Self-Modularized Transformer: Learn to Modularize Networks for Systematic Generalization

Yuichi Kamata[1], Moyuru Yamada[1] and Takayuki Okatani[2]
[1]*Fujitsu Ltd., Kawasaki, Kanagawa, Japan*
[2]*Graduate School of Information Sciences, Tohoku University, Sendai, Japan*

Keywords: Neural Module Network, Systematic Generalization, Visual Question Answering.

Abstract: Visual Question Answering (VQA) is a task of answering questions about images that fundamentally requires systematic generalization capabilities, i.e., handling novel combinations of known visual attributes (e.g., color and shape) or visual sub-tasks (e.g., `FILTER` and `COUNT`). Recent researches report that Neural Module Networks (NMNs), which compose modules that tackle sub-tasks with a given layout, are a promising approach for the systematic generalization in VQA. However, their performance heavily relies on the human-designed sub-tasks and their layout. Despite being crucial for training, most datasets do not contain these annotations. Self-Modularized Transformer (SMT), a novel Transformer-based NMN that concurrently learns to decompose the question into the sub-tasks and compose modules without such annotations, is proposed to overcome this important limitation of NMNs. SMT outperforms the state-of-the-art NMNs and multi-modal Transformers for the systematic generalization to the novel combinations of the sub-tasks in VQA.

1 INTRODUCTION

Recent studies suggest that systematic generalization remains challenging for state-of-the-art neural network models. Systematic generalization is the ability to generalize novel compositions of known concepts beyond the training distribution (Lake and Baroni, 2018; Bahdanau et al., 2019; Ruis et al., 2020). Even successful models for in-distribution, e.g., Transformer (Vaswani et al., 2017), largely degrades the performance for systematic generalization (Yamada et al., 2022; Bergen et al., 2021).

Visual Question Answering (VQA) (Antol et al., 2015) is the task of answering questions about images. The core of VQA is complex visual reasoning, a composition of sub-tasks, e.g., `FIND`, `FILTER`, and `COUNT`. This compositional structure yields a distribution of image-question pairs of combinatorial size, training data cannot fully capture. Thus, VQA fundamentally requires systematic generalization capabilities.

Neural Module Networks (NMNs) show promising performance for systematic generalization in VQA (Johnson et al., 2017a; 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 a *module*. NMNs take a sequence of sub-tasks, i.e., a layout of the sub-tasks, as an input instead of the question. NMNs alleviate the gap between in-distribution generalization and systematic generalization.

The performance of NMNs, however, significantly depends on the human-designed sub-tasks and their layouts (equivalently, programs), i.e., how to design a library of modules that covers all questions in a target dataset and how to compose them to provide correct answers to the questions. These annotations are essential to training the NMNs but are not included in most datasets such as VQA v2.0 (Goyal et al., 2017). A program generator can also be used to convert the questions into the programs in the test phase (Johnson et al., 2017b), but the program–question pairs are needed to train it. Due to these crucial limitations, NMNs cannot apply their excellent systematic generalization capabilities to the datasets that do not contain the programs.

In this paper, to eliminate the above limitations of NMNs, we propose a novel Transformer-based NMN called the Self-Modularized Transformer (SMT) that simultaneously learns to decompose the question into the sub-tasks and compose modules without the program. This is a significant challenge because the network must determine if the modules, their composition, or both are incorrect from an error between the predicted and actual answer. Two losses are introduced to facilitate them. SMT outperforms the state-

Kamata, Y., Yamada, M. and Okatani, T.
Self-Modularized Transformer: Learn to Modularize Networks for Systematic Generalization.
DOI: 10.5220/0011682100003417
In *Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2023) - Volume 5: VISAPP*, pages 599-606
ISBN: 978-989-758-634-7; ISSN: 2184-4321

of-the-art NMNs and multi-modal Transformers for the systematic generalization to the novel combinations of the sub-tasks in VQA.

2 RELATED WORK

Neural Module Networks (NMNs) (Andreas et al., 2016b) are commonly used to solve complex visual reasoning tasks such as CLEVR (Johnson et al., 2017a). NMNs decompose a question into a sequence of sub-tasks (i.e., program). With ground truth (GT) programs NMNs have solved CLEVR perfectly (Yi et al., 2018; Shi et al., 2019). Program generator (Andreas et al., 2016a; Yi et al., 2018; Akula et al., 2021), which infers a program from a question, is proposed to apply the NMNs to the questions for which the GT programs are not provided. However, the question and GT program pairs are required to train the program generator.

Stack-NMN (Hu et al., 2018) detects weights of applicability for each module, and similar to our approach it learns the soft weights to select modules. While Stack-NMN implements modules specialized to the sub-tasks defined in CLEVR, we use larger modules consisting of Transformer blocks to realize flexible acquisition of sub-tasks through the training process without pre-defining them.

