

# Dynamic Memory Networks for Visual and Textual Question Answering

Caiming Xiong\*, Stephen Merity\*, Richard Socher  
MetaMind, Palo Alto, CA USA

{CMXIONG, SMERITY, RICHARD} METAMIND.IO

\*indicates equal contribution.

## Abstract

Neural network architectures with memory and attention mechanisms exhibit certain reasoning capabilities required for question answering. One such architecture, the dynamic memory network (DMN), obtained high accuracy on a variety of language tasks. However, it was not shown whether the architecture achieves strong results for question answering when supporting facts are not marked during training or whether it could be applied to other modalities such as images. Based on an analysis of the DMN, we propose several improvements to its memory and input modules. Together with these changes we introduce a novel input module for images in order to be able to answer visual questions. Our new DMN+ model improves the state of the art on both the Visual Question Answering dataset and the bAbI-10k text question-answering dataset without supporting fact supervision.

## 1. Introduction

Neural network based methods have made tremendous progress in image and text classification (Krizhevsky et al., 2012; Socher et al., 2013b). However, only recently has progress been made on more complex tasks that require logical reasoning. This success is based in part on the addition of memory and attention components to complex neural networks. For instance, memory networks (Weston et al., 2015b) are able to reason over several facts written in natural language or (subject, relation, object) triplets. Attention mechanisms have been successful components in both machine translation (Bahdanau et al., 2015; Luong et al., 2015) and image captioning models (Xu et al., 2015).

The dynamic memory network (Kumar et al., 2015) (DMN) is one example of a neural network model that has both a memory component and an attention mechanism. The DMN yields state of the art results on question answering with supporting facts marked during training, sentiment analysis, and part-of-speech tagging.

We analyze the DMN components, specifically the input

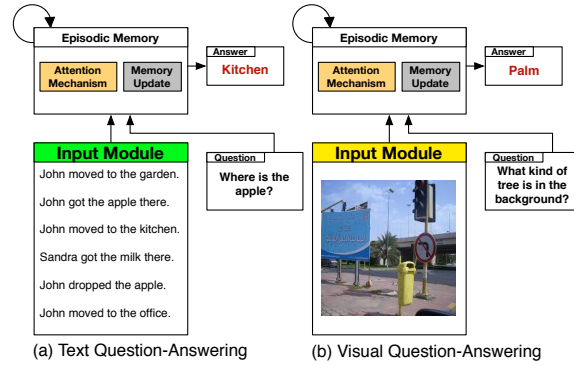


Figure 1. Question Answering over text and images using a Dynamic Memory Network.

module and memory module, to improve question answering. We propose a new input module which uses a two level encoder with a sentence reader and input fusion layer to allow for information flow between sentences. For the memory, we propose a modification to gated recurrent units (GRU) (Chung et al., 2014). The new GRU formulation incorporates attention gates that are computed using global knowledge over the facts. Unlike before, the new DMN+ model does not require that supporting facts (i.e. the facts that are relevant for answering a particular question) are labeled during training. The model learns to select the important facts from a larger set.

In addition, we introduce a new input module to represent images. This module is compatible with the rest of the DMN architecture and its output is fed into the memory module. We show that the changes in the memory module that improved textual question answering also improve visual question answering. Both tasks are illustrated in Fig. 1.

## 2. Dynamic Memory Networks

We begin by outlining the DMN for question answering and the modules as presented in Kumar et al. (2015).

The DMN is a general architecture for question answering (QA). It is composed of modules that allow different aspects such as input representations or memory components to be analyzed and improved independently. The modules, depicted in Fig. 1, are as follows:

**Input Module:** This module processes the input data about which a question is being asked into a set of vectors termed facts, represented as  $F = [f_1, \dots, f_N]$ , where  $N$  is the total number of facts. These vectors are ordered, resulting in additional information that can be used by later components. For text QA in Kumar et al. (2015), the module consists of a GRU over the input words.

As the GRU is used in many components of the DMN, it is useful to provide the full definition. For each time step  $i$  with input  $x_i$  and previous hidden state  $h_{i-1}$ , we compute the updated hidden state  $h_i = GRU(x_i, h_{i-1})$  by

$$u_i = \sigma \left( W^{(u)}x_i + U^{(u)}h_{i-1} + b^{(u)} \right) \quad (1)$$

$$r_i = \sigma \left( W^{(r)}x_i + U^{(r)}h_{i-1} + b^{(r)} \right) \quad (2)$$

$$\tilde{h}_i = \tanh \left( Wx_i + r_i \circ U h_{i-1} + b^{(h)} \right) \quad (3)$$

$$h_i = u_i \circ \tilde{h}_i + (1 - u_i) \circ h_{i-1} \quad (4)$$

where  $\sigma$  is the sigmoid activation function,  $\circ$  is an element-wise product,  $W^{(z)}, W^{(r)}, W \in \mathbb{R}^{n_H \times n_I}$ ,  $U^{(z)}, U^{(r)}, U \in \mathbb{R}^{n_H \times n_H}$ ,  $n_H$  is the hidden size, and  $n_I$  is the input size.

**Question Module:** This module computes a vector representation  $q$  of the question, where  $q \in \mathbb{R}^{n_H}$  is the final hidden state of a GRU over the words in the question.

**Episodic Memory Module:** Episode memory aims to retrieve the information required to answer the question  $q$  from the input facts. To improve our understanding of both the question and input, especially if questions require transitive reasoning, the episode memory module may pass over the input multiple times, updating episode memory after each pass. We refer to the episode memory on the  $t^{th}$  pass over the inputs as  $m^t$ , where  $m^t \in \mathbb{R}^{n_H}$ , the initial memory vector is set to the question vector:  $m^0 = q$ .

