THINK VISUALLY: QUESTION ANSWERING THROUGH VIRTUAL IMAGERY

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper we study the problem of visual reasoning in the context of textual question answering. We introduce Dynamic Spatial Memory Networks (DSMN), a new deep network architecture that specializes in answering questions that admit latent visual representations, and learns to generate and reason over such representations. Further, we propose two synthetic benchmarks, HouseQA and ShapeIntersection, to evaluate the visual reasoning capability of textual QA systems. Experimental results validate the effectiveness of our proposed DSMN for visual reasoning tasks.

1 INTRODUCTION

The ability to reason is a hallmark of intelligence and a requirement for building question-answering systems. In AI research, reasoning has been strongly associated with logic and symbol manipulation, as epitomized by work in automated theorem proving (Fitting, 2012). But for humans, reasoning involves not only symbols and logic, but also images and shapes. Einstein famously wrote: “The psychical entities which seem to serve as elements in thought are certain signs and more or less clear images which can be ‘voluntarily’ reproduced and combined... Conventional words or other signs have to be sought for laboriously only in a secondary state...” And the history of science abounds with discoveries from visual thinking, from the Benzene ring to the structure of DNA (Pinker, 2003).

There are also plenty of ordinary examples of human visual thinking. Consider a square room with a door in the middle of its southern wall. Suppose you are standing in the room such that the eastern wall of the room is behind you. Where is the door with respect to you? The answer is “left.” Note that in this case both the question and answer are just text. But in order to answer the question, it is natural to construct a mental picture of the room and use it in the process of reasoning.

In this paper, we investigate how to model the process of visual reasoning using deep networks. In particular, we address the task of answering textual questions through visual reasoning—both the question and answer are in text (or symbols in general), but a visual representation is created and used as part of the reasoning process.

We propose Dynamic Spatial Memory Network (DSMN), a novel deep architecture that uses virtual imagery to answer textual questions. DSMN is similar to existing memory networks (Kumar et al., 2016; Sukhbaatar et al., 2015; Henaff et al., 2016) in that it uses vector embeddings of questions and memory modules to perform reasoning. The main novelty of DSMN is that it creates virtual images for the input question and uses a spatial memory to aid the reasoning processing. To the best of our knowledge, DSMN is the first deep architecture that “thinks visually” in order to answer textual questions. Among the deep neural networks developed for textual question-answering, the use of visual representation and spatial memory is unique to DSMN.

To evaluate visual thinking, we introduce two datasets, HouseQA and ShapeIntersection, designed to test a system’s ability to reason visually. In the HouseQA dataset, we provide the blueprint of a house in text, and ask questions about location and orientation of objects in it. For ShapeIntersection, we give symobolic representation of various shapes, and ask at how many places do they intersect. In both the datasets, a reference visual representation is provided for each question, which allows the option to explicitly supervise a network to create visual representations from textual input.

We show through experiments that with the aid of an internal visual representation and a spatial memory, DSMN outperforms strong textual question answering baselines on both HouseQA and

ShapeIntersection. In addition, we demonstrate that explicitly learning to create visual representations further improves QA performance.

Our contributions are three-fold: First, we present Dynamic Spatial Memory Network (DSMN), a novel deep neural network architecture that performs visual reasoning for textual question answering. Second, we introduce two new datasets that evaluate a system's visual thinking ability. Third, we demonstrate that DSMNs achieve superior performance for answering questions that require visual thinking.

2 Related Work

Textual QA: Several text-based datasets have been proposed to test AI systems' reasoning ability (Levesque et al., 2011; Lin & Parikh, 2015). In particular, the bAbI dataset (Weston et al., 2015) has driven the development of several recent deep learning QA systems (Kumar et al., 2016; Sukhbaatar et al., 2015; Henaff et al., 2016). bAbI consists of 20 different tasks to evaluate different kinds of reasoning abilities. Two tasks, Task 17 on Positional Reasoning and Task 19 on Path Finding, are related to visual reasoning. However, each question in Task 17 contains only two sentences, and can be solved through simple logical deduction such as "A is left of B implies B is right of A". Similarly, Task 19 involves a search problem that requires simple spatial deductions such as "A is east of B implies B is west of A". In contrast, our datasets are focused on evaluating visual reasoning, with more sophisticated and challenging questions.

