Transfer Learning in Visual and Relational Reasoning

T.S. Jayram* Vincent Marois† Tomasz Kornuta‡
Vincent Albouy* Emre Sevgen§ Ahmet S. Ozcan*

IBM Research AI

Abstract

Transfer learning is becoming the de facto solution for vision and text encoders in the front-end processing of machine learning solutions. Utilizing vast amounts of knowledge in pre-trained models and subsequent fine-tuning allows achieving better performance in domains where labeled data is limited. In this paper, we analyze the efficiency of transfer learning in visual reasoning by introducing a new model (SAMNet) and testing it on two datasets: COG and CLEVR. Our new model achieves state-of-the-art accuracy on COG and shows significantly better generalization capabilities compared to the baseline. We also formalize a taxonomy of transfer learning for visual reasoning around three axes: feature, temporal, and reasoning transfer. Based on extensive experimentation of transfer learning on each of the two datasets, we show the performance of the new model along each axis.

1 Introduction

In recent years, neural networks, being at the epicenter of the Deep Learning [**?**] revolution, became the dominant solutions across many domains, from Speech Recognition [**?**], Image Classification [**?**], Object Detection [**?**], to Question Answering [**?**] and Machine Translation [**?**] among others. At their core, being statistical models [**?**, **?**], neural networks rely on the assumption that training and testing samples are independent and identically distributed (*iid*); i.e. sampled from a common input space under similar data distribution characteristics. However, in many real-world scenarios, this assumption does not hold. Moreover, as modern neural models often have millions of trainable parameters, training them requires vast amounts of data, which for some domains (e.g., medical) can be very expensive and/or extremely difficult to collect. One of the widely used solutions for the above mentioned problems is Transfer Learning [**?**, **?**], a technique which enhances model performance by transferring *information* from one domain to another.

In Computer Vision, it is now standard practice to pretrain an image encoder (such as VGG [**?**] or ResNet [**?**]) on large-scale datasets (such as ImageNet [**?**]), and reuse the weights in unrelated domains and tasks, such as segmentation of cars [**?**] or Visual Question Answering (VQA) in a medical domain [**?**]. Such performance improvements are appealing, especially in cases where both the domain (natural vs. medical images) and the task (image classification vs. image segmentation vs VQA) change significantly.

Similar developments have emerged in the Natural Language Processing (NLP) community. Using shallow word embeddings, such as word2vec [**?**] or GloVe [**?**], pretrained on large corpuses from e.g. Wikipedia or Twitter, has become a standard procedure when working with different NLP domains and tasks. Recently,

*IBM Almaden Research Center, San Jose, CA, USA.
†IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA.
‡Currently at NVIDIA. *Santa Clara, CA, USA*.
§Currently at Citrine Informatics.

there is a clear, growing trend of utilization of deep contextualized word representations such as ELMo [?] (based on bidirectional LSTMs [?]) or BERT [?] (based on the Transformer [?] architecture), where entire deep networks (not just the input layer) are pretrained on very large corporas. In analogy to pretrained image encoders, the NLP community has also started to create model repositories, some with dozens of pretrained models ready to be downloaded and used. HuggingFace [?] is one of the most notable examples.

The success of transfer learning raises several research questions, such as the characteristics which make a dataset more favorable to be used in pretraining (notably ImageNet [?]), or regarding the observed performance correlation of models with different architectures between the source and target domains [?]. One of the most systematic works in this area is the computational taxonomic map for task transfer learning [?], which aimed at discovering the dependencies between twenty-six 2D, 2.5D, 3D, and semantic computer vision tasks.

In this work we focus on transfer learning in multi-modal tasks combining vision and language [?]. More precisely, we narrow the scope to transfer learning between visual reasoning tasks that have a "nice" logical structure, e.g., [?, ?, ?]. While models such as BERT and ResNet can be transferred efficiently in the same modality they were pretrained on, challenges arise once the modalities have been fused. For example, the CoGenT (Constrained Generalization Test) variant of the CLEVR dataset [?] contains two sets with similar questions, but differing on combinations of object-attribute values in images (Section 5.4). In this case, training on the first variant might yield entangled feature representations that may fail reasoning tasks on the second one. In video reasoning, an additional challenge in the temporal dimension is whether a model trained on shorter video sequences will transfer over to longer ones, e.g., the Canonical and the Hard variants of the COG dataset [?] (Section 5.3). To address these challenges, mechanisms such as attention [?] and external memory [?, ?, ?] which facilitate higher-level abstractions, seem more promising.

Motivated by these considerations:

1. We propose a new model, called SAMNet (Selective Attention Memory Network), which achieves state-of-the-art results on COG [?], a Video QA reasoning dataset.

2. We propose a taxonomy of transfer learning, inspired from [?], applied to the domain of visual reasoning. Articulated around 3 main axes, we illustrate it through the COG dataset, as well as the CLEVR [?] diagnostic dataset for Image QA.

3. Subsequently, we measure the impact of transferring the whole pretrained SAMNet model in the 3 proposed transfer learning settings: feature transfer, temporal transfer and reasoning transfer. This analysis is supported by an extensive set of experiments using the COG and CLEVR datasets, as well as their variants. Several of these experiments show significant transfer learning capabilities of SAMNet.

2 Related work

Visual Question Answering (VQA) [?, ?] is a challenging multimodal task that combines vision and language. Most Image Question Answering datasets and tasks focus on identifying object attributes, counting, and reasoning about their spatial-relations. Prior work generally relied on dense visual features produced by either CNNs [?, ?] or object detection modules [?], and increasingly, recent models utilize the relationships among objects to augment those features with contextual information from each object's surroundings [?, ?].

Another research focus area has been multimodal fusion and attention for VQA. For multimodal fusion, earlier methods used concatenation or element-wise multiplication between multimodal features [?, ?]. Others proposed more sophisticated methods such as different approximated bilinear pooling methods to effectively integrate the multimodal features with second-order feature interactions [?, ?]. Attention research in VQA has demonstrated that learning question-guided visual attention on image regions is the most promising approach [?, ?, ?]. Visual reasoning tasks such as the CLEVR dataset, also benefit from multi-hop