The episodic memory module consists of two separate components: the attention mechanism and the memory update mechanism. The attention mechanism is responsible for producing a contextual vector  $c^t$ , where  $c^t \in \mathbb{R}^{n_H}$  is a summary of relevant input for pass  $t$ , with relevance inferred by the question  $q$  and previous episode memory  $m^{t-1}$ . The memory update mechanism is responsible for generating the episode memory  $m^t$  based upon the contextual vector  $c^t$  and previous episode memory  $m^{t-1}$ . By the final pass  $T$ , the episodic memory  $m^T$  should contain all the information required to answer the question  $q$ .

**Answer Module:** The answer module receives both  $q$  and  $m^T$  to generate the model's predicted answer. For simple answers, such as a single word, a linear layer with softmax activation may be used. For tasks requiring a sequence output, an RNN may be used to decode  $a = [q; m^T]$ , the concatenation of vectors  $q$  and  $m^T$ , to an ordered set of tokens. The cross entropy error on the answers is used for training

and backpropagated through the entire network.

### 3. Improved Dynamic Memory Networks: DMN+

We propose and compare several modeling choices for two crucial components: input representation, attention mechanism and memory update. The final DMN+ model obtains the highest accuracy on the bAbI-10k dataset without supporting facts and the VQA dataset (Antol et al., 2015). Several design choices are motivated by intuition and accuracy improvements on that dataset.

#### 3.1. Input Module for Text QA

In the DMN specified in Kumar et al. (2015), a single GRU is used to process all the words in the story, extracting sentence representations by storing the hidden states produced at the end of sentence markers. The GRU also provides a temporal component by allowing a sentence to know the content of the sentences that came before them. Whilst this input module worked well for bAbI-1k with supporting facts, as reported in Kumar et al. (2015), it did not perform well on bAbI-10k without supporting facts (Sec. 6.1).

We speculate that there are two main reasons for this performance disparity, all exacerbated by the removal of supporting facts. First, the GRU only allows sentences to have context from sentences before them, but not after them. This prevents information propagation from future sentences. Second, the supporting sentences may be too far away from each other on a word level to allow for these distant sentences to interact through the word level GRU.

#### Input Fusion Layer

For the DMN+, we propose replacing this single GRU with two different components. The first component is a sentence reader, responsible only for encoding the words into a sentence embedding. The second component is the input fusion layer, allowing for interactions between sentences. This resembles the hierarchical neural auto-encoder architecture of Li et al. (2015) and allows content interaction between sentences. We adopt the bi-directional GRU for this input fusion layer because it allows information from both past and future sentences to be used. As gradients do not need to propagate through the words between sentences, the fusion layer also allows for distant supporting sentences to have a more direct interaction.

Fig. 2 shows an illustration of an input module, where a positional encoder is used for the sentence reader and a bi-directional GRU is adopted for the input fusion layer. Each sentence encoding  $f_i$  is the output of an encoding scheme taking the word tokens  $[w_1^i, \dots, w_{M_i}^i]$ , where  $M_i$  is the length of the sentence.

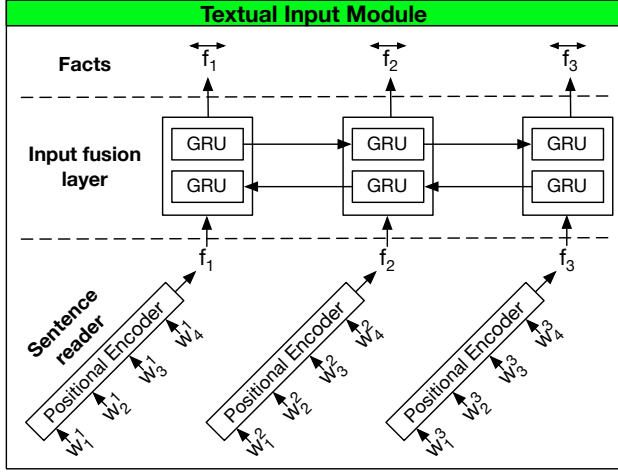


Figure 2. The input module with a “fusion layer”, where the sentence reader encodes the sentence and the bi-directional GRU allows information to flow between sentences.

The sentence reader could be based on any variety of encoding schemes. We selected positional encoding described in Sukhbaatar et al. (2015) to allow for a comparison to their work. GRUs and LSTMs were also considered but required more computational resources and were prone to overfitting if auxiliary tasks, such as reconstructing the original sentence, were not used.

For the positional encoding scheme, the sentence representation is produced by  $f_i = \sum_{j=1}^M l_j \circ w_j^i$ , where  $\circ$  is element-wise multiplication and  $l_j$  is a column vector with structure  $l_{jd} = (1 - j/M) - (d/D)(1 - 2j/M)$ , where  $d$  is the embedding index and  $D$  is the dimension of the embedding.

The input fusion layer takes these input facts and enables an information exchange between them by applying a bi-directional GRU.

$$\vec{f}_i = GRU_{fwd}(f_i, \vec{f}_{i-1}) \quad (5)$$

$$\overleftarrow{f}_i = GRU_{bwd}(f_i, \overleftarrow{f}_{i+1}) \quad (6)$$

$$\overleftrightarrow{f}_i = \vec{f}_i + \overleftarrow{f}_i \quad (7)$$

where  $f_i$  is the input fact at timestep  $i$ ,  $\vec{f}_i$  is the hidden state of the forward GRU at timestep  $i$ , and  $\overleftarrow{f}_i$  is the hidden state of the backward GRU at timestep  $i$ . This allows contextual information from both future and past facts to impact  $\overleftrightarrow{f}_i$ .

