CENTER FOR Brains Minds+ Machines

CBMM Memo No. 121 **February 3, 2022**

Transformer Module Networks for Systematic Generalization in Visual Question Answering

Moyuru Yamada[1], **Vanessa D'Amario**[2,3,4], **Kentaro Takemoto**[1], **Xavier Boix**[3,4*] **and Tomotake Sasaki**[1,4*]

[1]Fujitsu Limited, Kawasaki, Kanagawa, Japan
[2]Fujitsu Research of America, Inc., Sunnyvale, CA, USA
[3]Massachusetts Institute of Technology, Cambridge, MA, USA
[4]Center for Brains, Minds and Machines, Cambridge, MA, USA
yamada.moyuru@fujitsu.com, vanessad@mit.edu, k.takemoto@fujitsu.com, xboix@mit.edu, tomotake.sasaki@fujitsu.com
[*]Equally contributed

Abstract

Transformer-based models achieve great performance on Visual Question Answering (VQA). However, when we evaluate them on systematic generalization, i.e., handling novel combinations of known concepts, their performance degrades. Neural Module Networks (NMNs) are a promising approach for systematic generalization that consists on composing modules, i.e., neural networks that tackle a sub-task. Inspired by Transformers and NMNs, we propose Transformer Module Network (TMN), a novel Transformer-based model for VQA that dynamically composes modules into a question-specific Transformer network. TMNs achieve state-of-the-art systematic generalization performance in three VQA datasets, namely, CLEVR-CoGenT, CLOSURE and GQA-SGL, in some cases improving more than 30% over standard Transformers.

This material is based upon work supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216.

Transformer Module Networks for Systematic Generalization in Visual Question Answering

Moyuru Yamada[1] , **Vanessa D'Amario**[2,3,4] , **Kentaro Takemoto**[1] ,
Xavier Boix[3,4*] and **Tomotake Sasaki**[1,4*]
[1]Fujitsu Limited, Kawasaki, Kanagawa, Japan
[2]Fujitsu Research of America, Inc., Sunnyvale, CA, USA
[3]Massachusetts Institute of Technology, Cambridge, MA, USA
[4]Center for Brains, Minds and Machines, Cambridge, MA, USA
yamada.moyuru@fujitsu.com, vanessad@mit.edu, k.takemoto@fujitsu.com,
xboix@mit.edu, tomotake.sasaki@fujitsu.com
[*]Equally contributed

Abstract

Transformer-based models achieve great performance on Visual Question Answering (VQA). However, when we evaluate them on systematic generalization, i.e., handling novel combinations of known concepts, their performance degrades. Neural Module Networks (NMNs) are a promising approach for systematic generalization that consists on composing modules, i.e., neural networks that tackle a sub-task. Inspired by Transformers and NMNs, we propose Transformer Module Network (TMN), a novel Transformer-based model for VQA that dynamically composes modules into a question-specific Transformer network. TMNs achieve state-of-the-art systematic generalization performance in three VQA datasets, namely, CLEVR-CoGenT, CLOSURE and GQA-SGL, in some cases improving more than 30% over standard Transformers.

1 Introduction

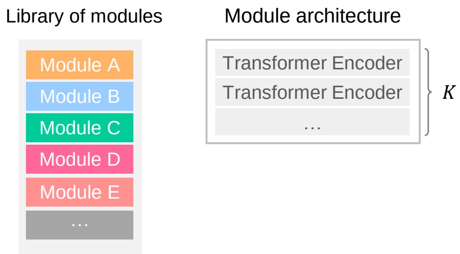
Visual Question Answering (VQA) aims to answer questions about an image [Andreas *et al.*, 2016], and it has become the *de facto* testbed to assess the capability of learning machines to perform complex visual reasoning. The compositional structure inherent to visual reasoning is at the core of VQA—visual reasoning is a composition of visual sub-tasks, and also, visual scenes are compositions of objects, composed of attributes such as textures, shapes and colors. This compositional structure of VQA renders a distribution of question-image pairs that is of combinatorial size and cannot be fully reflected by training datasets.

Systematic generalization is the ability to generalize to novel compositions of known concepts [Lake and Baroni, 2018; Bahdanau *et al.*, 2019; Ruis *et al.*, 2020], and it is a hallmark of intelligence. A learning machine capable of systematic generalization is still a distant goal, which contrasts with the exquisite ability of current learning machines to generalize in-distribution (i.e., unseen question-image pairs from the training distribution). In fact, the most successful learning machines, i.e., Transformer-based models, have been tremendously effective for VQA when evaluated in-distribution [Tan and Bansal, 2019; Li *et al.*, 2020]. Yet, recent studies stressed the need to evaluate them for systematic generalization instead of in-distribution generalization [Gontier *et al.*, 2020; Tsarkov *et al.*, 2021; Bergen *et al.*, 2021], as the learning mechanisms that enable systematic generalization remain as an outstanding question at the heart of science of intelligence.

