Transformer Module Networks for Systematic Generalization in Visual Question Answering

Moyuru Yamada ©, Vanessa D’Amario ©, Kentaro Takemoto ©, Xavier Boix, and Tomotake Sasaki ©, Member, IEEE

Abstract—Transformers achieve great performance on Visual Question Answering (VQA). However, their systematic generalization capabilities, i.e., handling novel combinations of known concepts, is unclear. We reveal that Neural Module Networks (NMNs), i.e., question-specific compositions of modules that tackle a sub-task, achieve better or similar systematic generalization performance than the conventional Transformers, even though NMNs’ modules are CNN-based. In order to address this shortcoming of Transformers with respect to NMNs, in this paper we investigate whether and how modularity can bring benefits to Transformers. Namely, we introduce Transformer Module Network (TMN), a novel NMN based on compositions of Transformer modules. TMNs achieve state-of-the-art systematic generalization performance in three VQA datasets, improving more than 30% over standard Transformers for novel compositions of sub-tasks. We show that not only the module composition but also the module specialization for each sub-task are the key of such performance gain.

Index Terms—Neural module network, systematic generalization, transformer, visual question answering.

I. INTRODUCTION

VISUAL Question Answering (VQA) [1] is a fundamental 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, which are composed of attributes such as textures, shapes and colors. This compositional structure yields a distribution of image-question pairs of combinatorial size, which cannot be fully reflected in an unbiased way by training distributions.

Systematic generalization is the ability to generalize to novel compositions of known concepts beyond the training distribution [2], [3], [4]. 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. In fact, the most successful learning machines, i.e., Transformer-based models, have been tremendously effective for VQA when evaluated in-distribution [5], [6], [7]. Yet, recent studies stressed the need to evaluate systematic generalization instead of in-distribution generalization [8], [9], [10], as the systematic generalization capabilities of Transformers for VQA are largely unknown.

A recent strand of research for systematic generalization in VQA investigates Neural Module Networks (NMNs) [3], [11], [12]. NMNs decompose a question in VQA into sub-tasks, and each sub-task is tackled with a shallow neural network called module. Thus, NMNs use a question-specific composition of modules to answer novel questions. This approach leverages the compositional nature of the questions and reasonably assume that the question can be represented as a sequence of sub-tasks that lead to the answer (Fig. 1). NMNs alleviate the gap between in-distribution generalization and systematic generalization due to its inherent compositional structure. In our experiments, we found that CNN-based NMNs outperform Transformers on systematic generalization to novel compositions of sub-tasks. This begs the question of whether we can combine the strengths of Transformers and NMNs in order to improve the systematic generalization capabilities of learning machines.

In this paper, we introduce Transformer Module Network (TMN), a novel NMN for VQA based on compositions of Transformer modules. In this way, we take the best of both worlds: the capabilities of Transformers given by attention mechanisms, and the flexibility of NMNs to adjust to questions based on novel compositions of modules. TMN allows us to investigate whether and how modularity brings benefits to Transformers in VQA.

To foreshadow the results, we find that TMNs achieve state-of-the-art systematic generalization accuracy in the following...
three VQA datasets: CLEVR-CoGenT [13], CLOSURE [11], and a novel test set based on GQA [14] that we introduce to evaluate systematic generalization performance with natural images, which we call GQA-SGL (Systematic Generalization to Linguistic combinations). Remarkably, TMNs improve systematic generalization accuracy over standard Transformers more than 30% in the CLOSURE dataset, i.e., systematic generalization to novel combinations of known linguistic constructs (equivalently, sub-tasks). Our results also show that both module composition and module specialization to a sub-task are key to TMN’s performance gain.

II. RELATED WORK

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

A. 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 [15] and GQA [14]. It has been reported that these datasets are biased and models achieve high accuracy by relying on spurious correlations instead of performing visual reasoning [16], [17].

Out-of-distribution generalization: VQA-CP [16] and GQA-OOD [17] were proposed to evaluate generalization under shifted distribution of question-answer pairs. 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 [13] and CLOSURE [11] are datasets that require systematic generalization as models need to tackle novel combinations of visual attributes and sub-tasks. Since these datasets include only synthetic images, we introduce GQA-SGL, a novel test set based on GQA to evaluate systematic generalization with natural images.

