

# COMPOSITIONAL ATTENTION NETWORKS FOR MACHINE REASONING

**Drew A. Hudson**

Department of Computer Science  
Stanford University  
dorarad@cs.stanford.edu

**Christopher D. Manning**

Department of Computer Science  
Stanford University  
manning@cs.stanford.edu

## ABSTRACT

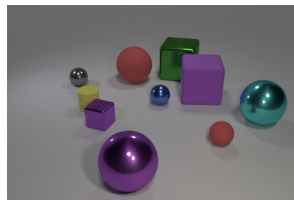
We present the MAC network, a novel fully differentiable neural network architecture, designed to facilitate explicit and expressive reasoning. MAC moves away from monolithic black-box neural architectures towards a design that encourages both transparency and versatility. The model approaches problems by decomposing them into a series of attention-based reasoning steps, each performed by a novel recurrent **Memory, Attention, and Composition (MAC)** cell that maintains a separation between control and memory. By stringing the cells together and imposing structural constraints that regulate their interaction, MAC effectively learns to perform iterative reasoning processes that are directly inferred from the data in an end-to-end approach. We demonstrate the model’s strength, robustness and interpretability on the challenging CLEVR dataset for visual reasoning, achieving a new state-of-the-art 98.9% accuracy, halving the error rate of the previous best model. More importantly, we show that the model is computationally-efficient and data-efficient, in particular requiring 5x less data than existing models to achieve strong results.

## 1 INTRODUCTION

Reasoning, the ability to manipulate previously acquired knowledge to draw novel inferences or answer new questions, is one of the fundamental building blocks of the intelligent mind. As we seek to advance neural networks beyond their current great success with sensory perception towards tasks that require more deliberate thinking, conferring them with the ability to move from facts to conclusions is thus of crucial importance. To this end, we consider here how best to design a neural network to perform the structured and iterative reasoning necessary for complex problem solving.

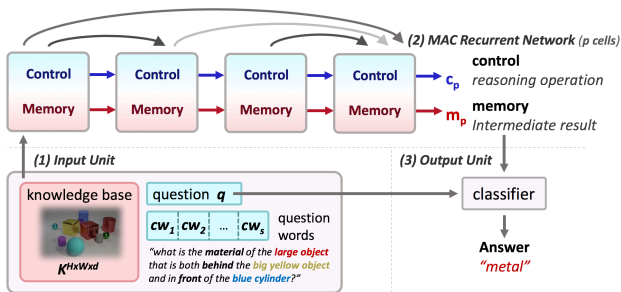
Concretely, we develop a novel model that we apply to the CLEVR task (Johnson et al., 2017a) of visual question answering (VQA). VQA (Antol et al., 2015; Gupta, 2017) is a challenging multimodal task that requires responding to natural language questions about images. However, Agrawal et al. (2016) show how the first generation of successful VQA models tends to acquire only superficial comprehension of both the image and the question, exploiting dataset biases rather than capturing a sound perception and reasoning process that would lead to the correct answer (cf. Sturm (2014)). CLEVR was created to address this problem. As illustrated in figure 1, the dataset features unbiased, highly compositional questions that require an array of challenging reasoning skills, such as transitive and logical relations, counting and comparisons, without allowing any shortcuts around such reasoning.

However, deep learning approaches often struggle to perform well on tasks with a compositional and structured nature (Garnelo et al., 2016; Lake et al., 2017). Most neural networks are essentially very large correlation engines that will hone in on any statistical, potentially spurious pattern that allows them to model the observed data more accurately. The depth, size and statistical nature that allows them to cope with noisy and diverse data often limits their interpretability and hinders their capacity



**Q:** Do the block in front of the tiny yellow cylinder and the tiny thing that is to the right of the large green shiny object have the same color? **A:** No

**Figure 1:** A CLEVR example. Color added for illustration.

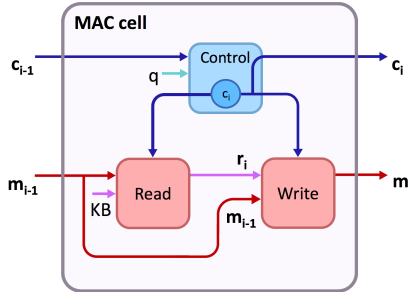


**Figure 2: Model Overview.** The MAC network consists of an input unit, a core recurrent network and an output unit. (1) The input unit transforms the raw image and question into distributed vector representations. (2) The core recurrent network reasons sequentially over the question by decomposing it into a series of operations (*control*) that retrieve information from the image (knowledge base) and aggregate the results into a recurrent *memory*. (3) The output classifier computes the final answer using the question and the final memory state.

to perform explicit and sound inference procedures that are vital for problem solving tasks. To mitigate this issue, some recent approaches adopt symbolic structures, resembling the expression trees of programming languages, that compose neural *modules* from a fixed predefined collection (Andreas et al., 2016a; Johnson et al., 2017b). However, they consequently rely on externally provided structured representations and functional programs, brittle handcrafted parsers or expert demonstrations, and require relatively complex multi-stage reinforcement learning training schemes. The rigidity of these models’ structure and the use of an inventory of specialized operation-specific modules ultimately undermines their robustness and generalization capacities.

Seeking a balance between the versatility and robustness of end-to-end neural approaches on the one hand and the need to support more explicit and structured reasoning on the other, we propose the MAC network, a novel fully differentiable architecture for reasoning tasks. Our model performs structured and explicit reasoning by sequencing a new recurrent **Memory, Attention and Composition** (MAC) cell. The MAC cell was deliberately designed to capture the inner workings of an elementary, yet general-purpose reasoning step, drawing inspiration from the design principles of computer architectures. The cell explicitly separates out memory from control, both represented recurrently, and consists of three operational units that work in tandem to perform a reasoning step: the control unit updates the control state to attend at each iteration to some aspect of a given question; the read unit extracts information out of a knowledge base, guided by the control and memory states; and the write unit integrates the retrieved information into the memory state, iteratively computing the answer. This universal design of the MAC cell serves as a structural prior that encourages the network to solve problems by decomposing them into a sequence of attention-based reasoning operations that are directly inferred from the data, without resorting to any strong supervision. With self-attention connections between the cells, the MAC network is capable of representing arbitrarily complex acyclic reasoning graphs in a soft manner, while still featuring a physically sequential structure and end-to-end differentiability, amenable to training simply by backpropagation.

We demonstrate the model’s quantitative and qualitative performance on the CLEVR task and its associated datasets. The model achieves state-of-the-art accuracy across a variety of reasoning tasks and settings, both for the primary dataset as well as the more difficult human-authored questions. Notably, it performs particularly well on questions that involve counting and aggregation skills, which tend to be remarkably challenging for other VQA models (Santoro et al., 2017; Hu et al., 2017; Johnson et al., 2017b). Moreover, we show that the MAC network learns rapidly and generalizes effectively from an order of magnitude less data than other approaches. Finally, extensive ablation studies and error analysis demonstrate MAC’s robustness, versatility and generalization capacity. These results highlight the significance and value of imposing strong structural priors to guide the network towards compositional reasoning. The model contains structures that encourage it to explicitly perform a chain of operations that build upon each other, allowing MAC to develop reasoning skills from the ground up, realizing the vision of an algebraic, compositional model of inference as proposed by Bottou (2014). Although each cell’s functionality has only a limited range of possible continuous behaviors, geared to perform a simple reasoning operation, when chained together in a MAC network, the whole system becomes expressive and powerful. TensorFlow implementation of the model is available at <https://github.com/stanfordnlp/mac-network>.



**Figure 3: The MAC cell architecture.** The MAC recurrent cell consists of a control unit, read unit, and write unit, that operate over dual *control* and *memory* hidden states. The **control unit** successively attends to different parts of the task description (question), updating the control state to represent at each timestep the reasoning operation the cell intends to perform. The **read unit** extracts information out of a knowledge base (here, image), guided by the control state. The **write unit** integrates the retrieved information into the memory state, yielding the new intermediate result that follows from applying the current reasoning operation.

## 2 THE MAC NETWORK

A MAC network is an end-to-end differentiable architecture primed to perform an explicit multi-step reasoning process, by stringing together  $p$  recurrent MAC cells, each responsible for performing one reasoning step. Given a knowledge base  $K$  (for VQA, an image) and a task description  $q$  (for VQA, a question), the model infers a decomposition into a series of  $p$  reasoning operations that interact with the knowledge base, iteratively aggregating and manipulating information to perform the task at hand. It consists of three components: (1) an input unit, (2) the core recurrent network, composed out of  $p$  MAC cells, and (3) an output unit, all described below.