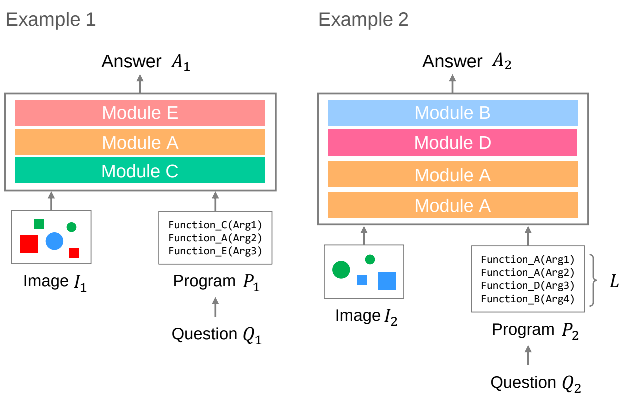
A recent, promising strand of research investigates Neural Module Networks (NMNs) for systematic generalization in VQA [Bahdanau *et al.*, 2019; Bahdanau *et al.*, 2020; D'Amario *et al.*, 2021]. NMNs decompose a question in VQA into sub-tasks, and each sub-task is tackled with a shallow neural network called *module*. Modules are composed into a question-specific network. NMNs alleviate the gap between in-distribution generalization and systematic generalization. However, current NMNs cannot rival with Transformer-based networks, which leverage large amounts of data by using attention mechanisms that boost their overall accuracy. This begs the question of whether we can improve the systematic generalization capabilities of Transformer-based models by borrowing NMNs' strategies.

In this paper, we introduce *Transformer Module Network (TMN)*, a novel Transformer-based model for VQA that composes modules *à la* NMN into a question-specific Transformer network. In this way, we take the best of both worlds: the capabilities of Transformer-based models given by attention mechanisms, and the flexibility of NMNs to adjust to questions based on novel compositions of modules. An overview of TMNs is depicted in Fig. 1.

To foreshadow the results, we find that TMNs achieve state-of-the-art systematic generalization accuracy in three VQA datasets: CLEVR-CoGenT [Johnson *et al.*, 2017a], CLOSURE [Bahdanau *et al.*, 2020], and a novel test set based on GQA [Hudson and Manning, 2019] that we introduce for evaluating systematic generalization performance with natural images, which we call GQA-SGL. Remarkably, TMNs improve systematic generalization accuracy over standard Transformers more than 30% in the CLOSURE dataset,



(a) Library and module architecture



(b) Examples of question-specific networks

Figure 1: *Overview of Transformer Module Network (TMN)*. It consists of neural modules and takes as inputs an image and a program that corresponds to a question. Each module is a small stack of Transformer encoders. The modules are dynamically taken out of a library of the modules and composed into a question-specific Transformer network. The total number of Transformer layers is the number of Transformer layers of each module (K) $\times$ the program length (L).

and also achieve state-of-the-art in-distribution accuracy. Furthermore, our results highlight the importance to control the pre-training of the networks in order to ensure a fair evaluation of systematic generalization.

2 Related work

We review previous works on systematic generalization in VQA. We first revisit the available benchmarks and then introduce existing approaches.

2.1 Benchmarking VQA

Even though systematic generalization capabilities are the crux of VQA, attempts to benchmark these capabilities are only recent. The first VQA datasets evaluated in-distribution generalization, and later ones evaluated generalization under distribution shifts that do not require systematicity. In the following, we review progress made towards benchmarking systematic generalization in VQA.

In-distribution generalization. There is a plethora of datasets to evaluate in-distribution generalization, e.g., VQA-v2 [Goyal *et al.*, 2019]. It has been reported that these datasets are biased and models achieve high accuracy by relying on spurious correlations instead of performing visual reasoning [Agrawal *et al.*, 2018].

Out-of-distribution generalization. VQA-CP [Agrawal *et al.*, 2018] and GQA-OOD [Kervadec *et al.*, 2021] were proposed to evaluate generalization under shifted distribution of question-answers. While this requires a stronger form of generalization than in-distribution, it does not require tackling the combinatorial nature of visual reasoning, and models can leverage biases in the images and questions.

Systematic generalization. CLEVR-CoGenT [Johnson *et al.*, 2017a] and CLOSURE [Bahdanau *et al.*, 2020] are datasets that require systematic generalization as models need to tackle novel combinations of visual attributes and subtasks. Since these datasets include only synthetic images, we introduce GQA-SGL, a novel test set based on GQA [Hudson and Manning, 2019] to evaluate systematic generalization with natural images.

2.2 Approaches for Systematic Generalization

We now revisit Transformer-based models and NMNs for systematic generalization in VQA as they are the basis of TMNs.

Transformer-Based Models. Transformers were originally introduced to effectively handle long-range dependencies in natural language processing [Vaswani *et al.*, 2017]. Currently, most approaches to VQA are based on Transformers, e.g., [Lu *et al.*, 2019; Tan and Bansal, 2019]. These works focus on in-distribution generalization, and it was not until recently that Transformers for systematic generalization have been investigated. To the best of our knowledge, in VQA the only related work is MDETR [Kamath *et al.*, 2021], which uses a novel pre-training approach that captures the long tail of visual concepts and achieves state-of-the-art performance on many vision-and-language datasets, including CLEVR-CoGenT. As we show in the sequel, our approach improves such systematic generalization capabilities in CLEVR-CoGenT without requiring pre-trained Transformer encoders, as it leverages insights from NMNs.

