ are vectors, the addition between a matrix and a vector is performed by adding each column of the matrix by the vector.

Based on the attention distribution, we calculate the weighted sum of the image vectors, each from a region, $\tilde{v}_i$ as in Eq. 17. We then combine $\tilde{v}_i$ with the question vector v_Q to form a refined query vector u as in Eq. 18. u is regarded as a refined query since it encodes both question information and the visual information that is relevant to the potential answer:

$$\tilde{v}_I = \sum_i p_i v_i, \tag{17}$$
$$u = \tilde{v}_I + v_Q. \tag{18}$$

Compared to models that simply combine the question vector and the global image vector, attention models construct a more informative u since higher weights are put on the visual regions that are more relevant to the question. However, for complicated questions, a single attention layer is not sufficient to locate the correct region for answer prediction. For example, the question in Fig. 1 `what are sitting in the basket on a bicycle` refers to some subtle relationships among multiple objects in an image. Therefore, we iterate the above query-attention process using multiple attention layers, each extracting more fine-grained visual attention information for answer prediction. Formally, the SANs take the following formula: for the k-th attention layer, we compute:

$$h_A^k = \tanh(W_{I,A}^k v_I \oplus (W_{Q,A}^k u^{k-1} + b_A^k)), \tag{19}$$
$$p_I^k = \text{softmax}(W_P^k h_A^k + b_P^k). \tag{20}$$

where u^0 is initialized to be v_Q. Then the aggregated image feature vector is added to the previous query vector to form a new query vector:

$$\tilde{v}_I^k = \sum_i p_i^k v_i, \tag{21}$$
$$u^k = \tilde{v}_I^k + u^{k-1}. \tag{22}$$

That is, in every layer, we use the combined question and image vector u^{k-1} as the query for the image. After the image region is picked, we update the new query vector as $u^k = \tilde{v}_I^k + u^{k-1}$. We repeat this K times and then use the final u^K to infer the answer:

$$p_{\text{ans}} = \text{softmax}(W_u u^K + b_u). \tag{23}$$

Fig. 1b illustrates the reasoning process by an example. In the first attention layer, the model identifies roughly the area that are relevant to `basket, bicycle`, and `sitting in`. In the second attention layer, the model focuses more sharply on the region that corresponds to the answer `dogs`. More examples can be found in Sec. 4.

4. Experiments

4.1. Data sets

We evaluate the SAN on four image QA data sets.

DAQUAR-ALL is proposed in [18]. There are $6,795$ training questions and $5,673$ test questions. These questions are generated on 795 and 654 images respectively. The images are mainly indoor scenes. The questions are categorized into three types including *Object*, *Color* and *Number*. Most of the answers are single words. Following the setting in [21, 17, 19], we exclude data samples that have multiple words answers. The remaining data set covers 90% of the original data set.

DAQUAR-REDUCED is a reduced version of DAQUAR-ALL. There are $3,876$ training samples and 297 test samples. This data set is constrained to 37 object categories and uses only 25 test images. The single word answers data set covers 98% of the original data set.

COCO-QA is proposed in [21]. Based on the Microsoft COCO data set, the authors first parse the caption of the image with an off-the-shelf parser, then replace the key components in the caption with question words for form questions. There are 78736 training samples and 38948 test samples in the data set. These questions are based on $8,000$ and $4,000$ images respectively. There are four types of questions including *Object*, *Number*, *Color*, and *Location*. Each type takes $70\%, 7\%, 17\%$, and 6% of the whole data set, respectively. All answers in this data set are single word.

VQA is created through human labeling [1]. The data set uses images in the COCO image caption data set [16]. Unlike the other data sets, for each image, there are three questions and for each question, there are ten answers labeled by human annotators. There are $248,349$ training questions and $121,512$ validation questions in the data set. Following [1], we use the top 1000 most frequent answer as possible outputs and this set of answers covers 82.67% of all answers. We first studied the performance of the proposed model on the validation set. Following [6], we split the validation data set into two halves, val1 and val2. We use training set and val1 to train and validate and val2 to test locally. The results on the val2 set are reported in Table. 6. We also evaluated the best model, SAN(2, CNN), on the standard test server as provided in [1] and report the results in Table. 5.