B. 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: Currently, most approaches to VQA are based on Transformers [18], e.g., [5], [6], [7], and pre-training is at the core of these approaches. 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 [19], which uses a novel 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 shows better systematic generalization capabilities in CLEVR-CoGenT and CLOSURE without requiring pre-trained Transformer encoders.

Recent breakthroughs in Large Multi-modal Models (LMMs), which are mostly Transformer-based models pre-trained on a huge amount of data, have demonstrated their outperforming capabilities in VQA [20], [21]. LMMs can handle any pair of images and questions as a text generation task, while they struggle with hallucination and a lack of explainability. They also require high computational costs regardless of the complexity of the questions. It is very difficult to correctly evaluate their systematic generalization performance since training and test data must be carefully designed for it.

Neural Module Networks (NMNs): They represent a question in the form of a program in which each sub-task is implemented with a neural module. Thus, modules are composed into a network specific for the question [22]. Some of the most successful modular approaches include NS-VQA [23], which uses a symbolic execution engine instead of the neural modules, and the Meta-Module Network (MMN) [24], which introduces a neural module that adjusts to novel sub-tasks. TbD [25] is another trial which uses attention as intermediate representations for transparent reasoning process. While they show their effectiveness in terms of in-distribution generalization, they are not designed for systematic generalization. NS-CL [26] introduces neural operators which map the object representation into an attribute embedding and shows superior generalization performance, while it requires handcrafted modules. Vector-NMN [11] is the state-of-the-art NMN on systematic generalization and outperforms all previous modular approaches. We present evidence that the systematic generalization capabilities of modular...
Overview of Transformer Module Network (TMN)

II. TRANSFORMER MODULE NETWORKS (TMNs)

In this section we introduce TMNs, i.e., our novel Transformer-based model for VQA that composes neural modules into a question-specific Transformer network. An overview of TMNs is depicted in Fig. 2. Unlike previous NMNs [11], [22] that use the visual features as input to each module, TMNs regard the tokens that represent the input image as a workspace shared among the modules, which transform the token representations. This architectural change is not straightforward and thus we need to architect NMNs taking into account the philosophy of tokens to leverage the attention mechanism.

A. Question-Specific Composition of Modules

Fig. 2 provides an overview. TMNs use a library of modules, and each module tackles a different sub-task we defined (e.g., FILTER, COUNT). TMNs represent the question in VQA with a program, which is a sequence of sub-tasks each of which is implementable with a module from the library. For instance, a question “How many red squares are there?” can be converted to a program \( \text{FILTER(square)}, \text{FILTER(red)}, \text{COUNT} \), and then the modules corresponding to the sub-tasks are composed to form a network to answer the question. The program can be regarded as a question with low ambiguity.

In TMN, each module is a stack of Transformer encoders. Unlike the standard Transformers, which are a stack of a fixed number of Transformer encoders (typically 12) regardless of the input question, TMN composes the Transformer modules into a question-specific network, and thus, the total number of encoder layers in TMN varies according to the question. Let \( K \) be the number of Transformer encoder layers in each module, and let \( L \) be the number of sub-tasks in the program. Then, TMN has a total of \( K \times L \) layers to implement a program of length \( L \).

We follow the same approach in previous works that directly use the program rather than the raw question [3], [11], [12]. This allows to analyse systematic generalization in isolation by focusing only on visual reasoning aspects, and omits the performance of the language parser.

B. Transformer-Based Architecture

The aforementioned question-specific composition of modules is preceded by a feature extractor and followed by an output classifier, as shown in Fig. 3(a).

We use a feature extractor to obtain the visual features of the input image. The visual features include grid features, patch features, or object features extracted from a backbone (e.g., ResNet), a vision Transformer [27], or an object detector (e.g., Faster R-CNN), respectively. We follow the common procedure in Transformers: The visual features are transformed into the visual feature embeddings and then summed with the position embedding, which yields the input visual representations of the first module. The initial head token representation is the average of all the visual representations.