Multi-modal Transformers (Lu et al., 2019; Kamath et al., 2021; Tan and Bansal, 2019) are recent successful models on vision and language tasks (Goyal et al., 2017; Hudson and Manning, 2019; Yu et al., 2016), coupled with the effect of pretraining with large amounts of data (Krishna et al., 2017; Sharma et al., 2018), and are reported with high accuracy on CLEVR without using any GT programs as well (Kamath et al., 2021). The architecture of Transformer is also adopted for the program generator for NMNs (Chen et al., 2021). We use Transformer blocks to build modules for each sub-task.

The CLOSURE dataset (Bahdanau et al., 2020) provides novel combinations of sub-tasks in the CLEVR dataset. The authors of CLOSURE have also proposed Vector-NMN. Vector-NMN outperforms all the previous NMNs and achieves promising performance for systematic generalization with GT programs. Recently, some NMNs under the condition of using GT programs (Bahdanau et al., 2020; Yamada et al., 2022) or program generator (Akula et al., 2021) have improved the performance on CLOSURE. However, they still rely on the GT programs to train the model or program generator, unlike our approach.

3 APPROACH

In this section, we introduce Self-Modularized Transformer (SMT) that consists of a set of Transformer modules and a controller network to select the modules for the sub-task at each layer.

3.1 Architecture

Figure 1 depicts an overview of SMT, which is composed of Transformer blocks as modules. We design this architecture based on the preliminary experimental results (see Sec. 4.5). Conventional L-layer Transformer models stack L Transformer blocks sequentially, while our M-module SMT arranges all M Transformer blocks in parallel and uses that set of modules repeatedly to stack L layers (i.e., all Transformer blocks operate their sub-tasks L times). Transformer modules at l-th layer receive as their input the output of the previous layer S^{l-1} and produce a weighted summation of each module's output S^l_m:

$$\begin{aligned} S^l_m &= \mathrm{Transformer}_m(S^{l-1}), \\ S^l &= \sum_{m=1}^{M} w^l_m \cdot S^l_m, \end{aligned} \tag{1}$$

where w^l_m is a weight for the output of the m-th module at l-th layer.

Following the approach of conventional vision and language Transformers (Lu et al., 2019; Li et al., 2020), the image features $f = \{f_{(1)}, \cdots, f_{(O)}\}$ and regional positions $r = \{r_{(1)}, \cdots, r_{(O)}\}$ of the O objects extracted by the object detector are embedded by each of the linear layers $\mathrm{FC_I}$ and $\mathrm{FC_R}$, and they are summed up to create a sequence of object features with a layer normalization:

$$V = \mathrm{LayerNorm}(\mathrm{FC_I}(f) + \mathrm{FC_R}(r)). \tag{2}$$

Also, a sequence of question embeddings Q is created by converting W tokens of question sentences $q = \{q_{(1)}, \cdots, q_{(W)}\}$ with the embedding layer $\mathrm{E_w}$ and adding the positional embeddings. In addition, the question sentence q is fed into a dependency parser to obtain dependency tags and dependency heads for each word in the sentences as shown in Figure 2. The dependency tag is converted to embeddings $T = \{T_{(1)}, \cdots, T_{(W)}\}$, and is added to question embeddings Q to obtain extended question embeddings:

$$Q_{\mathrm{ex}} = \mathrm{LayerNorm}(Q + T). \tag{3}$$

The dependency heads are transformed to a matrix A_{head} that aligns attention masks for the Transformer modules such that only a single phrase of dependency parsing is present sequentially for the modules of each