Neural Module Networks (NMNs). They are an effective approach to alleviate the gap between in-distribution generalization and systematic generalization [Bahdanau *et al.*, 2019; Bahdanau *et al.*, 2020; D'Amario *et al.*, 2021]. NMNs consists of a collection of shallow neural networks called modules that are specialized to tackle a sub-task [Andreas *et al.*, 2016]. A question is represented in the form of a program which each instruction can be implemented with a module. Thus, modules are composed into a network specific for the question. Vector-NMN [Bahdanau *et al.*, 2020] is the state-of-the-art NMN on systematic generalization and outperforms all previous NMNs. NMNs could further benefit from the powerful attentional mechanisms of Transformer architectures. Meta-Module Network (MMN) [Chen *et al.*, 2021] is a recently proposed NMN which employs attention mechanism in each of its modules. Yet, it has been only eval-

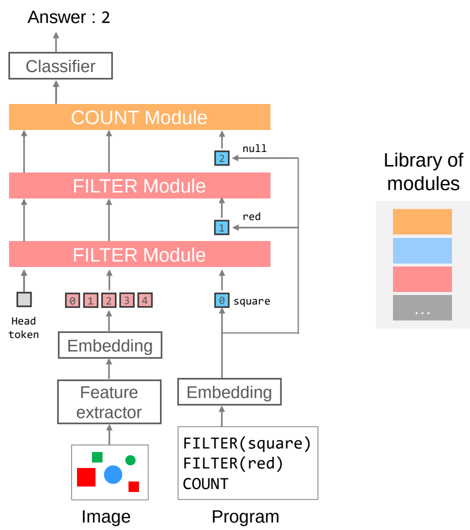
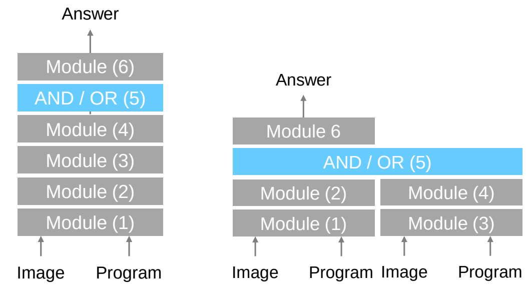


Figure 2: *TMN Architecture.* Given an image and a program related to a question, TMNs firstly constructs a question-specific Transformer network with modules based on the program and extracts visual features from the image. Each module transforms representations of the visual features and a special head token with arguments. A classifier outputs an answer from the final representation of the head token.



(a) Stack (b) Tree

Figure 3: *Stack and Tree modular structures.* (a) Stack structure simply stacks all modules. (b) Tree structure runs two threads (Module 1 $\rightarrow$ Module 2 and Module 3 $\rightarrow$ Module 4) and then merges their outputs. (*) denotes the order of each function in the program.

uated for in-distribution generalization. Our proposed approach provides substantial gains of systematic generalization capabilities also over previous NMNs.

3 Transformer Module Networks (TMNs)

In this section we introduce TMNs, i.e., our novel Transformer-based model for VQA that dynamically composes neural modules into a question-specific Transformer network.

3.1 Dynamic Composition of Modules

Fig. 1 provides an overview of the dynamic composition of modules in TMNs.

TMNs use a library of neural modules, each of which tackles a sub-task. TMNs represent the question in VQA with a program, which is a sequence of sub-tasks we defined (e.g., {`FILTER`, `COUNT`}). For instance, a question "How many red squares are there?" can be converted to a program {`FILTER(square)`, `FILTER(red)`, `COUNT`}, and then the modules corresponding to the sub-tasks are taken out of the library and composed to form a Transformer network.

Each module is a stack of Transformer encoders. Thus, the modules are dynamically composed into a question-specific Transformer network. Unlike standard Transformers, which are a stack of a fixed number of Transformer encoders (typically 12) regardless of the input question, TMN dynamically composes the layers and thus the total number of layers varies according to the question in our approach. When the module consists of K Transformer encoder layers and the program length is L, TMN has a total of $K \times L$ Transformer encoder layers (six in the above case with $K = 2$).

We follow the same approach in previous works that directly use the program rather than the raw question [Johnson *et al.*, 2017b; Bahdanau *et al.*, 2019; D'Amario *et al.*, 2021]. This allows to analyse systematic generalization in isolation by focusing only on visual reasoning aspects, and omits the performance of the language parser.

3.2 Transformer-based Architecture

TMN implements a Transformer-based architecture as shown in Fig. 2.

For each input image, we use a feature extractor to extract visual features of the input image. Following the common procedure in Transformers, the visual features are transformed into the visual feature embeddings. Initial visual representations are the sum of the visual features embeddings and the position embeddings. The initial representation of the head token is an average of all the visual representations.

Each module is a stack of Transformer encoders that takes the representations of the head token, the visual features, and arguments corresponding to the module as an input sequence of tokens, and outputs the transformed representations of the head token and visual features. The output of the previous module are fed into the next module. Only the first module receives initial representations of the head token and visual features. Unlike Vector-NMN, whose modules always receive the initial visual representations, only the first module in the dynamically composed network receives the initial visual representations in our approach. Also, due to the versatility of Transformer architectures, in TMN we can easily change the number of Transformer encoders in each module according to the complexity of the sub-task.

Finally, a classifier outputs an answer from the final representation of the head token.

3.3 Stack and Tree Modular Structures

Some modules with two inputs may be useful in constructing a network corresponding to a complex program. For instance, a logical sub-task (`AND` or `OR`) or sub-tasks that compare two objects. For programs containing such sub-tasks, we investigate two modular structures as for the example shown in