Then, the first Transformer module in the question-specific network takes as input the sequence of tokens consisting of the representations of the visual features, the head token, and the module’s argument coming from the program. The Transformer module outputs the transformed representations of the head token and visual features. This output is fed into the next module, which also receives its corresponding argument coming from the program. Unlike Vector-NMN, whose modules always receive the initial visual representations, in TMN only the first module receives the initial visual representations. 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.

C. Stack and Tree Modular Structures

We further analyze the effect of modular structures. Some modules with two inputs may be useful for constructing a network corresponding to a complex program. For instance, a logical sub-task (AND or OR) is used to combine two conditions (see

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Fig. 3. TMN Architecture and Modular Structures. (a) Given an image and a program related to a question, TMNs construct a question-specific Transformer network as a composition of modules that tackle a sub-task. Each module takes as input the representations of the visual features, the head token and the module’s argument from the program. A classifier outputs an answer from the final representation of the head token. (b) We investigate two modular structures. Stack structure simply stacks all the modules. Tree structure runs two threads and then merges their outputs with a module (AND/OR in this example). The numbers in () denotes the order of each sub-task in the program.

IV. EXPERIMENTAL SETUP

We use three datasets for VQA: two existing datasets and a novel test set we built. We compare our TMN with two Transformer-based baselines and several state-of-the-art models on those datasets.

A. 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 [13] 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. Using this dataset, we can test systematic generalization to novel combinations of visual attributes.

CLOSURE: This is a test set for models trained with CLEVR, and provides 7 benchmarks [11]. The benchmarks test the systematic generalization to novel combinations of known linguistic constructs, which can be considered as sub-tasks, e.g., “is the same size as” can be a sub-task SAME_SIZE. CLOSURE uses the same synthetic images in CLEVR but contains questions which require the models to recombine the known sub-tasks in a novel way.

GQA-SGL: GQA is a VQA dataset that consists of complex natural images and questions, associated with ground-truth programs [14]. GQA-SGL (Systematic Generalization to Linguistic combinations) is a novel test set we built based on GQA to test systematic generalization to novel combinations of known linguistic constructs with natural images. We generate new questions with ground-truth programs by combining the existing programs as shown in Fig. 4. To build the test set, we use the test-dev set from the balanced split. For the train set, in all experiments we always use the balanced split that controls bias. 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 process. On the left, we first list out all ground-truth programs that belong to these question types. On the right, 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. See Appendix A, available online, for details.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Fig. 4. **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 to four question types (verify, query, choose, and logical) by combining the existing programs. \(<A,...,D>\) and \(<Attr>\) denote an object (e.g., chair) and an attribute (e.g., blue). \(<\text{Spatial referring}>\) and \(<\text{Matching referring}>\) are a referring expression, such as “is right of” and “is the same color as”, respectively.

Fig. 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. \(\text{Emb}\) denotes visual/word embeddings. These approaches use the same visual inputs and Transformer encoder structure.

### B. Methods and Settings

We compare TMNs with two Transformer-based baseline methods shown in Fig. 5. In (a), standard Transformer (Transformer) takes an image and a question as the 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 an input and executes each module with corresponding arguments as shown in Fig. 3. Thus, comparing Transformer w/ PR with TMNs allow us to assess the effectiveness of our Transformer modules and question-specific compositions of the modules. We treat VQA as a classification task and train the models with cross-entropy loss across the TMNs and our baselines.

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

**Sub-tasks and programs:** For CLEVR-CoGenT and CLOSURE, we use the sub-tasks and programs defined in the dataset. For GQA-SGL, we follow the definitions proposed in [24].

**Visual features:** For CLEVR-CoGenT and CLOSURE, we use grid features extracted by a 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 the popular regional features [29] extracted by a Faster R-CNN [30] (with a ResNet-101 backbone) pre-trained on Visual Genome dataset [31]. We consistently keep 36 regions for each image. Each region has 2048-dimensional visual feature and 5-dimensional spatial location feature.

**Embeddings:** The input representation of the program is a sum of word embeddings, segment embeddings and position embeddings. They represent a sub-task or an argument (e.g., \(\text{FILTER}\) or circle), a thread index, and word position, respectively. For the standard Transformer, we adopt BERT tokenizer and the standard sentence embeddings described in [32].