### 2.1 THE INPUT UNIT

The input unit transforms the raw inputs given to the model into distributed vector representations. Naturally, this unit is tied to the specifics of the task we seek to perform. For the particular case of VQA, it receives a question and an image and processes each of them respectively:

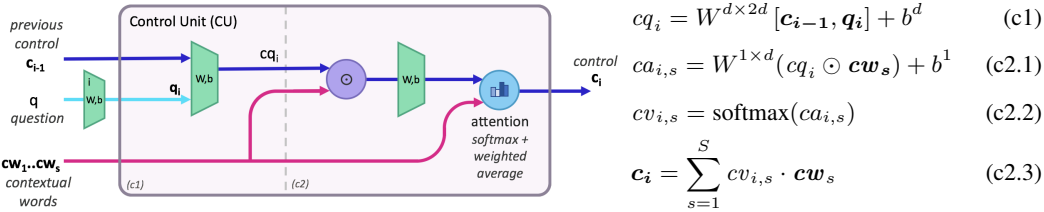
**The question** string, of length  $S$ , is converted into a sequence of learned word embeddings that is further processed by a  $d$ -dimensional biLSTM yielding: (1) *contextual words*: a series of output states  $\mathbf{c}w_1, \dots, \mathbf{c}w_S$  that represent each word in the context of the question, and (2) the question representation:  $q = [\overleftarrow{\mathbf{c}w}_1, \overrightarrow{\mathbf{c}w}_S]$ , the concatenation of the final hidden states from the backward and forward LSTM passes. Subsequently, for each step  $i = 1, \dots, p$ , the question  $q$  is transformed through a learned linear transformation into a position-aware vector  $\mathbf{q}_i = W_i^{d \times 2d} q + b_i^d$ , representing the aspects of the question that are relevant to the  $i^{\text{th}}$  reasoning step.

**The image** is first processed by a fixed feature extractor pre-trained on ImageNet (Russakovsky et al., 2015) that outputs *conv4* features from ResNet101 (He et al., 2016), matching prior work for CLEVR (Hu et al., 2017; Santoro et al., 2017; Perez et al., 2017). The resulting tensor is then passed through two CNN layers with  $d$  output channels to obtain a final image representation, the *knowledge base*  $\mathbf{K}^{H \times W \times d} = \{\mathbf{k}_{h,w}^d |_{h,w=1,1}^{H,W}\}$ , where  $H = W = 14$  are the height and width of the processed image, corresponding to each of its regions.

### 2.2 THE MAC CELL

The MAC cell is a recurrent cell designed to capture the notion of an atomic and universal reasoning operation and formulate its mechanics. For each step  $i = 1, \dots, p$  in the reasoning process, the  $i^{\text{th}}$  cell maintains dual hidden states: control  $\mathbf{c}_i$  and memory  $\mathbf{m}_i$ , of dimension  $d$ , initialized to learned parameters  $\mathbf{m}_0$  and  $\mathbf{c}_0$ , respectively.

**The control**  $\mathbf{c}_i$  represents the *reasoning operation* the cell should accomplish in the  $i^{\text{th}}$  step, selectively focusing on some aspect of the question. Concretely, it is represented by a soft attention-based weighted average of the question words  $\mathbf{c}w_s$ ;  $s = 1, \dots, S$ .



**Figure 4: The Control Unit (CU) architecture.** The control unit attends at each iteration to some part of the question, by applying soft attention over the question words, and updates the control state accordingly. The unit’s inputs and outputs are in **bold**. See section 2.2.1 for details.

The **memory**  $m_i$  holds the *intermediate result* obtained from the reasoning process up to the  $i^{th}$  step, computed recurrently by integrating the preceding hidden state  $m_{i-1}$  with new information  $r_i$  retrieved from the image, performing the  $i^{th}$  reasoning operation  $c_i$ . Analogously to the control,  $r_i$  is a weighted average over its regions  $\{k_{h,w}^{H,W} |_{h,w=1,1}\}$ .

Building on the design principles of computer organization, the MAC cell consists of three operational units: control unit CU, read unit RU and write unit WU, that work together to accomplish tasks by performing an iterative reasoning process: The control unit identifies a series of operations, represented by a recurrent control state; the read unit extracts relevant information from a given knowledge base to perform each operation, and the write unit iteratively integrates the information into the cell’s memory state, producing a new intermediate result.

Through their operation, the three units together impose an interface that regulates the interaction between the control and memory states. Specifically, the control state, which is a function of the question, guides the integration of content from the image into the memory state only through indirect means: soft-attention maps and sigmoidal gating mechanisms. Consequently, the interaction between these two modalities – visual and textual, or knowledge base and query – is mediated through probability distributions only. This stands in stark contrast to common approaches that fuse the question and image together into the same vector space through linear combinations, multiplication, or concatenation. As we will see in section 4, maintaining a strict separation between the representational spaces of question and image, which can interact only through interpretable discrete distributions, greatly enhances the generalizability of the network and improves its transparency.

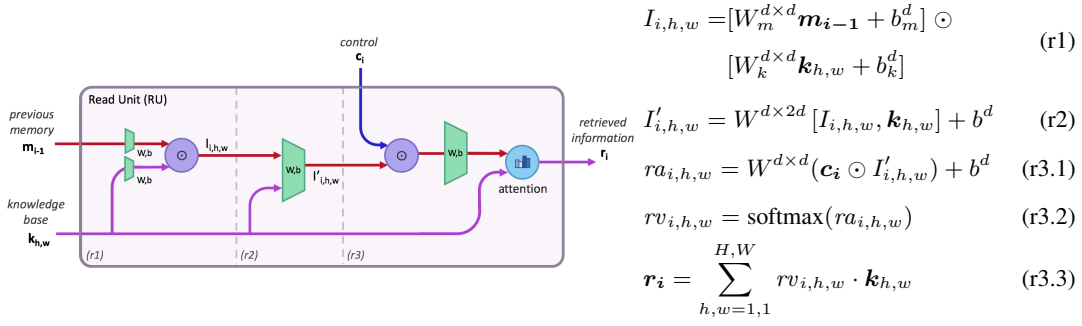
In the following, we describe the cell’s three components: control, read and write units, and detail their formal specification. Unless otherwise stated, all the vectors are of dimension  $d$ .

### 2.2.1 THE CONTROL UNIT

The control unit (see figure 4) determines the reasoning operation that should be performed at each step  $i$ , attending to some part of the question and updating the control state  $c_i$  accordingly. It receives the contextual question words  $cw_1, \dots, cw_S$ , the question position-aware representation  $q_i$ , and the control state from the preceding step  $c_{i-1}$  and consists of two stages:

1. First, we combine  $q_i$  and  $c_{i-1}$  through a linear transformation into  $cq_i$ , taking into account both the overall question representation  $q_i$ , biased towards the  $i^{th}$  reasoning step, as well as the preceding reasoning operation  $c_{i-1}$ . This allows the cell to base its decision for the  $i^{th}$  reasoning operation  $c_i$  on the previously performed operation  $c_{i-1}$ .
2. Subsequently, we *cast*  $cq_i$  onto the space of the question words. Specifically, this is achieved by measuring the similarity between  $cq_i$  and each question word  $cw_s$  and passing the result through a softmax layer, yielding an attention distribution over the question words  $cw_1, \dots, cw_S$ . Finally, we sum the words according to this distribution to produce the reasoning operation  $c_i$ , represented *in terms of* the question words.

The casting of  $cq_i$  onto question words serves as a form of regularization that restricts the space of the valid reasoning operations by anchoring them back in the original question words, and due to the use of soft attention may also improve the MAC cell transparency, since we can interpret the control state content and the cell’s consequent behavior based on the words it attends to.



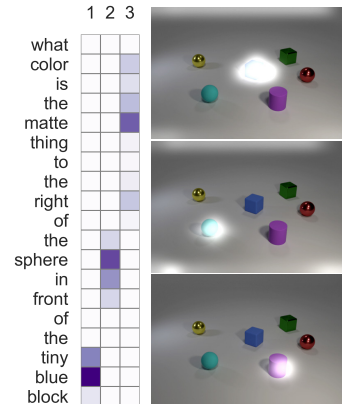
**Figure 5: The Read Unit (RU) architecture.** The read unit retrieves information from the knowledge base that is necessary for performing the current reasoning operation (control) and potentially related to previously obtained intermediate results (memory). It extracts the information by performing a two-stage attention process over the knowledge base elements. See section 2.2.2 for details.

