The Scattering Compositional Learner: Discovering Objects, Attributes, Relationships in Analogical Reasoning

Anonymous authors
Paper under double-blind review

Abstract

In this work, we focus on an analogical reasoning task that contains rich compositional structures, Raven's Progressive Matrices (RPM). To discover compositional structures of the data, we propose the Scattering Compositional Learner (SCL), an architecture that composes neural networks in a sequence. Our SCL achieves state-of-the-art performance on two RPM datasets, with a 48.7% relative improvement on Balanced-RAVEN and 26.4% on PGM over the previous state-of-the-art. We additionally show that our model discovers compositional representations of objects' attributes (e.g., shape color, size), and their relationships (e.g., progression, union). We also find that the compositional representation makes the SCL significantly more robust to test-time domain shifts and greatly improves zero-shot generalization to previously unseen analogies.

1 Introduction

As humans, we routinely use compositions of simpler concepts to make sense of the world. For instance, consider the mundane feature of everyday language which lets us compose concepts for living things and the activities they are engaged in: "a running cat", "a running elephant", "a sleeping cat", "a sleeping elephant". By exploiting the compositional structure of concepts, human learners enjoy strong generalization from limited data in various domains (Lake et al., 2018), making "infinite use of finite means" (Chomsky, 1965). We are capable of comprehending novel situations as compositions of simple and known elements. For example, we can visualize the concept of "a flying elephant" despite never having seen one, by combining the familiar notions of "flying" and "an elephant".

The state-of-the-art deep neural networks have achieved superhuman performance in many vision and language tasks. However, discovering the underlying structure remains a challenge, even for modern deep learning methods (Fodor and Pylyshyn, 1988; Lake and Baroni, 2018; Loula et al., 2018; Bahdanau et al., 2019b;a; Keysers et al., 2020). Previous studies (Lake and Baroni, 2018; Loula et al., 2018) attribute poor generalization to the lack of composition rules in the neural architecture design. In this paper, we introduce a novel neural network architecture to learn a compositional hidden representation. The explicit factorization allows the network to generalize to novel situations.

In this paper, we focus on an analogy completion task that contains rich compositional structures, the Raven's Progressive Matrices (RPM) task (Raven, 1936; John and Raven, 2003). In one of the RPM datasets, RAVEN (Zhang et al., 2019a), one is given a 3×3 matrix of panels, with the last panel missing. The task is to choose a panel from eight candidates to fill in the missing panel so that three rows form consistent analogies (see Fig. 2 for an example). To succeed in this task, the model needs to be capable of extracting compositions of three distinct components: (1) the objects in every panel, (2) the visual attributes (such as color, size, and shape) of these objects, and (3) the relationships among attributes, such as "progression", "union", and "constant".

In order to discover rich compositional structures of the data, we propose to build compositions of neural networks. In particular, we compose three types of networks: the object networks $\{\mathcal{N}_i^o\}$, the attribute networks $\{\mathcal{N}_i^a\}$, and the relationship networks $\{\mathcal{N}_i^r\}$. Each composition $\mathcal{N}_k^r \circ \mathcal{N}_j^a \circ \mathcal{N}_i^o$ computes if the kth relationship holds among the jth attributes of the ith objects of the input. We computes all possible compositions of those three neural networks $\{\mathcal{N}_k^r \circ \mathcal{N}_j^a \circ \mathcal{N}_i^o\}_{i,j,k}$ to predict
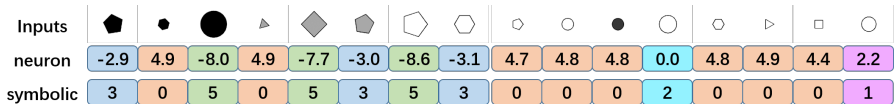
Inputs	⬠	•	●	▲	◈	⬟	⬡	⬡	◌	○	●	○	○	▷	▫	○
neuron	-2.9	4.9	-8.0	4.9	-7.7	-3.0	-8.6	-3.1	4.7	4.8	4.8	0.0	4.8	4.9	4.4	2.2
symbolic	3	0	5	0	5	3	5	3	0	0	0	2	0	0	0	1