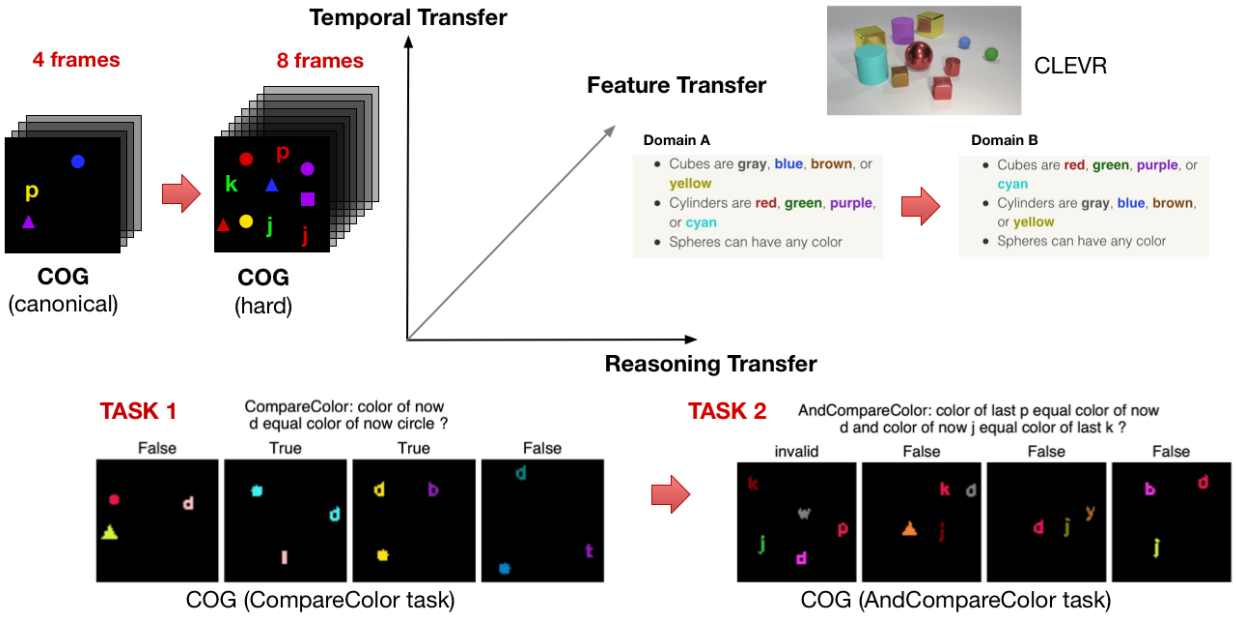


Figure 1: Transfer learning taxonomy.

attention and reasoning methods [**?**, **?**]. Visualization of attention maps also provides interpretability, which is an increasingly important aspect. The MAC model [**?**] is a great example of such an approach. The follow-on model by the same authors [**?**], called Neural State Machine (NSM) takes a slightly different approach and reason over graph structures rather than directly over spatial maps of visual features. Representing objects in a scene and their relationship as a graph is an obvious choice, which is another growing research direction [**?**, **?**, **?**].

Video question answering, aside from spatial queries, also focuses on questions that require *temporal* reasoning. Early works [**?**, **?**, **?**] used LSTMs to encode video frames and text queries to leverage temporal attention to selectively attend to essential frames in a video. These approaches might be sufficient for action-recognition type of tasks but fall short when spatio-temporal reasoning is required. Learning long-term dependencies is also another challenge that LSTMs may struggle with. Yin et al. [**?**] recently proposed a Memory-Augmented Neural Network (MANN) architecture for video QA which leverages the external memory for storing and retrieving useful information in questions and videos and modeling long-term visual-textual dependencies.

Our work is mostly related to the MAC [**?**] as well as neural networks with external memory [**?**, **?**] that support flexible addressing mechanisms including sequential and content-based addressing. An important distinction of our approach is the frame-by-frame processing of the video input. Prior studies typically divide the whole video into clips; e.g., in [**?**], the model extracts visual features from each frame and aggregates features first into clips, followed by aggregation over clips to form a single video representation. Nevertheless, when reasoning and producing the answer, the system has *access to all frames*. Similarly, in Visual Dialog [**?**], the system memorizes the whole dialog history. However, in real-time dialog or video monitoring, it is not always possible to keep the entire history of conversation nor all frames from the beginning of the recording.

3 Transfer Learning

We follow the notations in the survey by [**?**]. A *domain* is a pair $\mathcal{D} = (\mathcal{X}, P(X))$, where $\mathcal{X}$ is a feature space and $P(X)$ is a marginal probability distribution. For visual reasoning problems considered in this paper, $\mathcal{X}$ will consist of purely visual inputs, i.e., either images or videos in some cases, or a combination of both visual inputs and questions in other cases. A *task* is a pair $\mathcal{T} = (\mathcal{Y}, f(\cdot))$, where $\mathcal{Y}$ is a label space and $f : \mathcal{X} \to \mathcal{Y}$ is a prediction function. When the domain elements consist of both the question and the visual input, there is only one task, namely, to answer the question[1]. If the domain elements consist of just the visual inputs, then the task is defined by the question so that each question defines a separate task.

Definition ([**?**])**.** *Given a source domain $\mathcal{D}_S$ and a source learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and a target learning task $\mathcal{T}_T$, transfer learning aims to help improve the learning of the target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$, or $\mathcal{T}_S \neq \mathcal{T}_T$.*

In all our applications, $\mathcal{X}_S = \mathcal{X}_T$, so $\mathcal{D}_S \neq \mathcal{D}_T$ means that the marginal distributions P_S and P_T are different. Similarly, $\mathcal{T}_S \neq \mathcal{T}_T$ means that either $Y_S \neq Y_T$ or that the associated prediction functions are different.

Although Section 3 is quite general, it does not adequately capture all artifacts present in visual reasoning. For example, consider the transfer learning setting where the tasks $\mathcal{T}_S$ and $\mathcal{T}_T$ are the same but the marginal distributions P_S and P_T are different (referred to as *domain adaptation*). As mentioned in the introduction, one setting is the case of static images, where this could be due to having different feature combinations in the source and target. A different setting is in the context of video reasoning where the number of frames can increase significantly going from source to target. These require possibly very different methods: the first involves building disentangled feature representations that can generalize across domains; the second might need external memory to remember relevant objects to generalize across frame lengths. Another situation is when the questions themselves can be grouped into families such as count-based queries, comparison of objects, or existence of objects with certain features etc. This entails studying transfer learning between families of tasks which requires extending the above definition.

Broadly, we consider 3 kinds of transfer learning problems in this work, as illustrated in Figure 1. Let $\mathcal{Q}$ denote the set of questions and $\mathcal{V}$ denote the set of visual inputs.

Feature Transfer: In this setting of domain adaptation, $\mathcal{X}_S = \mathcal{X}_T \subseteq \mathcal{Q} \times \mathcal{V}$ and the task $f(q, v)$ is just the answer to the question q on visual input v. The output set $\mathcal{Y}$ is the union of legitimate answers over all questions in $\mathcal{Q}$. The marginal distributions P_S and P_T differ in the feature attributes such as shape, color, and size, or their combinations thereof.

Temporal Transfer: This setting is similar to attribute adaptation in that $\mathcal{X}_S = \mathcal{X}_T \subseteq \mathcal{Q} \times \mathcal{V}$ and there is a single task. The key difference is that we introduce a notion of complexity $C(v) = (n, m)$ for a visual input v, where n equals the maximum number of objects n in an image, and m equals the number of frames in a video. For any visual input v_S coming from $\mathcal{X}_S$ with $C(v_S) = (n_S, m_S)$ and for any visual input v_T coming from $\mathcal{X}_T$ with $C(v_T) = (n_T, m_T)$, we require that $n_T \geq n_S$ and $m_T \geq m_S$ with at least one inequality being a strict one. Thus, we necessarily increase the complexity of the visual input going from the source to the target domain.

Reasoning Transfer: This setting requires an extension of Section 3 above to investigate transfer learning when grouping questions into families. Let $\mathcal{V}$ be the feature space consisting of visual inputs only, shared by all tasks, with a common marginal distribution $P(X)$. For each question $q \in \mathcal{Q}$, we define the task

[1]For the COG dataset, the answer is a tuple, one for each frame in the video, whereas for typical video answering datasets, only a single answer is needed for the entire video.