### 2.2.2 THE READ UNIT

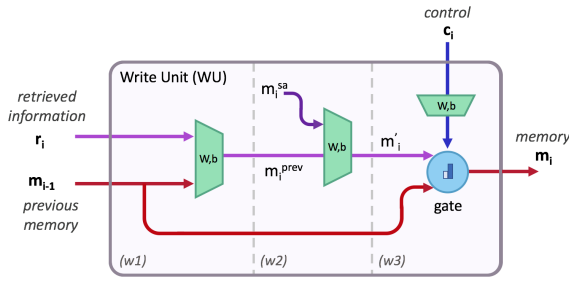
For the  $i^{th}$  step, the read unit (see figure 5) inspects the knowledge base (the image) and retrieves the information  $\mathbf{r}_i$  that is required for performing the  $i^{th}$  reasoning operation  $\mathbf{c}_i$ . The content’s relevance is measured by an attention distribution  $rv_i$  that assigns a probability to each element in the knowledge base  $\mathbf{k}_{h,w}^d$ , taking into account the current reasoning operation  $\mathbf{c}_i$  and the prior memory  $\mathbf{m}_{i-1}$ , the intermediate result produced by the preceding reasoning step. The attention distribution is computed in several stages:

1. First, we compute the direct interaction between the knowledge-base element  $\mathbf{k}_{h,w}$  and the memory  $\mathbf{m}_{i-1}$ , resulting in  $I_{i,h,w}$ . This term measures the relevance of the element to the preceding intermediate result, allowing the model to perform transitive reasoning by considering content that now seems important in light of information obtained from the prior computation step.
2. Then, we concatenate the element  $\mathbf{k}_{h,w}$  to  $I_{i,h,w}$  and pass the result through a linear transformation, yielding  $I'_{i,h,w}$ . This allows us to also consider new information that is *not* directly related to the prior intermediate result, as sometimes a cogent reasoning process has to combine together *independent* facts to arrive at the answer (e.g., for a logical OR operation, set union and counting).
3. Finally, aiming to retrieve information that is relevant for the reasoning operation  $\mathbf{c}_i$ , we measure its similarity to each of the interactions  $I_{i,h,w}$  and pass the result through a softmax layer. This produces an attention distribution over the knowledge base elements, which we then use to compute a weighted average over them –  $\mathbf{r}_i$ .

To give an example of the read unit operation, consider the question in figure 6, which refers to the purple cylinder in the image. Initially, no cue is provided to the model to attend to the cylinder, since no direct mention of it is given in the question. Instead, the model approaches the question in steps: in the first iteration it attends to the “*tiny blue block*”, updating  $\mathbf{m}_1$  accordingly to the visual representation of the block. At the following step, the control unit realizes it should now look for “*the sphere in front*” of the block, storing that in  $\mathbf{c}_2$ . Then, when considering *both*  $\mathbf{m}_1$  and  $\mathbf{c}_2$ , the read unit realizes it should look for “the sphere in front” ( $\mathbf{c}_2$ ) of the blue block (stored in  $\mathbf{m}_1$ ), thus finding the cyan sphere and updating  $\mathbf{m}_2$ . Finally, a similar process repeats in the next iteration, allowing the model to traverse from the cyan ball to the final objective – *the purple cylinder*, and answer the question correctly.



**Figure 6: Attention maps produced by a MAC network of length 3.**



$$m_i^{info} = W^{d \times 2d} [r_i, m_{i-1}] + b^d \quad (w1)$$

$$sa_{ij} = \text{softmax} \left( W^{1 \times d} (c_i \odot c_j) + b^1 \right) \quad (w2.1)$$

$$m_i^{sa} = \sum_{j=1}^{i-1} sa_{ij} \cdot m_j \quad (w2.2)$$

$$m_i' = W_s^{d \times d} m_i^{sa} + W_p^{d \times d} m_i^{info} + b^d \quad (w2.3)$$

$$c_i' = W^{1 \times d} c_i + b^1 \quad (w3.1)$$

$$m_i = \sigma(c_i') m_{i-1} + (1 - \sigma(c_i')) m_i' \quad (w3.2)$$

**Figure 7: The Write Unit (WU) architecture.** The write unit integrates the information retrieved from the knowledge base into the recurrent memory state, producing a new intermediate result  $m_i$  that corresponds to the reasoning operation  $c_i$ . See section 2.2.3 for details.

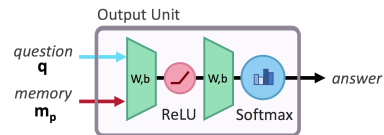
### 2.2.3 THE WRITE UNIT

The write unit (see figure 7) is responsible for computing the  $i^{th}$  intermediate result of the reasoning process and storing it in the memory state  $m_i$ . Specifically, it integrates the information retrieved from the read unit  $r_i$  with the preceding intermediate result  $m_{i-1}$ , guided by the  $i^{th}$  reasoning operation  $c_i$ . The integration proceeds in three steps, the first mandatory while the others are optional<sup>1</sup>:

1. First, we combine the new information  $r_i$  with the prior intermediate result  $m_{i-1}$  by a linear transformation, resulting in  $m_i^{info}$ .
2. **Self-Attention** (Optional). To support non-sequential reasoning processes, such as trees or graphs, we allow each cell to consider all previous intermediate results, rather than just the preceding one  $m_{i-1}$ : We compute the similarity between the  $i^{th}$  operation  $c_i$  and the previous ones  $c_1, \dots, c_{i-1}$  and use it to derive an attention distribution over the prior reasoning steps  $sa_{i,j}$  for  $j = 0, \dots, i-1$ . The distribution represents the relevance of each previous step  $j$  to the current one  $i$ , and is used to compute a weighted average of the memory states, yielding  $m_i^{sa}$ , which is then combined with  $m_i^{info}$  to produce  $m_i'$ . Note that while we compute the attention based on the *control states*, we use it to average over the *memory states*, in a way that resembles Key-Value Memory Networks (Miller et al., 2016).
3. **Memory Gate** (Optional). Not all questions are equally complex – some are simpler while others are more difficult. To allow the model to dynamically adjust the reasoning process length to the given question, we add a sigmoidal gate over the memory state that interpolates between the previous memory state  $m_{i-1}$  and the new candidate  $m_i'$ , *conditioned on the reasoning operation*  $c_i$ . The gate allows the cell to skip a reasoning step if necessary, passing the previous memory value further along the network, dynamically reducing the effective length of the reasoning process as demanded by the question.

### 2.3 THE OUTPUT UNIT

The output unit predicts the final answer to the question based on the question representation  $q$  and the final memory  $m_p$ , which represents the final intermediate result of the reasoning process, holding relevant information from the knowledge base.<sup>2</sup> For CLEVR, where there is a fixed set of possible answers, the unit processes the concatenation of  $q$  and  $m_p$  through a 2-layer fully-connected softmax classifier that produces a distribution over the candidate answers.



**Figure 8: The output unit.** A classifier that predicts an answer based on the question and the final memory state.

<sup>1</sup>Both self-attention connections as well as the memory gate serve to reduce long-term dependencies. However, note that for the CLEVR dataset we were able to maintain almost the same performance with the first step only, and so we propose the second and third ones as optional extensions of the basic write unit, and explore their impact on the model’s performance in section 4.3.

<sup>2</sup>Note that some questions refer to important aspects that do not have counterpart information in the knowledge base, and thus considering both the question and the memory is critical to answer them.

**Table 1:** CLEVR and CLEVR-Humans Accuracy by baseline methods, previous methods, and our method (MAC). For CLEVR-Humans, we show results before and after fine-tuning. (\*) denotes use of extra supervisory information through program labels. (†) denotes use of data augmentation. (‡) denotes training from raw pixels.

Model	CLEVR Overall	Count	Exist	Compare Numbers	Query Attribute	Compare Attribute	Humans before FT	Humans after FT
Human (Johnson et al., 2017b)	92.6	86.7	96.6	86.5	95.0	96.0	-	-
Q-type baseline (Johnson et al., 2017b)	41.8	34.6	50.2	51.0	36.0	51.3	-	-
LSTM (Johnson et al., 2017b)	46.8	41.7	61.1	69.8	36.8	51.8	27.5	36.5
CNN+LSTM (Johnson et al., 2017b)	52.3	43.7	65.2	67.1	49.3	53.0	37.7	43.2
CNN+LSTM+SA+MLP (Johnson et al., 2017a)	73.2	59.7	77.9	75.1	80.9	70.8	50.4	57.6
N2NMN* (Hu et al., 2017)	83.7	68.5	85.7	84.9	90.0	88.7	-	-
PG+EE (9K prog.)* (Johnson et al., 2017b)	88.6	79.7	89.7	79.1	92.6	96.0	-	-
PG+EE (18K prog.)* (Johnson et al., 2017b)	95.4	90.1	97.3	96.5	97.4	98.0	54.0	66.6
PG+EE (700K prog.)* (Johnson et al., 2017b)	96.9	92.7	97.1	98.7	98.1	98.9	-	-
CNN+LSTM+RN†‡ (Santoro et al., 2017)	95.5	90.1	97.8	93.6	97.9	97.1	-	-
CNN+GRU+FiLM (Perez et al., 2017)	97.7	94.3	99.1	96.8	99.1	99.1	56.6	75.9
CNN+GRU+FiLM‡ (Perez et al., 2017)	97.6	94.3	99.3	93.4	99.3	99.3	-	-
<b>MAC</b>	<b>98.9</b>	<b>97.1</b>	<b>99.5</b>	<b>99.1</b>	<b>99.5</b>	<b>99.5</b>	<b>57.4</b>	<b>81.5</b>

### 3 RELATED WORK

There have been several prominent models that address the CLEVR task. By and large they can be partitioned into two groups: module networks, which in practice have all used the strong supervision provided in the form of structured functional programs that accompany each data instance, and large, relatively unstructured end-to-end differentiable networks that complement a fairly standard stack of CNNs with components that aid them in performing reasoning tasks. In contrast to modular approaches (Andreas et al., 2016a;b; Hu et al., 2017; Johnson et al., 2017b), our model is fully differentiable and does not require additional supervision, making use of a single computational cell chained in sequence rather than a collection of custom modules deployed in a rigid tree structure. In contrast to augmented CNN approaches (Santoro et al., 2017; Perez et al., 2017), we suggest that our approach provides an ability for relational reasoning with better generalization capacity, higher computational efficiency and enhanced transparency. These approaches and other related work are discussed and contrasted in more detail in the supplementary material in appendix D.