4.2. Baselines and evaluation methods

We compare our models with a set of baselines proposed recently [21, 1, 18, 19, 17] on image QA. Since the results of these baselines are reported on different data sets in different literature, we present the experimental results on different data sets in different tables.

For all four data sets, we formulate image QA as a classification problem since most of answers are single words. We evaluate the model using classification accuracy as reported in [1, 21, 19]. The reference models also report the Wu-Palmer similarity (WUPS) measure [29]. The WUPS measure calculates the similarity between two words based on their longest common subsequence in the taxonomy tree. We can set a threshold for WUPS, if the similarity is less than the threshold, then it is zeroed out. Following the reference models, we use WUPS0.9 and WUPS0.0 as evaluation metrics besides the classification accuracy. The evaluation on the VQA data set is different from other three data sets, since for each question there are ten answer labels that may or may not be the same. We follow [1] to use the following metric: $min(\text{\# human labels that match that answer}/3, 1)$, which basically gives full credit to the answer when three or more of the ten human labels match the answer and gives partial credit if there are less matches.

4.3. Model configuration and training

For the image model, we use the VGGNet to extract features. When training the SAN, the parameter set of the CNN of the VGGNet is fixed. We take the output from the last pooling layer as our image feature which has a dimension of $512 \times 14 \times 14$.

For DAQUAR and COCO-QA, we set the word embedding dimension and LSTM's dimension to be 500 in the question model. For the CNN based question model, we set the unigram, bigram and trigram convolution filter size to be 128, 256, 256 respectively. The combination of these filters makes the question vector size to be 640. For VQA dataset, since it is larger than other data sets, we double the model size of the LSTM and the CNN to accommodate the large data set and the large number of classes. In evaluation, we experiment with SAN with one and two attention layers. We find that using three or more attention layers does not further improve the performance.

In our experiments[1], all the models are trained using stochastic gradient descent with momentum 0.9. The batch size is fixed to be 100. The best learning rate is picked using grid search. Gradient clipping technique [8] and dropout [24] are used.

4.4. Results and analysis

The experimental results on DAQUAR-ALL, DAQUAR-REDUCED, COCO-QA and VQA are presented in Table. 1 to 6 respectively. Our model names explain their settings: SAN is short for the proposed stacked attention networks, the value 1 or 2 in the brackets refer to using one or two attention layers, respectively. The keyword LSTM or CNN refers to the question model that SANs use.

The experimental results in Table. 1 to 6 show that the two-layer SAN gives the best results across all data sets and the two kinds of question models in the SAN, LSTM and CNN, give similar performance. For example, on DAQUAR-ALL (Table. 1), both of the proposed two-layer SANs outperform the two best baselines, the IMG-CNN in [17] and the Ask-Your-Neuron in [19], by 5.9% and 7.6% absolute in accuracy, respectively. Similar range of improvements are observed in metrics of WUPS0.9 and

[1]Our code is publicly available at https://github.com/zcyang/imageqa-san

Methods	Accuracy	WUPS0.9	WUPS0.0
Multi-World: [18]			
Multi-World	7.9	11.9	38.8
Ask-Your-Neurons: [19]			
Language	19.1	25.2	65.1
Language + IMG	21.7	28.0	65.0
CNN: [17]			
IMG-CNN	23.4	29.6	63.0
Ours:			
SAN(1, LSTM)	28.9	34.7	68.5
SAN(1, CNN)	29.2	35.1	67.8
SAN(2, LSTM)	**29.3**	34.9	68.1
SAN(2, CNN)	**29.3**	**35.1**	**68.6**
Human :[18]			
Human	50.2	50.8	67.3