$\mathcal{T}_q = (\mathcal{Y}_q, f_q(\cdot))$ where the output set $\mathcal{Y}_q$ is the set of legitimate answers to q and $f_q(v)$, for a visual input v, is the answer to question q on visual input v. Thus, tasks are in a 1-1 correspondence with questions. A *task family* is a probability distribution on tasks which in our case can be obtained by defining the distribution on $\mathcal{Q}$. Given a task family, the goal is to learn a prediction function that gives an answer to $f_q(v)$ for $v \in \mathcal{V}$ chosen according to the feature space distribution and q chosen according to the task probability distribution. Suppose $\mathcal{F}_S$ is the source task family and $\mathcal{F}_T$ is the target task family. Transfer learning aims to help improve the learning of the predictive function for the target task family using the knowledge in the source task family.

If labeled data is available for $\mathcal{X}_T$, a training algorithm distinction we make is between *zero-shot learning* and *finetuning*. Finetuning entails the use of labeled data in the target domain $\mathcal{D}_T$, foreseeing performance gain on the target task $\mathcal{X}_T$, after initial training on $\mathcal{X}_S$ and additional training on $\mathcal{X}_T$. Zero-shot learning thus refers to immediate test on $\mathcal{X}_T$ after initial training on $\mathcal{X}_S$.

4 Selective Attention Memory (SAM) Network

SAM Network (SAMNet for short) is an end-to-end differentiable recurrent model equipped with an external memory (Figure 2). At the conceptual level, SAMNet draws from two core ideas: iterative reasoning as proposed e.g. in MAC (Memory-Attention-Composition) Network [**?**, **?**] and use of an external memory, as in Memory-Augmented Neural Networks such as NTM (Neural Turing Machine) [**?**], DNC (Differentiable Neural Computer) [**?**] or DWM (Differentiable Working Memory) [**?**].

A distinctive feature of the SAM Network is its frame-by-frame temporal processing approach, where a single frame can be accessed at once. This is a notable difference from [**?**], which uses graph traversal reasoning. The recurrent nature of SAMNet does not prevent frame sequences longer than those used for training. The memory locations store relevant objects representing contextual information about words in text and visual objects extracted from video. Each location of the memory stores a d-dimensional vector. The memory can be accessed through either content-based addressing, via dot-product attention, or location-based addressing. Using gating mechanisms, correct objects can be retrieved in order to perform multi-step spatio-temporal reasoning over text and video. A notable feature of this design is that the number of addresses N can be changed between training and testing, to fit the data characteristics.

SAMNet's core is a recurrent cell called the SAM Cell. Unrolling a new series of T cells for each frame allows T steps of iterative reasoning, similar to [**?**]. Information flows between frames through the external memory. During the t-th reasoning step, for $t = 1, 2, \ldots, T$, SAM Cell maintains the following information as part of its recurrent state: (a) $\mathbf{c}_t \in \mathbb{R}^d$, the control state used to drive reasoning over objects in the frame and memory; and (b) $\mathbf{so}_t \in \mathbb{R}^d$, the summary visual object representing the relevant object for step t. Let $\mathbf{M}_t \in \mathbb{R}^{N \times d}$ denote the external memory with N slots at the end of step t. Let $\mathbf{wh}_t \in \mathbb{R}^N$ denote an attention vector over the memory locations; in a trained model, $\mathbf{wh}_t$ points to the location of the first empty slot in memory for adding new objects.

Question-driven Controller. This module drives attention over the question to produce k control states, one per reasoning operation. The control state $\mathbf{c}_t$ at step t is then fed to a *temporal classifier*, a two-layer feedforward network with ELU activation in the hidden layer of d units. The output $\boldsymbol{\tau}_t$ of the classifier is intended to represent the different temporal contexts (or lack thereof) associated with the word in focus for that reasoning step. For the COG dataset, we pick 4 classes to capture the terms labeled "last", "latest", "now", and "none".

The visual retrieval unit uses the information generated above to extract a relevant object $\mathbf{vo}_t$ from the frame. A similar operation on memory yields the object $\mathbf{mo}_t$. The memory operation is based on attention mechanism, and resembles content-based addressing. Therefore, we obtain an attention vector over memory addresses that we interpret to be the *read head*, denoted by $\mathbf{rh}_t$.

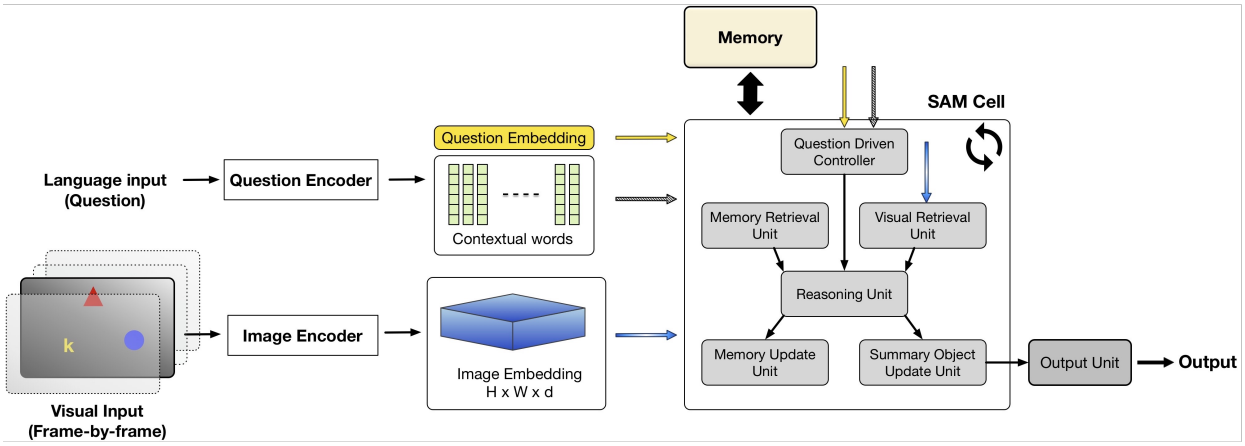


Figure 2: General architecture of SAMNet.

Reasoning Unit. This module is the backbone of SAMNet, which determines the gating operations to be performed on the external memory, as well as determining the correct object's location for reasoning. To resolve whether we have a valid object from the frame (and similarly for memory), we execute the following reasoning procedure. First, we compute a simple aggregate[2] of the visual attention vector $\mathbf{va}_t$ of dimension L (L denotes the number of feature vectors for the frame): $vs_t = \sum_{i=1}^{L}[\mathbf{va}_t(i)]^2$. It can be shown that the more localized the attention vector, the higher the aggregate value. We perform a similar computation on the read head $\mathbf{rh}_t$ over memory locations. We input these two values, along with the temporal class weights $\boldsymbol{\tau}_t$, into a 3-layer feedforward classifier with hidden ELU units to extract 4 gating values (in $[0, 1]$) for the current reasoning step: (a) g_t^{v}, which determines whether there is a valid visual object; (b) g_t^{m}, which determines whether there is a valid memory object. (c) h_t^{r}, which determines whether the memory should be updated by replacing a previously stored object with a new one; and (d) h_t^{a}, which determines whether a new object should be added to memory. We stress that the network has to learn via training how to correctly implement these behaviors.