We explored a variety of encoding schemes for the sentence reader, including GRUs, LSTMs, and the positional encoding scheme described in Sukhbaatar et al. (2015). For simplicity and speed, we selected the positional encoding scheme. GRUs and LSTMs were also considered but required more computational resources and were prone to overfitting if auxiliary tasks, such as reconstructing the

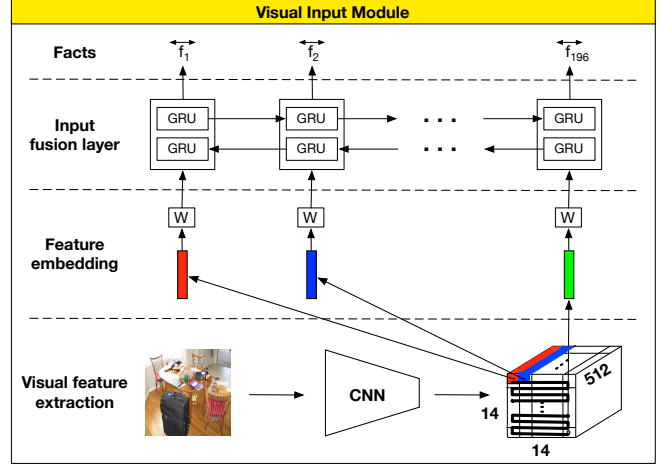


Figure 3. VQA input module to represent images for the DMN.

original sentence, were not used.

### 3.2. Input Module for VQA

To apply the DMN to visual question answering, we introduce a new input module for images. The module splits an image into small local regions and considers each region equivalent to a sentence in the input module for text. The input module for VQA is composed of three parts, illustrated in Fig. 3: local region feature extraction, visual feature embedding, and the input fusion layer introduced in Sec. 3.1.

**Local region feature extraction:** To extract features from the image, we use a convolutional neural network (Krizhevsky et al., 2012) based upon the VGG-19 model (Simonyan & Zisserman, 2014). We first rescale the input image to  $448 \times 448$  and take the output from the last pooling layer which has dimensionality  $d = 512 \times 14 \times 14$ . The pooling layer divides the image into a grid of  $14 \times 14$ , resulting in 196 local regional vectors of  $d = 512$ .

**Visual feature embedding:** As the VQA task involves both image features and text features, we add a linear layer with tanh activation to project the local regional vectors to the textual feature space used by the question vector  $q$ .

**Input fusion layer:** The local regional vectors extracted from above do not yet have global information available to them. Without global information, their representational power is quite limited, with simple issues like object scaling or locational variance causing accuracy problems.

To solve this, we add an input fusion layer similar to that of the textual input module described in Sec. 3.1. First, to produce the input facts  $F$ , we traverse the image in a snake like fashion, as seen in Figure 3. We then apply a bi-directional GRU over these input facts  $F$  to produce the

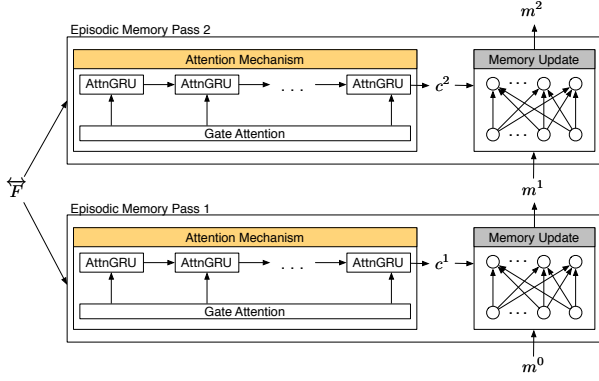


Figure 4. The episodic memory module of the DMN+ when using two passes. The  $\overleftrightarrow{F}$  is the output of the input module.

globally aware input facts  $\overleftrightarrow{F}$ . The bi-directional GRU allows for information propagation from neighboring image patches, capturing spatial information.

### 3.3. The Episodic Memory Module

The episodic memory module, as depicted in Fig. 4, retrieves information from the input facts  $\overleftrightarrow{F} = [\overleftrightarrow{f}_1, \dots, \overleftrightarrow{f}_N]$  provided to it by focusing attention on a subset of these facts. We implement this attention by associating a single scalar value, the attention gate  $g_i^t$ , with each fact  $\overleftrightarrow{f}_i$  during pass  $t$ . This is computed by allowing interactions between the fact and both the question representation and the episode memory state.

$$z_i^t = [\overleftrightarrow{f}_i \circ q; \overleftrightarrow{f}_i \circ m^{t-1}; |\overleftrightarrow{f}_i - q|; |\overleftrightarrow{f}_i - m^{t-1}|] \quad (8)$$

$$Z_i^t = W^{(2)} \tanh(W^{(1)} z_i^t + b^{(1)}) + b^{(2)} \quad (9)$$

$$g_i^t = \frac{\exp(Z_i^t)}{\sum_{k=1}^{M_i} \exp(Z_k^t)} \quad (10)$$

where  $\overleftrightarrow{f}_i$  is the  $i^{th}$  fact,  $m^{t-1}$  is the previous episode memory,  $q$  is the original question,  $\circ$  is the element-wise product,  $|\cdot|$  is the element-wise absolute value, and  $;$  represents concatenation of the vectors.

The DMN implemented in Kumar et al. (2015) involved a more complex set of interactions within  $z$ , containing the additional terms  $[f; m^{t-1}; q; f^T W^{(b)} q; f^T W^{(b)} m^{t-1}]$ . After an initial analysis, we found these additional terms were not required.

#### Attention Mechanism

Once we have the attention gate  $g_i^t$  we use an attention mechanism to extract a contextual vector  $c^t$  based upon the current focus. We focus on two types of attention: soft attention and a new attention based GRU. The latter improves performance and is hence the final modeling choice for the DMN+.