Figure 1: SCL learned a "size neuron" that is highly predictive of the input object size. From top to bottom: input images from RAVEN, the neuron's activations of each image, the labels of symbolic representation of "size" of each image. The neuron learned to represent "size" by an approximate linear transformation: $y \approx -\frac{5}{2}x + 5$.

the label. By explicitly computing all possible compositions of three types of neural networks, we force neural networks of each type to be compatible with each other. For example, a relationship neural network that extracts "progression" needs to be compatible with all attribute neural networks. Namely, it needs to recognize "progression" for any input feature, regardless of whether the input feature represents "color", "size" or "shape". Hence, it is encouraged to learn the exact notion of "progression" (a process of gradual advancement), instead of an attribute-dependent concept (e.g., "color getting darker"). The same reasoning applies to all of the networks of three types. Therefore, the network of each type learns precisely its intended functionality, and the underlying compositional structures are discovered.

We term our proposed model Scattering Compositional Learner (SCL). Our SCL achieved the state-of-the-art in two datasets on the RPM task: a 48.7% relative improvement on Balanced-RAVEN (Hu et al., 2020) and 26.4% on PGM (Santoro et al., 2018) over the previous state-of-the-art. We further show SCL trained with standard end-to-end backpropagation discovers interpretable factorized structures on the Balanced-RAVEN dataset. We found SCL learns to extract symbolic features such as the shape, color, and size for each object. We highlight an example of the learned representation in Figure 1, which shows the responses of one of the learned neurons to a set of inputs. This neuron learned a linear transformation of the scalar representation of the symbolic attribute "size", captured by the equation $y \approx -\frac{5}{2}x + 5$. As a consequence of learning compositional structures, we further demonstrate that our model is also capable of generalizing to novel test distributions, whereas the previous state-of-the-art model suffered severely from the distribution shift.

2 RELATED WORKS

There has been an old debate over the limits of connectionism for compositional generalization. In a set of influential and controversial papers, Jerry Fodor and other researchers criticized neural networks for failing to model the intricacy of mind because they cannot capture systematic compositionality (Fodor and Pylyshyn, 1988; Marcus, 1998; 2001; Fodor and Lepore, 2002). These issues have received less attention since deep learning revolutionized various domains. However, there have been a few papers recently bringing up these issues again, after observing non-robustness in the systematic generalization of neural networks (Lake and Baroni, 2018; Loula et al., 2018; Bahdanau et al., 2019b;a; Keysers et al., 2020). Our work is an attempt to address the issues of systematic generalization with an architectural bias that respects compositions. We advocate for the same principle as those methods in the past that add modularity for better generalization, such as Neural Module Networks(NMN) (Andreas et al., 2015), NTPT (Gaunt et al., 2017), MAC (Hudson and Manning, 2018), Neural Symbolic Concept Learner (Mao et al., 2019), CRL (Chang et al., 2019). Our work is differentiated from the past work, in that our architecture is specifically designed for learning compositions of a sequence of neural modules, whereas the past work relied on symbolic information (e.g., language in NMN, NSCL, MAC; source code in NTPT), or non-differentiable search (e.g., NSCL, CRL) to compose neural modules in task-specific layout. A more detailed comparison of these approaches is provided in Appendix D.

SCL also shares some similarities with the group convolution architecture (Krizhevsky et al., 2012) and its followup work such as ResNeXt (Xie et al., 2017). Similarly, they perform the "split" operation, dividing the computation into multiple groups. However, the transformations in these architectures do not share parameters among groups, whereas "share" is a key component in our architecture (see Scattering Transformation in Section 3). We present ablation studies to show that our model without parameter sharing could not learn at all (in Appendix E), demonstrating the importance of this distinction.