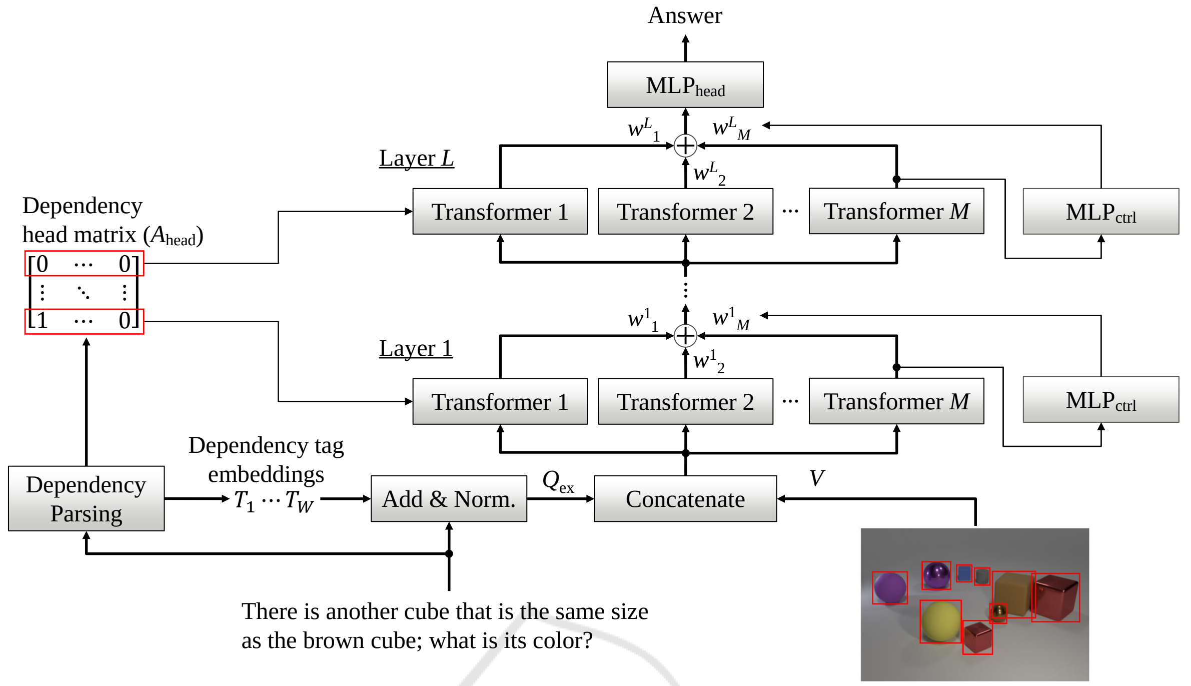


Figure 1: Overview of Self-Modularized Transformer (SMT). It takes an image and a question as the input and outputs the answer by selecting one from predefined candidates. All Transformer modules are aligned in parallel, and that set of modules is stacked in L layers repeatedly. A dependency parser divides the question into sentences for the modules. Selections of the modules are determined by a controller network (MLP$_{\mathrm{ctrl}}$).

layer. For example, in the case of the question sentence $q = \{\text{There is another cube that } \cdots\}$, the attention mask at the position of layer 1 of $[1\ 1\ 0\ 1\ 0\ \cdots]$ indicates the phrase "There is cube" as hard attention to each word in the question. To keep the model size (i.e., the number of layers) the same in the whole dataset, every row of the dependency head matrix A_{head} which exceeds the number of the phrase is padded with the attention mask of the language tokens set to zero.

We add embeddings of special tokens to the head and tail of the object embeddings V and the extended question embeddings Q_{ex} (i.e, tokens of $\langle\mathrm{BOS}\rangle$, $\langle\mathrm{EOS}\rangle$, $\langle\mathrm{BOI}\rangle$, and $\langle\mathrm{EOI}\rangle$). Finally, the initial input for the first layer is as follows:

$$\begin{aligned} S^0 = \{&\mathrm{E_w}(\langle\mathrm{BOS}\rangle) \frown Q_{\mathrm{ex}} \frown \mathrm{E_w}(\langle\mathrm{EOS}\rangle) \\ &\frown \mathrm{E_w}(\langle\mathrm{BOI}\rangle) \frown V \frown \mathrm{E_w}(\langle\mathrm{EOI}\rangle)\} \end{aligned} \tag{4}$$

where "$\frown$" denotes a concatenation operation and $\mathrm{E_w}$ indicates the embedding layer for question tokens.

The transformed first tokens (i.e., at the position of $\langle\mathrm{BOS}\rangle$) obtained from each module are fed into a controller network MLP$_{\mathrm{ctrl}}$ to predict the weights for the module selection. The controller network is a multi-layer perceptron (MLP). The weights of modules at the l-th layer are

$$\begin{aligned} \{w^l_1, \cdots, w^l_M\} = \mathrm{softmax}(\{&\mathrm{MLP_{ctrl}}(S^l_1[0]), \cdots, \\ &\mathrm{MLP_{ctrl}}(S^l_M[0])\}) \end{aligned} \tag{5}$$

where $S^l_m[0]$ denotes the first token in the output of the m-th module.

3.2 Losses for Module Selectivity

We introduce two losses, i.e., a sparsity loss and a diversity loss to improve the module selectivity.