Mental Imagery and Visual Reasoning: The importance of visual reasoning has been long recognized in AI research (Forbus et al., 1991; Lathrop & Laird, 2007).

Recently, Lin & Parikh (2015) takes advantage of visual commonsense to tackle textual question-answering problems. They propose to form "mental images" and use cues from it to perform fill-in-the-blank (FITB) and visual paraphrasing tasks. Seo et al. (2015) use diagrams to help their machine learning system solve SAT geometry problems. Our approach involves forming mental images as well, but unlike Lin & Parikh (2015) and Seo et al. (2015), our models are based on deep networks instead of hand-designed features.

Image Generation: Our work is related to image generation using deep networks, for wich there is a large body of literature, with a diverse set of approaches (Reed et al., 2016; Gregor et al., 2015). Our approach is connected to image generation in that a virtual image is generated from textual input. But in our task, image generation is in the service of reasoning rather than an end goal in itself—as a result, photorealism or artistic style of generated images is irrelevant and not considered.

Visual Question Answering: Our work is also related to visual question-answering (VQA) (Johnson et al., 2016; Antol et al., 2015). Our task is different from VQA because our questions are completely textual whereas in VQA the questions are visual, consisting of both text descriptions and images. The images involved in our task are internal and virtual, and are not part of the input or output.

Memory and Attention: Memory and attention have been increasingly incorporated into deep networks, especially for tasks involving algorithmic inference and/or natural language (Graves et al., 2014; Vaswani et al., 2017). For QA tasks, memory and attention play an important role in state of the art approaches. End-To-End Memory Networks (Sukhbaatar et al., 2015) introduced a neural network with memory and recurrent attention mechanism, that can be trained end-to-end for diverse tasks like textual question answering and language modeling. Concurrently, Kumar et al. (2016) introduced Dynamic Memory Networks (DMN), which also used attention and memory. Xiong et al. (2016) proposed the DMN+, which proposed several improvements over the previous version of DMN, and achieved state-of-the-art results on the VQA (Antol et al., 2015) and the bAbI (Weston et al., 2015) datasets. Our proposed DSMN is a generalization of DMNs (see Sec. 4.6). On removing the images and spatial memory from DMN, our model reduces to a version of DMN.

Recently Gupta et al. (2017) also used spatial memory in their deep learning system, but for visual navigation. We are using spatial memory for textual question-answering.

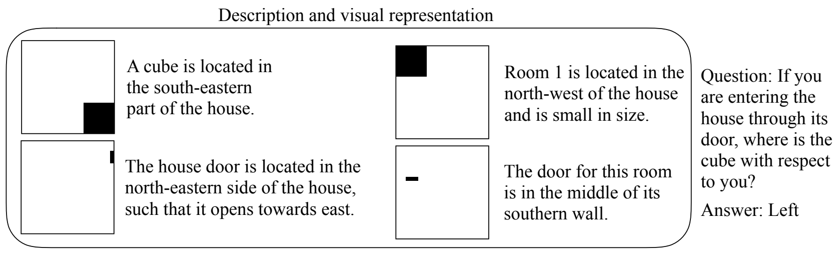

Description and visual representation
A cube is located in the south-eastern part of the house.
Room 1 is located in the north-west of the house and is small in size.
The house door is located in the north-eastern side of the house, such that it opens towards east.
The door for this room is in the middle of its southern wall.
Question: If you are entering the house through its door, where is the cube with respect to you?
Answer: Left

Figure 1: An example in the HouseQA dataset.

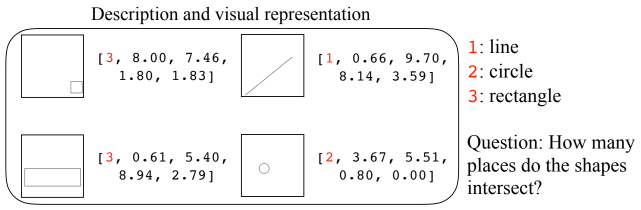


Figure 2: An example in the ShapeIntersection dataset.