Memory Update Unit. The unit first determines the memory location where an object could be added:

$$\mathbf{w}_t = h^{\mathrm{r}} \cdot \mathbf{rh}_t + h^{\mathrm{a}} \cdot \mathbf{wh}_{t-1}$$

Above, $\mathbf{w}_t$ denotes the pseudo-attention vector that represents the "location" where the memory update should happen. $\mathbf{w}_t$ sums up to at most 1 and can be zero, indicating in this case there is no need adding a new object nor replacing an existing object. We then update the memory accordingly as:

$$\mathbf{M}_t = \mathbf{M}_{t-1} \odot (\mathbf{J} - \mathbf{w}_t \otimes \mathbf{1}) + \mathbf{w}_t \otimes \mathbf{vo}_t,$$

where $\mathbf{vo}_t$ denotes the object returned by the visual retrieval unit. Here $\mathbf{J}$ denotes the all ones matrix, $\odot$ denotes the Hadamard product and $\otimes$ denotes the Kronecker product. Note that the memory is unchanged in the case where $\mathbf{w}_t = 0$, i.e., $\mathbf{M}_t = \mathbf{M}_{t-1}$. We finally update the write head so that it points to the succeeding address if an object was added to memory or otherwise stay the same. Let $\mathbf{wh}'_{t-1}$ denote the circular shift to the right of $\mathbf{wh}_{t-1}$ which corresponds to the soft version of the head update. Then:

$$\mathbf{wh}_t = h^{\mathrm{a}} \cdot \mathbf{wh}'_{t-1} + (1 - h^{\mathrm{a}}) \cdot \mathbf{wh}_{t-1}$$

Summary Update Unit. This unit updates the (recurrent) summary object to equal the outcome of the t-th reasoning step. We first determine whether the relevant object $\mathbf{ro}_t$ should be obtained from memory or the

[2]This is closely related to Rényi entropy and Tsallis entropy of order 2.

frame according to:

$$\mathbf{ro}_t = g_t^{\mathrm{v}} \cdot \mathbf{vo}_t + g_t^{\mathrm{m}} \cdot \mathbf{mo}_t$$

Note that $\mathbf{ro}_t$ is allowed to be a null object (i.e. 0 vector) in case neither of the gates evaluate to true. Finally, $\mathbf{so}_t$ is the output of a linear layer whose inputs are $\mathbf{ro}_t$ and the previous summary object $\mathbf{so}_{t-1}$. This serves to retain additional information in $\mathbf{so}_{t-1}$, e.g., if it held the partial result of a complex query with Boolean connectives.

5 Experiments

We implemented and trained our SAMNet model using MI-Prometheus [**?**], a framework based on Pytorch [**?**]. We evaluated the model on the COG dataset [**?**], a video reasoning [**?**] dataset developed for the purpose of research on relational and temporal reasoning, as well as on the CLEVR dataset [**?**], created for Image Question Answering research. There are 23 classification question categories in COG and the 5 original question categories in CLEVR. Our experiments were designed to study SAMNet's performance in terms of generalization and transfer learning abilities in different settings, and involved incorporating variants of both datasets. For COG, the Canonical (easy) and Hard variant differ mainly on the number of frames in the video, the maximum amount of look-back in frame history containing relevant information for reasoning, and the number of object distractors (see Table 1). For CLEVR, we consider the CoGenT (Constrained Generalization Test) variant which contains two conditions, differing on the combinations of attribute values (details in Section 5.3).

Variant	Frames	History	Distractors
Canonical (Easy)	4	3	1
Hard	8	7	10

Table 1: Details of the Canonical and Hard variants of COG.

More details on the composition of these datasets is available in Appendix A.

5.1 Performance of SAMNet vs baseline model

We trained SAMNet using 8 reasoning steps and external memory of 8 address locations, each storing an array of 128 floats. We compared our results with the baseline model introduced in the same paper as the COG dataset [**?**]. The most important results are highlighted in Figure 3; full comparison can be found in Appendix B.

For the Canonical variant (top row), we achieve similar accuracies for the majority of tasks (with a total average accuracy of 98.0%, compared to 97.6% for the baseline model), with significant improvements (around 13 points) for *AndCompare* tasks. As these tasks focus on compositional questions referring to two objects, we hypothesize that our model achieves better accuracy due to its ability to selectively pick and store relevant objects from the past frames in memory. For the Hard variant, we achieve a total average accuracy of 96.1% compared to 80.1% for the baseline model, demonstrating that our model can adapt to larger number of frames and distractors. Despite there being some tasks in the Canonical variant where SAMNet achieved slightly lower accuracies, when comparing performances on the Hard variant, it improves upon the baseline model on all tasks, with improvements varying from 0.5 to more than 30 points, and especially on the more complex tasks in the dataset.

Appendix B also includes the task-by-task performance of "SoftPaths" model [**?**] on the Canonical variant of the COG dataset, on which it achieves on overall accuracy of 96.1%. This model is however not

recurrent, therefore, it is not immediately applicable for temporal transfer learning—in contrast to SAMNet or the baseline model of [**?**]—but we include it here for completeness.

5.2 Reasoning transfer on CLEVR and COG

In these experiments using the CLEVR and COG datasets, we focus on analyzing the impact of each task relative to others using appropriate groupings of tasks.

5.2.1 CLEVR

For the CLEVR dataset, we consider the question categories defined by the authors: *Exist*, *Count*, *CompareInteger*, *CompareAttribute*, *QueryAttribute*. For performance reasons, we deactivated the external memory and temporally-related modules in SAMNet as they are unnecessary while reasoning about static images. We conduct the following experiments:

- Train and test SAMNet on a single task group t. These 5 experiments fit into the traditional ML setup of single-task learning;
- Train SAMNet on all task groups jointly and evaluate its performance on each task group t separately. This is a transfer learning setting where for the source task family, the task is sampled from all questions, while for the target task family, the samples consist of questions from group t only;
- Finally, for each task group t, we train SAMNet on all tasks but t, and test its performance on t. This can also be viewed as a transfer learning setting similar to the previous case.

Noticeable results are shown in Figure 4, while the complete set is available in Appendix C.

Looking at Figure 4, SAMNet does well on *Count* and *QueryAttribute*, but poorer on the 3 other tasks in the single-task learning setting (blue). Indeed, *Exist*, *CompareInteger* and *CompareAttribute* are binary tasks; *Count* has output labels digits 0 through 10 (so $<$10% accuracy by chance) and *QueryAttribute* maps to the set of object-attribute values (15 labels).

Nevertheless, significant accuracy gains are noted when training jointly on all tasks (gray), ranging from 18 points to 37 points on 4 out of 5 tasks. These improvements suggest that related tasks benefit from joint training. *QueryAttribute* only sees an increase of one point. One could qualify it as *self-sufficient* as it does not appear to benefit from joint training with other tasks.

Finally, the "all-tasks-but-t" experiments (yellow) demonstrate that while tasks are related, one does not subsume another in terms of learning. Indeed, we can observe that for *CompareAttribute*, while *Exist* and *CompareInteger* share the same output space, including them and holding out *CompareAttribute* from the training set results in poor accuracy. We also observe no learning for *Count* and *QueryAttribute*. As these categories have labels that do not overlap with other categories, the model cannot predict these labels.