According to conventional approaches, the loss for the VQA task L_{T} uses the cross-entropy loss as classification with predetermined answer candidates. The first token in the output of the last layer $S^L[0]$ is fed into the head network MLP$_{\mathrm{head}}$ for classification, and class probabilities of the answers p_a are identified via the softmax function:

$$p_a = \mathrm{softmax}(\mathrm{MLP_{head}}(S^L[0])). \tag{6}$$

Then, we adopt a sparsity loss so that each layer has a high module selectivity:

$$L_{\mathrm{S}} = \frac{1}{B}\sum_{b=1}^{B}\sum_{l=1}^{L}\sum_{m=1}^{M}\left(w^{l}_{m}{}^{(b)}\right)^{1/2} - L \tag{7}$$

where B is batch size. For each layer, since the weights are non-negative and add up to 1, the sum

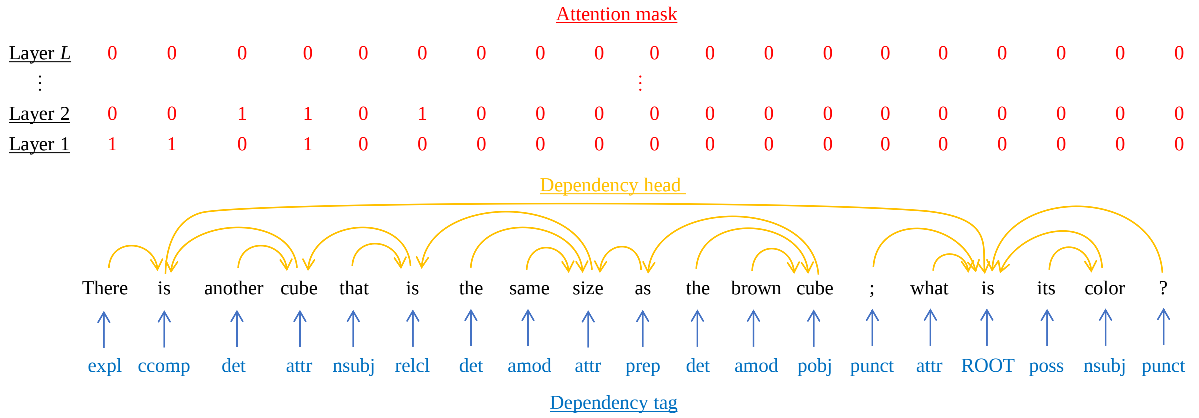

Attention mask
Layer L 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
⋮ ⋮
Layer 2 0 0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
Layer 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Dependency head
There is another cube that is the same size as the brown cube ; what is its color ?
expl ccomp det attr nsubj relcl det amod attr prep det amod pobj punct attr ROOT poss nsubj punct
Dependency tag

Figure 2: An example of the approach to dependency parsing. Dependency tag embeddings are added to word embeddings, and dependency heads transform into attention masks to present a single phrase sequentially for each layer.

of that weights in powers less than one is limited to 1 (i.e., sparse) or more.

In addition, a diversity loss is added so that all the modules can be selected uniformly ($\approx L/M$) throughout the dataset:

$$L_{\mathrm{D}} = \frac{1}{B}\sum_{m=1}^{M}\left|\sum_{b=1}^{B}\sum_{l=1}^{L} w_m^{l\,(b)} - \frac{L}{M}\right| \tag{8}$$

Then, the final loss function is

$$L_{\mathrm{T}} + \gamma L_{\mathrm{S}} + \varepsilon L_{\mathrm{D}} \tag{9}$$

where γ and ε are coefficients for adjusting the sparsity and diversity.

4 EXPERIMENTS

4.1 Preprocess and Parameters

We use Faster R-CNN with ResNet-101 backbone pretrained by the bottom-up attention method (Anderson et al., 2018) to extract object features from the input images, and we set a fixed number of objects ($O = 36$). The byte-level Byte-Pair-Encoding and Transformer block (1032 hidden units and 12 attention heads) as RoBERTa (Liu et al., 2019) are applied to our model. We use spaCy[1] to extract the dependencies from the input question. Moreover, the number of layers is set to 22, the maximum number of phrases in the question sentence obtained by dependency parsing. We set γ and ε of loss adjustments to $5e^{-4}$ and $5e^{-5}$, respectively, and train the model using Adam optimizer with the maximum learning rate of $8e^{-5}$ by 4,000 steps warm-up and linear decay. The training is executed for 40 epochs using 16 A100 GPUs with a batch size of 512 in total. Since the loss decreases rapidly, the gradients from the controller $\mathrm{MLP}_{\mathrm{ctrl}}$ to the first tokens are not allowed to back-propagate.

4.2 Datasets

CLEVR is a dataset of VQA tasks that consists of images depicting simple 3D objects with a finite number of attributes and questions of combinations of fundamental sub-tasks, and is used to focus on the evaluation of reasoning and systematic generalization. It contains three splits of data, i.e, a training set of 70k images and 700k questions, a validation set of 15k images and 150k questions, and a test set of 15k images and 15k questions.