**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 the 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. We use this experimental setup and hyperparameter tuning for all experiments.
TABLE I
RESULTS ON SYSTEMATIC GENERALIZATION TO NOVEL COMBINATIONS OF KNOWN VISUAL ATTRIBUTES AND LINGUISTIC CONSTRUCTS

| Methods            | Visual attributes | Linguistic constructs | Overall |
|--------------------|-------------------|-----------------------|---------|
|                    | CoGen1-T (In-Dist.) | CoGen1-B (Syst. Gen.) | CLEVR (In-Dist.) | CLOSURE (Syst. Gen.) | (Syst. Gen.) |
| Transformer*       | 97.5 ± 0.18       | 78.9 ± 0.80           | 97.4 ± 0.23   | 57.4 ± 1.6           | 68.2 |
| Transformer w/ PR  | 97.4 ± 0.56       | 81.7 ± 1.1            | 97.1 ± 0.14   | 64.5 ± 2.5           | 73.1 |
| TMN-Stack (ours)   | 97.9 ± 0.03       | 80.6 ± 0.21           | 98.0 ± 0.03   | 90.9 ± 0.49          | 85.3 |
| TMN-Tree (ours)    | 98.0 ± 0.02       | 80.1 ± 0.72           | 97.9 ± 0.01   | 95.4 ± 0.20          | 87.8 |
| Vector-NMN         | 98.0 ± 0.2        | 73.2 ± 0.2            | 98.0 ± 0.07   | 94.4                 | 83.8 |
| NS-VQA*            | 99.8              | 63.9                  | 99.8          | 76.4                 | 70.2 |
| MDETR*             | 99.7              | 76.2                  | 99.7          | 53.3                 | 64.8 |

Mean and standard deviation of accuracy (%) across at least three repetitions when possible (only one trained model is provided in NS-VQA and MDETR). 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). We trained and test Vector-NMN on CLEVR-CoGenT in our environment while its performances on CLEVR and CLOSURE are cited from the original paper [11]. “Overall” at the right end is the mean value of accuracies on CoGen-T and CLOSURE. *Model takes the question as input rather than the dataset program.

TABLE II
RESULTS ON AN APPLICATION TO COMPLEX NATURAL IMAGES WITH NOVEL COMBINATIONS OF KNOWN LINGUISTIC CONSTRUCTS

| Methods            | GQA (In-Dist.) | GQA-SGL (Syst. Gen.) |
|--------------------|----------------|----------------------|
| Transformer*       | 54.9 ± 0.004 (67.6) | 47.7 ± 2.1           |
| Transformer w/ PR  | 54.7 ± 0.22 (67.0)  | 52.2 ± 3.2           |
| TMN-Stack (ours)   | 52.8 ± 0.10 (64.7)  | 50.7 ± 0.94          |
| TMN-Tree (ours)    | 53.5 ± 0.24 (65.2)  | 53.7 ± 1.7           |
| MMN                | 55.5 ± 0.12 (67.8)  | 51.0 ± 1.5           |

Mean and standard deviation of accuracy (%) on GQA test-dev and GQA-SGL. (Systematic generalization performance) across at least three repetitions. We use a pre-trained object detector as a feature extractor. The numbers in ( ) denote the mean accuracy on only four question types contained in GQA-SGL (verify, query, choose, and logical). *Model takes the question as input rather than the dataset program.

Other methods: We also experiment with other state-of-the-art methods, including Vector-NMN [11], MDETR [19] and MMN [24] (LXMERT [5] in Appendix G, available online). We use the hyperparameters described in the original papers. For a fair comparison, we train Vector-NMN and MMN using their publicly available code on the same training set as the rest of the methods. We also use the ground-truth programs as in TMN. MDETR was already trained on the same training set as the rest of the methods, and hence, we use the publicly available trained model. Also, we report the performance of NS-VQA [23] from its original paper and [33].

V. RESULTS AND ANALYSIS

In this section, we first report the systematic generalization performance to novel combinations of visual attributes or linguistic constructs with synthetic images (CLEVR-CoGenT and CLOSURE) and complex natural images (GQA-SGL). Then, we analyze the effect of module composition, module specialization, and pre-training.