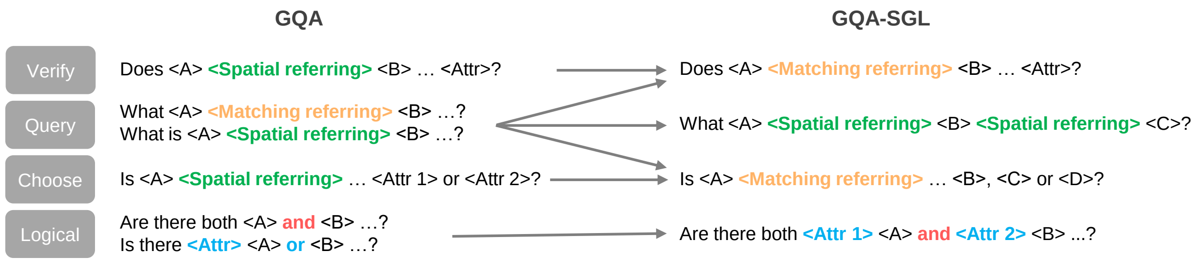


Figure 4: *The GQA-SGL test dataset for systematic generalization with natural images.* GQA-SGL is a novel test set we built based on GQA to evaluate systematic generalization performance. We generate new questions with ground-truth programs that belong four question types by combining the existing programs. <A,...,D> and <Attr> denote an object (e.g., chair) and an attribute (e.g., blue). <Spatial referring> and <Matching referring> are a referring expression such as " is right of " and " is the same color as ", respectively.

Fig. 3: (a) Stack structure simply stacks all modules and executes the modules in sequence. (b) Tree structure runs two threads in parallel (Module 1 → Module 2 and Module 3 → Module 4) and then merges their outputs. We compare these two structures in the experiments.

4 Experimental Setup

We introduce three datasets for VQA including a novel test set we built. We compare our TMN with two baseline models and state-of-the-art models on those datasets.

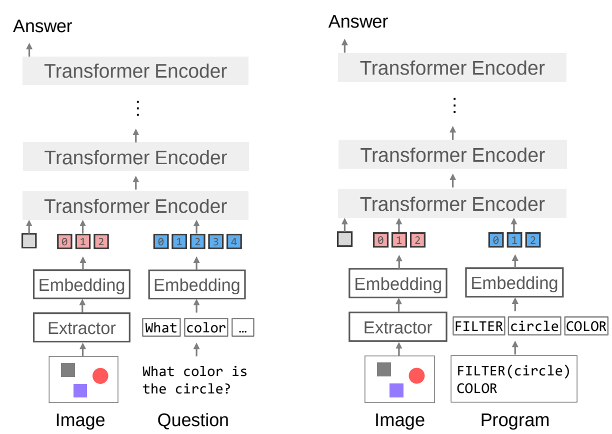
4.1 Datasets

To evaluate the models for systematic generalization to novel combinations of known visual attributes or linguistic constructs, we use the following three datasets.

CLEVR-CoGenT. CLEVR is a diagnostic dataset for VQA models [Johnson *et al.*, 2017a] that consists of synthetic 3D scenes with multiple objects and automatically generated questions, associated with a ground-truth program formed by multiple sub-tasks. This dataset comes with additional splits to test systematic generalization, namely the Compositional Generalization Test (CLEVR-CoGenT). CLEVR-CoGenT is divided in two conditions where objects appear with limited color and shape combinations, that are inverted between training and testing. With this dataset, we can test systematic generalization to novel combinations of visual attributes.

CLOSURE. This is a test set for models trained with CLEVR [Bahdanau *et al.*, 2020], and provides 7 benchmarks to test systematic generalization to novel combinations of known linguistic constructs. CLOSURE uses the same synthetic images in CLEVR but contains questions which require the models to recombine the known constructs in a novel way.

GQA-SGL. GQA is a VQA dataset that consists of complex natural images and questions, associated with ground-truth programs [Hudson and Manning, 2019]. GQA-SGL (Systematic Generalization to Linguistic combinations) is a novel test set we built based on GQA to test systematic generalization to novel linguistic combinations with natural images. We generate new questions with ground-truth programs by combining the existing programs as shown in Fig. 4. We only use test-dev balanced split to build this test set, and in all experiments with GQA, we always use the balanced split that controls bias. See Appendix A for a detailed description.



(a) Transformer (b) Transformer w/ PR

Figure 5: *Transformer-based baseline methods.* (a) Standard Transformer (Transformer) takes an image and a question as inputs. (b) Standard Transformer with programs (Transformer w/ PR) takes a program as an input instead of the question. Visual inputs and the Transformer encoder layers are exactly the same.

4.2 Methods and Settings

We compare TMNs with two baseline methods shown in Fig. 5. In (a), standard Transformer (Transformer) takes an image and a question as inputs. In (b), standard Transformer with programs (Transformer w/ PR) takes a program as an input instead of the question and treat it as a sequence of tokens. TMNs also take the program as input and executes each module with corresponding arguments as shown in Fig. 2. The programs allow us to focus on evaluating how effective our Transformer encoder modules and dynamic compositions of Transformer layers are.

Architecture. We use vanilla Transformer encoder for both standard Transformers and our TMNs for fair comparison. Standard Transformers consist of 12 Transformer layers. In TMN, the number of Transformer layers in each module is 1 for CLEVR-CoGenT and CLOSURE, and 2 for GQA-SGL

Methods	Visual attributes		Linguistic constructs	
	CoGenT-Condition A (In-Distribution)	CoGenT-Condition B (Syst. generalization)	CLEVR (In-Distribution)	CLOSURE (Syst. generalization)
Transformer	97.5 ± 0.18	78.9 ± 0.80	97.4 ± 0.23	57.4 ± 1.6
Transformer w/ PR	97.4 ± 0.56	**81.7** ± 1.1	97.1 ± 0.14	64.5 ± 2.5
TMN-Stack (ours)	97.9 ± 0.030	80.6 ± 0.21	**98.0** ± 0.030	90.9 ± 0.49
TMN-Tree (ours)	**98.0** ± 0.016	80.1 ± 0.72	97.9 ± 0.0098	**95.4** ± 0.20
Vector-NMN	98.0 ± 0.2	73.2 ± 0.2	98.0 ± 0.066	94.4
MDETR	99.7	76.2	99.7	53.3

Table 1: *Results on systematic generalization to novel visual and linguistic combinations.* Mean and standard deviation of accuracy (%) across more than three repetitions. All methods are tested on in-distribution (CLEVR-CoGenT validation condition A and CLEVR validation) and systematic generalization (CLEVR-CoGenT validation condition B and CLOSURE, indicated in yellow). We trained and test Vector-NMN on CLEVR-CoGenT on our environment while its performances on CLEVR and CLOSURE are cited from the original paper. We test MDETR with a official trained model on our environment.