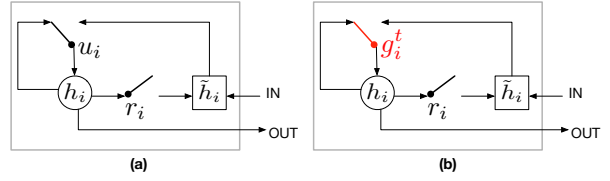


Figure 5. (a) The traditional GRU model, and (b) the proposed attention-based GRU model

**Soft attention:** Soft attention produces a contextual vector  $c^t$  through a weighted summation of the sorted list of vectors  $\overleftrightarrow{F}$  and corresponding attention gates  $g_i^t$ :  $c^t = \sum_{i=1}^N g_i^t \overleftrightarrow{f}_i$ . This method has two advantages. First, it is easy to compute. Second, if the softmax activation is spiky it can approximate a hard attention function by selecting only a single fact for the contextual vector whilst still being differentiable. However the main disadvantage to soft attention is that the summation process loses both positional and ordering information. Whilst multiple attention passes can retrieve some of this information, this is inefficient.

**Attention based GRU:** For more complex queries, we would like for the attention mechanism to be sensitive to both the position and ordering of the input facts  $\overleftrightarrow{F}$ . An RNN would be advantageous in this situation except they cannot make use of the attention gate from Equation 10.

We propose a modification to the GRU architecture by embedding information from the attention mechanism. The update gate  $u_i$  in Equation 1 decides how much of each dimension of the hidden state to retain and how much should be updated with the transformed input  $x_i$  from the current timestep. As  $u_i$  is computed using only the current input and the hidden state from previous timesteps, it lacks any knowledge from the question or previous episode memory.

By replacing the update gate  $u_i$  in the GRU (Equation 1) with the output of the attention gate  $g_i^t$  (Equation 10) in Equation 4, the GRU can now use the attention gate for updating its internal state. This change is depicted in Fig 5.

$$h_i = g_i^t \circ \tilde{h}_i + (1 - g_i^t) \circ h_{i-1} \quad (11)$$

An important consideration is that  $g_i^t$  is a scalar, generated using a softmax activation, as opposed to the vector  $u_i \in \mathbb{R}^{n_H}$ , generated using a sigmoid activation. This allows us to easily visualize how the attention gates activate over the input, later shown for visual QA in Fig. 6. Though not explored, replacing the softmax activation in Equation 10 with a sigmoid activation would result in  $g_i^t \in \mathbb{R}^{n_H}$ . To produce the contextual vector  $c^t$  used for updating the episodic memory state  $m^t$ , we use the final hidden state of the attention based GRU.

#### Episode Memory Updates

After each pass through the attention mechanism, we wish

to update the episode memory  $m^{t-1}$  with the newly constructed contextual vector  $c^t$ , producing  $m^t$ . In the DMN, a GRU with the initial hidden state set to the question vector  $q$  is used for this purpose. The episodic memory for pass  $t$  is computed by

$$m^t = GRU(c^t, m^{t-1}) \quad (12)$$

The work of Sukhbaatar et al. (2015) suggests that using different weights for each pass through the episodic memory may be advantageous. When the model contains only one set of weights for all episodic passes over the input, it is referred to as a **tied model**, as in the “Mem Weights” row in Table 1.

Following the memory update component used in Sukhbaatar et al. (2015) and Peng et al. (2015) we experiment with using a ReLU layer for the memory update, calculating the new episode memory state by

$$m^t = ReLU(W^t[m^{t-1}; c^t; q] + b) \quad (13)$$

where  $;$  is the concatenation operator,  $W^t \in \mathbb{R}^{n_H \times n_H}$ ,  $b \in \mathbb{R}^{n_H}$ , and  $n_H$  is the hidden size. The untying of weights and using this ReLU formulation for the memory update improves accuracy by another 0.5% as shown in Table 1 in the last column. The final output of the memory network is passed to the answer module as in the original DMN.

## 4. Related Work

The DMN is related to two major lines of recent work: memory and attention mechanisms. We work on both visual and textual question answering which have, until now, been developed in separate communities.

**Neural Memory Models** The earliest recent work with a memory component that is applied to language processing is that of memory networks (Weston et al., 2015b) which adds a memory component for question answering over simple facts. They are similar to DMNs in that they also have input, scoring, attention and response mechanisms. However, unlike the DMN their input module computes sentence representations independently and hence cannot easily be used for other tasks such as sequence labeling. Like the original DMN, this memory network requires that supporting facts are labeled during QA training. End-to-end memory networks (Sukhbaatar et al., 2015) do not have this limitation. In contrast to previous memory models with a variety of different functions for memory attention retrieval and representations, DMNs (Kumar et al., 2015) have shown that neural sequence models can be used for input representation, attention and response mechanisms. Sequence models naturally capture position and temporality of both the inputs and transitive reasoning steps.

**Neural Attention Mechanisms** Attention mechanisms allow neural network models to use a question to selectively pay attention to specific inputs. They can benefit image classification (Stollenga et al., 2014), generating captions for images (Xu et al., 2015), among others mentioned below, and machine translation (Cho et al., 2014; Bahdanau et al., 2015; Luong et al., 2015). Other recent neural architectures with memory or attention which have proposed include neural Turing machines (Graves et al., 2014), neural GPUs (Kaiser & Sutskever, 2015) and stack-augmented RNNs (Joulin & Mikolov, 2015).

**Question Answering in NLP** Question answering involving natural language can be solved in a variety of ways to which we cannot all do justice. If the potential input is a large text corpus, QA becomes a combination of information retrieval and extraction (Yates et al., 2007). Neural approaches can include reasoning over knowledge bases, (Bordes et al., 2012; Socher et al., 2013a) or directly via sentences for trivia competitions (Iyyer et al., 2014).