A. Systematic Generalization Performance

Tables I and II show the mean and standard deviation accuracy of TMNs and the rest of approaches for both in-distribution and systematic generalization (see Appendix B, C, D and J, available online, for further details). In the following, we first introduce the results in the three datasets, and then, we discuss them:

**CLEVER-CoGenT**: It evaluates the systematic generalization performance to novel combinations of known visual attributes. All models are trained on CLEVR-CoGenT condition A. Our results show that both standard Transformers and TMNs largely outperform the state-of-the-art modular approaches (Vector-NMN and NS-VQA). TMNs’ systematic generalization performance also surpasses MDETR, 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 (Transformer w/ PR). As expected, Transformer w/ PR outperforms standard Transformer as the use of program facilitates visual grounding.

**CLOSURE**: It evaluates 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 novel compositions of known linguistic constructs even when the program is provided (Transformer w/ PR). TMNs achieve much better performance than standard Transformers and MDETR, and also outperform the other modular approaches (Vector-NMN and NS-VQA). Remarkably, TMN-Tree improves systematic generalization accuracy over standard Transformers more than 30% in this dataset. Tree structure seems effective for CLOSURE because the questions often have tree structure.

**GQA-SGL**: We evaluate the systematic generalization performance to novel combinations of known linguistic constructs with natural images. All models are trained on GQA. In Table II, we evaluate the exact matching between the predicted and ground-truth answers, on both GQA and GQA-SGL. TMN-Tree achieves superior performance on systematic generalization with a smaller drop of in-distribution accuracy compared to standard Transformers. Although MMN is the recently proposed NMN for GQA and achieves great in-distribution performance, TMN-Tree outperforms MMN on systematic generalization.

Discussion of the Results: Our results in CLOSURE and GQA-SGL show that TMNs (i.e., introducing modularity to Transformers) improve the systematic generalization performance to novel combinations of known linguistic constructs (i.e., novel compositions of sub-tasks) of Transformers and NMNs. The performance gain in CLOSURE is relatively larger than that in GQA-SGL tests. One reason to explain this is 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. TMN-Tree and Vector-NMN, which also employs the tree-structure, outperform TMN-Stack in the CLOSURE dataset. This result implies the importance of the inductive bias due to the tree-structure. Note that unlike Vector-NMN which employs learnable neural modules, NS-VQA uses hand-crafted modules designed to operate each sub-tasks. Thus, it should accomplish the task almost perfectly given a program on CLOSURE. Since this work focuses on whether modularity can bring benefits to Transformers, as a reference, we show NS-VQA performance where it depends on the program generator required to convert the out-of-distribution questions into the programs.

Compared to MMN, TMN also employs an attention-based module network and uses the same visual features. However, MMN instantiates all the modules from the meta-module (designed for VQA) and they have shared parameters, while our TMN assigns a module (vanilla Transformer encoder) for each sub-task. Thus, each of our modules have different parameters, and as a result the modules of TMN can learn their task more effectively, as demonstrated by our experiments.

Results in CLEVR-CoGenT show that TMNs do not improve systematic generalization of Transformers for novel compositions of visual attributes. This may be expected as [12] demonstrated that the visual feature extractor is the key component for generalization to novel combinations of visual attributes, and Transformer and TMNs have the same visual feature extractor. Note that for CoGenT dataset, systematic generalization requires disentangled representations of the shape and color, while for CLOSURE dataset, systematic generalization requires addressing compositions of sub-tasks. Thus, it is reasonable to expect that Transformers achieve higher systematic generalization accuracy than Vector-NMN in CoGenT, as Transformers have been shown to be able to learn more generalizable visual representations than CNNs. Also, it is reasonable to expect that Vector-NMN outperforms Transformers in CLOSURE dataset, because Vector-NMN uses compositions of modules to address the compositions of sub-tasks. Finally, we observe that since TMNs combines the “best of worlds” by combining Transformers and NMMs, they achieve the best accuracy across all tested methods in CLOSURE and GQA-SGL, and competitive accuracy with the best performing method in CLEVR-CoGenT.