Raven's Progressive Matrices (Raven, 1936) (RPM) are studied extensively in cognitive science (Carpenter et al., 1990). It is considered as a diagnostic of human's abstract and structural reasoning
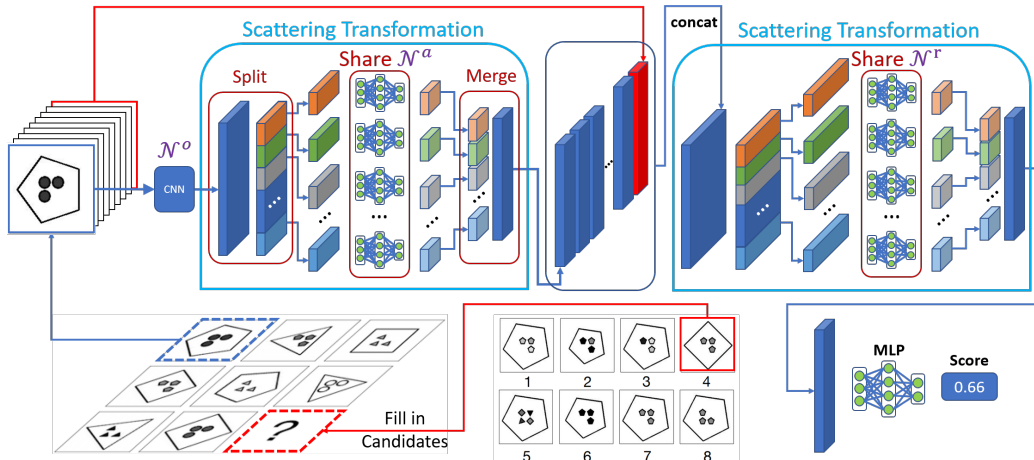
Figure 2: An illustration of the proposed architecture SCL with the scattering transformation.

ability (Carpenter et al., 1990), and characterizes a central feature of high-level intelligence, namely fluid intelligence (Jaeggi et al., 2008). In the machine learning community, following on the work by Wang and Su (2015), Santoro et al. (2018) introduced the Procedurally Generating Matrices (PGM) dataset of RPM-like problems. They proposed an advanced version of Relational Network (Santoro et al., 2017), Wild Relational Network (WReN), and studied its generalization on this dataset. Followup work (Hill et al., 2019) showed the importance of candidate sets in learning analogies. Zhang et al. (2019a) extended PGM and created the dataset RAVEN, with a larger number of relationships in each problem, a task that is considered to be harder than PGM in symbolic reasoning. Several methods have been proposed since then to solve these tasks, including (Zheng et al., 2019), (Zhang et al., 2019b) and (Wang et al., 2020). However, recently, Hu et al. (2020) pointed out a short-cut solution in RAVEN which exploited artifacts in the candidate generation procedure. They then created an unbiased version called Balanced-RAVEN which eliminated these artifacts; this is the dataset we have conducted most of our experiments on.

Lastly, copycat (Hofstadter and Mitchell, 1995) is a cognitive architecture designed to discover insightful analogies, a classical AI approach for analogical reasoning. Its followup Phaeaco (Foundalis, 2006) was able to demonstrate limited successes in a problem similar to RPM – the Bongard problems (Bongard, 1970). These systems use hand-crafted features and patterns detectors, hence they require heavy engineering efforts to scale to challenging analogical reasoning tasks.

3 METHODS

The Raven's Progressive Matrices (RPM) task (Raven, 1936; John and Raven, 2003) contains rich compositional structures. To illustrate, in RAVEN (Zhang et al., 2019a), one is given a 3×3 matrix of panels, with the last panel missing. Each panel is an image that contains a number of objects with visually discernible attributes (such as color, size, and shape). The task is to choose a panel from eight candidates to fill in the missing panel so that three rows form consistent analogies (see Figure 2 for an example). To succeed in this task, the model needs to be capable of extracting a composition of three distinct components. It first needs to extract objects in every panel. It then needs to extract the visual attributes of these objects. Lastly, it needs to recognize the relationships among attributes, such as "progression", "union", and "constant". In this section, we describe an architecture that is designed to discover the underlying compositional structure of the data.

3.1 THE SCATTERING COMPOSITIONAL LEARNER

In order to discover the underlying compositional structure of the data, we employ three types of neural networks: object networks $\{\mathcal{N}_i^o\}$, attribute networks $\{\mathcal{N}_i^a\}$, and relationship networks $\{\mathcal{N}_i^r\}$. The object networks are convolutional neural networks that extract object representations of from panels; the attribute networks extract attribute features (such as color, size, and shape) from a object representation; the relationship network decides whether certain relationship holds among attributes.