3 Datasets

We introduce two question-answering datasets: HouseQA and ShapeIntersection. They are created synthetically for benchmarking a system's ability to think visually. In both datasets, questions and answers are represented as text and symbols. Each dataset also provides visual representations of the questions. Each dataset has 38,400 questions, evenly split into a training set, a validation set and a test set (12,800 each).

3.1 HouseQA

Each sample in the HouseQA dataset involves the layout of a house that has rooms and doors. Objects like cubes, cuboids, spheres and cones are placed at different locations in the house.

Each sample in the dataset has four components, *a textual description, a question, an answer* and *a visual representation*. Figure 1 shows an example.

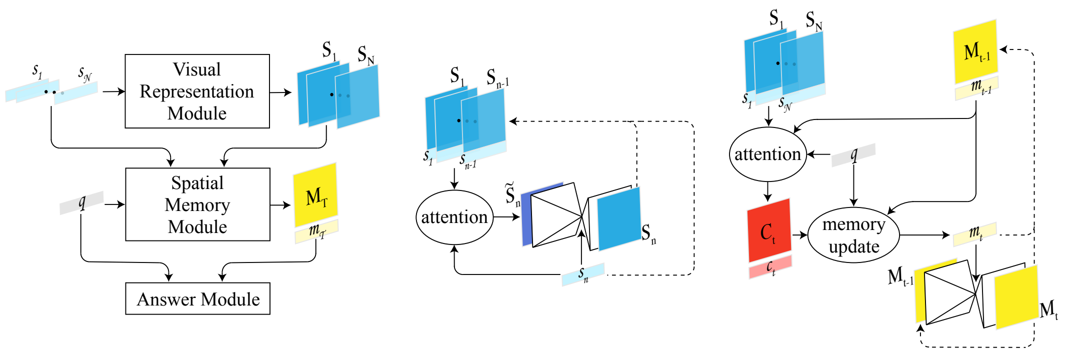
Note that each sentence in the description describes either a room, a door or an object. On an average, a house description has six sentences. A question is always of the following template: *Suppose you are entering the* {*house, room 1, room 2, room 3*}*, where is the* {*house door, room 1 door, room 2 door, room 3 door, cube, cuboid, sphere, cone*} *with respect to you?*. The answer is either of *left, right, front,* or *back*. The visual representation associated with each house consists of an ordered set of image channels, one channel per sentence in the description. An image channel pictorially represents the location and/or orientation of the described item (room, door, object) with respect to the house. An example is shown in Figure 1.

To generate samples for the HouseQA dataset, we define a probabilistic generative process that produces tree structures that represent layouts of houses, similar to scene graphs used in computer graphics. The root node of a tree represents an entire house, and the leaf nodes represent rooms. We do rejection sampling to ensure that all the answers are equally likely in the training, validation, and test set.

3.2 ShapeIntersection

As the name suggests, the ShapeIntersection dataset is concerned with counting the number of intersection points between shapes. In this dataset, the description consists of symbols representing various shapes, and the question is always "how many intersections are there among these shapes?"

There are three types of shapes in our intersection dataset: rectangles, circles, and lines. The description of shapes is provided in the form of a sequence of 1D vectors, where each vector represents one shape. Note that a vector in the ShapeIntersection dataset is analogous to a sentence in the HouseQA dataset, so for the ShapeIntersection dataset, the term 'sentence' actually refers to a vector. Further, each sentence describing a shape consists of 5 real numbers. The first number stands for the type of



(a) Overall architecture (b) Visual represenation module (c) Spatial memory module

Figure 3: The architecture of the proposed Dynamic Spatial Memory Network (DSMN).

shape: 1 for a line, 2 for a circle, and 3 for a rectangle. The subsequent four real numbers specify the size and location of the shape. For example, in case of a rectangle they represent its height, its width, and coordinates of its bottom-left corner.