Table 1: DAQUAR-ALL results, in percentage

Methods	Accuracy	WUPS0.9	WUPS0.0
Multi-World: [18]			
Multi-World	12.7	18.2	51.5
Ask-Your-Neurons: [19]			
Language	31.7	38.4	80.1
Language + IMG	34.7	40.8	79.5
VSE: [21]			
GUESS	18.2	29.7	77.6
BOW	32.7	43.2	81.3
LSTM	32.7	43.5	81.6
IMG+BOW	34.2	45.0	81.5
VIS+LSTM	34.4	46.1	82.2
2-VIS+BLSTM	35.8	46.8	82.2
CNN: [17]			
IMG-CNN	39.7	44.9	83.1
Ours:			
SAN(1, LSTM)	45.2	49.6	84.0
SAN(1, CNN)	45.2	49.6	83.7
SAN(2, LSTM)	**46.2**	**51.2**	**85.1**
SAN(2, CNN)	45.5	50.2	83.6
Human :[18]			
Human	60.3	61.0	79.0

Table 2: DAQUAR-REDUCED results, in percentage

WUPS0.0. We also observe significant improvements on DAQUAR-REDUCED (Table. 2), i.e., our SAN(2, LSTM) outperforms the IMG-CNN [17], the 2-VIS+BLSTM [21], the Ask-Your-Neurons approach [19] and the Multi-World [18] by 6.5%, 10.4%, 11.5% and 33.5% absolute in accuracy, respectively. On the larger COCO-QA data set, the proposed two-layer SANs significantly outperform the best baselines from [17] (IMG-CNN) and [21] (IMG+BOW and 2-VIS+BLSTM) by 5.1% and 6.6% in accuracy (Table. 3). Table. 5 summarizes the performance of various models on

Methods	Accuracy	WUPS0.9	WUPS0.0
VSE: [21]			
GUESS	6.7	17.4	73.4
BOW	37.5	48.5	82.8
LSTM	36.8	47.6	82.3
IMG	43.0	58.6	85.9
IMG+BOW	55.9	66.8	89.0
VIS+LSTM	53.3	63.9	88.3
2-VIS+BLSTM	55.1	65.3	88.6
CNN: [17]			
IMG-CNN	55.0	65.4	88.6
CNN	32.7	44.3	80.9
Ours:			
SAN(1, LSTM)	59.6	69.6	90.1
SAN(1, CNN)	60.7	70.6	90.5
SAN(2, LSTM)	61.0	71.0	90.7
SAN(2, CNN)	**61.6**	**71.6**	**90.9**

Table 3: COCO-QA results, in percentage

Methods	Objects	Number	Color	Location
VSE: [21]				
GUESS	2.1	35.8	13.9	8.9
BOW	37.3	43.6	34.8	40.8
LSTM	35.9	45.3	36.3	38.4
IMG	40.4	29.3	42.7	44.2
IMG+BOW	58.7	44.1	52.0	49.4
VIS+LSTM	56.5	46.1	45.9	45.5
2-VIS+BLSTM	58.2	44.8	49.5	47.3
Ours:				
SAN(1, LSTM)	62.5	49.0	54.8	51.6
SAN(1, CNN)	63.6	48.7	56.7	52.7
SAN(2, LSTM)	63.6	**49.8**	57.9	52.8
SAN(2, CNN)	**64.5**	48.6	**57.9**	**54.0**

Table 4: COCO-QA accuracy per class, in percentage

	test-dev				test-std
Methods	All	Yes/No	Number	Other	All
VQA: [1]					
Question	48.1	75.7	36.7	27.1	-
Image	28.1	64.0	0.4	3.8	-
Q+I	52.6	75.6	33.7	37.4	-
LSTM Q	48.8	78.2	35.7	26.6	-
LSTM Q+I	53.7	78.9	35.2	36.4	54.1
SAN(2, CNN)	**58.7**	**79.3**	**36.6**	**46.1**	**58.9**

Table 5: VQA results on the official server, in percentage

Methods	All	Yes/No 36%	Number 10%	Other 54%
SAN(1, LSTM)	56.6	78.1	41.6	44.8
SAN(1, CNN)	56.9	78.8	42.0	45.0
SAN(2, LSTM)	57.3	78.3	**42.2**	45.9
SAN(2, CNN)	**57.6**	78.6	41.8	**46.4**

Table 6: VQA results on our partition, in percentage

VQA, which is the largest among the four data sets. The overall results show that our best model, SAN(2, CNN), outperforms the LSTM Q+I model, the best baseline from [1], by 4.8% absolute. The superior performance of the SANs across all four benchmarks demonstrate the effectiveness of using multiple layers of attention.