An additional set of experiments, for which results are available in Appendix C, fine-tune the model trained on all tasks on each task t respectively. Fine-tuning did not demonstrate a clear benefit (except for *Count*, where the accuracy increased by 1.5 pt) without hurting performance on the other tasks. Nevertheless, these experiments leave open the possibility that joint training of tasks may potentially benefit from using weighted sampling towards the tail end with more emphasis on samples from less performing task groups, similar to [**?**, **?**].

5.2.2 COG

Since the number of task classes (=23) in the COG dataset is large, we designed a 2-level hierarchy of task groups using the description of these tasks, as shown in Figure 5.

For groups at the lowest level, we chose the following task classes to be placed in those groups. Below, substitute each of *Shape* and *Color* for X to obtain the task class.

Basic: *Exist*X, *Get*X and *Exist*;

Obj-Attr: *SimpleCompare*X and *AndSimpleCompare*X;

Compare: *Compare*X, *AndCompare*X & *Exist*X*Of*;

Spatial: *ExistSpace*, *Exist*X*Space*, and *Get*X*Space*;

Cognitive: *ExistLastColorSameShape*, *ExistLastShapeSameColor* and *ExistLastObjectSameObject*

We then conducted the following experiments using the Canonical variant of COG to study whether transfer learning was effective in leveraging information gained by training a task family at a higher level of the hierarchy:

- Train and test SAMNet on each of the 5 task groups at the lowest level of the hierarchy (Single-task training);
- Jointly train on:
  - **Group A** and test on each task from the lowest groups (i.e. **Basic**, **Obj-Attr** and **Compare**) separately;
  - **Group B** and test separately on **Spatial** and **Cognitive**;
- As a baseline, we compared the above results to the earlier experiment shown in Figure 3, which can be viewed as training jointly on **All** and testing on each group at the leaf level separately (All-task training).

The results of these experiments are shown in Figure 6.

First, notice that for each of the **Basic** and **Obj-Attr** task families, the accuracy is near-perfect in all cases, suggesting that each contains the most primitive tasks and therefore do not benefit from training with other task families. With **Spatial**, we see a small improvement showing that there is some benefit due to joint training with other task families. Two groups that demonstrated a huge improvement of more than 25 points are **Compare** and **Cognitive**. The former saw an accuracy jump from 68.6% by training on samples from that family alone to 97.0% when training on all samples. To further emphasize this behavior, notice that just joining **Compare** with **Obj-Attr** and **Basic** already causes a significant accuracy jump to 92.3%. In hindsight, this is not surprising, as the questions in **Compare** are composed of fragments of questions given by **Basic** and **Obj-Attr**, and therefore can leverage the reasoning strategies developed there to reason about questions in **Compare**. Lastly, for the **Spatial** family, we again see the benefits of joint training with all questions (68.6% to 95.7%) but in this case there is a slight loss incurred by including everything. As seen in the figure, just jointly training with **Spatial** alone is sufficient to get a boost in accuracy (97.1%). To summarize, while joint training helps, one needs to determine how much of correlation is present with the other tasks.

5.3 Temporal transfer in COG

The goal here is to test the transfer learning ability concerning the frame sequence length, frame history required for reasoning, and the number of object distractors. For that purpose, we compare both models when trained on the Canonical variant but tested on the Hard variant (Figure 7). The light gray color indicates original accuracies of the baseline model from paper, whereas dark gray indicates accuracies of the baseline model obtained by running the original code provided by the authors [**?**].

The first column displays the scores of the traditional ML setup when training and testing on the Canonical variant. The observed close scores in light and dark gray underscore the baseline model reproducibility. For both cases of zero-shot learning (second column–91.6% vs 65.9%) and fine-tuning using a single epoch (third column–96.7% vs. 78.1%), SAMNET outperforms the baseline model significantly. Interestingly, this fine-tuning yields a mild boost of +0.6% on the earlier reported accuracy in Section 5.1 (fourth column). These results suggest that it suffices to train SAMNet on simpler videos to enable learning of good memory usage and attention on relevant entities in order to achieve comparable, if not better, performance on longer video frames with more complex scenes.

5.4 Feature transfer on CLEVR-CoGenT

The final set of experiments we consider is related to feature transfer. The CoGenT-A & -B variants of CLEVR differ by the available combinations of 8 object color-attributes. The colors are partitioned into two complementary families: Gray, Blue, Brown and Yellow in **Family A** and Red, Green, Purple, Cyan in **Family B**. The cubes and cylinders take colors from complementary families in each variant with opposite configurations while the spheres can take any color (See Table 2). As the input domain consist of the set of objects with all attribute values, both variants differ by their marginal distributions P_S and P_T.

Dataset	Cubes	Cylinders	Spheres
CoGenT A	Family A	Family B	Any color
CoGenT B	Family B	Family A	Any color

Table 2: Restrictions on feature combinations in A & B conditions of the CoGenT variant of the CLEVR dataset.

We conduct 2 experiments with SAMNet trained on CoGenT-A:

- an immediate test (zero-shot learning) from A to B; and
- fine-tuning for a single epoch on 30k samples from CoGenT-B (following [**?**, **?**, **?**, **?**]).

Performance of SAMNet on CoGenT (Figure 8) is clearly worse on CoGenT-B than CoGenT-A. Fine-tuning for a single epoch allows an observable increase of 15 pts on CoGenT-B, and a slight drop on CoGenT-A. Both are consistent with findings from the literature [**?**, **?**, **?**].

6 Summary

In this paper, we quantified the impact of Transfer Learning on Visual Reasoning. We have proposed a new taxonomy of transfer learning for Visual Reasoning, articulated around three axes: feature, temporal and reasoning transfer. We have also introduced a novel Memory-Augmented Neural Network model called SAMNet. SAMNet, designed to learn to reason over sequences of frames, shows significant improvements over SOTA models on the COG dataset for Video Reasoning and achieves comparable performance on the CLEVR dataset for Image Reasoning.

Taking that as a starting point and leveraging the proposed taxonomy, we have conducted an extensive set of experiments, focusing on the impact of transfer learning in areas that might be considered as higher-level reasoning. SAMNet demonstrates very good generalization capabilities along certain axes and, through the cautious use of fine-tuning, can see its performance advanced even further.

Finally, we note that some of the proposed tasks (e.g. Train on all but *t*) are complementary to ones already well established in the literature, e.g. in Taskonomy [**?**]. We hope these contributions will bolster

new research directions to acutely apply transfer learning to areas in need of data and continue exploring the conceivable performance improvements.

A Datasets: CLEVR-CoGenT and COG

The CoGenT dataset [**?**] contains:

– Training set of 70,000 images and 699,960 questions in Condition A,

– Validation set of 15,000 images and 149,991 questions in Condition A,

– Test set of 15,000 images and 149,980 questions in Condition A (without answers),

– Validation set of 15,000 images and 150,000 questions in Condition B,

– Test set of 15,000 images and 149,992 questions in Condition B (without answers),

– Scene graphs and functional programs for all training/validation images/questions.

The combinations of attributes in CoGenT-A and CoGenT-B are shown in Table 3.