There are at most 6 lines, 3 rectangles and 3 circles in a given description. In generating the dataset, we do rejection sampling to ensure that the number of intersections is uniformly distributed from 0 to the maximum possible number of intersections, regardless of the number of lines, rectangles and circles. This ensures that the number of intersections cannot be estimated from the number of lines, circles or rectangles.

Similar to the HouseQA dataset, the visual representation for a sample in this dataset is an ordered set of image channels. Each channel is associated with a sentence, and it plots the shape described by that sentence. An example is shown in Figure 2.

4 Dynamic Spatial Memory Network

We propose Dynamic Spatial Memory Networks (DSMN), a novel deep learning architecture that is built upon Dynamic Memory Networks (DMN) (Kumar et al., 2016; Xiong et al., 2016) and designed for visual reasoning. What differentiates DSMN from other question-answering deep architectures is that it forms an internal visual representation of its input. It then uses a spatial memory to reason over this visual representation.

A DSMN and can be divided into five modules: *the input module, visual representation module, question module, spatial memory module* and *answer module*. The input module generates an embedding for each sentence in the description. The visual representation module uses these sentence embeddings to produce an intermediate visual representation for each sentence. In parallel, the question module produces an embedding for the question. The spatial memory module then goes over the question embedding, the sentence embeddings, and the visual representation multiple times to update the spatial memory. Finally, the answer module uses the spatial memory to output the answer. Figure 3 illustrates overall architecture of a DSMN.

4.1 Input Module

The role of the input module is to produce an embedding for each sentence in the input description. The input module can therefore be customized based on how the description is represented in a dataset. For example, in the HouseQA dataset the description is in natural language. Hence, we include a position encoding layer, which uses the word embeddings to produce the initial sentence embeddings. We include the positional encoding layer to enable a fair comparison with DMN+ (Xiong et al., 2016) and MemN2N (Sukhbaatar et al., 2015), which use positional encoding too. For ShapeIntersection, the input description is a sequence of vectors, each representing a shape. We therefore use a fully connected layer to obtain the initial sentence embeddings from the shape vectors.

These initial sentence embeddings are then fed into a bidirectional Gated Recurrent Unit (GRU) to fuse the information together. Let $\overrightarrow{\boldsymbol{s_i}}$ and $\overleftarrow{\boldsymbol{s_i}}$ be the respective output of the forward and backward GRU at i^{th} step. Then, the final sentence embedding for the i^{th} sentence is given by $\boldsymbol{s_i} = \overrightarrow{\boldsymbol{s_i}} + \overleftarrow{\boldsymbol{s_i}}$

4.2 Question Module

The question module produces an embedding for the question. This module is also customized to the specific dataset. In case of the HouseQA dataset, the question module consists of a GRU. The embeddings of the words in the question are fed into the GRU, and the final output of the GRU is used as the question embedding. For the ShapeIntersection dataset, the question is always fixed, so we use a vector with all zero elements as the question embedding.

4.3 Visual Representation Module

The visual representation module generates a visual representation for each sentence in the description. It consists of two sub-components: an attention network and an encoder-decoder network. The purpose of the attention network is to gather the visual information from the previous sentences that are important to produce the visual representation for the current sentence. For example, suppose the current sentence describes the location of an object with respect to a room. Then, in order to infer the location of the object with respect to the house, one needs the location of the room with respect to the house, which is described in a previous sentence.

The encoder-decoder network encodes the visual information gathered by the attention network, combines it with the information from the current sentence, and finally decodes it to obtain the visual representation of the current sentence. As encoders and decoders form an integral part in many components of the DSMN, we formally define them here. An encoder ($En(.)$) takes an image as input and produces an embedding as output, while a decoder ($De(.)$) takes an embedding as input and produces an image as output. An encoder is composed of series of convolution layers and a decoder is composed of series of deconvolution layers.