Methods	GQA (In-Distribution)	GQA-SGL (Syst. generalization)	Four question types in GQA-SGL			
			verify	query	choose	logical
Transformer	**54.9** ± 0.004 (67.6)	47.7 ± 2.1	50.0 ± 1.6	35.3 ± 5.0	50.0 ± 7.5	55.3 ± 4.1
Transformer w/ PR	54.7 ± 0.22 (67.0)	52.2 ± 3.2	51.6 ± 7.7	45.6 ± 5.6	**56.0** ± 6.6	55.6 ± 5.4
TMN-Stack (ours)	52.8 ± 0.10 (64.7)	50.7 ± 0.94	51.6 ± 4.1	**49.6** ± 5.0	44.8 ± 3.7	56.4 ± 7.1
TMN-Tree (ours)	53.5 ± 0.24 (65.2)	**53.7** ± 1.7	**56.0** ± 7.3	48.0 ± 5.5	47.2 ± 3.2	**64.0** ± 4.6

Table 2: *Results on an application to complex natural images with novel linguistic combinations.* Mean and standard deviation of accuracy (%) on GQA test-dev and GQA-SGL across more than three repetitions. We use a pre-trained object detector as a feature extractor. The numbers in () denote mean accuracy on only four question types in GQA. Systematic generalization performance is indicated in yellow.

because of the difference in the complexity of their sub-tasks. Transformer encoders in all models have a hidden state size of 768 and 12 attention heads.

Library of modules and programs. For CLEVR-CoGenT and CLOSURE, we use the modules and programs defined in the dataset. For GQA-SGL, we follow the definitions proposed in [Chen *et al.*, 2021].

Visual features. For CLEVR-CoGenT and CLOSURE, we use grid features extracted by ResNet-101 pre-trained with ImageNet to obtain the visual feature of $H \times W \times 2048$ dimensions, where H and W are the height and the width of the grid feature map. We add a positional encoding to the grid features and flatten them with a linear projection. We treat the flattened features as a sequence of tokens and thus the number of the tokens is $H \times W$ (150 on these datasets). The input images in GQA contain various objects and are much more complex than those in CLOSURE. Therefore, for GQA-SGL, we use popular regional features [Anderson *et al.*, 2018] extracted by Faster R-CNN [Ren *et al.*, 2015] (with ResNet-101 backbone) pre-trained on Visual Genome dataset [Krishna *et al.*, 2016]. We consistently keep 36 regions for each image. Each region has 2048-dimensional visual feature and 5-dimensional spatial location feature.

Embeddings. Input representation of the program is sum of word embeddings, segment embeddings and position embeddings. They represent a function or an argument (e.g., `FILTER` or `circle`), a thread index, and word position, respectively. For the standard Transformer, we adopt BERT tokenizer and a standard sentence embeddings described in [Li *et al.*, 2020].

Hyperparameters. We use the Adam optimizer for all cases. Learning rates for standard Transformers and TMNs are $2e^{-5}$ and $1e^{-5}$ for CLEVR-CoGenT, $2e^{-5}$ for CLEVR, and $1e^{-5}$ and $4e^{-5}$ for GQA. We search best learning rates for each model. We train all models with batch size of 128 on 4 Tesla V100 GPUs for 20 epochs except standard Transformer on CLEVR, where we use 30 epochs to achieve convergence. Training of standard Transformers and TMNs finished in 3 and 4 days, respectively.

Other methods. We also experiment with other state-of-the-art methods, including Vector-NMN [Bahdanau *et al.*, 2020], MDETR [Kamath *et al.*, 2021], LXMERT [Tan and Bansal, 2019] and MMN [Chen *et al.*, 2021]. We follow the parameters described in the original paper.

5 Results and Analysis

We report the systematic generalization performance to the novel visual or linguistic combinations with synthetic images (CLEVR-CoGenT and CLOSURE) and complex natural images (GQA-SGL). Then, we discuss the effect of pre-training.

5.1 Systematic Generalization Performance

Tables 1 and 2 show the mean and standard deviation accuracy of TMNs and rest of approaches on both in-distribution and systematic generalization.

CLEVR-CoGenT. With CLEVR-CoGenT, we evaluate the systematic generalization performance to novel combinations of known visual attributes. All models are trained on

Methods	CLEVR-CoGenT		
	w/o VG	w/ VG	Δ
Transformer	78.9	87.5	8.6
Transformer w/ PR	**81.7**	**88.7**	6.9
TMN-Stack (ours)	80.6	86.9	6.3
TMN-Tree (ours)	80.1	86.4	6.4

Methods	CLOSURE		
	w/o VG	w/ VG	Δ
Transformer	57.4	62.1	4.7
Transformer w/ PR	64.5	64.5	0.005
TMN-Stack (ours)	90.9	92.7	1.7
TMN-Tree (ours)	**95.4**	**96.4**	1.0

Table 3: *Effect of pre-training.* Mean accuracy (%) on CLEVR-CoGenT validation condition B and CLOSURE with two different feature extractors, which are an object detector pre-trained on Visual Genome (VG) and ResNet-101 (without pre-training on VG).