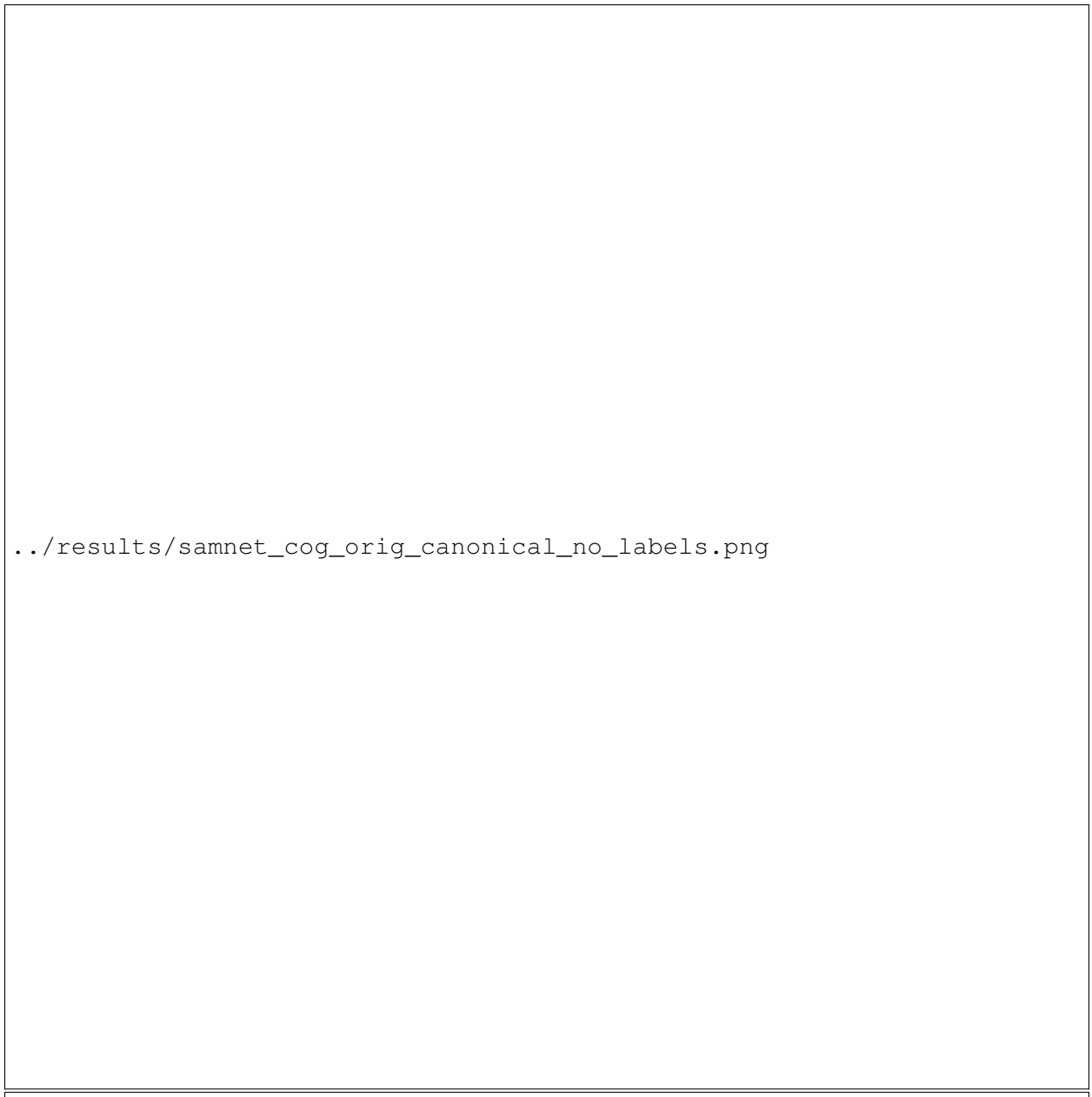
Dataset	Cubes	Cylinders	Spheres
CoGenT-A	Gray / Blue / Brown / Yellow	Red / Green / Purple / Cyan	Any color
CoGenT-B	Red / Green / Purple / Cyan	Gray / Blue / Brown / Yellow	Any color

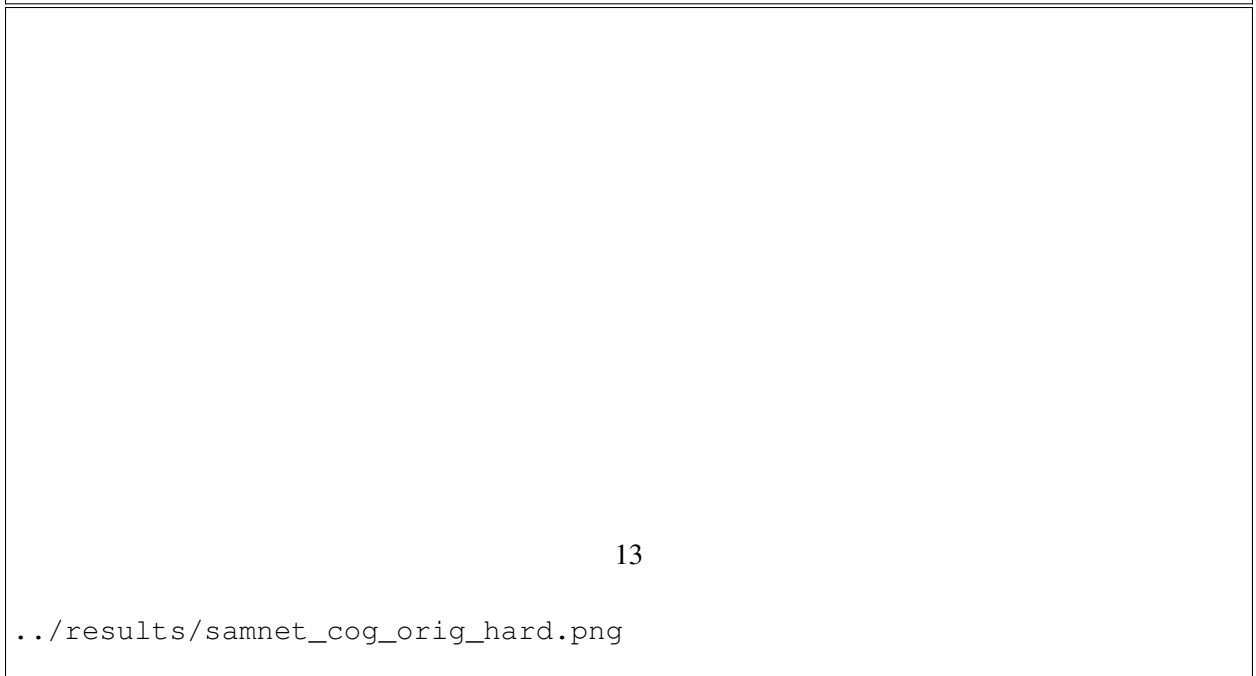
Table 3: Colors & shapes combinations present in CoGenT-A & -B datasets.

The Canonical and Hard variants of the COG dataset [**?**] are contrasted in Table 4.