### 4 EXPERIMENTS

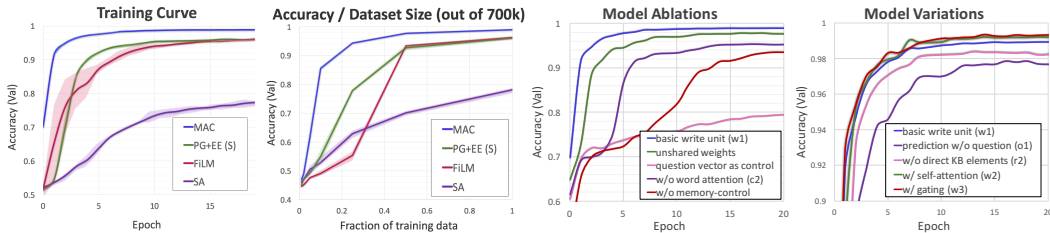
We evaluate our model on the recent CLEVR task for visual reasoning (Johnson et al., 2017a). The dataset consists of rendered images featuring 3D-objects of various shapes, materials, colors and sizes, coupled with machine-generated compositional multi-step questions that measure performance on an array of challenging reasoning skills such as following transitive relations, counting objects and comparing their properties. Each question is also associated with a tree-structured functional program that was used to generate it, specifying the reasoning operations that should be performed to compute the answer.

In the following experiments, our model’s training is cast as a supervised classification problem to minimize the cross-entropy loss of the predicted candidate answer out of the 28 possibilities. The model uses a hidden state size of  $d = 512$  and, unless otherwise stated, length of  $p = 12$  MAC cells.<sup>3</sup> While some prior work uses the functional programs associated with each question as additional supervisory information at training time (see table 1), we intentionally do not use these structured representations to train our model, aiming to infer coherent reasoning strategies directly from the question and answer pairs in an end-to-end approach.

We first perform experiments on the primary 700k dataset. As shown in table 1, our model outperforms all prior work both in overall accuracy, as well as in each of the categories of specific reasoning skills. In particular, for the overall performance, we achieve 98.94% accuracy, more than halving the error rate of the best prior model, FiLM (Perez et al., 2017).

**Counting and Numerical Comparison.** In particular, our performance on questions about counting and numerical comparisons is significantly higher than existing models, which consistently struggle

<sup>3</sup>We initialize the word embeddings of our model to random vectors using a uniform distribution. In an earlier version of this work, we used pretrained GloVe vectors, but found that they did not improve the performance for CLEVR and led to only a marginal improvement for CLEVR-Humans.



**Figure 9:** From left to right: (1) Learning curve of MAC and alternative approaches (accuracy / epoch). (2) Models’ performance as a function of the CLEVR subset size used for training, ranging from 1% to 100%. (3),(4) Learning curves for ablated MAC variants. See section 4.3 for details.

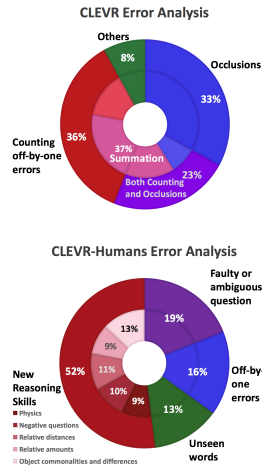
on these question types. Again, we nearly halve the corresponding error rate. These are significant results, as counting and aggregations are known to be particularly challenging in the area of VQA (Chattopadhyay et al., 2017). In contrast to CNNs, using attention enhances our model’s ability to perform reasoning operations such as counting that pertain to the global aggregation of information across different regions of the image.

#### 4.1 CLEVR HUMANS AND ERROR ANALYSIS

We analyze our model’s performance on the CLEVR-Humans dataset (Johnson et al., 2017b), consisting of natural language questions collected through crowdsourcing. As such, the dataset has a diverse vocabulary and linguistic variations, and it also demands more varied reasoning skills. Since the training set is relatively small, comprising 18k samples, we use it to finetune a model pre-trained on the primary CLEVR dataset, following prior work.

As shown in table 1, our model achieves state-of-the-art performance on CLEVR-Humans both before and after fine-tuning. It surpasses the next-best model by 5.6% percent, achieving 81.5%. The results substantiate the model’s robustness against linguistic variations and noise as well as its ability to adapt to new and more diverse vocabulary and reasoning skills. The soft attention performed over the question allows the model to focus on the words that are most critical to answer the question while paying less attention to irrelevant linguistic variations. See figure 11, and figures 16 and 17 in the appendix for examples.

In order to gain insight into the nature of the mistakes our model makes, we perform an error analysis for the CLEVR and CLEVR-Humans datasets (See figure 10). Overall, we see that most of the errors in the CLEVR dataset are either off-by-one counting mistakes or result from heavy object occlusions. For CLEVR-Humans, we observe many errors that involve new reasoning skills that the model has not been trained for, such as ones that relate to physical properties (stability and reflections), relative distances and amounts, commonalities and uniqueness of objects, or negative questions. See appendix B for further details. Nevertheless, the model does respond correctly to many of the questions that fall under these reasoning skills, as illustrated in figures 11 and 16, and so we speculate that the errors the model makes stem in part from the small size of the CLEVR-Human dataset.



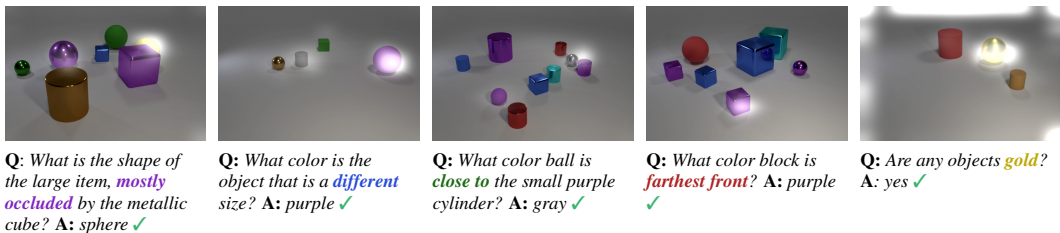
**Figure 10:** Error distribution for CLEVR and CLEVR-Humans.

#### 4.2 COMPUTATIONAL AND DATA EFFICIENCY

We examine the learning curves of MAC and compare them to previous models<sup>4</sup>: specifically, FiLM (Perez et al., 2017), the strongly-supervised PG+EE (Johnson et al., 2017b), and stacked-attention

<sup>4</sup>For previous models, we use the author’s original publicly available implementations. All the models were trained with an equal batch size of 64 (as in the original implementations) and using the same hardware – a single Maxwell Titan X GPU per model. To make sure the results are statistically significant, we run each model multiple (10) times, and plot the averages and confidence intervals.





**Figure 11:** CLEVR-Humans examples showing the model performs novel reasoning skills that do not appear in CLEVR, including: **obstructions**, **object uniqueness**, **relative distances**, **superlatives** and **new concepts**.

networks (SA) (Johnson et al., 2017b; Yang et al., 2016). As shown in figure 9, our model learns significantly faster than other approaches. While we do not have learning curves for the recent Relation Network model, Santoro et al. (2017) report 1.4 million iterations (equivalent to 125 epochs) to achieve 95.5% accuracy, whereas our model achieves a comparable accuracy after only 3 epochs, yielding a 40x reduction in the length of the training process. Likewise, Perez et al. (2017) report a training time of 4 days, equivalent to 80 epochs, to reach accuracy of 97.7%. In contrast, we achieve higher accuracy in 6 epochs, 9.5 hours overall, leading to a 10x reduction in training time.

In order to study the ability of MAC to generalize from a smaller amount of data, we explore its performance on subsets of CLEVR, sampled at random from the original 700k dataset. As shown in figure 9, MAC outperforms the other models by a wide margin: For 50% of the data, equivalent to 350k samples, other models obtain accuracies ranging between 70% and 93%, while our model achieves 97.6%. The gap becomes larger as the dataset size reduces: for 25% of the data, equivalent to 175k samples, the performance of other models is between 50% and 77%, while MAC maintains a high 94.3% accuracy.