CLEVR-CoGenT condition A. Our results show that both standard Transformers and TMNs largely outperform the state-of-the-art NMN (Vector-NMN [Bahdanau *et al.*, 2020]). TMNs' systematic generalization performance also surpasses MDETR [Kamath *et al.*, 2021], which is the-state-of-the-art Transformer-based model proposed to capture long-tail visual concepts. TMNs achieve superior performance over the standard Transformer but their performances are slightly lower than the Transformer with programs. As expected, the programs seem to make visual grounding easier. See Appendix B for further details.

CLOSURE. With CLOSURE, we evaluate the systematic generalization performance to novel combinations of known linguistic constructs. All models are trained on CLEVR. Our results show that standard Transformers struggle with the novel linguistic combinations even when the program is given (Transformer w/ PR). Our TMNs achieve much better performance than the standard Transformers, and also outperform the Vector-NMN. Remarkably, TMN-Tree improves systematic generalization accuracy over standard Transformers more than 30% in this dataset. Tree structure seems effective for the novel linguistic constructs because the questions often have tree structure. See Appendix C for further details.

GQA-SGL. With GQA-SGL, we evaluate the systematic generalization performance to novel linguistic combinations with natural images. All models are trained on GQA. In Table 2, we evaluate the exact matching between the predicted and ground-truth answers, on both GQA and GQA-SGL. The in-distribution accuracy of TMNs is slightly lower than the one of the standard Transformers, but TMN-Tree achieves superior performance on systematic generalization. The performance gain in this test set is relatively smaller than that in CLOSURE tests. One reason could be the complexity of the questions. The maximum program length is 26 in CLOSURE, while it is 9 in GQA. CLOSURE provides more complex questions which require stronger systematic generalization capabilities.

Methods	Pre-training data	GQA	GQA-SGL
TMN (ours)	-	65.2	53.7
MDETR †	VG, COCO, Fk, Gu	73.9	59.0
LXMERT †	VG, COCO, VQA	68.9	58.0
MMN †	Gu	71.3	48.0

Table 4: *State-of-the-art VQA methods.* Mean accuracy (%) on four question types in GQA test-dev and GQA-SGL. The pre-training refers to datasets with image-text pairs used during training besides GQA. All methods except TMN-Tree are pre-trained with a large amount of image-text pairs. † We use official trained models. VG, COCO, Fk, Gu and VQA denote Visual Genome, MS COCO, Flickr30k, GQA unbalanced split, and VQA-v2, respectively.

5.2 Effect of Pre-Training

Since pre-training is a very common approach in VQA, we investigate how the additional image data from other domains affects systematic generalization to novel combinations of visual attributes or linguistic constructs, i.e., CLEVR-CoGenT and CLOSURE, respectively. Table 3 shows the mean accuracy with two different visual features, namely, regional features extracted by an object detector pre-trained on Visual Genome (VG) and grid features extracted by ResNet-101 without pre-training on VG. Across all tested models, we observe that the performance gains from pre-training the visual feature extractor are larger for generalization to novel visual combinations (CLEVR-CoGenT) and smaller for novel linguistic constructs (CLOSURE). These improvement cannot be attributed to using regional features because these do not contribute to the performance improvements without pre-training on VG (see Appendix D). Thus, as expected, pre-training can lead to large improvements of systematic generalization capabilities when the pre-training data is relevant to the aspects evaluated in the dataset. This highlights the need to control pre-training in order to ensure a fair comparison of different methods for systematic generalization. Also, it shows that the pre-training data needs to target the systematic generalization task at hand in order to be effective.

Table 4 reports results of TMNs and state-of-the-art Transformer-based models (MDETR [Kamath *et al.*, 2021] and LXMERT [Tan and Bansal, 2019]) and Neural Module Networks (Meta-Module Network [Chen *et al.*, 2021]) on GQA and GQA-SGL. The table also indicates the pre-training datasets with image-text pairs. The methods use different pre-training datasets, and hence, they cannot be directly compared. Yet, we observe that TMNs strike a balance between amount of pre-training data for the Transformer encoders and systematic generalization performance.

6 Conclusions

We have compared the systematic generalization capabilities of TMNs with the most promising approaches in the literature, i.e., Transformers and NMNs, on three VQA datasets. TMNs achieve state-of-the-art systematic generalization performance across all benchmarks. Namely, all Transformer-based models (including TMNs) outperform the state-of-the-art NMNs, with TMNs surpassing the standard Transformers on novel linguistic constructs. We have also shown that controlling pre-training is essential to ensure a fair comparison, and that TMNs are capable of improving systematic generalization without the need of pre-training datasets. We hope that these results unleash the potential of modular approaches for systematic generalization as they motivate promising future work towards creating large-scale modular architectures.

Acknowledgements and Disclosure of Funding

We would like to thank Pawan Sinha and Tomaso Poggio for warm encouragement and insightful advice. This work has been supported by Fujitsu Limited (Contract No. 40009105) and by the Center for Brains, Minds and Machines (funded by NSF STC award CCF-1231216). XB is supported by the R01EY020517 grant from the National Eye Institute (NIH).

References

[Agrawal *et al.*, 2018] Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. Don't just assume; look and answer: Overcoming priors for visual question answering. In *CVPR*, pages 4971–4980, 2018.

[Anderson *et al.*, 2018] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, pages 6077–6086, 2018.