Suppose we are currently processing the sentence $\boldsymbol{s_n}$. Let us assume that we have already processed the sentences $\boldsymbol{s_1}, \boldsymbol{s_2}, \ldots, \boldsymbol{s_{n-1}}$ and produced the corresponding visual representations $\boldsymbol{S_1}, \boldsymbol{S_2}, \ldots, \boldsymbol{S_{n-1}}$. Note that we also add $\boldsymbol{s_0}$ and $\boldsymbol{S_0}$, which are all-zero element vectors, to the set of processed sentences and representations respectively. Here $\boldsymbol{s_0}$ represents the null sentence. The attention network produces scalar values $a_i \; \forall \; i \in [0, \ldots, n-1]$ to represent the attention for the i^{th} sentence, where a_i is calculated using the following equations:

$$\boldsymbol{z_i} = [|\boldsymbol{s_i} - \boldsymbol{s_t}|; \boldsymbol{s_i} \circ \boldsymbol{s_t}]; a_i = \text{Softmax}(\boldsymbol{w_s}^t \boldsymbol{z_i} + b_s).$$

Here, $\boldsymbol{w_s}$ is a vector, b_s is a scalar, $\circ$ represents element-wise multiplication, $|.|$ represents element-wise absolute value, and $[\boldsymbol{v1}; \boldsymbol{v2}]$ represents the concatenation of vectors $\boldsymbol{v1}$ and $\boldsymbol{v2}$.

The gathered visual information is given by $\bar{\boldsymbol{S}}_{\boldsymbol{t}} = \sum_{i=0}^{t-1} a_i \boldsymbol{S_i}$. It is then fed into the encoder-decoder network. The produced visual representation of the t^{th} sentence is given by:

$$\boldsymbol{S_t} = De_s([\boldsymbol{s_t}; En_s(\bar{\boldsymbol{S}}_{\boldsymbol{t}})])$$

Note that the parameters of $En_s(.)$, $De_s()$, $\boldsymbol{w_s}$, and b_s are shared across multiple iterations. Also, note that in the proposed model, we make the simplifying assumption that the visual representation of the current sentence does not depend on any future sentence. In other words, it can be completely determined from the previous sentences in the description. Both the HouseQA and ShapeIntersection datasets satisfy this assumption.

4.4 Spatial Memory Module

The spatial memory module collects the important information from the description, and updates its memory depending on the gathered information. Similar to DMN+ (Xiong et al., 2016) and MemN2N (Sukhbaatar et al., 2015), it performs the step of collecting information and updating memory multiple times in order to perform transitive reasoning. An iteration of information collection and memory update is referred to as one memory "hop".

The memory consists of two components: a 2D spatial memory and a tag vector. The 2D spatial memory can be thought of as a visual scratch pad on which the network "sketches" out the visual information. The tag vector is meant to represent what is 'sketched' on the 2D spatial memory. For example, the network can sketch the location of room 1 on its 2D spatial memory, and store the fact that it has sketched room 1 in the tag vector.

As mentioned earlier, each step of the spatial memory module involves the gathering of relevant information and the updating of memory. The network gathers the relevant information by calculating the attention value for each sentence based on the question and the current memory. Suppose we are at step t. Let $\boldsymbol{M^{(t-1)}}$ represent the 2D spatial memory after step $t-1$ and $\boldsymbol{m^{(t-1)}}$ represent the tag vector. Then for each sentence embedding $\boldsymbol{s_i}$, we find the scalar attention value $g_i^{(t)}$:

$$\begin{aligned} \boldsymbol{p_i^{(t)}} &= [|\boldsymbol{m^{(t-1)}} - \boldsymbol{s_i}|;\ \boldsymbol{m^{(t-1)}} \circ \boldsymbol{s_i};\ |\boldsymbol{q} - \boldsymbol{s_i}|;\ \boldsymbol{q} \circ \boldsymbol{s_i}; \\ &\quad En_{p_1}^{(t)}(|\boldsymbol{M^{(t-1)}} - \boldsymbol{S_i}|);\ En_{p_2}^{(t)}(\boldsymbol{M^{(t-1)}} \circ \boldsymbol{S_i})] \\ g_i^{(t)} &= \text{Softmax}(\boldsymbol{w_y^t} \boldsymbol{p_i^{(t)}} + b_y) \end{aligned}$$