**Visual Question Answering (VQA)** In comparison to QA in NLP, VQA is still a relatively young task that is feasible only now that objects can be identified with high accuracy. The first large scale database with unconstrained questions about images was introduced by Antol et al. (2015). While VQA datasets existed before they did not include open-ended, free-form questions about general images (Geman et al., 2014). Others were too small to be viable for a deep learning approach (Malinowski & Fritz, 2014). The only VQA model which also has an attention component is the stacked attention network (Yang et al., 2015). Their work also uses CNN based features. However, unlike our input fusion layer, they use a single layer neural network to map the features of each patch to the dimensionality of the question vector. Hence, the model cannot easily incorporate adjacency of local information in its hidden state. A model that also uses neural modules, albeit logically inspired ones, is that by Andreas et al. (2016) who evaluate on knowledgebase reasoning and visual question answering. We compare directly to their method on the latter task and dataset.

Related to visual question answering is the task of describing images with sentences (Kulkarni et al., 2011). Socher et al. (2014) used deep learning methods to map images and sentences into the same space in order to describe images with sentences and to find images that best visualize a sentence. This was the first work to map both modalities into a joint space with deep learning methods, but it could only select an existing sentence to describe an image. Shortly thereafter, recurrent neural networks were used to generate often novel sentences based on images (Karpathy & Fei-Fei, 2015; Chen & Zitnick, 2014; Fang et al., 2015; Xu et al., 2015).

## 5. Datasets

To analyze our proposed model changes and compare our performance with other architectures, we use three datasets.

### 5.1. bAbI-10k

For evaluating the DMN on textual question answering, we use bAbI-10k English (Weston et al., 2015a), a synthetic dataset which features 20 different tasks. Each example is composed of a set of facts, a question, the answer, and the supporting facts that lead to the answer. The dataset comes in two sizes, referring to the number of training examples each task has: bAbI-1k and bAbI-10k. The experiments in Sukhbaatar et al. (2015) found that their lowest error rates on the smaller bAbI-1k dataset were on average three times higher than on bAbI-10k.

### 5.2. DAQUAR-ALL visual dataset

The DATaset for QUEStion Answering on Real-world images (DAQUAR) (Malinowski & Fritz, 2014) consists of 795 training images and 654 test images. Based upon these images, 6,795 training questions and 5,673 test questions were generated. Following the previously defined experimental method, we exclude multiple word answers (Malinowski et al., 2015; Ma et al., 2015). The resulting dataset covers 90% of the original data. The evaluation method uses classification accuracy over the single words. We use this as a development dataset for model analysis (Sec. 6.1).

### 5.3. Visual Question Answering

The Visual Question Answering (VQA) dataset was constructed using the Microsoft COCO dataset (Lin et al., 2014) which contained 123,287 training/validation images and 81,434 test images. Each image has several related questions with each question answered by multiple people. This dataset contains 248,349 training questions, 121,512 validation questions, and 244,302 for testing. The testing data was split into test-development, test-standard and test-challenge in Antol et al. (2015).

Evaluation on both test-standard and test-challenge are implemented via a submission system. test-standard may only be evaluated 5 times and test-challenge is only evaluated at the end of the competition. To the best of our knowledge, VQA is the largest and most complex image dataset for the visual question answering task.

## 6. Experiments

### 6.1. Model Analysis

To understand the impact of the proposed module changes, we analyze the performance of a variety of DMN models on textual and visual question answering datasets.

The original DMN (ODMN) is the architecture presented in Kumar et al. (2015) without any modifications. DMN2 only replaces the input module with the input fusion layer (Sec. 3.1). DMN3, based upon DMN2, replaces the soft attention mechanism with the attention based GRU proposed in Sec. 3.3. Finally, DMN+, based upon DMN3, is an untied model, using a unique set of weights for each pass and a linear layer with a ReLU activation to compute the memory update. We report the performance of the model variations in Table 1.

A large improvement to accuracy on both the bAbI-10k textual and DAQUAR visual datasets results from updating the input module, seen when comparing ODMN to DMN2. On both datasets, the input fusion layer improves interaction between distant facts. In the visual dataset, this improvement is purely from providing contextual information from neighboring image patches, allowing it to handle objects of varying scale or questions with a locality aspect. For the textual dataset, the improved interaction between sentences likely helps the path finding required for logical reasoning when multiple transitive steps are required.

The addition of the attention GRU in DMN3 helps answer questions where complex positional or ordering information may be required. This change impacts the textual dataset the most as few questions in the visual dataset are likely to require this form of logical reasoning. Finally, the untied model in the DMN+ overfits on some tasks compared to DMN3, but on average the error rate decreases.

From these experimental results, we find that the combination of all the proposed model changes results, culminating in DMN+, achieves the highest performance across both the visual and textual datasets.

### 6.2. Comparison to state of the art using bAbI-10k

We trained our models using the Adam optimizer (Kingma & Ba, 2014) with a learning rate of 0.001 and batch size of 128. Training runs for up to 256 epochs with early stopping if the validation loss had not improved within the last 20 epochs. The model from the epoch with the lowest validation loss was then selected. Xavier initialization was used for all weights except for the word embeddings, which used random uniform initialization with range  $[-\sqrt{3}, \sqrt{3}]$ . Both the embedding and hidden dimensions were of size  $d = 80$ . We used  $\ell_2$  regularization on all weights except bias and used dropout on the initial sentence encodings and the an-



Dynamic Memory Networks for Visual and Textual Question Answering

Model	ODMN	DMN2	DMN3	DMN+
Input module	GRU	Fusion	Fusion	Fusion
Attention	$\sum g_i f_i$	$\sum g_i f_i$	AttnGRU	AttnGRU
Mem update	GRU	GRU	GRU	ReLU
Mem Weights	Tied	Tied	Tied	Untied
bAbI English 10k dataset				
QA2	36.0	0.1	0.7	0.3
QA3	42.2	19.0	9.2	1.1
QA5	0.1	0.5	0.8	0.5
QA6	35.7	0.0	0.6	0.0
QA7	8.0	2.5	1.6	2.4
QA8	1.6	0.1	0.2	0.0
QA9	3.3	0.0	0.0	0.0
QA10	0.6	0.0	0.2	0.0
QA14	3.6	0.7	0.0	0.2
QA16	55.1	45.7	47.9	45.3
QA17	39.6	5.9	5.0	4.2
QA18	9.3	3.8	0.1	2.1
QA20	1.9	0.0	0.0	0.0
Mean error	11.8	3.9	3.3	2.8
DAQUAR-ALL visual dataset				
Accuracy	27.54	28.43	28.62	28.79