Variant	number of frames	maximum memory duration	maximum number of distractors	size of training set	size of validation set	size of test set
Canonical	4	3	1	10000320	500016	500016
Hard	8	7	10	10000320	500016	500016

Table 4: Details of the Canonical and Hard variants of the COG dataset.


../results/samnet_cog_orig_canonical_no_labels.png


../results/samnet_cog_orig_hard.png

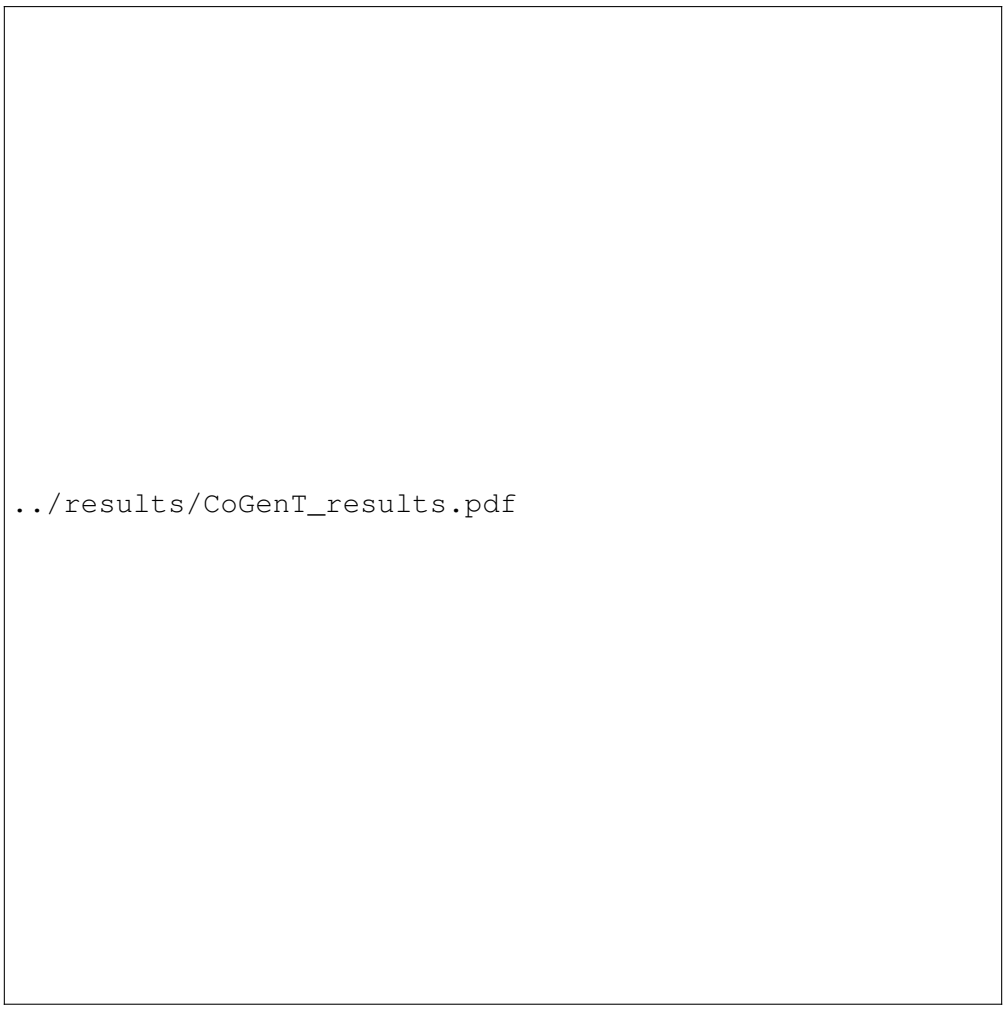


Figure 4: CLEVR-CoGenT accuracies for all tasks t when training on t only, training on all tasks jointly and training on all tasks but t.

Figure 5: Hierarchy of Task Groups.

Figure 6: COG accuracies for all task groups t when training on t only; training on Group A or B; and on all tasks.

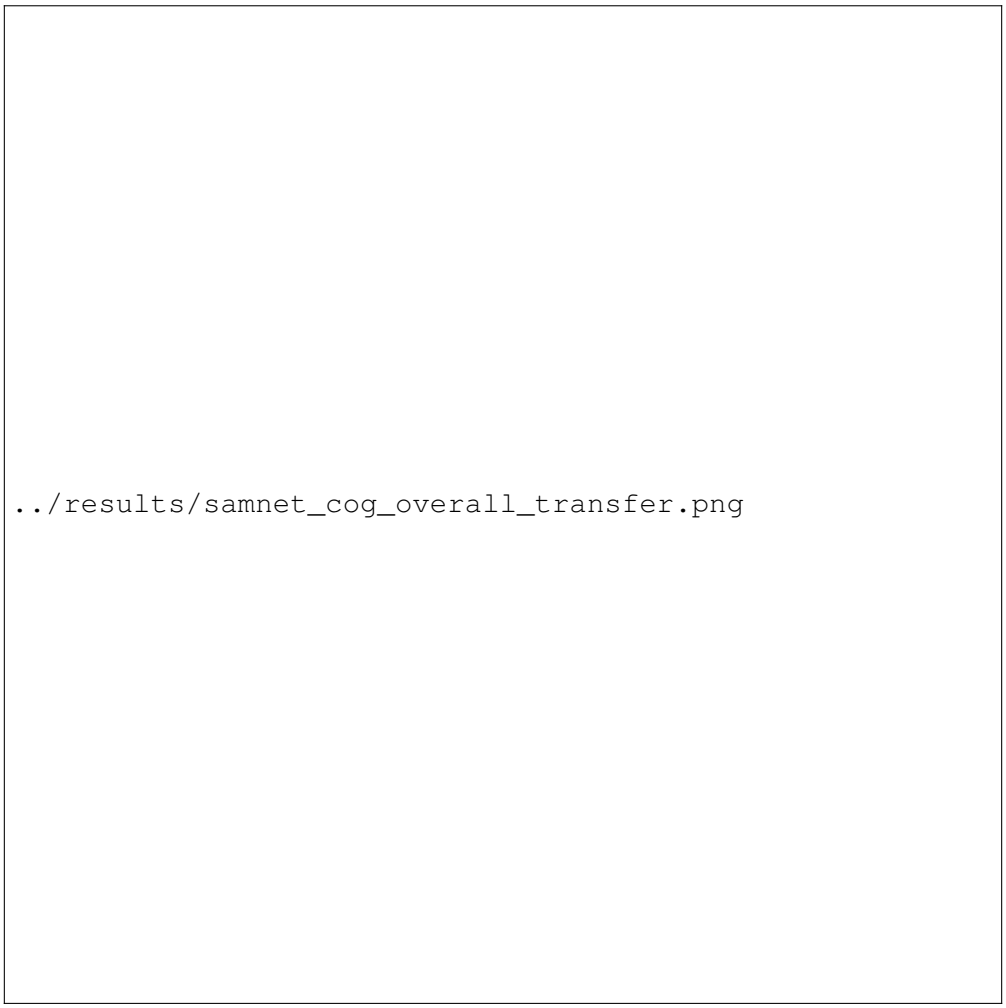


Figure 7: Total accuracies of SAMNet (blue) and baseline models (light/dark gray) when testing generalization from Canonical to Hard variants of the dataset.