Note that all elements of $\boldsymbol{M^{(0)}}$ and $\boldsymbol{m^{(0)}}$ are zero. This is meant to represent initial blank memory.Once the attention values are obtained, the gathered information is represented as a 2D context and a context tag vector.

$$\boldsymbol{C^{(t)}} = \sum_{i=0}^{n} {g_i}^{(t)} \boldsymbol{S_i}\ ;\ \ \boldsymbol{c^{(t)}} = \sum_{i=0}^{n} {g_i}^{(t)} \boldsymbol{s_i}$$

Finally, we use the 2D context and context tag vector to update the memory. The memory update rule is given as follows:

$$\begin{aligned} \boldsymbol{m^{(t)}} &= \boldsymbol{W}_m{}^{(t)}[\boldsymbol{m^{(t-1)}};\ \boldsymbol{q};\ \boldsymbol{c^{(t)}};\ En_c(\boldsymbol{C^{(t)}})] + \boldsymbol{b_m}{}^{(t)} \\ \boldsymbol{M^{(t)}} &= De_m^{(t)}([\boldsymbol{m^{(t)}};\ En_m^{(t)}(\boldsymbol{M^{(t-1)}})]) \end{aligned}$$

4.5 Answer Module

The answer module uses the final memory and question embedding to generate the output. The feature vector used for predicting the answer is given by $\boldsymbol{f}$.

$$\boldsymbol{f} = [En_f(\boldsymbol{M^{(T)}});\ \boldsymbol{m^{(T)}};\ \boldsymbol{q}],$$

where $\boldsymbol{M^{(T)}}$ and $\boldsymbol{m^{(T)}}$ represent the final memory.

4.6 DSMN as a strict generalization of DMN

A DSMN is a strict generalization of a DMN (Kumar et al., 2016; Xiong et al., 2016). If we remove the visual representation of the input along with the 2D spatial memory, and just use vector representations with memory tags, then a DSMN reduces to a version of DMN.

4.7 DSMN with or without intermediate visual supervision

As described in previous sections, a DSMN forms an intermediate visual representation of the input. Therefore, if we have a "ground-truth" visual representation for the training data, we could use it to train our network better. This leads to two different ways for training a DSMN, one with intermediate visual supervision and one without it. Without intermediate visual supervision, we train the network in an end-to-end fashion just by using a loss (L_{w/o_vi}) that compares the predicted answer with the ground truth. With intermediate visual supervision, we train our network using an additional visual representation loss (L_{vi}) that measures how close the formed intermediate visual representation of the input is to the ground-truth visual representation. Thus, the loss used for training with intermediate supervision is given by

$$L_{w\_vi} = \lambda_{vi} L_{vi} + (1 - \lambda_{vi}) L_{w/o\_vi},$$

where λ_{vi} is a hyperparameter which can be tuned for each dataset. Note that in neither case do we need any visual input once the network is trained. During testing, the only input to the network is the description and question.

5 Experiments

5.1 Baselines

LSTM (Hochreiter & Schmidhuber, 1997) is a popular neural network model for sequence processing tasks. We use two versions of LSTM-based baselines to evaluate our model. LSTM-1 is a common version that is used as baseline for textual question answering (Sukhbaatar et al., 2015). In LSTM-1, we concatenate all the sentences and the question into a long string. For the HouseQA dataset, we do word embedding look-up, while for ShapeIntersection dataset we project each real number into higher dimension via a series of fully connected layers. The sequence of vectors is fed into an LSTM. The final output vector of the LSTM is then used to do prediction.

We also develop another version of LSTM we call LSTM-2, which uses a two-level hierarchy to embed the question and description. In this version, we first extract an embedding for the question and each sentence. For HouseQA, we use an LSTM to get the sentence embeddings, and for ShapeIntersection, we use a series of fully connected layers. Then we feed the question embedding and sentence embeddings into an LSTM, whose output is used to do prediction.