CLOSURE is a complementary dataset to CLEVR that consists of images in the CLEVR validation set and questions of novel combinations of the same fundamental sub-tasks as CLEVR. It provides questions of a validation set, a test set, and a small training set for few-shot learning. Validation and test sets consist of seven cases of questions depending on the combination of phrases, and each case of questions contains 36k questions. By training on CLEVR and evaluating on CLOSURE, it is possible to evaluate the ability of systematic generalization on novel combinations of known sub-tasks.

4.3 Results

Table 1 shows the evaluation results on the CLEVR validation set and the CLOSURE test set for the models trained on the CLEVR training set. Our SMT achieved new state-of-the-art accuracy on CLOSURE compared to the models without GT programs cited in the first-row group. Moreover, our performance on CLOSURE cannot be achieved by simply inputting tokens of questions and images into 22-layer

[1] https://spacy.io/

Table 1: Performance on the systematic generalization of novel linguistic combinations. For the proposed model, the mean ± standard deviation over five runs is reported. The result of MDETR from (Yamada et al., 2022) and the others from (Akula et al., 2021) is cited. [†]Variance is reported.

Model	use GT programs training	 evaluation	CLEVR val	CLOSURE
MAC (Hudson and Manning, 2018)			99.1	71.6
ViLBERT (Lu et al., 2019)			95.3	51.2
MDETR (Kamath et al., 2021)			**99.7**	53.3
RoBERTa, 22-layer			98.5 ± 0.03	65.7 ± 2.1
SMT(**Ours**)			98.3 ± 0.05	**77.0** ± 2.2
NS-VQA (Yi et al., 2018)	✓		**99.2**	76.4
PG-Vector-NMN (Bahdanau et al., 2020)	✓		98.8	71.0
NMN w/ CoSAtt (Akula et al., 2021)	✓		98.9 ± 0.1[†]	**88.0** ± 0.2[†]

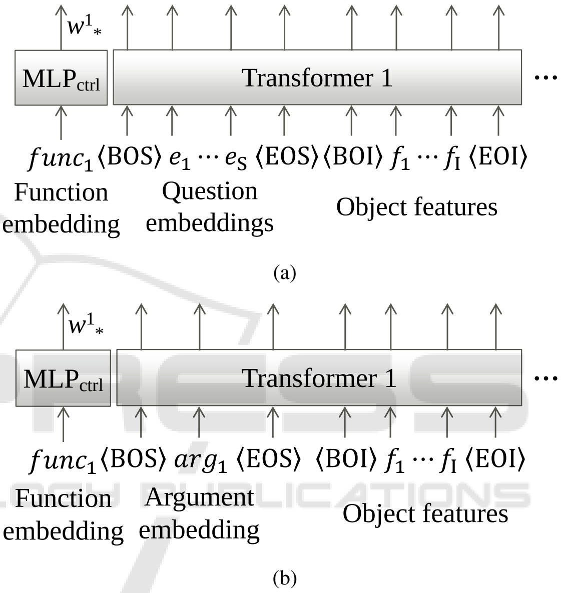
RoBERTa, as shown in the second-row group. On the other hand, in comparison to the models using the program generator cited in the third-row group, the proposed model is largely inferior to NMN w/ CoSAtt but is comparable and superior to NS-VQA and PG-Vector-NMN. Therefore our model shows a promising performance for systematic generalization on novel linguistic compositions without any program annotations.

4.4 Ablation Studies

Table 2 shows the effect of approaches applied to SMT. If the information of dependency parsing is not used (the second-row), the performance of CLOSURE is significantly decreased. This result shows that the decomposition of phrases based on dependency parsing instead of GT programs greatly contributes to systematic generalization for novel combinations of linguistic constructs. Alternatively, SMT causes degradation in “and_mat_spa”, a major issue to address. In the case our sparsity loss and diversity loss are not set (the third row), the performance on CLOSURE is slightly decreased; therefore, we believe that improving the module selectivity with our proposed losses contributes to the systematic generalization performance.

4.5 Preliminary Experiments

We first investigated whether the proposed approach can obtain a systematic generalization using GT programs. To this end, we evaluate the performance on CLOSURE with the following two models. Figure 3 illustrates the two models that use GT programs. In Fig. 3 (a), we input the embedding of the function name in the GT programs (e.g., `FIND`, `FILTER`, or `COUNT`) to the controller $\mathrm{MLP}_{\mathrm{ctrl}}$ instead of the first token in each module’s output. This setting gives the oracle for the selection of modules. In Fig. 3 (b), in addition to the previous setting the modules take the embedding of the argument name in the GT programs (e.g., “red”, “left”, or “sphere”) as linguistic input instead of the question. This setting gives the oracle for linguistic compositions of the question as well. The number of layers is set to 26 (i.e., the maximum program length), and the functions and the arguments fill with a ⟨PAD⟩ token (i.e., padded to align the size of the input tokens).