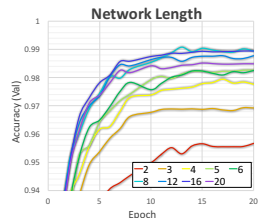
Finally, for just 10% of the data, amounting to 70k samples, our model is the only one to generalize well, with performance of 85.5% on average, whereas the other leading models fail, achieving 49.0%-54.9%. Note that, as pointed out by Johnson et al. (2017a), a simple baseline that predicts the most frequent answer for each question type already achieves 42.1%, suggesting that answering only half of the questions correctly means that the other models barely learn to generalize from this smaller subset. These results demonstrate the robustness and generalization capacity of our architecture and its key role as a structural prior guiding MAC to learn the intended reasoning skills.

### 4.3 ABLATIONS

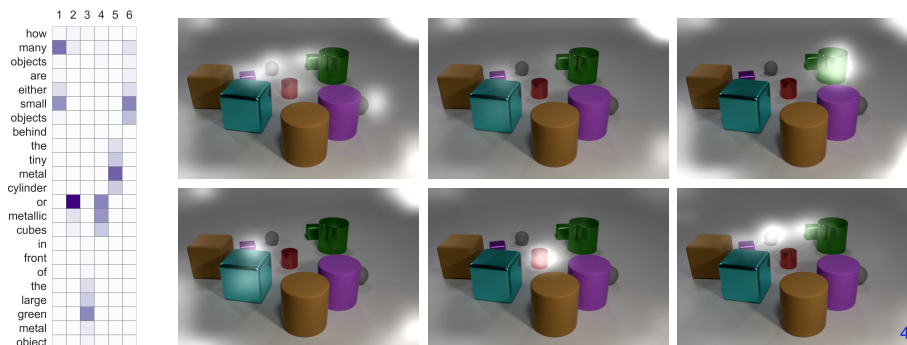
To gain better insight into the relative contribution of the design choices we made, we perform extensive ablation studies. See figure 9 and appendix C for accuracies and learning curves. The experiments demonstrate the robustness of the model to hyperparameter variations such as network dimension and length, and shed light on the significance of various aspects and components of the model, as discussed below:

**Question Attention.** The ablations show that using attention over the question words (see section 2.2.1) is highly effective in accelerating learning and enhancing generalization capacity. Using the complete question  $q$  instead of the attention-based control state leads to a significant drop of 18.5% in the overall accuracy. Likewise, using unconstrained recurrent control states, without casting them back onto the question words space (step (3) in section 2.2.1) leads to a 6x slowdown in the model convergence rate. These results illustrate the importance and usefulness of decomposing the question into a series of simple operations, such that a single cell is faced with learning the semantics of one or a few words at a time, rather than grasping all of the question at once. They provide evidence for the efficacy of *using attention as a regularization mechanism*, by restricting the input and output spaces of each MAC cell.

**Control and Memory.** Maintaining separation between control and memory proves to be another key property that contributes significantly to the model’s accuracy, learning rate and data efficiency. We perform experiments for a variant of the MAC cell in which we maintain one hidden state that



**Figure 12:** Model performance as a function of the network length.



**Figure 14:** Attention maps produced by MAC which provide some evidence for the ability of the model to perform counting and summation of small numbers. Note how the first iterations focus on the key structural question words “many” and “or” that inform the model of the required reasoning operation it has to perform.

plays both the roles of the control and memory, iteratively attending and integrating information from both the question and the image. While this approach achieves a final accuracy of 93.75%, it leads to a sharp drop in the convergence rate, as shown in figure 9, and a 20.2% reduction in the final accuracy for a smaller 10% subset of CLEVR. The results make a strong case for our model’s main design choice, namely, splitting the computation into two dual paths: one that decomposes the linguistic information and another that reconstructs the corresponding visual information.

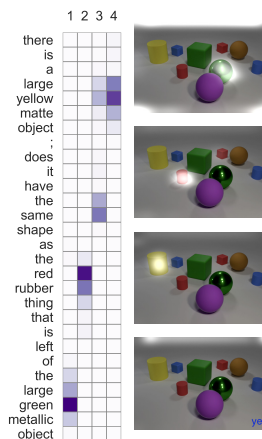
The design choices discussed above were found to be the most significant to the model’s overall accuracy, convergence rate and generalization. Other design choices that were found beneficial include (1) predicting the final answer based on both the final memory state and the question (see section 2.3), and (2) considering knowledge base elements directly (step (2) in section 2.2.2), resulting in 19.8% and 11.1% improvement for a 10% subset of CLEVR, respectively. Please refer to appendix C for further discussion and results.

#### 4.4 INTERPRETABILITY

To obtain better insight into the underlying reasoning processes MAC learns to perform, we study visualizations of the attention distributions produced by the model during its iterative computation, and provide examples in figures 13, 14, 17, and 18. Examining the sequence of attention maps over the image and the question reveals several qualitative patterns and properties that characterize MAC’s mode of operation.

First, we observe that both the linguistic and visual attentions of the model are very focused on specific terms or regions in the image, and commonly refer to concrete objects (“the shiny red cube” or the “metallic cylinder”) or question structural keywords (“or”, “and” or “how many”). More importantly, the attention maps give evidence of the ability of the model to capture the underlying semantic structure of the question, traversing the correct transitive relations between the objects it refers to. For instance, we see in figure 13 how the model explicitly decomposes the question into the correct reasoning steps: first identifying the *green ball*, then focusing on the *red cylinder* that is located left of the ball, and finally attending to the *yellow cylinder*. In the second step, note how the model attends only to the relevant red cylinder and not to other *red rubber things*, correctly resolving the indirect reference in the question. This shows strong evidence of the ability of the model to perform transitive reasoning, integrating information from prior steps that allows it to focus only on the relevant objects, even when they are not mentioned explicitly.

In figure 14, we further see how the model interprets a multi-step counting question, apparently summing up the amounts of two referenced object groups to produce the correct overall count. These observations suggest that the model infers and effectively performs complex reasoning processes in a transparent manner.



**Figure 13:** Attention maps produced by MAC, showing how it tracks transitive relations between objects.

## 5 CONCLUSION

We have introduced the Memory, Attention and Composition (MAC) network, an end-to-end differentiable architecture for machine reasoning. The model solves problems by decomposing them into a series of inferred reasoning steps that are performed successively to accomplish the task at hand. It uses a novel recurrent MAC cell that aims to formulate the inner workings of a single universal reasoning operation by maintaining a separation between memory and control. These MAC cells are chained together to produce explicit and structured multi-step reasoning processes. We demonstrate the versatility, robustness and transparency of the model through quantitative and qualitative studies, achieving state-of-the-art results on the CLEVR task for visual reasoning, and generalizing well even from a 10% subset of the data. The experimental results further show that the model can adapt to novel situations and diverse language, and generate interpretable attention-based rationales that reveal the underlying reasoning it performs. While CLEVR provides a natural testbed for our approach, we believe that the architecture will prove beneficial for other multi-step reasoning and inference tasks, including reading comprehension, textual question answering, and real-world VQA.

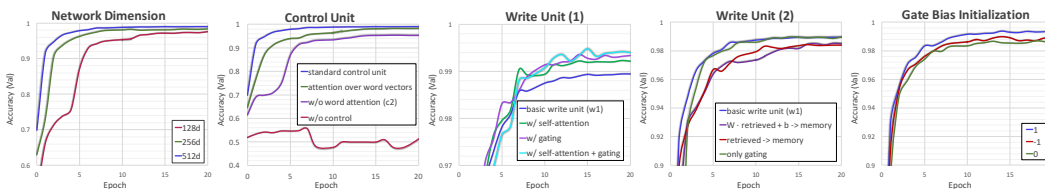
## 6 ACKNOWLEDGMENTS

We wish to thank Justin Johnson, Aaron Courville, Ethan Perez, Harm de Vries, Mateusz Malinowski, Jacob Andreas, and the anonymous reviewers for the helpful suggestions, comments and discussions. Stanford University gratefully acknowledges the support of the Defense Advanced Research Projects Agency (DARPA) Communicating with Computers (CwC) program under ARO prime contract no. W911NF15-1-0462 for supporting this work.

## REFERENCES

- Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. Analyzing the behavior of visual question answering models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1955–1960, 2016.
- Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 39–48, 2016a.
- Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Learning to compose neural networks for question answering. In *Proceedings of NAACL-HLT*, pp. 1545–1554, 2016b.
- Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. VQA: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2425–2433, 2015.
- Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. 2015.
- Léon Bottou. From machine learning to machine reasoning. *Machine learning*, 94(2):133–149, 2014.
- Prithvijit Chattopadhyay, Ramakrishna Vedantam, Ramprasaath R Selvaraju, Dhruv Batra, and Devi Parikh. Counting everyday objects in everyday scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1135–1144, 2017.
- Marta Garnelo, Kai Arulkumaran, and Murray Shanahan. Towards deep symbolic reinforcement learning. *arXiv preprint arXiv:1609.05518*, 2016.
- Alex Graves, Greg Wayne, and Ivo Danihelka. Neural Turing machines. *arXiv preprint arXiv:1410.5401*, 2014.
- Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, 2016.
- Akshay Kumar Gupta. Survey of visual question answering: Datasets and techniques. *arXiv preprint arXiv:1705.03865*, 2017.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 804–813, 2017.
- Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456, 2015.
- 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. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pp. 1988–1997. IEEE, 2017a.
- Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Inferring and executing programs for visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2989–2998, 2017b.