In order to study the strength and weakness of the SAN in detail, we report performance at the question-type level on the two large data sets, COCO-QA and VQA, in Table. 4 and 5, respectively. We observe that on COCO-QA, compared to the two best baselines, IMG+BOW and 2-VIS+BLSTM, out best model SAN(2, CNN) improves 7.2% in the question type of *Color*, followed by 6.1% in *Objects*, 5.7% in *Location* and 4.2% in *Number*. We observe similar trend of improvements on VQA. As shown in Table. 5, compared to the best baseline LSTM Q+I, the biggest improvement of SAN(2, CNN) is in the *Other* type, 9.7%, followed by the 1.4% improvement in *Number* and 0.4% improvement in *Yes/No*. Note that the *Other* type in VQA refers to questions that usually have the form of "what color, what kind, what are, what type, where" etc., which are similar to question types of *Color*, *Objects* and *Location* in COCO-QA. The VQA data set has a special *Yes/No* type of questions. The SAN only improves the performance of this type of questions slightly. This could due to that the answer for a *Yes/No* question is very question dependent, so better modeling of the visual information does not provide much additional gains. This also confirms the similar observation reported in [1], e.g., using additional image information only slightly improves the performance in *Yes/No*, as shown in Table. 5, Q+I vs Question, and LSTM Q+I vs LSTM Q.

Our results demonstrate clearly the positive impact of using multiple attention layers. In all four data sets, two-layer SANs always perform better than the one-layer SAN. Specifically, on COCO-QA, on average the two-layer SANs outperform the one-layer SANs by 2.2% in the type of *Color*, followed by 1.3% and 1.0% in the *Location* and *Objects* categories, and then 0.4% in *Number*. This aligns to the order of the improvements of the SAN over baselines. Similar trends are observed on VQA (Table. 6), e.g., the two-layer SAN improve over the one-layer SAN by 1.4% for the *Other* type of question, followed by 0.2% improvement for *Number*, and flat for *Yes/No*.

4.5. Visualization of attention layers

In this section, we present analysis to demonstrate that using multiple attention layers to perform multi-step reasoning leads to more fine-grained attention layer-by-layer in locating the regions that are relevant to the potential answers. We do so by visualizing the outputs of the attention layers of a sample set of images from the COCO-QA test set. Note the attention probability distribution is of size 14×14 and the original image is 448×448, we up-sample the attention probability distribution and apply a Gaussian filter to make it the same size as the original image.

Fig. 5 presents six examples. More examples are presented in the appendix. They cover types as broad as *Object*, *Numbers*, *Color* and *Location*. For each example, the three images from left to right are the original image, the output of the first attention layer and the output of the second attention layer, respectively. The bright part of the image is the detected attention. Across all those examples, we see that in the first attention layer, the attention is scattered on many objects in the image, largely corresponds to the objects and concepts referred in the question, whereas in the second layer, the attention is far more focused on the regions that lead to the correct answer. For example, consider the question `what is the color of the horns`, which asks the color of the horn on the woman's head in Fig. 5(f). In the output of the first attention layer, the model first recognizes a woman in the image. In the output of the second attention layer, the attention is focused on the head of the woman, which leads to the answer of the question: the color of the horn is `red`.

4.6. Errors analysis

We randomly sample 100 images from the COCO-QA test set that the SAN make mistakes. We group the errors into four categories: (i) the SANs focus the attention on the wrong regions (22%), e.g., the example in Fig. 6(a); (ii) the SANs focus on the right region but predict a wrong answer (42%), e.g., the examples in Fig. 6(b)(c)(d); (iii) the answer is ambiguous, the SANs give answers that are different from labels, but might be acceptable (31%). E.g., in Fig. 6(e), the answer label is `pot`, but out model predicts `vase`, which is also visually reasonable; (iv) the labels are clearly wrong (5%). E.g., in Fig. 6(f), our model gives the correct answer `trains` while the label `cars` is wrong.