Further, we compare our model against DMN+ (Xiong et al., 2016) and MemN2N (Sukhbaatar et al., 2015), which achieved state-of-the-art results on the bAbI dataset (Weston et al., 2015). In particular, we compare the 3-hop versions of DSMN, DMN+ and MemN2N.

5.2 Training Details

We used ADAM (Kingma & Ba, 2014)to train all models, and the learning rate for each model is tuned for each dataset. We tune the embedding dimension and l_2 regularization weight for each model and dataset pair separately. As reported by (Sukhbaatar et al., 2015; Kumar et al., 2016; Henaff et al., 2016), we also observe that the results of memory networks are unstable across multiple runs. Therefore for each hyperparameter choice we run all the models 10 times, and select the run with the best performance on the validation set. For the HouseQA dataset, all models are trained up to a maximum of 1600 epochs, with early stopping after 80 epochs if the validation accuracy did not increase. The maximum number of epochs for ShapeIntersection is 800 epochs, with early stopping after 80 epochs. Additionally, we modify the input module and question module of DMN+ and MemN2N to be same as ours for the ShapeIntersection dataset. We use publicly available implementations of DMN+[1] and MemN2N [2]. For MemN2N, while the reported best result is on the version with positional-encoding, linear start training, and random-injection of time index noise (Sukhbaatar et al., 2015), the version we use has only positional encoding. Note that the comparison is still meaningful because linear start straining and time index noise are not used in DMN (and as a result neither in our proposed DSMN)

5.3 Results

The results for HouseQA and ShapeIntersection dataset are summarized in Table 1a. For brevity, we will refer to DSMN trained without intermediate visual supervison as DSMN, and DSMN trained with intermediate visual supervision as DSMN*. We observe similar trends in both datasets. We see that DSMN outperforms DMN, MemN2N, and LSTM baslines. This demonstrates that an intermediate visual representation and a spatial memory are useful for visual reasoning, with or without extra supervision on image generation. Finally, DSMN* (i.e. SMN with intermediate visual supervision) performs the best on both datasets, and outperforms all other approaches by a large margin, suggesting the importance of visual reasoning for the tasks.

Table 1b shows the effect of the number of hops for DSMN and DSMN* on the HouseQA dataset. For the 1-hop and 2-hop case, the differnece between performance of DSMN and DSMN* is larger than for the 3-hop case. The 2-hop DSMN* performs slightly better than the 3-hop case. Even the 1-hop DSMN* performs very well. This is likely because DSMN* has learned to generate good visual representations through additional supervision and can complete "one hop of reasoning" in the visual representation module itself. For example, suppose one needs to find the location of an

[1]`https://github.com/barronalex/Dynamic-Memory-Networks-in-TensorFlow`
[2]`https://github.com/domluna/memn2n`

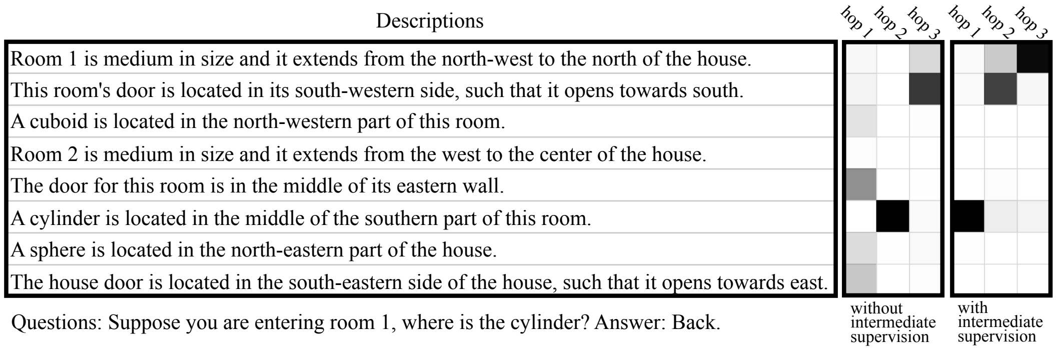
Table 1: Experiment results.