We explicitly compute the compositions of all these neural networks, $\{\mathcal{N}_k^r \circ \mathcal{N}_j^a \circ \mathcal{N}_i^o\}_{ijk}$ (e.g., whether the kth relationship holds among the jth attributes of the ith objects of the input images), and aggregate the results using an output network for making the prediction.

By explicitly computing all possible compositions of three types of neural networks, we force the networks of each type to be compatible with each other. For example, a relationship network that extracts "progression" needs to be compatible with all attribute networks. Namely, it needs to recognize "progression" for any input feature, regardless of whether the input feature represents "color", "size" or "shape". Hence, it is encouraged to learn the general notion of "progression" (a process of gradual advancement), instead of an attribute-dependent concept (e.g., "color getting darker"). The same reasoning applies to all neural networks of three types. As a result, each type of neural network learns precisely its intended functionality, and the underlying compositional structure can be discovered. Furthermore, since the neural networks of different types are learned to be compatible with each other, unseen analogies can also be grasped by novel compositions of these neural networks without further training, as we show in Section 4.5.

Multi-Head Instantiating all neural networks of three types creates nontrivial engineering burdens. Instead, we represent all neural networks of the same type by one multi-head architecture neural network. For example, we use the multi-head attribute neural network $\mathcal{N}^a$ with M heads, to represent all attribute neural networks $\{\mathcal{N}_i^a\}_{i=1}^M$. Namely, we treat the output of the neural network $\mathcal{N}^a$ as the concatenation of the outputs of $\{\mathcal{N}_i^a\}_{i=1}^M$. As a result, we only need to instantiate 3 multi-head neural networks in total, instead of $3M$ distinct neural networks.

Scattering transformation With the multi-ahead architecture, we propose an operation called the *scattering transformation*[1] to compute the all possible compositions of neural network components in one single forward pass, largely reducing the computational overhead. We define the scattering transformation of an output vector (from $\mathcal{N}^o$) by the attribute neural network $\mathcal{N}^a$ with the following three steps: (1) *Split* the output vector from $\mathcal{N}^o$ into M groups. (2) *Share* the neural transformation $\mathcal{N}^a$ for each group. (3) *Merge* all previous results into a single vector. An illustration is shown in Figure 2. The first step corresponds to splitting $\mathcal{N}^o$ into multiple networks $\mathcal{N}_i^o$. The second step computes all compositions of $\mathcal{N}_i^o$ and $\mathcal{N}^a$, where $\mathcal{N}^a$ is implicitly a multi-head architecture that computes the concatenation of $\mathcal{N}_j^a$. Therefore, the resulting vector after the third step contains outputs from all compositions $\{\mathcal{N}_j^a \circ \mathcal{N}_i^o\}$. In general, the scattering transformation can be applied repeatedly to compose a sequence of neural network components, as in the case of RAVEN.

Full Architecture We describe the full architecture as follows. We first make eight copies of the 3×3 matrix with the missing panel and fill in a different candidate in each of them. The job of SCL is to choose the correctly filled matrix out of eight copies. Every panel of the eight matrices is fed into $\mathcal{N}^o$. We perform the first scattering with $\mathcal{N}^o$ and $\mathcal{N}^a$ for every image, obtaining every attribute of every object. We then perform a second scattering for the output of the first scattering with $\mathcal{N}^r$. In the second scattering, since the relationship is defined over the matrix, we feed a concatenation of the attributes of all panels in the matrix into $\mathcal{N}^r$ to extract the relationship. Lastly, we aggregate the results by an output MLP that produces a score for each matrix. The predicted probability is computed by taking a softmax over the 8 scores. The full diagram is shown in Figure 2.