- 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 *International Conference on Machine Learning*, pp. 1378–1387, 2016.
- Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40, 2017.
- Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. In *Advances In Neural Information Processing Systems*, pp. 289–297, 2016.
- Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1400–1409, 2016.
- Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. *arXiv preprint arXiv:1709.07871*, 2017.
- Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.
- Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Tim Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pp. 4974–4983, 2017.
- Bob L Sturm. A simple method to determine if a music information retrieval system is a horse. *IEEE Transactions on Multimedia*, 16(6):1636–1644, 2014.
- Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic memory networks for visual and textual question answering. In *International Conference on Machine Learning*, pp. 2397–2406, 2016.
- Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 21–29, 2016.



**Figure 15:** Learning curves for ablated MAC variants (accuracy / epoch). See appendix C for details.

## SUPPLEMENTARY MATERIAL

### A IMPLEMENTATION AND TRAINING DETAILS

We train our model using Adam (Kingma & Ba, 2014), with a learning rate of  $10^{-4}$  and a batch size of 64. We use gradient clipping, and employ early stopping based on the validation accuracy, resulting in a training process of 10–20 epochs, equivalent to roughly 15–30 hours on a single Maxwell Titan X GPU. Word vectors have dimension 300 and were initialized randomly using a standard uniform distribution. The exponential moving averages of the model weights are maintained during training, with a decay rate of 0.999, and used at test time instead of the raw weights. We use variational dropout of 0.15 across the network, along with ELU as non-linearity, which, in our experience, accelerates training and performs favorably compared to the more standard ReLU.

### B ERROR ANALYSIS

To gain insight into the model’s failure cases, we perform error analysis for the CLEVR and CLEVR-Humans datasets. For CLEVR, we see that many of the errors arise from object occlusions, which may make it harder for the model to recognize the objects’ material or shape. Most of the other errors are off-by-one counting mistakes, oftentimes for questions that ask to sum up two groups of objects (see examples in figure 16). Interestingly, we noticed that when the model is required to count objects that are heavily occluded, it may lead the model to slightly underestimate the correct number of objects, suggesting that it may perform some sort of “continuous” counting rather than discrete.

For CLEVR-Humans, the errors made by the model are more diverse. About half of the errors result from questions about reasoning skills that the model has not been exposed to while training on CLEVR. These include questions about physical properties: lighting and shadows, reflections and objects stability (“How many of the items are casting a shadow?”, “Can a ball stay still on top of one another?”); relative distances (“How many objects... are almost touching?”, “What color is the sphere positioned closest to...”, “What object is in between...”); relative amounts (“Are half the items...”, “Are the objects mainly...”); commonalities (“What color are the identical objects?”, “What shape do the items that... have in common?”); and negative questions, which refer to objects that do not maintain some property (“How many items do not...”). In addition, we observed some cases where the model misinterprets unseen words, capturing plausible but incorrect semantics: for instance, in some cases it interpreted a “caramel” object as yellow whereas the original question referred to a brown one, or considered a cylinder to be “circle” while the question referred to the spheres only. We believe that these errors may arise from diversity in the semantics of these words across the dataset, making the model learn those potentially valid but incorrect interpretations. Similarly to CLEVR, we observed some off-by-one errors in CLEVR-Humans. Finally, in one fifth of the cases, the errors result from faulty or ambiguous questions, which may mistakenly regard a cyan object as blue or mention references that cannot be uniquely resolved to a specific object.

### C ABLATION STUDIES

Based on the validation set, we have conducted an ablation study for MAC to better understand the impact of each of its components on the overall performance. We have tested each setting for the primary 700K CLEVR dataset as well as on a 10% subset of it. See table 2, figure 9 and figure 15 for

final accuracies and training curves. The following discussion complements the main conclusions presented in section 4.3:

**Network Length.** We observe a positive correlation between the network length and its performance, with significant improvements up to length  $p = 8$ . These results stand out from other multi-hop architectures that tend to benefit from a lower number of iterations, commonly 2–3 only (Yang et al., 2016; Kumar et al., 2016), and suggest that MAC makes effective use of the recurrent cells to perform compositional reasoning.

**Weight Sharing.** Weight sharing across the  $p$  cell instances has also proven to be useful both for the primary CLEVR task as well as for settings with limited data. In contrast to alternative approaches that apply specialized modules for different parts of the reasoning, these results provide some evidence for the ability of the same MAC cell to adapt its behavior to the task at hand and demonstrate different behaviors as inferred from the context.

**Control Unit.** We have performed several ablations in the control unit architecture to identify its contribution to the model behavior and performance. First, we observe that, as would be expected, an ablated model that reasons over the image alone with no access to the question, performs poorly, achieving accuracy of 51.1%. As discussed in section 4.3, the ablations further show the importance of applying attention over the question words to decompose it into an explicit sequence of steps. Finally, we find that using the “contextual words” – the output states of a biLSTM processing the question – results in better performance and faster learning than attending directly to the learned word vectors. This implies that the model benefits from interpreting the word semantics and entailed behaviors in the broader context of the question rather than as a sequence of independent entities.

**Write Unit.** The basic MAC write unit integrates new information  $r_i$  with the previous memory state  $m_{i-1}$  through a linear transformation (step (1) in section 2.2.3). In this experiment, we explore other variants of the unit. We begin by measuring the impact of the self-attention and gating mechanisms, both aiming to reduce long-range dependencies in the reasoning process. Compared to the basic MAC model, which achieves 98.94% on the validation set, self-attention yields an accuracy of 99.23%, memory gating – 99.36%, and adding both results in 99.48%. While we can see some gain from using these components for CLEVR, we speculate that they may prove more useful for tasks that necessitate longer or more complex reasoning processes over larger knowledge bases.

Next, we examine ablated write unit variants that assign the newly retrieved information  $r_i$  (or its linear transformation) to  $m_i$  directly, ignoring the prior memory content  $m_{i-1}$ . Notably, the results show that in fact such variants are only slightly worse than the default basic write unit, reducing accuracy by 0.4% only. We perform further experiments in which we compute the new memory state  $m_i$  by averaging the retrieved information  $r_i$  with the previous memory state  $m_{i-1}$  using a sigmoidal gate alone. This architecture results in equivalent performance to that of the standard basic write unit variant.

**Gate Bias Initialization.** Finally, we test the impact of the gate bias (step (3) in section 2.2.3), initializing it to either  $-1$ ,  $0$  or  $1$ . Intuitively, initialization of  $-1$  amounts to biasing the model to retain previous memory states and thereby shortening the effective reasoning process, while initialization of  $1$  is equivalent to using all new intermediate results derived by each of the cells. The experiments show that a bias of  $1$  is optimal for training on the full dataset while  $0$  is ideal for settings of limited data (training on 10% of the data). These results demonstrate that when enough data is available, the MAC network benefits from utilizing its full capacity, whereas biasing the model towards using less cells helps to mitigate overfitting when data is more scarce.

## D RELATED WORK

In this section we provide detailed discussion of related work. Several models have been applied to the CLEVR task. These can be partitioned into two groups, module networks that use the strong supervision provided as a tree-structured functional program associated with each instance, and end-to-end, fully differentiable networks that combine a fairly standard stack of CNNs with components that aid them in performing reasoning tasks. We also discuss the relation of MAC to other approaches, such as memory networks and neural computers.

**Table 2:** Accuracies for ablated MAC models, measured for the validation set after training on the full CLEVR dataset (left) and 10% subset of it (right).

Model	Standard CLEVR	10% CLEVR
MAC	98.9	84.5
state dimension 256	98.4	76.3
state dimension 128	97.6	77.0
unshared weights	97.8	67.5
attention over word vectors	98.3	61.4
w/o word-attention	95.3	63.2
question vector as control	80.7	65.0
w/o control	55.6	51.5
w/o memory-control separation	93.9	64.7
w/o direct KB elements	98.4	73.4
retrieved $\rightarrow$ memory	98.2	84.5
$W \cdot$ retrieved + $b \rightarrow$ memory	98.5	83.7
only memory gate	99.3	83.1
w/ self-attention	99.2	83.2
w/ memory gate	99.4	83.1
w/ self-attention and memory gate	99.5	85.5
gate bias 0	98.7	84.9
gate bias 1	99.4	68.5
gate bias $-1$	99.0	77.1
prediction w/o question	97.8	64.7

## D.1 MODULE NETWORKS

The modular approach (Andreas et al., 2016a;b; Hu et al., 2017; Johnson et al., 2017b) first translates a given question into a tree-structured action plan, aiming to imitate the question underlying structural representation externally provided as strong supervision. Then, it constructs a tailor-made network that progressively executes the plan over the image. The network is composed of discrete units selected out of a fixed collection of predefined “modules”, each responsible for an elementary reasoning operation, such as identifying an object’s color, filtering them for their shape, or comparing their amounts. Each module has its own set of learned parameters (Johnson et al., 2017b), or even a hand-crafted design (Andreas et al., 2016a) that guides it towards its intended behavior. Overall, this approach makes discrete choices at two levels: the identity of each module – the behavior it should learn among a fixed set of possible behavior types, and the network layout – the way in which the modules are wired together to compute the answer. The model differentiability is thus confined to the boundaries of a single module.