[Andreas *et al.*, 2016] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *CVPR*, pages 39–48, 2016.

[Bahdanau *et al.*, 2019] Dzmitry Bahdanau, Shikhar Murty, Michael Noukhovitch, Thien Huu Nguyen, Harm de Vries, and Aaron Courville. Systematic generalization: What is required and can it be learned? In *ICLR*, 2019.

[Bahdanau *et al.*, 2020] Dzmitry Bahdanau, Harm de Vries, Timothy J. O'Donnell, Shikhar Murty, Philippe Beaudoin, Yoshua Bengio, and Aaron C. Courville. CLOSURE: assessing systematic generalization of CLEVR models. arXiv preprint, arXiv:1912.05783v2, 2020.

[Bergen *et al.*, 2021] Leon Bergen, Timothy J. O'Donnell, and Dzmitry Bahdanau. Systematic generalization with edge transformers. In *NeurIPS*, 2021.

[Chen *et al.*, 2021] Wenhu Chen, Zhe Gan, Linjie Li, Yu Cheng, William Wang, and Jingjing Liu. Meta module network for compositional visual reasoning. In *WACV*, pages 655–664, 2021.

[D'Amario *et al.*, 2021] Vanessa D'Amario, Tomotake Sasaki, and Xavier Boix. How modular should neural module networks be for systematic generalization? In *NeurIPS*, 2021.

[Gontier *et al.*, 2020] Nicolas Gontier, Koustuv Sinha, Siva Reddy, and Christopher Pal. Measuring systematic generalization in neural proof generation with transformers. In *NeurIPS*, pages 22231–22242, 2020.

[Goyal *et al.*, 2019] Yash Goyal, Tejas Khot, Aishwarya Agrawal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *CVPR*, pages 398–414, 2019.

[Hudson and Manning, 2019] Drew A. Hudson and Christopher D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *CVPR*, pages 6700–6709, 2019.

[Johnson *et al.*, 2017a] 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 *CVPR*, pages 2901–2910, 2017.

[Johnson *et al.*, 2017b] 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 *ICCV*, 2017.

[Kamath *et al.*, 2021] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. MDETR - modulated detection for end-to-end multi-modal understanding. In *ICCV*, pages 1780–1790, 2021.

[Kervadec *et al.*, 2021] Corentin Kervadec, Grigory Antipov, Moez Baccouche, and Christian Wolf. Roses are red, violets are blue... but should VQA expect them to? In *CVPR*, pages 2776–2785, 2021.

[Krishna *et al.*, 2016] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David Shamma, Michael Bernstein, and Fei Fei Li. Visual Genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 123:32–73, 2016.

[Lake and Baroni, 2018] Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *ICML*, pages 2873–2882, 2018.

[Li *et al.*, 2020] Gen Li, Nan Duan, Yuejian Fang, Ming Gong, Daxin Jiang, and Ming Zhou. Unicoder-VL: A universal encoder for vision and language by cross-modal pretraining. In *AAAI*, pages 11336–11344, 2020.

[Lu *et al.*, 2019] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, pages 13–23, 2019.

[Ren *et al.*, 2015] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015.

[Ruis *et al.*, 2020] Laura Ruis, Jacob Andreas, Marco Baroni, Diane Bouchacourt, and Brenden M Lake. A benchmark for systematic generalization in grounded language understanding. In *NeurIPS*, pages 19861–19872, 2020.

[Tan and Bansal, 2019] Hao Tan and Mohit Bansal. LXMERT: Learning cross-modality encoder representations from transformers. In *EMNLP-IJCNLP*, pages 5100–5111, 2019.

[Tsarkov *et al.*, 2021] Dmitry Tsarkov, Tibor Tihon, Nathan Scales, Nikola Momchev, Danila Sinopalnikov, and Nathanael Schärli. *-CFQ: Analyzing the scalability of machine learning on a compositional task. In *AAAI*, pages 9949–9957, 2021.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

Appendix

A GQA-SGL

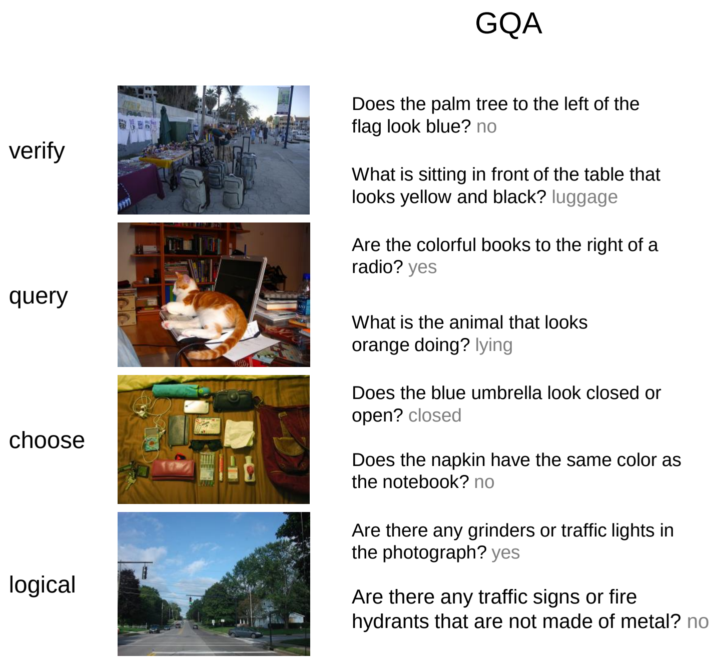


Figure 6: Examples of test questions from GQA and GQA-SGL for the same images.