Table 1. Test error rates of various model architectures on the bAbI-10k dataset, and accuracy performance on the DAQUAR-ALL visual dataset. The skipped bAbI questions (1,4,11,12,13,15,19) achieved 0 error across all models.

swer module, keeping the input with probability  $p = 0.9$ . The last 10% of the training data on each task was chosen as the validation set. For all tasks, three passes were used for the episodic memory module, allowing direct comparison to other state of the art methods. Finally, we limited the input to the last 70 sentences for all tasks except QA3 for which we limited input to the last 130 sentences, similar to Sukhbaatar et al. (2015).

On some tasks, the accuracy was not stable across multiple runs. This was particularly problematic on QA3, QA17, and QA18. To solve this, we repeated training 10 times using random initializations and evaluated the model that achieved the lowest validation set loss.

### Text QA Results

We compare our best performing approach, DMN+, to two state of the art question answering architectures: the end to end memory network (E2E) (Sukhbaatar et al., 2015) and the neural reasoner framework (NR) (Peng et al., 2015). Neither approach use supporting facts for training.

The end-to-end memory network is a form of memory network (Weston et al., 2015b) tested on both textual question answering and language modeling. The model features both explicit memory and a recurrent attention mechanism. We select the model from the paper that achieves the lowest mean error over the bAbI-10k dataset. This model utilizes positional encoding for input, RNN-style tied weights for the episode module, and a ReLU non-linearity for the

Task	DMN+	E2E	NR
2: 2 supporting facts	0.3	0.3	-
3: 3 supporting facts	1.1	2.1	-
5: 3 argument relations	0.5	0.8	-
6: yes/no questions	0.0	0.1	-
7: counting	2.4	2.0	-
8: lists/sets	0.0	0.9	-
9: simple negation	0.0	0.3	-
11: basic coreference	0.0	0.1	-
14: time reasoning	0.2	0.1	-
16: basic induction	45.3	51.8	-
17: positional reasoning	4.2	18.6	0.9
18: size reasoning	2.1	5.3	-
19: path finding	0.0	2.3	1.6
Mean error (%)	2.8	4.2	-
Failed tasks (err > 5%)	1	3	-

Table 2. Test error rates of various model architectures on tasks from the the bAbI English 10k dataset. E2E = End-To-End Memory Network results from Sukhbaatar et al. (2015). NR = Neural Reasoner with original auxiliary task from Peng et al. (2015). DMN+ and E2E achieve an error of 0 on bAbI question sets (1,4,10,12,13,15,20).

memory update component.

The neural reasoner framework is an end-to-end trainable model which features a deep architecture for logical reasoning and an interaction-pooling mechanism for allowing interaction over multiple facts. While the neural reasoner framework was only tested on QA17 and QA19, these were two of the most challenging question types at the time.

In Table 2 we compare the accuracy of these question answering architectures, both as mean error and error on individual tasks. The DMN+ model reduces mean error by 1.4% compared to the the end-to-end memory network, achieving a new state of the art for the bAbI-10k dataset.

One notable deficiency in our model is that of QA16: Basic Induction. In Sukhbaatar et al. (2015), an untied model using only summation for memory updates was able to achieve a near perfect error rate of 0.4. When the memory update was replaced with a linear layer with ReLU activation, the end-to-end memory network’s overall mean error decreased but the error for QA16 rose sharply. Our model experiences the same difficulties, suggesting that the more complex memory update component may prevent convergence on certain simpler tasks.

The neural reasoner model outperforms both the DMN and end-to-end memory network on QA17: Positional Reasoning. This is likely as the positional reasoning task only involves minimal supervision - two sentences for input, yes/no answers for supervision, and only 5,812 unique examples after removing duplicates from the initial 10,000

Method	test-dev				test-std
	All	Y/N	Other	Num	All
VQA					
Image	28.1	64.0	3.8	0.4	-
Question	48.1	75.7	27.1	36.7	-
Q+I	52.6	75.6	37.4	33.7	-
LSTM Q+I	53.7	78.9	36.4	35.2	54.1
ACK	55.7	79.2	40.1	36.1	56.0
iBOWIMG	55.7	76.5	42.6	35.0	55.9
DPPnet	57.2	80.7	41.7	37.2	57.4
D-NMN	57.9	80.5	43.1	37.4	58.0
SAN	58.7	79.3	46.1	36.6	58.9
DMN+	<b>60.3</b>	80.5	48.3	36.8	<b>60.4</b>

Table 3. Performance of various architectures and approaches on VQA test-dev and test-standard data. VQA numbers are from Antol et al. (2015); ACK - Wu et al. (2015); iBOWIMG - Zhou et al. (2015); DPPnet - Noh et al. (2015); D-NMN - Andreas et al. (2016); SAN - Yang et al. (2015)

training examples. Peng et al. (2015) add an auxiliary task of reconstructing both the original sentences and question from their representations. This auxiliary task likely improves performance by preventing overfitting.

### 6.3. Comparison to state of the art using VQA

For the VQA dataset, each question is answered by multiple people and the answers may not be the same, the generated answers are evaluated using human consensus. For each predicted answer  $a_i$  for the  $i_{th}$  question with target answer set  $T^i$ , the accuracy of VQA:  $Acc_{VQA} = \frac{1}{N} \sum_{i=1}^N \min(\frac{\sum_{t \in T^i} \mathbb{1}(a_i == t)}{3}, 1)$  where  $\mathbb{1}(\cdot)$  is the indicator function. Simply put, the answer  $a_i$  is only 100% accurate if at least 3 people provide that exact answer.