4 EXPERIMENTS

We design experiments to answer the following questions: **1) I.I.D. Generalization:** How does SCL perform on analogical reasoning tasks compared to existing methods? **2) Learning compositional Structure:** Can SCL discover the underlying compositional structure of the task? How strongly correlated is the learning of compositional structures and the test accuracy? **3) Compositional generalization:** Can SCL generalize to novel analogies not seen at training time? The implementation of the architecture and the experiments is available at `https://github.com/dhh1995/SCL`.

4.1 DATASET

PGM PGM (Santoro et al., 2018) is a visual analogical reasoning task. The task mimics how Raven Progressive Matrices were designed as a test for Intelligence Quotient (IQ) for humans (Raven, 1936).

---

[1]The name is inspired by the scattering transform (Bruna and Mallat, 2011).

Table 1: Results on Balanced-RAVEN dataset (joint training).

| | Test Accuracy(%) | | | | | | |
Model	Avergae	Center	2Grid	3Grid	L-R	U-D	O-IC	O-IG
LSTM (Hu et al.)	18.9	26.2	16.7	15.1	14.6	16.5	21.9	21.1
WReN (Hu et al.)	23.8	29.4	26.8	23.5	21.9	21.4	22.5	21.5
Resnet (Hu et al.)	40.3	44.7	29.3	27.9	51.2	47.4	46.2	35.8
Wild ResNet (Hu et al.)	44.3	50.9	33.1	30.8	53.1	52.6	50.9	38.7
LEN (Zheng et al.)	39.0	45.5	27.9	26.6	44.2	43.6	50.5	34.9
CoPINet (Zhang et al.)	46.3	54.4	33.4	30.1	56.8	55.6	54.3	39.0
HriNet (Hu et al.)	63.9	80.1	53.3	46.0	72.8	74.5	71.0	49.6
SCL (ours)	**95.0**	**99.0**	**96.2**	**89.5**	**97.9**	**97.1**	**97.6**	**87.7**
Human[2] (Zhang et al.)	84.4	95.5	81.8	79.6	86.4	81.8	86.4	81.8

PGM consists of 8 different subdatasets, each containing 1,222,000 questions with 119,552,000 images. Due to the size of the dataset, we conduct experiments on only one of the sub-datasets, "neutral". Each image may contain a variable number of objects of type "shape" or "line". Five relationships (progression, XOR, OR, AND, and union) are defined over 5 attributes (size, type, color, position, and number) of the objects. There are on average 1-2 relationships defined in each problem.

RAVEN/Balanced-RAVEN   RAVEN (Zhang et al., 2019a) is an extension to the PGM dataset. While RAVEN shares the same set of attributes as PGM, it introduces a different set of relationships over these attributes: progression, constant, union and arithmetic. The dataset is designed to be a more difficult task than PGM, as each problem contains 6.3 relationships on average, compared to 1-2 in PGM. The dataset consists of 7 distinct configurations: Center, 2x2Grid, 3x3Grid, L-R, U-D, O-IC, O-IG. Each contains 10,000 problems. An illustration of each configuration is shown in Appendix B along with a detailed explanation. We evaluated our method on this dataset in addition to PGM because RAVEN also provides labeled symbolic representation of the attributes. This helps us verify the learning of compositional structures in Section 4.4. The symbolic representation is an integer-valued scalar for each feature, ranging from 0 to 9.

Recently, Hu et al. (2020) observed that there are defects in the design of candidate set generation in RAVEN dataset. In particular, they discovered a short-cut solution that predicts the label perfectly solely based on the candidate set without any information of the context images. Hence, they introduced a revised dataset called Balanced-RAVEN, which generates the candidate set in a different way to eliminate the short-cut solution. Since the results on RAVEN can be untrustworthy, in the following experiments, we mainly compare results on Balanced-RAVEN. Results on the original RAVEN are also included for completeness in Appendix C.2.