$w^1{}_*$
$\mathrm{MLP}_{\mathrm{ctrl}}$
Transformer 1
⋯
$func_1 \langle\mathrm{BOS}\rangle\ e_1 \cdots e_S\ \langle\mathrm{EOS}\rangle\langle\mathrm{BOI}\rangle\ f_1 \cdots f_I\ \langle\mathrm{EOI}\rangle$
Function embedding
Question embeddings
Object features
$w^1{}_*$
$\mathrm{MLP}_{\mathrm{ctrl}}$
Transformer 1
⋯
$func_1 \langle\mathrm{BOS}\rangle\ arg_1\ \langle\mathrm{EOS}\rangle\ \langle\mathrm{BOI}\rangle\ f_1 \cdots f_I\ \langle\mathrm{EOI}\rangle$
Function embedding
Argument embedding
Object features

(a)

(b)

Figure 3: Illustration of models using the GT programs for preliminary experiments. (a) Instead of ⟨BOS⟩ tokens, an embedding of the GT function is input to $\mathrm{MLP}_{\mathrm{ctrl}}$; (b) furthermore, instead of question tokens, an embedding of the GT argument is input to each module.

The results in Table 3 show that the performance on CLOSURE improves enough, except for the cases of “or_mat” and “or_mat_spa”, by using both functions and arguments (i.e., with GT programs) for selecting modules with the functions and for restricting the tokens of linguistic input with the arguments.

Table 2: Ablation studies for SMT. The mean ± standard deviation over five runs is reported.

Model	CLEVR	CLOSURE							
	val	and_mat_spa	or_mat	or_mat_spa	compare_mat	compare_mat_spa	embed_spa_mat	embed_mat_spa	overall
SMT	98.3 ± 0.05	66.1 ± 16.1	**77.4** ± 3.4	46.5 ± 1.8	**93.3** ± 2.3	**94.6** ± 1.4	96.1 ± 0.8	**64.8** ± 2.0	**77.0** ± 2.2
w/o parsing	**98.5** ± 0.04	**91.6** ± 3.6	26.0 ± 5.6	36.4 ± 5.9	81.3 ± 6.2	77.8 ± 4.0	**98.5** ± 0.2	62.7 ± 0.7	67.8 ± 2.6
w/o S&D loss	98.3 ± 0.01	60.9 ± 9.2	77.3 ± 3.9	**48.4** ± 4.8	89.9 ± 1.6	92.0 ± 2.1	96.1 ± 0.7	62.6 ± 0.8	75.3 ± 1.8

Table 3: Preliminary experiments leading to our proposed model.

Model	and_mat_spa	or_mat	or_mat_spa	compare_mat	compare_mat_spa	embed_spa_mat	embed_mat_spa
SMT, 26-layer w/o parsing	86.0	25.6	30.1	83.3	79.5	98.3	62.1
w/ GT func.	90.3	41.3	44.0	80.7	76.5	98.6	63.7
w/ GT prog.	**97.4**	**59.0**	**44.7**	**93.8**	**95.9**	**99.1**	**89.3**

4.6 Visualization of Module Selectivity

For each question case on the CLOSURE test set, we visualize the weight maps of modules' selection in SMT with different loss adjustments for sparsity and diversity (i.e., γ and ε) in Fig. 4. First, the weights of modules' selection show different maps according to the cases of question in CLOSURE, and the selectivity of modules becomes higher as increasing loss adjustments for sparsity and diversity. We show the original setting in Fig. 4 (a) and a higher selectivity setting in Fig 4 (b). With the high module selectivity, the accuracy of CLOSURE appears to be slightly reduced. It is hypothesized that their learning capacity will be constrained if the arrangement of modules is strictly controlled before they understand how to complete the task to a certain extent through trial and error.

5 CONCLUSION

We proposed Self-Modularized Transformer (SMT) that learns to decompose questions into sub-tasks and compose modules without those annotations. SMT outperforms a previous state-of-the-art NMN without the human-annotated programs on the CLOSURE dataset that provide novel combinations of the sub-tasks. Our analysis reveals that restricting input tokens to be a part of the question sentence is the key to learning the sub-tasks effectively. This approach helps to achieve systemic generalization even for datasets not provided with human-designed annotations, to which NMNs cannot apply.

With the proposed SMT, the standard deviation of CLOSURE accuracy is large ("and_mat_spa", in particular); therefore, we intend to improve the control approach for module selection (MLP_{ctrl}) in our future work. In addition, since dependency parsing does not significantly contribute to the performance improvement on "and_mat_spa" and "embed_ mat_spa", it is necessary to analyze key factors to learning those sub-tasks individually.