Module Specialization: While the question-specific composition of modules is a key property of NMNs, a recent work reports that a degree of modularity of the NMN also has large influence on systematic generalization [12]. Here we also investigate this aspect for TMNs. Table III shows the mean accuracy of TMN-Stack with four different libraries of modules, each with a different degree of specialization. “Individual” denotes that each module is assigned to a single sub-task (26 sub-tasks defined in CLEVR in total). This is the library used in the previous sections. “Semantic group” denotes that each module tackles a group of sub-tasks that are semantically similar (e.g., FILTER_COLOR, FILTER_SIZE,...). “Random group” denotes that each module tackles a group of sub-tasks randomly selected (e.g., FILTER_COLOR, COUNT,...). “Order” denotes that each module is always placed in the same position of the question-specific network, independently of the sub-task performed in that position. If there are more sub-tasks in the program than modules in the library, the modules are repeated following the same order. In this way, we compare the importance of modules tackling sub-tasks with the importance of having a question-specific network. See Appendix E, available online, for further details about each library.

Results shows that the systematic generalization accuracy largely drops when the modules are not specialized, even if the number of modules is the same (compare “Order” with “Semantic group” and “Random group”). The degree of modularity (Individual vs Semantic group) and how sub-tasks are grouped (Semantic vs Random) also affects the systematic generalization performance. These results suggest that question-specific composition of modules is not enough and specialization of modules is crucially important. Finally, note that the results of Semantic group and Random group also show that TMN performs better than the standard Transformer (93.7 vs 64.5) even if the total number of parameters that all modules have is the same as the number of parameters of the standard Transformer (i.e., TMN has 12 modules of one layer each, while Transformers have 12 layers).

Ablation Analysis: We investigate the contribution of TMN’s specialization of modules (SoM) to TMNs’ superior performance over standard Transformer. Besides SoM, TMNs also have a variable number of layers, while standard Transformer have a fixed number of layers. We analyze if it is this difference in number of layers or the SoM that is causing TMNs’ higher accuracy. To do so, we add a property named Variable number of layers (VL) to the Transformer w/ PR. With the VL, the Transformer w/ PR uses as many encoder layers as the program length (e.g., 18 layers are used to output an answer for an input program with a length of 18). VL can be regarded as an imitation of the question-specific compositions of the modules in TMN, but each layer is not specialized for specific sub-tasks. Another key difference between TMNs and Transformers besides SoM is that TMN modules takes as argument tokens related to the sub-task as indicated by the program, while every layer of standard Transformer takes all input tokens in the program, because Transformers’ layers are not specialized to sub-tasks. Splitting the program tokens may help the model to steer the computations that should be executed at each layer. To investigate this, we add this property, named as Split Tokens (ST), to the Transformer w/ PR.
Table IV shows that any of the aforementioned properties do not improve the systematic generalization performance without the specialization of modules to sub-tasks, i.e., SoM. Even with Split tokens, the performance does not improve because the layers can not be specialized and splitting the token can spoil the flexibility of the standard Transformer.

Pre-training: We analyze 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 V 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 a ResNet-101 without pre-training on VG. Across all the tested models, we observe, as expected, that the performance gains from pre-training the visual feature extractor are larger for generalization to novel visual combinations and smaller for novel linguistic combinations. This highlights the need to control pre-training in order to ensure a fair comparison of different methods for systematic generalization. See Appendix F and G, available online for further experiments.

VI. CONCLUSION

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 tested benchmarks. We have shown that the question-specific composition of Transformer modules and the specialization of the modules to each sub-task are the key of the performance gains. Also, our results highlight that a fair comparison for systematic generalization requires controlling that all compared methods are pre-trained in the same way. We hope that our investigation unleash the potential of modular approaches for systematic generalization as they motivate promising future work towards creating large-scale modular architectures.

Limitations: TMNs have similar limitations as NMNs, i.e., it is challenging to decompose a question into sub-tasks. We have used the sub-tasks and the programs provided in the datasets generated by humans. This makes it difficult to apply NMNs and TMNs to other datasets that do not contain the programs. Another limitation is that we only investigated two types of systematic generalization (novel compositions of visual attributes and linguistic constructs). These limitations are key open questions that we will tackle in future work.

Reproducibility Statement

The source code and new dataset used in this study are publicly available at https://github.com/FujitsuResearch/transformer_module_networks. URLs of other existing assets (datasets, source code and trained models) used in this study.
are provided in the reference section. Experimental setup is described in Section IV.