Several key differences exist between our approaches. First, MAC replaces the fixed and specialized modules inventory with one universal cell that adapts its operation to the task at hand, selected from a continuous range of reasoning behaviors. Therefore, in contrast to module networks, our cell can be applied across all the reasoning steps, sharing both its parameters and architecture. Second, we replace the dynamic recursive tree structures with a sequential topology, augmented by soft attention mechanisms, inspired by Bahdanau et al. (2015). This confers the network with the capacity to represent arbitrarily complex Directed Acyclic Graphs (DAGs) in a soft way, while still having efficient and readily deployed physical sequential structure. Together, these relaxations allow us to effectively train our model end-to-end by backpropagation alone, whereas module networks demand more involved training schemes that rely on the strongly-supervised programs at the first stage, and on various reinforcement learning (RL) techniques at the second. Finally, since the only source of supervision in training our model arises from the answer to each question, MAC is free to acquire more robust and adaptive reasoning strategies from the bottom up – inferred directly from the data, rather than trying to imitate the behaviors dictated by brittle parsers or closed domain functional programs, and is thus more applicable to real-world settings.

## D.2 AUGMENTED CONVOLUTIONAL NEURAL NETWORKS

Alternative approaches for the CLEVR task that do not rely on the provided programs as a strong supervision signal are Santoro et al. (2017) and Perez et al. (2017). Both complement standard multi-layer Convolutional Neural Networks (CNNs) with components that aid them in handling compositional and relational questions.

**Relation Networks.** Santoro et al. (2017) appends a Relation Network (RN) layer to the CNN. This layer inspects all pairs of pixels in the image, thereby enhancing the network capacity to reason over binary relations between objects. While this approach is very simple and elegant conceptually, it



suffers from quadratic computational complexity, in contrast to our approach, which is linear. But beyond that, closer inspection reveals that this direct pairwise comparison might be unnecessary. Based on the analogy suggested by Santoro et al. (2017), according to which pixels are equivalent to objects and their pairwise interactions to relations, an RN layer attempts to grasp the induced graph between objects all at once in one shallow and broad layer. Conversely, our attention-based model proceeds in steps, iteratively comparing the image to a memory state that had aggregated information from the image in prior iterations. By the same analogy, MAC traverses a narrow and deep reasoning “path” that progressively follows transitive relations. Consequently, our model exhibits a relational capacity while circumventing the computational inefficiency.

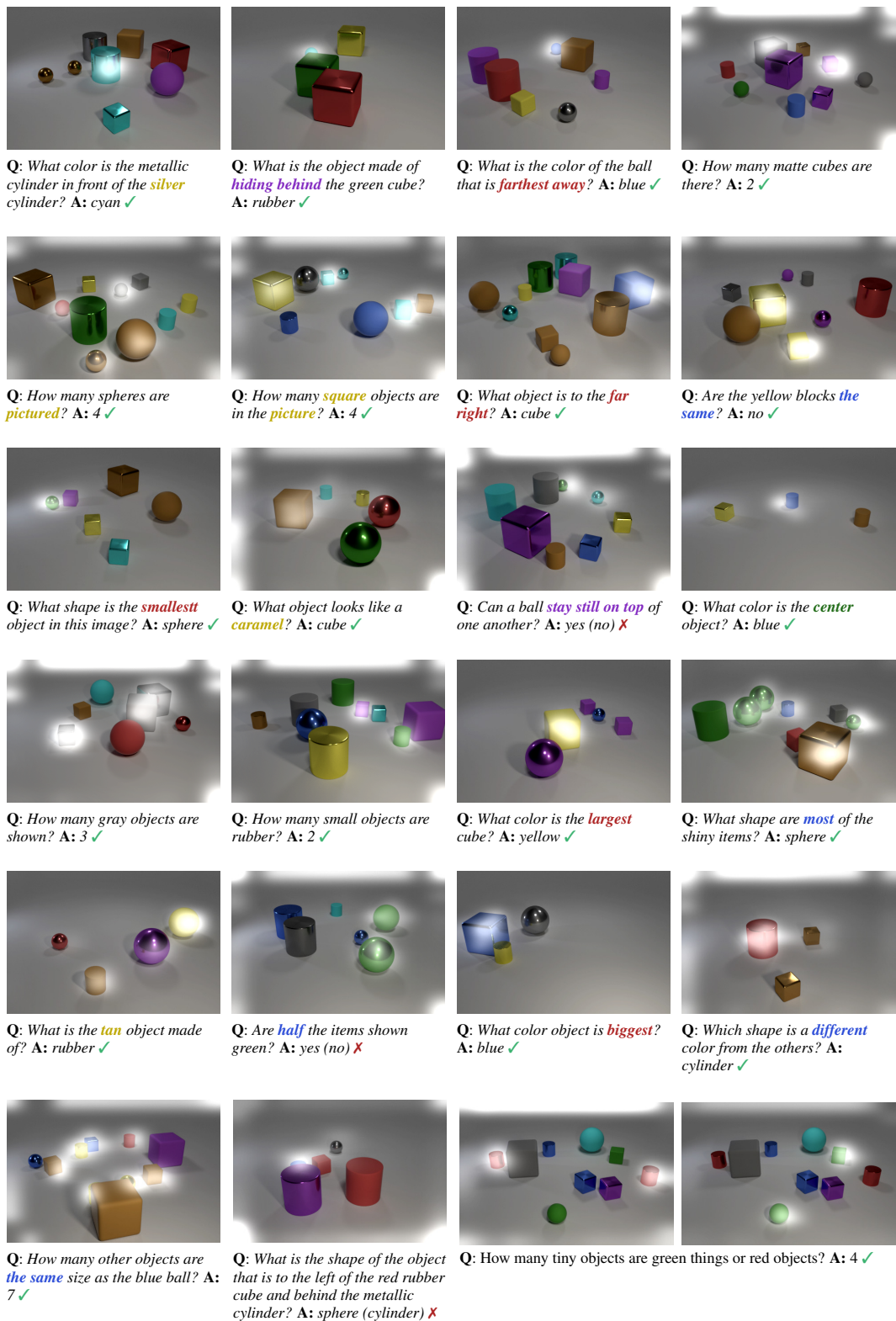
**FiLM.** Perez et al. (2017) proposes a model for visual reasoning that interleaves standard CNN layers with linear layers, reminiscent of layer normalization techniques (Ba et al., 2016; Ioffe & Szegedy, 2015). Each of these layers, called FiLM, is conditioned on the question, which is translated into matching bias and variance terms that tilt the layer’s activations to reflect the specifics of the given question, thus influencing the computation done over the image. Similarly to our model, this approach features distant modulation between the question and the image, where the former can affect the latter only through constrained means. However, since the same normalization is applied across all the activations homogeneously, agnostic to both their spatial location as well as their feature values, FiLM does not allow the question to differentiate between regions in the image based on their semantics – the objects or concepts they represent.

This stands in stark contrast to our attention-based model, which readily allows and actually encourages the question to inform the model about relevant regions to focus on. As supported by section 4, this more selective interaction between the question and the image facilitates learning and increases the model’s generalizability. Indeed, since attention is commonly used in models designed for standard VQA (Antol et al., 2015; Gupta, 2017; Lu et al., 2016; Yang et al., 2016), it is reasonable to assume that it would be beneficial to incorporate such methods into visual reasoning systems for the CLEVR task as well. In fact, attention mechanisms should be especially useful for multi-step reasoning questions such as those present in CLEVR. Such questions refer to several relations between different objects in the image and feature compositional structure that may be approached one step at a time. Thus, it should be beneficial for a cogent responder to have the capacity to selectively focus on one or some objects at each step, traversing the relevant relational links one after the other, both at the image level, and at the question level.