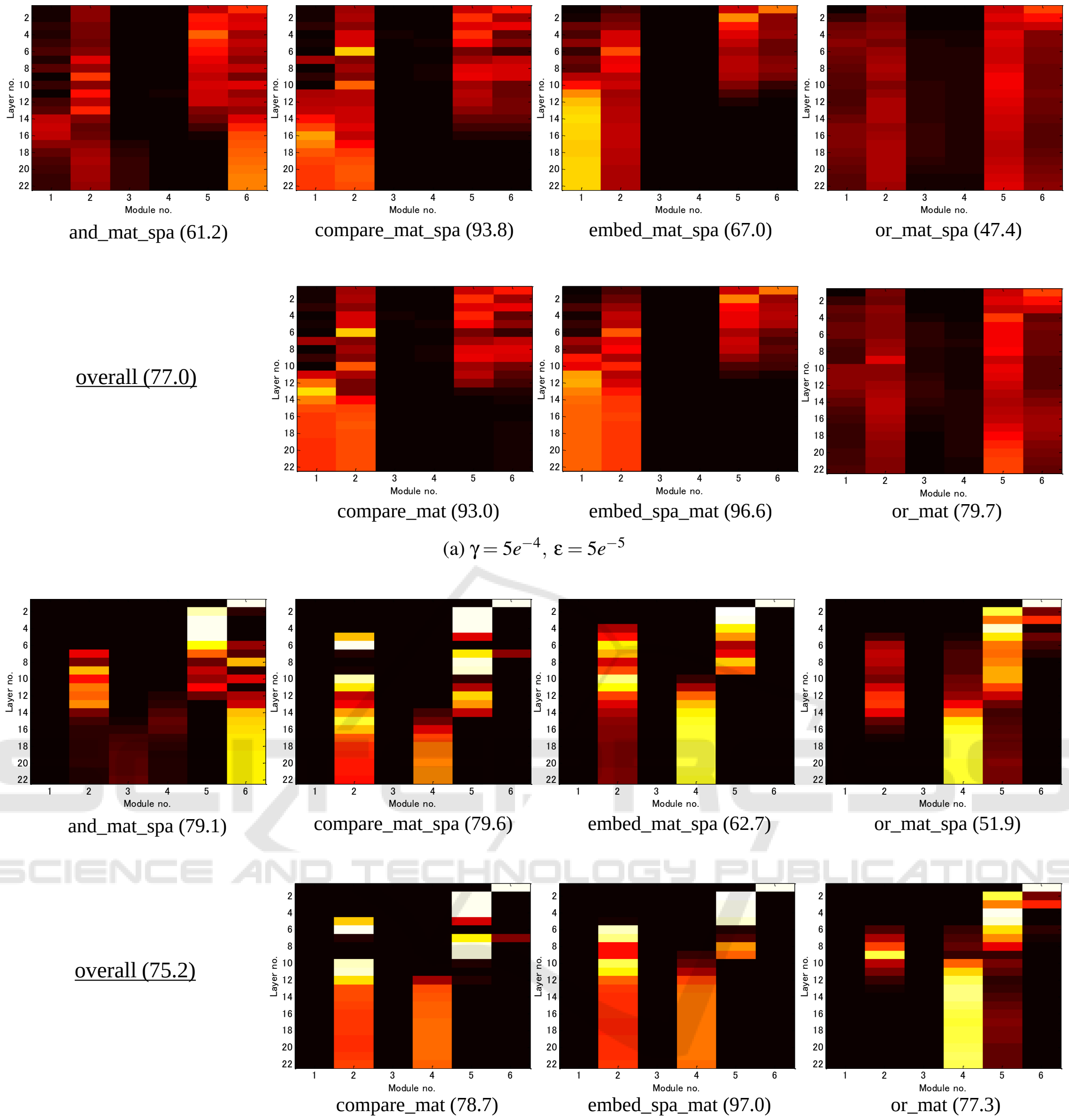
REFERENCES

Akula, A., Jampani, V., Changpinyo, S., and Zhu, S.-C. (2021). Robust visual reasoning via language guided neural module networks. In *NeurIPS*, pages 11041–11053.

Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., and Zhang, L. (2018). Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, pages 6077–6086.

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. (2016a). Learning to compose neural networks for question answering. In *NAACL*, pages 1545–1554.