5. Conclusion

In this paper, we propose a new stacked attention network (SAN) for image QA. SAN uses a multiple-layer attention mechanism that queries an image multiple times to locate the relevant visual region and to infer the answer progressively. Experimental results demonstrate that the proposed SAN significantly outperforms previous state-of-the-art approaches by a great margin on all four image QA data.

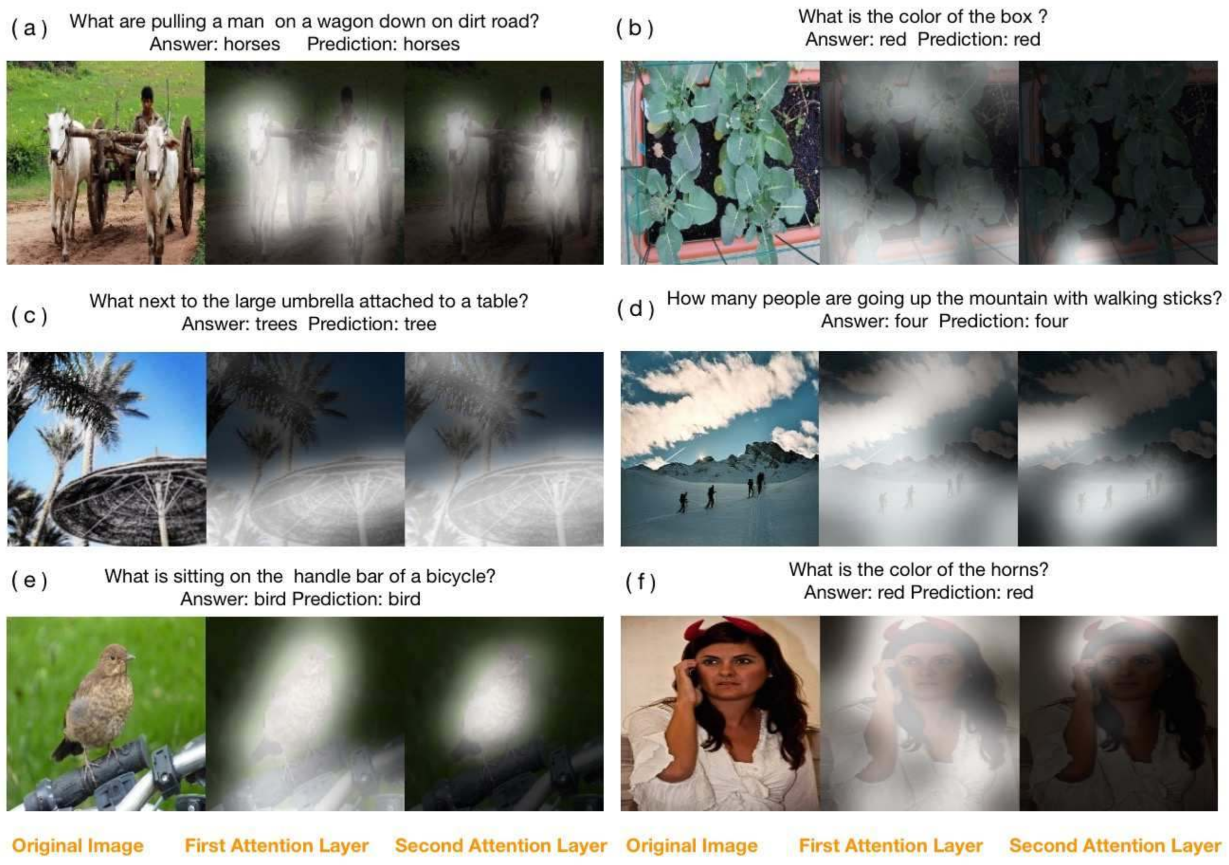

(a)
What are pulling a man on a wagon down on dirt road?
Answer: horses Prediction: horses
(b)
What is the color of the box ?
Answer: red Prediction: red
(c)
What next to the large umbrella attached to a table?
Answer: trees Prediction: tree
(d)
How many people are going up the mountain with walking sticks?
Answer: four Prediction: four
(e)
What is sitting on the handle bar of a bicycle?
Answer: bird Prediction: bird
(f)
What is the color of the horns?
Answer: red Prediction: red
Original Image
First Attention Layer
Second Attention Layer
Original Image
First Attention Layer
Second Attention Layer