Acknowledgment

The authors would like to thank Pawan Sinha, Serban Georgescu, Hisanaga Akima, and Tomaso Poggio for warm encouragement and insightful advice.

References

[1] S. Antol et al., “VQA: Visual question answering,” in Proc. 15th IEEE Int. Conf. Comput. Vis., 2015, pp. 2425–2433.
[2] B. Lake and M. Baroni, “Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks,” in Proc. 35th Int. Conf. Mach. Learn., 2018, pp. 2873–2882.
[3] D. Bahdanau, S. Mruty, M. Noukhovitch, T. H. Nguyen, H. de Vries, and A. Courville, “Systematic generalization: What is required and can it be learned?,” in Proc. 7th Int. Conf. Learn. Representations, 2019. [Online]. Available: https://openreview.net/forum?id=HkZeZxN9YY
[4] L. Ruis, J. Andreas, M. Baroni, D. Bouchacourt, and B. M. Lake, “A benchmark for systematic generalization in grounded language understanding,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 19861–19872.
[5] H. Tan and M. Bansal, “LXMERT: Learning cross-modality encoder representations from transformers,” in Proc. Conf. Empirical Methods Natural Lang. Process., 9th Int. Joint Conf. Natural Lang. Process., 2019, pp. 5100–5111.
[6] Y.-C. Chen et al., “UNITER: Universal image-Text representation learning,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 104–120.
[7] P. Zhang et al., “VinVL: Revisiting visual representations in vision-language models,” in Proc. 34th IEEE/ACM Conf. Comput. Vis. Pattern Recognit., 2021, pp. 5579–5588.
[8] N. Gontier, K. Sinha, S. Reddy, and C. Pal, “Measuring systematic generalization in neural proof generation with transformers,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 22231–22242.
[9] D. Tsarkov, T. Thon, N. Scales, N. Morniche, D. Sinopinakiou, and N. Schärlä, “CFQ: Analyzing the scalability of machine learning on a compositional task,” in Proc. 35th AAAI Conf. Artif. Intell., 2021, pp. 9949–9957.
[10] L. Bergen, T. J. O’Donnell, and D. Bahdanau, “Systematic generalization with edge transformers,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 1390–1402.
[11] D. Bahdanau et al., “CLOSURE: Assessing systematic generalization of CLEVR models,” 2022, arXiv: 1912.05783v2.
[12] V. D’Amario, T. Sasaki, and X. Boix, “How modular should neural module networks be for systematic generalization?,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 23374–23385.
[13] J. Johnson, B. Harirhan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and A. Kembhavi, “Don’t just assume; look and answer: Overcoming priors for visual question answering,” in Proc. 32nd IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 6700–6709.
[14] V. Goyal, T. Khot, A. Agrawal, D. Summers-Stay, D. Batra, and D. Parikh, “Making the V in VQA matter: Elevating the role of image understanding in visual question answering,” in Proc. 32nd IEEE/ACM Conf. Comput. Vis. Pattern Recognit., 2019, pp. 398–414.
[15] A. Agrawal, D. Batra, D. Parikh, and A. Kemnhavi, “Don’t just assume; look and answer: Overcoming priors for visual question answering,” in Proc. 31st IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2018, pp. 4942–4950.
[16] V. D’Amario et al., “Multimodal chatbot for medical applications: An experimental study of GPT-4/5,” 2023, arXiv:2310.19061. [Online]. Available: https://doi.org/10.48850/arXiv.2310.19061
[17] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, “Neural module networks,” in Proc. 29th IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2016, pp. 39–48.
[18] K. Yi, J. Wu, C. Gan, A. Torralba, P. Kohli, and J. B. Tenenbaum, “Neural-symbolic VQA: Disentangling reasoning from vision and language understanding,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 1039–1050.
[19] W. Chen, Z. Gan, L. Li, Y. Cheng, W. Wang, and J. Liu, “Meta module network for compositional visual reasoning,” in Proc. IEEE Winter Conf. Appl. Comput. Vis., 2021, pp. 655–664.
[20] D. Mascharka, P. Tran, R. Soklaski, and A. Majumdar, “Transparency by design: Closing the gap between performance and interpretability in visual reasoning,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 4942–4950.
[21] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu, “The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision,” in Proc. Int. Conf. Learn. Representations, 2019.
[22] A. Dosovitskiy et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” in Proc. Int. Conf. Learn. Representations, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy
[23] R. Hu, J. Andreas, M. Rohrbach, T. Darrell, and K. Saenko, “Learning to reason: End-to-end module networks for visual question answering,” in Proc. 16th IEEE Int. Conf. Comput. Vis., 2017, pp. 804–813.
[24] P. Anderson et al., “Bottom-up and top-down attention for image captioning and visual question answering,” in Proc. 31st IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2018, pp. 6077–6086.
[25] S. Sen, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in Proc. Adv. Neural Inf. Process. Syst., 2015, pp. 91–99.
[26] R. Krishna et al., “Visual genome: Connecting language and vision using crowdsourced dense image annotations,” Int. J. Comput. Vis., vol. 123, pp. 32–73, 2016.
[27] G. Li, N. Duan, Y. Fang, M. Gong, D. Jiang, and M. Zhou, “Unicoder-VL: A universal encoder for vision and language by cross-modal pre-training,” in Proc. 34th AAAI Conf. Artif. Intell., 2020, pp. 11336–11344.
[28] A. R. Akula, V. Jampani, S. Chakkpinyo, and S.-C. Zhu, “Robust visual reasoning via language guided neural module networks,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 11041–11053. [Online]. Available: https://proceedings.neurips.cc/paper/2021/hash/5bd53571b97884635d13910db49626bc-Abstract.html