4.2 EXPERIMENTAL PROTOCOL

For Balanced-RAVEN, we used 70,000 examples in total, and used a training, validation, and test split of 60%, 20% and 20% following Zhang et al. (2019a). We ran every model for 300 epochs, with 5 random seeds. We took the best-validated model out of 5 runs and evaluated on the test set. We used the same architectural hyperparameters for our model in all the tasks, as detailed in Appendix A. In training our model, we used the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.0007 for PGM and 0.005 for Balanced-RAVEN/RAVEN, while otherwise keeping the default Adam hyperparameters. We also applied a weight decay of 0.01. The hyperparameters were found after a search over the learning rate from $\{0.007, 0.005, 0.001, 0.0007, 0.0005\}$ and weight decay from $\{0.0, 0.001, 0.01\}$. In all of our experiments, we used one Nvidia P100 GPU with 12 GB RAM with 8 CPU cores.

4.3 I.I.D. GENERALIZATION TASKS

In this section, we focus on i.i.d. generalization, where the training data and the test data come from the same distribution. As we focus on architectural comparisons in the following evaluations, we consider the following baselines: LSTM (Hochreiter and Schmidhuber, 1997), WReN (Santoro et al., 2018), CNN+MLP, ResNet-18 (He et al., 2016a), MXGNet (Wang et al., 2020), LEN (Zheng et al.,

---

[2]The human experiments were conducted on the original RAVEN dataset, obtained from Zhang et al. (2019a).

2019), T-LEN (Zheng et al., 2019), CoPINet (Zhang et al., 2019b), HriNet (Hu et al., 2020) [3]. We do not consider baselines that make use of auxiliary labels, or techniques for optimizing the training data. We were not able to run MXGNet on Balanced-RAVEN due to the lack of an open-source implementation.

4.3.1 PGM

We followed the training setting and experimental protocol described in Section 4.2, and the results are shown in Table 2. One can see that our SCL achieved a substantial improvement over the baselines. The SCL achieved a relative improvement of 26.4% over the previous state-of-the-art method T-LEN.

Table 2: Results on PGM dataset.

Model	Acc(%)
LSTM (Santoro et al.)	33.0
CNN+MLP (Santoro et al.)	35.8
Resnet-50 (Santoro et al.)	42.0
W-ResNet-50 (Santoro et al.)	48.0
WReN (Santoro et al.)	62.8
MXGNet (Wang et al.)	66.7
CoPINet (Zhang et al.)	56.4
LEN (Zheng et al.)	68.1
T-LEN (Zheng et al.)	70.3
SCL (ours)	**88.9**

4.3.2 Balanced-RAVEN/RAVEN

Due to the aforementioned shortcut solution in RAVEN, we evaluated all methods on the unbiased Balanced-RAVEN dataset. Zhang et al. (2019a) introduced the *joint* training setting, where the model is trained and evaluated on all configurations jointly. The results are shown in Table 1. One can see that our proposed method SCL achieved a substantial improvement over the baselines. On average, SCL achieved 31.1% absolute improvement over the previous state-of-the-art HriNet, from 63.9% to 95.0% in test accuracy, which is a 48.7% relative improvement. In addition, we also introduce *single* training setting, where the model is trained and evaluated on each configuration separately. Due to the lack of space, the results are shown in the Appendix, Table 6. Our model achieves perfect 100% test accuracy on 4 out of 7 configurations, significantly outperforming the baselines, which obtain test accuracy around 50%.

4.3.3 Which Methods Exploit the Shortcut Solution?

As pointed out by Hu et al. (2020), there exists a shortcut solution to the original RAVEN dataset. To investigate which methods exploit the shortcut, we first compared the results on RAVEN to Balanced-RAVEN. The results on the original RAVEN dataset are shown in Appendix C.2. We observed that most of the existing baselines' performances degraded significantly. In particular, the previous state-of-the-art method CoPINet (Zhang et al., 2019b), achieving a test accuracy of

Table 3: Sanity check on RAVEN when context images are masked out.

Model \Context	Yes	No
CoPINet (Zhang et al.)	91.4	95.0
SCL (ours)	91.6	12.2

91.4% on RAVEN, was only able to obtain a test accuracy of 46.3% on Balanced-RAVEN. This indicates that the existing baselines very likely exploited the short-cut solution. In contrast, the performance of SCL was robust to the change in candidate generation (we will also show how SCL solved the task exactly in the following section). We further verified this by masking out all 8 panels that provide contextual cues to the correct answer on RAVEN. We found CoPINet achieved an even higher test accuracy of 95.0%, suggesting that this model exploited the short-cut solution (Hu et al., 2020) to obtain good performances on this task, whereas our model fell to 12.2% (corresponding to chance accuracy), as shown in Table 3.