Figure 5: Visualization of two attention layers

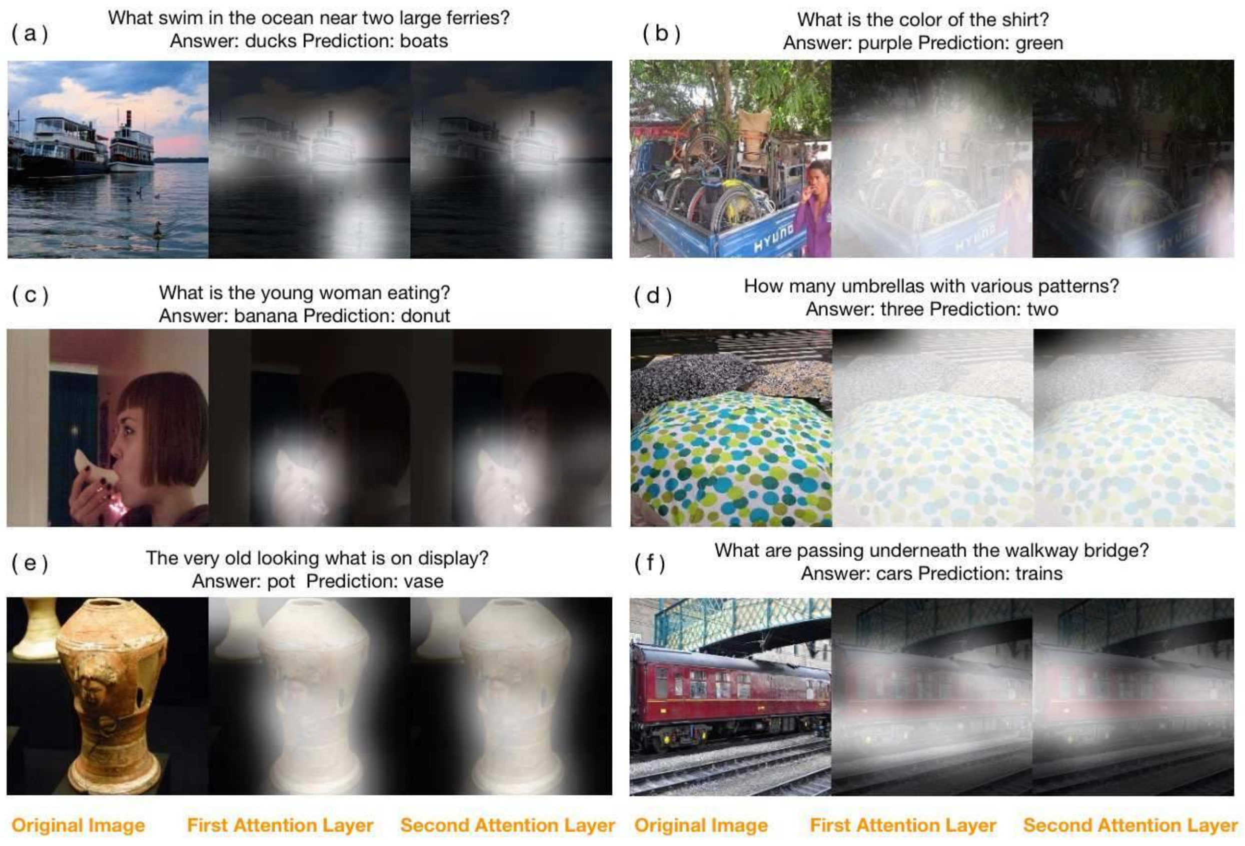

(a)
What swim in the ocean near two large ferries?
Answer: ducks Prediction: boats
(b)
What is the color of the shirt?
Answer: purple Prediction: green
(c)
What is the young woman eating?
Answer: banana Prediction: donut
(d)
How many umbrellas with various patterns?
Answer: three Prediction: two
(e)
The very old looking what is on display?
Answer: pot Prediction: vase
(f)
What are passing underneath the walkway bridge?
Answer: cars Prediction: trains
Original Image
First Attention Layer
Second Attention Layer
Original Image
First Attention Layer
Second Attention Layer

Figure 6: Examples of mistakes

References

[1] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. Vqa: Visual question answering. *arXiv preprint arXiv:1505.00468*, 2015. 1, 2, 5, 6, 7

[2] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 1

[3] J. Berant and P. Liang. Semantic parsing via paraphrasing. In *Proceedings of ACL*, volume 7, page 92, 2014. 1

[4] A. Bordes, S. Chopra, and J. Weston. Question answering with subgraph embeddings. *arXiv preprint arXiv:1406.3676*, 2014. 1

[5] X. Chen and C. L. Zitnick. Learning a recurrent visual representation for image caption generation. *arXiv preprint arXiv:1411.5654*, 2014. 2

[6] H. Fang, S. Gupta, F. Iandola, R. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J. Platt, et al. From captions to visual concepts and back. *arXiv preprint arXiv:1411.4952*, 2014. 2, 3, 5

[7] H. Gao, J. Mao, J. Zhou, Z. Huang, L. Wang, and W. Xu. Are you talking to a machine? dataset and methods for multilingual image question answering. *arXiv preprint arXiv:1505.05612*, 2015. 1, 2

[8] A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013. 5

[9] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1

[10] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. *arXiv preprint arXiv:1412.2306*, 2014. 2

[11] Y. Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014. 3

[12] R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014. 2

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 2