**Training Details** We use the Adam optimizer (Kingma & Ba, 2014) with a learning rate of 0.003 and batch size of 100. Training runs for up to 256 epochs with early stopping if the validation loss has not improved in the last 10 epochs. For weight initialization, we sampled from a random uniform distribution with range  $[-0.08, 0.08]$ . Both the word embedding and hidden layers were vectors of size  $d = 512$ . We apply dropout on the initial image output from the VGG convolutional neural network (Simonyan & Zisserman, 2014) as well as the input to the answer module, keeping input with probability  $p = 0.5$ .

### Results and Analysis

The VQA dataset is composed of three question domains: Yes/No, Number, and Other. This enables us to analyze the performance of the models on various tasks that require different reasoning abilities.

The comparison models are separated into two broad

classes: those that utilize a full connected image feature for classification and those that perform reasoning over multiple small image patches. Only the SAN and DMN approach use small image patches, while the rest use the fully-connected whole image feature approach.

Here, we show the quantitative and qualitative results in Table 3 and Fig. 6, respectively. The images in Fig. 6 illustrate how the attention gate  $g_i^t$  selectively activates over relevant portions of the image according to the query. In Table 3, our method outperforms baseline and other state-of-the-art methods across all question domains (**All**) in both test-dev and test-std, and especially for **Other** questions, achieves a wide margin compared to the other architectures, which is likely as the small image patches allow for finely detailed reasoning over the image.

However, the granularity offered by small image patches does not always offer an advantage. The Number questions may be not solvable for both the SAN and DMN architectures, potentially as counting objects is not a simple task when an object crosses image patch boundaries.

## 7. Conclusion

We have proposed new modules for the DMN framework to achieve strong results without supervision of supporting facts. These improvements include the input fusion layer to allow interactions between input facts and a novel attention based GRU that allows for logical reasoning over ordered inputs. Our resulting model obtains state of the art results on both the VQA dataset and the bAbI-10k text question-answering dataset, proving the framework can be generalized across input domains.

## References

- Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. Learning to Compose Neural Networks for Question Answering. *arXiv preprint arXiv:1601.01705*, 2016.
- Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., and Parikh, D. VQA: Visual Question Answering. *arXiv preprint arXiv:1505.00468*, 2015.
- Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.
- Bordes, A., Glorot, X., Weston, J., and Bengio, Y. Joint Learning of Words and Meaning Representations for Open-Text Semantic Parsing. *AISTATS*, 2012.
- Chen, X. and Zitnick, C. L. Learning a recurrent visual representation for image caption generation. *arXiv preprint arXiv:1411.5654*, 2014.