We generate GQA-SGL, a novel GQA split to test Systematic Generalization to Linguistic combinations of models trained on GQA. GQA-SGL contains 200 new questions with their ground-truth programs. 97.4% of the original GQA questions can be divided into four question types: verify, query, choose, and logical. Fig. 4 describes the GQA-SGL data generation. On the left, we first list out all ground-truth programs that belong these question types. The arguments in parenthesis $<\cdot>$ denote either objects (*e.g.*, A and B), attributes (*e.g.*, Attr), or referring expression. Those can be *spatial referring*, where an object or group of objects is selected based on the relation with another object or group of objects (*e.g., the object A which is left to objects B*); *matching referring* where an objects/group is selected depending on some matching properties with another object/group; *and* or *or* where either or one between two objects/groups of objects are selected.

On the right of Fig. 4, we generate new ground-truth programs by combining attributes and referring expressions which never appeared in GQA. We emphasize the systematic shift between GQA and GQA-SGL linguistic construct by highlighting referring expressions and attributes through different colors.

We generate 50 questions for each question type as shown in Fig. 6.

B Supplemental Results on CLEVR-CoGenT

We report the systematic generalization performance for each question type on CLEVR-CoGenT in Table 5. All methods are trained on CLEVR-CoGenT condition A and tested on CLEVR-CoGenT validation condition B.

C Supplemental Results on CLOSURE

We report the systematic generalization performance for each test in CLOSURE in Table 6. All methods are trained on CLEVR and tested on CLOSURE tests.

D Effect of Regional Features

We compare the systematic generalization performances on CLEVR-CoGenT with grid features and regional features, as shown in Table 7. We use ResNet-101 and region of interest pooling to extract regional features, while we use only the same ResNet-101 to extract grid features. In case of the standard Transformer and our TMNs, the performance gaps are less than 1.5%. In case of the standard Transformer with ground-truth programs, the performance gap is slightly larger than it (3.6%).

Methods	Exist	Count	Compare Integer	Compare Attribute	Query Attribute
Transformer	87.8 $\pm$ 0.69	73.3 $\pm$ 0.13	88.6 $\pm$ 0.41	85.8 $\pm$ 0.68	73.2 $\pm$ 1.7
Transformer w/ PR	**90.4** $\pm$ 0.90	**76.8** $\pm$ 0.99	**89.8** $\pm$ 0.45	**88.3** $\pm$ 1.1	76.3 $\pm$ 1.6
TMN-Stack (ours)	88.4 $\pm$ 0.069	75.1 $\pm$ 0.27	88.1 $\pm$ 0.20	86.3 $\pm$ 0.22	**76.6** $\pm$ 0.28
TMN-Tree (ours)	87.8 $\pm$ 0.62	74.3 $\pm$ 0.83	89.0 $\pm$ 0.66	87.8 $\pm$ 0.69	75.3 $\pm$ 0.75
Vector-NMN	84.4 $\pm$ 0.35	70.4 $\pm$ 0.44	81.3 $\pm$ 0.52	78.1 $\pm$ 0.79	66.5 $\pm$ 1.05
MDETR	85.4	75.2	81.3	82.6	68.9

Table 5: Mean and standard deviation of accuracy (%) on each question type in CLEVR-CoGenT Condition B.

Methods	and_mat _spa	or_mat	or_mat_spa	embed _spa_mat	embed _mat_spa	compare _mat	compare _mat_spa
Transformer	77.0 $\pm$ 5.2	19.1 $\pm$ 0.55	25 $\pm$ 11	97.0 $\pm$ 1.3	55.2 $\pm$ 4.4	65.7 $\pm$ 2.8	62.7 $\pm$ 2.3
Transformer w/ PR	77.6 $\pm$ 6.9	39.7 $\pm$ 1.9	36.4 $\pm$ 5.8	59.7 $\pm$ 2.8	78.1 $\pm$ 5.0	85.7 $\pm$ 4.4	74.4 $\pm$ 6.6
TMN-Stack (ours)	98.4 $\pm$ 0.19	77.8 $\pm$ 1.5	72.7 $\pm$ 2.6	**98.8** $\pm$ 0.00	92.7 $\pm$ 0.84	77.8 $\pm$ 1.5	72.7 $\pm$ 2.6
TMN-Tree (ours)	**98.5** $\pm$ 0.03	89.5 $\pm$ 0.62	**90.1** $\pm$ 0.8	98.7 $\pm$ 0.16	92.8 $\pm$ 0.42	**99.0** $\pm$ 0.03	**99.3** $\pm$ 0.11
Vector-NMN	86.3 $\pm$ 2.5	**91.5** $\pm$ 0.77	88.6 $\pm$ 1.2	98.5 $\pm$ 0.13	**98.7** $\pm$ 0.19	98.5 $\pm$ 0.17	98.4 $\pm$ 0.3
MDETR	8.63	34.1	18.4	99.3	77.4	69.3	66.0

Table 6: Mean and standard deviation of accuracy (%) on each CLOSURE test.

Methods	Grid features	Regional features
Transformer	78.9 $\pm$ 0.80	77.4 $\pm$ 0.68
Transformer w/ PR	**81.7** $\pm$ 1.1	78.2 $\pm$ 0.44
TMN-Stack (ours)	80.6 $\pm$ 0.21	79.4 $\pm$ 0.37
TMN-Tree (ours)	80.1 $\pm$ 0.72	**80.9** $\pm$ 0.25

Table 7: Mean and standard deviation of accuracy (%) on CLEVR-CoGenT validation condition B with grid features and regional features.