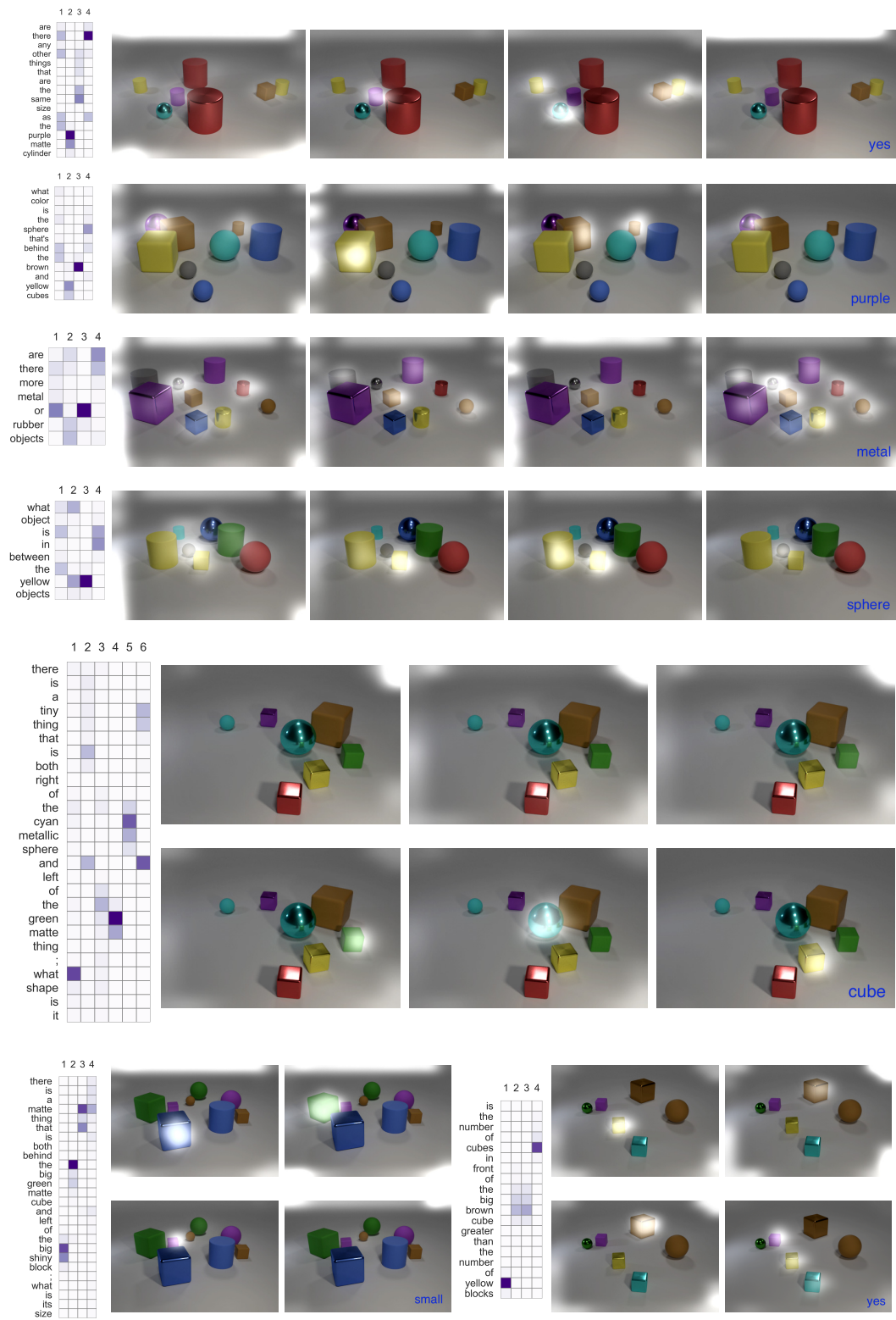
### D.3 MEMORY AND ATTENTION

Our architecture draws inspiration from recent research on mechanisms for neural memory and attention (Kumar et al., 2016; Xiong et al., 2016; Graves et al., 2014; 2016). Kumar et al. (2016) and Xiong et al. (2016) propose the Dynamic Memory Network (DMN) model that proceeds in an iterative process, attending to relevant information from a given knowledge base, which is then successively accumulated into the model’s memory state. However, the DMN views the question as one atomic unit, whereas our model decomposes it into a multi-step action plan that informs each cell of its specific objective. Another key difference is the distant interaction between the question and the knowledge base that characterizes our model. Conversely, DMN fuses their representations together into the same vector space.

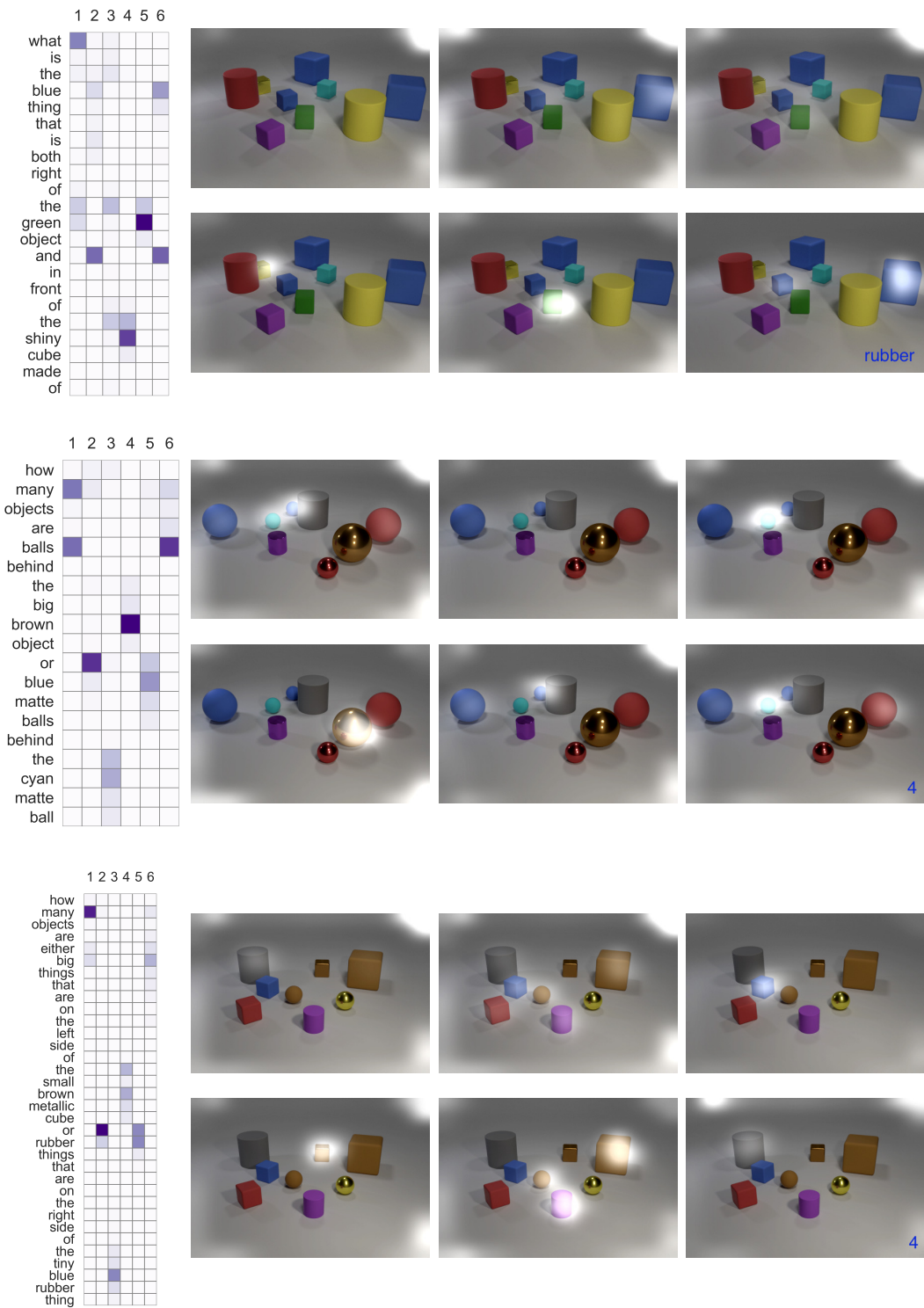
Graves et al. (2014; 2016) complement a neural network with an external memory it can interact with through the means of soft attention. Similarly to our approach, the model consists of a controller that performs read and write operations over a fixed-size memory array. However, in contrast to Graves et al. (2014; 2016), we employ a recurrent memory structure, where each MAC cell is associated with its own memory state. Rather than reading and writing iteratively into multiple slots in a shared memory resource, each cell creates a new memory, building upon the contents of the prior ones. This allows us to avoid potential issues of content blurring due to multiple global write operations, while still supporting the emergence of complex reasoning processes that progressively interact with preceding memories and intermediate results to accomplish the task at hand.



**Figure 16:** The first five rows show examples of the final attention map produced by the model for CLEVR-Human questions, demonstrating the ability of the model to perform novel reasoning skills and cope with new concepts that have not been introduced in CLEVR. These include in particular: **obstructions**, **object uniqueness**, **relative distances**, **superlatives** and **new terms**. The final row shows examples from CLEVR with object occlusions and summation.



**Figure 17:** Attention maps produced by MAC networks of lengths 4 and 6, providing evidence for the ability of the model to track transitive relations and perform logical operations. Note how the model tends to proceed from the end of the question backwards, tracking the relevant objects iteratively.



**Figure 18:** Attention maps produced by a MAC network of length 6, providing evidence for the ability of the model to track transitive relations, and perform logical operations, counting and summation. Note how the first iterations focus on the key structural question words “many” and “or” that serve as indicators for the model of the required reasoning operation it has to perform. Also note how the model correctly sums up two object groups in the second example, while correctly accounting for the intersection between them.