(a) The test set performance of different models on the HouseQA and ShapeIntersection dataset. DSMN* refers to the with intermediate supervision model.

MODEL	HouseQA (accuracy in %)	ShapeIntersection (rmse)
LSTM-1	41.36	3.28
LSTM-2	50.69	2.99
MemN2N	45.92	3.52
DMN+	63.92	2.90
DSMN	92.05	2.86
DSMN*	96.43	2.18

(b) The validation set performance from our ablation study on HouseQA dataset.

MODEL	HouseQA (accuracy in %)
1-Hop DSMN	63.91
2-Hop DSMN	59.09
3-Hop DSMN	92.69
1-Hop DSMN*	85.23
2-Hop DSMN*	97.39
3-Hop DSMN*	96.84



Room 1 is medium in size and it extends from the north-west to the north of the house.
This room's door is located in its south-western side, such that it opens towards south.
A cuboid is located in the north-western part of this room.
Room 2 is medium in size and it extends from the west to the center of the house.
The door for this room is in the middle of its eastern wall.
A cylinder is located in the middle of the southern part of this room.
A sphere is located in the north-eastern part of the house.
The house door is located in the south-eastern side of the house, such that it opens towards east.

Questions: Suppose you are entering room 1, where is the cylinder? Answer: Back.

Figure 4: Attention values for each sentence from a sample in House dataset. Darker color indicates more attention is given.

object, which is placed in a room, with respect to the house. To do so, one first needs to find the location of the room with respect to the house, and then the location of the object with respect to the room. However, if one has already "sketched" out the location of the object, one can directly use it. It is during sketching the object's location that one has completed one hop of reasoning. To illustrate this point, we visualize the attention maps in the spatial memory module of 3-hop DSMN and 3-hop DSMN*. Fig. 4 shows an example. When asked about the cylinder, DSMN* directly jumps to sentence 6. It completes the reasoning in two hops. This also indicates why the 2-hop DSMN* performs so well. For the DSMN model (without the additional visual supervision), the visual representations are not perfect, so it first tries to look for the location of room in which the cylinder is kept, and then looks for it in the second hop.

6 Conclusion

We have investigated how to use deep neural networks for modeling the process of visual thinking. We have introduced two question-answering datasets, HouseQA and ShapeIntersection, that test a systems ability to think visually. We have developed Dynamic Spatial Memory Networks (DSMN), a novel deep learning architecture that reasons in the visual space for answering textual questions. Experimental results have demonstrated the effectiveness of DSMN for visual reasoning.

References

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *ICCV*, pp. 2425–2433, 2015.

Melvin Fitting. *First-order logic and automated theorem proving*. Springer Science & Business Media, 2012.

Kenneth D Forbus, Paul Nielsen, and Boi Faltings. Qualitative spatial reasoning: The clock project. *Artificial Intelligence*, 51(1-3):417–471, 1991.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. *arXiv preprint arXiv:1702.03920*, 2017.

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. Tracking the world state with recurrent entity networks. *arXiv preprint arXiv:1612.03969*, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. *arXiv preprint arXiv:1612.06890*, 2016.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In *ICML*, pp. 1378–1387, 2016.

Scott D Lathrop and John E Laird. Towards incorporating visual imagery into a cognitive architecture. In *Proceedings of the eighth international conference on cognitive modeling*, pp. 25, 2007.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, volume 46, pp. 47, 2011.

Xiao Lin and Devi Parikh. Don't just listen, use your imagination: Leveraging visual common sense for non-visual tasks. In *ICCV*, pp. 2984–2993, 2015.

Steven Pinker. *The language instinct: How the mind creates language*. Penguin UK, 2003.

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016.

Minjoon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. Solving geometry problems: Combining text and diagram interpretation. In *EMNLP*, pp. 1466–1476, 2015.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *NIPS*, pp. 2440–2448, 2015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*, 2015.

Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic memory networks for visual and textual question answering. In *ICML*, pp. 2397–2406, 2016.