[14] A. Kumar, O. Irsoy, J. Su, J. Bradbury, R. English, B. Pierce, P. Ondruska, I. Gulrajani, and R. Socher. Ask me anything: Dynamic memory networks for natural language processing. *arXiv preprint arXiv:1506.07285*, 2015. 1

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[16] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014*, pages 740–755. Springer, 2014. 5

[17] L. Ma, Z. Lu, and H. Li. Learning to answer questions from image using convolutional neural network. *arXiv preprint arXiv:1506.00333*, 2015. 2, 5, 6

[18] M. Malinowski and M. Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in Neural Information Processing Systems*, pages 1682–1690, 2014. 1, 2, 4, 5, 6

[19] M. Malinowski, M. Rohrbach, and M. Fritz. Ask your neurons: A neural-based approach to answering questions about images. *arXiv preprint arXiv:1505.01121*, 2015. 1, 2, 5, 6

[20] J. Mao, W. Xu, Y. Yang, J. Wang, and A. Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv preprint arXiv:1412.6632*, 2014. 2

[21] M. Ren, R. Kiros, and R. Zemel. Exploring models and data for image question answering. *arXiv preprint arXiv:1505.02074*, 2015. 1, 2, 5, 6

[22] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 101–110. ACM, 2014. 3

[23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014. 5

[25] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014. 3

[26] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014. 2

[27] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. *arXiv preprint arXiv:1411.4555*, 2014. 2

[28] J. Weston, S. Chopra, and A. Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014. 1

[29] Z. Wu and M. Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138. Association for Computational Linguistics, 1994. 5

[30] K. Xu, J. Ba, R. Kiros, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2015. 1, 2

[31] W.-t. Yih, M.-W. Chang, X. He, and J. Gao. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the Joint Conference of the 53rd Annual Meeting of the ACL and the 7th International Joint Conference on Natural Language Processing of the AFNLP*, 2015. 1

[32] W.-t. Yih, X. He, and C. Meek. Semantic parsing for single-relation question answering. In *Proceedings of ACL*, 2014. 1