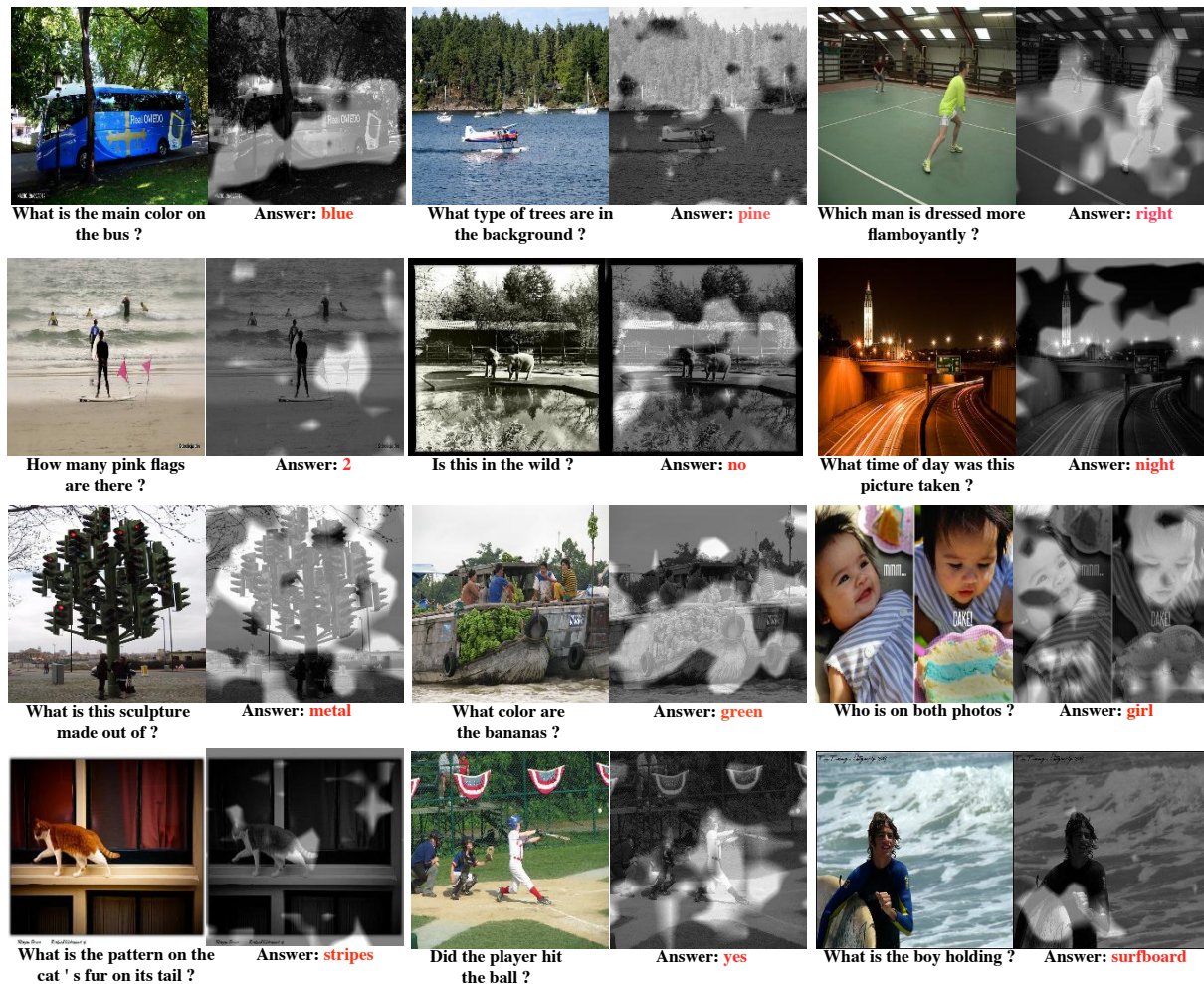


Figure 6. Examples of qualitative results of attention for VQA. The original images are shown on the left. On the right we show how the attention gate  $g_i^t$  activates given one pass over the image and query. White regions are the most active. Answers are given by the DMN+.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*, 2014.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Fang, H., Gupta, S., Iandola, F., Srivastava, R., Deng, L., Dollar, P., Gao, J., He, X., Mitchell, M., and Platt, J. From captions to visual concepts and back. In *CVPR*, 2015.

Geman, D., Geman, S., Hallonquist, N., and Younes, L. A Visual Turing Test for Computer Vision Systems. In *PNAS*, 2014.

Graves, A., Wayne, G., and Danihelka, I. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Iyyer, M., Boyd-Graber, J., Claudino, L., Socher, R., and Daumé III, H. A Neural Network for Factoid Question Answering over Paragraphs. In *EMNLP*, 2014.

Joulin, A. and Mikolov, T. Inferring algorithmic patterns with stack-augmented recurrent nets. In *NIPS*, 2015.

Kaiser, L. and Sutskever, I. Neural GPUs Learn Algorithms. *arXiv preprint arXiv:1511.08228*, 2015.

Karpathy, A. and Fei-Fei, L. Deep Visual-Semantic Alignments for Generating Image Descriptions. In *CVPR*, 2015.

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

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

- Kulkarni, G., Premraj, V., Dhar, S., Li, S., Choi, Y., Berg, A. C., and Berg, T. L. Baby talk: Understanding and generating image descriptions. In *CVPR*, 2011.
- Kumar, A., Irsoy, O., Ondruska, P., Iyyer, M., Bradbury, J., Gulrajani, I., and Socher, R. Ask Me Anything: Dynamic Memory Networks for Natural Language Processing. *arXiv preprint arXiv:1506.07285*, 2015.
- Li, J., Luong, M. T., and Jurafsky, D. A Hierarchical Neural Autoencoder for Paragraphs and Documents. *arXiv preprint arXiv:1506.01057*, 2015.
- Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft COCO: Common Objects in Context. In *ECCV 2014*, 2014.
- Luong, M. T., Pham, H., and Manning, C. D. Effective approaches to attention-based neural machine translation. In *EMNLP*, 2015.
- Ma, L., Lu, Z., and Li, H. Learning to Answer Questions From Image Using Convolutional Neural Network. *arXiv preprint arXiv:1506.00333*, 2015.
- Malinowski, M. and Fritz, M. A Multi-World Approach to Question Answering about Real-World Scenes based on Uncertain Input. In *NIPS*, 2014.
- Malinowski, M., Rohrbach, M., and Fritz, M. Ask your neurons: A neural-based approach to answering questions about images. In *ICCV*, 2015.
- Noh, H., Seo, P. H., and Han, B. Image question answering using convolutional neural network with dynamic parameter prediction. *arXiv preprint arXiv:1511.05756*, 2015.
- Peng, B., Lu, Z., Li, H., and Wong, K. Towards neural network-based reasoning. *arXiv preprint arXiv:1508.05508*, 2015.
- Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- Socher, R., Chen, D., Manning, C. D., and Ng, A. Y. Reasoning With Neural Tensor Networks For Knowledge Base Completion. In *NIPS*, 2013a.
- Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C., Ng, A., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 2013b.
- Socher, R., Karpathy, A., Le, Q. V., Manning, C. D., and Ng, A. Y. Grounded compositional semantics for finding and describing images with sentences. In *ACL*, 2014.
- Stollenga, M. F., J. Masci, F. Gomez, and Schmidhuber, J. Deep Networks with Internal Selective Attention through Feedback Connections. In *NIPS*, 2014.
- Sukhbaatar, S., Szlam, A., Weston, J., and Fergus, R. End-to-end memory networks. In *NIPS*, 2015.
- Weston, J., Bordes, A., Chopra, S., and Mikolov, T. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*, 2015a.
- Weston, J., Chopra, S., and Bordes, A. Memory networks. In *ICLR*, 2015b.
- Wu, Q., Wang, P., Shen, C., Hengel, A. van den, and Dick, A. Ask Me Anything: Free-form Visual Question Answering Based on Knowledge from External Sources. *arXiv preprint arXiv:1511.06973*, 2015.
- Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A. C., Salakhutdinov, R., Zemel, R. S., and Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015.
- Yang, Z., He, X., Gao, J., Deng, L., and Smola, A. Stacked attention networks for image question answering. *arXiv preprint arXiv:1511.02274*, 2015.
- Yates, A., Banko, M., Broadhead, M., Cafarella, M. J., Etzioni, O., and Soderland, S. Textrunner: Open information extraction on the web. In *HLT-NAACL (Demonstrations)*, 2007.
- Zhou, B., Tian, Y., Sukhbaatar, S., Szlam, A., and Fergus, R. Simple baseline for visual question answering. *arXiv preprint arXiv:1512.02167*, 2015.