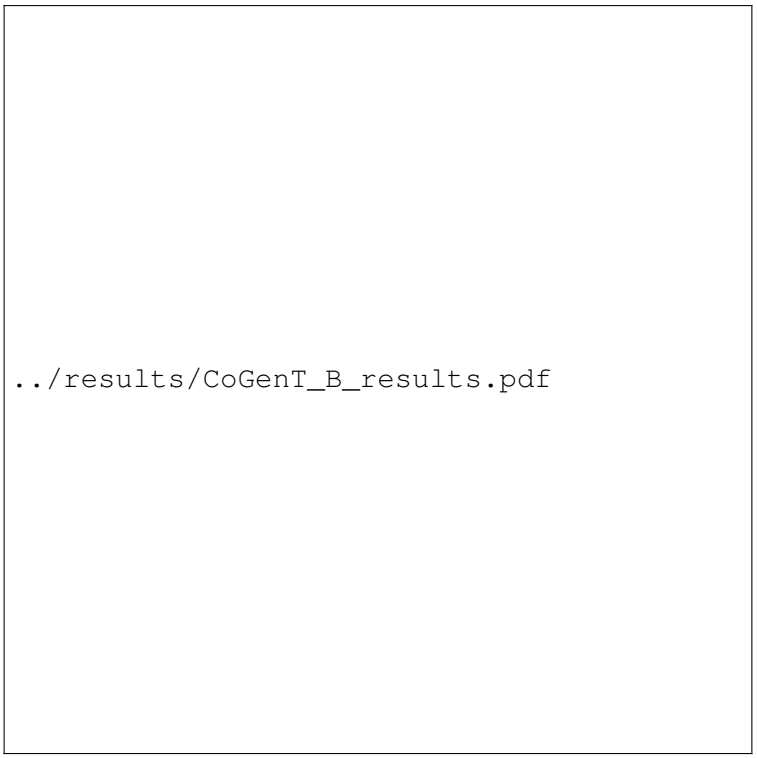


Figure 8: Test accuracy on CoGenT-A & -B when training on CoGenT-A (blue) and fine-tuning on CoGenT-B (gray).

B Complete COG results

Model	SAMNet				Baseline Model				SoftPaths
					paper	code	code	paper	
Trained on	canonical	canonical	canonical	hard	canonical	canonical	canonical	hard	canonical
Fine tuned on	-	-	hard	-	-	-	hard	-	-
Tested on	canonical	hard	hard	hard	canonical	hard	hard	hard	canonical
Overall accuracy	98.0	91.6	96.5	96.1	97.6	65.9	78.1	80.1	96.1
AndCompareColor	93.5	82.7	89.2	80.6	81.9	57.1	60.7	51.4	77.9
AndCompareShape	93.2	83.7	89.7	80.1	80.0	53.1	50.3	50.7	77.0
AndSimpleCompareColor	99.2	85.3	97.6	99.4	99.7	53.4	77.1	78.2	92.9
AndSimpleCompareShape	99.2	85.8	97.6	99.2	100.0	56.7	79.3	77.9	93.4
CompareColor	98.1	89.3	95.9	99.7	99.2	56.1	67.9	50.1	95.7
CompareShape	98.0	89.7	95.9	99.2	99.4	66.8	65.4	50.5	95.1
Exist	100.0	99.7	99.8	99.8	100.0	63.5	96.1	99.3	98.6
ExistColor	100.0	99.6	99.9	99.9	99.0	70.9	99.0	89.8	100.0
ExistColorOf	99.9	95.5	99.7	99.8	99.7	51.5	76.1	73.1	99.7
ExistColorSpace	94.1	88.8	91.0	90.8	98.9	72.8	77.3	89.2	97.8
ExistLastColorSameShape	99.5	99.4	99.4	98.0	100.0	65.0	62.5	50.4	100.0
ExistLastObjectSameObject	97.3	97.5	97.7	97.5	98.0	77.5	61.7	60.2	100.0
ExistLastShapeSameColor	98.2	98.5	98.8	97.5	100.0	87.8	60.4	50.3	100.0
ExistShape	100.0	99.5	100.0	100.0	100.0	77.1	98.2	92.5	100.0
ExistShapeOf	99.4	95.9	99.2	99.2	100.0	52.7	74.7	72.7	99.8
ExistShapeSpace	93.4	87.5	91.1	90.5	97.7	70.0	89.8	89.8	98.0
ExistSpace	95.3	89.7	93.2	93.3	98.9	71.1	88.1	92.8	98.6
GetColor	100.0	95.8	99.9	100.0	100.0	71.4	83.1	97.9	100.0
GetColorSpace	98.0	90.0	95.0	95.4	98.2	71.8	73.0	92.3	98.3
GetShape	100.0	97.3	99.9	99.9	100.0	83.5	89.2	97.1	100.0
GetShapeSpace	97.5	89.4	93.9	94.3	98.1	78.7	77.3	90.3	98.3
SimpleCompareShape	99.9	91.4	99.7	99.9	100.0	67.7	96.7	99.3	94.5
SimpleCompareColor	100.0	91.6	99.8	99.9	100.0	64.2	90.4	99.3	94.1

Table 5: COG test set accuracies for SAMNet, the baseline model [**?**] and the "SoftPaths" model [**?**]. For the baseline model, 'paper' denotes results reproduced from [**?**] along with some further clarifications by the authors (private communication) regarding the performance on individual task types while 'code' denotes results of our experiments using their implementation [**?**]. Similarly, the performance of the "SoftPaths" model on individual task types was obtained from the authors of that paper upon our request.

C Complete CLEVR-CoGenT results

Experiments	Test Accuracy (%) on valA – 150,000 samples					
	Overall	**Exist**	**Count**	**CompareInteger**	**CompareAttribute**	**QueryAttribute**
Exist only	26.07	**74.20**	0.0	59.79	59.15	0.0
Count only	14.89	0.0	**62.78**	0.0	0.0	0.0
CompareInteger only	23.44	48.15	0.0	**77.98**	54.08	0.0
CompareAttribute only	23.50	51.93	0.0	59.06	**61.73**	0.0
QueryAttribute only	34.64	0.0	0.0	0.0	0.0	**97.08**
All tasks	95.32	98.4	86.75	96.0	98.65	98.02
All tasks but Exist	90.33	**60.42**	86.12	96.18	98.52	98.60
All tasks but Count	74.59	97.51	**0.0**	94.37	98.42	98.47
All tasks but CompareInteger	91.53	98.22	86.09	**56.35**	98.78	98.40
All tasks but CompareAttribute	81.92	98.32	86.36	95.86	**23.18**	98.44
All tasks but QueryAttribute	42.17	76.79	54.64	79.87	64.13	**0.0**
Trained on all tasks – Finetune on t						
Exist	94.16	**98.04**	82.96	95.02	98.12	97.97
Count	95.10	96.46	**88.28**	94.78	97.01	98.26
CompareInteger	95.31	98.39	86.56	**96.07**	98.70	98.09
CompareAttribute	95.31	98.40	86.78	96.00	**98.66**	98.04
QueryAttribute	93.49	97.45	82.09	92.23	97.85	**97.76**

Table 6: Complete set of results for SAMNet on CLEVR-CoGenT.