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. (2016b). Neural module networks. In *CVPR*, pages 39–48.



and_mat_spa (61.2)

compare_mat_spa (93.8)

embed_mat_spa (67.0)

or_mat_spa (47.4)

overall (77.0)

compare_mat (93.0)

embed_spa_mat (96.6)

or_mat (79.7)

(a) $\gamma = 5e^{-4}$, $\varepsilon = 5e^{-5}$

and_mat_spa (79.1)

compare_mat_spa (79.6)

embed_mat_spa (62.7)

or_mat_spa (51.9)

overall (75.2)

compare_mat (78.7)

embed_spa_mat (97.0)

or_mat (77.3)

(b) $\gamma = 2e^{-3}$, $\varepsilon = 1e^{-3}$

Figure 4: Visualization of the mean weight maps of modules' selection for each question case on CLOSURE test set. Numbers in brackets indicate accuracy in each question case on CLOSURE.

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Lawrence Zitnick, C., and Parikh, D. (2015). VQA: Visual Question Answering. In *ICCV*, pages 2425–2433.

Bahdanau, D., de Vries, H., O'Donnell, T. J., Murty, S., Beaudoin, P., Bengio, Y., and Courville, A. C. (2020). CLOSURE: assessing systematic generalization of CLEVR models. *arXiv preprint, arXiv:1912.05783*.

Bahdanau, D., Murty, S., Noukhovitch, M., Nguyen, T. H., de Vries, H., and Courville, A. (2019). Systematic generalization: What is required and can it be learned? In *ICLR*.

Bergen, L., O'Donnell, T. J., and Bahdanau, D. (2021). Systematic generalization with edge transformers. In *NeurIPS*, pages 1390–1402.

Chen, W., Gan, Z., Li, L., Cheng, Y., Wang, W., and Liu, J. (2021). Meta module network for compositional visual reasoning. In *WACV*, pages 655–664.

D'Amario, V., Sasaki, T., and Boix, X. (2021). How modular should neural module networks be for systematic generalization? In *NeurIPS*, pages 23374–23385.

Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., and Parikh, D. (2017). Making the V in VQA matter: Elevating the role of image understanding in Visual

Question Answering. In *CVPR*, pages 6904–6913.

Hu, R., Andreas, J., Darrell, T., and Saenko, K. (2018). Explainable neural computation via stack neural module networks. In *ECCV*, pages 53–69.

Hudson, D. A. and Manning, C. D. (2018). Compositional attention networks for machine reasoning. In *ICLR*.

Hudson, D. A. and Manning, C. D. (2019). GQA: A new dataset for real-world visual reasoning and compositional question answering. In *CVPR*, pages 6693–6702.

Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., and Girshick, R. (2017a). CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, pages 2901–2910.

Johnson, J., Hariharan, B., van der Maaten, L., Hoffman, J., Fei-Fei, L., Zitnick, C. L., and Girshick, R. (2017b). Inferring and executing programs for visual reasoning. In *ICCV*.

Kamath, A., Singh, M., LeCun, Y., Synnaeve, G., Misra, I., and Carion, N. (2021). MDETR-modulated detection for end-to-end multi-modal understanding. In *ICCV*, pages 1780–1790.

Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.-J., Shamma, D. A., et al. (2017). Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. Journal of Computer Vision*, 123(1):32–73.

Lake, B. and Baroni, M. (2018). Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *ICML*, pages 2873–2882.

Li, L. H., Yatskar, M., Yin, D., Hsieh, C.-J., and Chang, K.-W. (2020). What does BERT with vision look at? In *ACL*, pages 5265–5275.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Lu, J., Batra, D., Parikh, D., and Lee, S. (2019). ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, pages 13–23.

Ruis, L., Andreas, J., Baroni, M., Bouchacourt, D., and Lake, B. M. (2020). A benchmark for systematic generalization in grounded language understanding. In *NeurIPS*, pages 19861–19872.

Sharma, P., Ding, N., Goodman, S., and Soricut, R. (2018). Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, volume 1, pages 2556–2565.

Shi, J., Zhang, H., and Li, J. (2019). Explainable and explicit visual reasoning over scene graphs. In *CVPR*, pages 8376–8384.

Tan, H. and Bansal, M. (2019). LXMERT: Learning cross-modality encoder representations from transformers. In *EMNLP and Proc. of the Conference on Empirical Methods in Natural Language Processing and ICNLP*, pages 5100–5111.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *NeurIPS*, pages 5998–6008.

Yamada, M., D'Amario, V., Takemoto, K., Boix, X., and Sasaki, T. (2022). Transformer module networks for systematic generalization in visual question answering. *arXiv preprint arXiv:2201.11316*.

Yi, K., Wu, J., Gan, C., Torralba, A., Kohli, P., and Tenenbaum, J. (2018). Neural-symbolic VQA: Disentangling reasoning from vision and language understanding. In *NeurIPS*, pages 1039–1050.

Yu, L., Poirson, P., Yang, S., Berg, A. C., and Berg, T. L. (2016). Modeling context in referring expressions. In *ECCV*, pages 69–85.