Vanessa D’Amario received the bachelor’s and master’s degrees in physics from the Università degli Studi di Genova, and the PhD degree in computer science from the Massachusetts Institute of Technology, in 2020. She is an assistant professor with the Massachusetts Institute of Technology from 2018 to 2019. His current research interests include visual scene understanding, human-machine interaction, and their applications.

Moyuru Yamada received the PhD degree in mechanical engineering from the Toyohashi University of Technology, Japan, in 2012. He is the principal researcher with the Fujitsu Research of India Private Limited. After he joined Fujitsu Laboratories Ltd. in 2012, he was a visiting scientist with the Massachusetts Institute of Technology from 2018 to 2019. His current research interests include visual scene understanding, machine learning, its application to healthcare, and neuroscience.
Kentaro Takemoto received the bachelor’s and master’s degrees in information science from the University of Tokyo, Japan, in 2018 and 2020, respectively. He joined Fujitsu Laboratories Ltd. in 2020 and he is currently a researcher with Fujitsu Limited. His current research interests include computer vision, natural language processing, deep learning, and their applications.

Xavier Boix received the PhD degree in machine learning from ETH Zurich, in 2014, and continued his postdoctoral training with MIT as part of the multidisciplinary Center for Brains, Minds, and Machines. He is the director of research of the Self-Improving Machine Learning group with Fujitsu Research of America. Prior to that, he was a research scientist with MIT, where he led a research group that investigated biologically-inspired machine learning. His research focuses on building the next generation of intelligent machines while also seeking to understand them. He integrates insights and techniques from theoretical machine learning, deep neural network engineering, and neuroscience to achieve this goal.

Tomotake Sasaki (Member, IEEE) received the bachelor’s degree in science from Waseda University, Japan, in 2004, and the master’s and PhD degrees in information science and technology from the University of Tokyo, Japan, in 2006 and 2010, respectively. After working with the University of Tokyo as a project academic support specialist, he joined Fujitsu Laboratories Ltd. in 2010. He was a visiting scientist with the Massachusetts Institute of Technology from 2017 to 2018. He is a recipient of the 73rd IEEJ Academic Promotion Award Technical Development Award. At the time this work was conducted, he was a senior researcher with Fujitsu Limited, and a research affiliate with Massachusetts Institute of Technology and Center for Brains, Minds and Machines. He is currently a student with Department of Applied Electronics Production, Tokyo Metropolitan, Chuo-Johoku Vocational Skills Development Center/Japan Electronics College. His current research interests include control theory, reinforcement learning, deep learning and their applications. He is a member of SICE, ISCIE, and IEEJ.