4.4 Does SCL Discover Compositional Structure?

Encouraged by the results shown in the previous section, we analyzed SCL to understand the source of the improvements over the baselines. We examined whether SCL learned the underlying compositional structures of the dataset. We conducted our analysis on Balanced-RAVEN instead of PGM, because RAVEN contains the labels of symbolic attributes of every object, which is crucial to verify whether it discovered compositional structures. In the following sections, we first investigate the learning of objects and their attributes by examining the outputs of neural networks $\mathcal{N}^o$ and $\mathcal{N}^a$. We then investigate whether SCL learned to identify the symbolic relationship by examining $\mathcal{N}^r$.
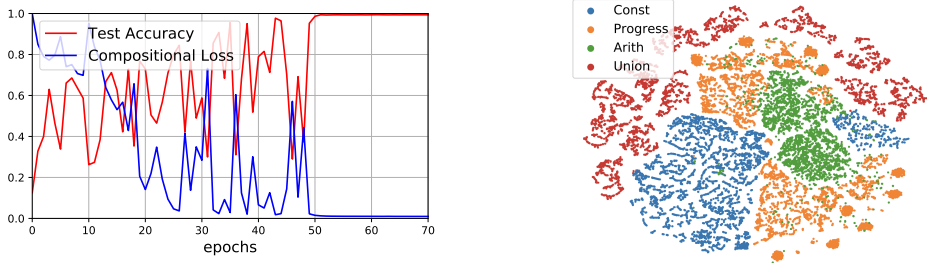
---

[3]We took the implementation of LSTM, CNN+MLP, and ResNet-18, WReN from `https://github.com/Fen9/WReN`, implementation of LEN from `https://github.com/zkcys001/distracting_feature` and implementation of CoPINet from `https://github.com/WellyZhang/CoPINet`.

(a) We show test accuracy and $\mathcal{L}_{comp}$ in task `L-R`. The test accuracy is a mirror image of the $\mathcal{L}_{comp}$, suggesting learning compositional structures is strongly correlated with generalization.

(b) t-SNE plot of the output vectors of the relationship neural network $\mathcal{N}^r$ in task `L-R`. Each cluster has clear boundary, indicating the neural network was able to identify every relationships clearly.

Figure 3: Visualization of compositional structure learning.

4.4.1 LEARNING OBJECTS AND ATTRIBUTES

Since the dataset provides the ground truth symbolic attributes of every object for each scene, we can examine whether the composition of neural networks $\mathcal{N}^o$ and $\mathcal{N}^a$ learned to extract attributes (i.e., "color", "type", and "size") of each object of the scene. Formally, we denote the ground truth i^{th} attribute of the jth object on an image X by $f_i^a \circ f_j^o(X) \in \mathbb{R}$. We let $\mathcal{N}_j^o$ represent the extraction of jth *object*, and $\mathcal{N}_i^a$ represent the extraction of ith *attribute*. We examine if the ground truth symbolic attributes match any of the compositions of neural modules $\{\mathcal{N}_i^a \circ \mathcal{N}_j^o\}_{i,j}$ up to permutation of the indices, followed by a linear transformation. Namely, we say that the model learned to extract the ith attribute of jth object, $f_i^a \circ f_j^o$, if there exists an linear transformation that maps a neural composition $\mathcal{N}_q^a \circ \mathcal{N}_p^o$ to the ground truth for every image X approximately,

$$f_i^a \circ f_j^o(X) \approx \text{Linear}(\mathcal{N}_q^a \circ \mathcal{N}_p^o(X); W, b), \forall X.$$

Therefore, to verify if the model learned the composition (i, j), we took the learned model and searched for the linear transformation and the pair by optimizing the following (i, j)-composition loss:

$$\mathcal{L}_{comp}(i, j) = \min_{\substack{W, b \\ p, q \in [1, \ldots, M]^2}} \frac{1}{|\mathcal{D}|} \sum_{X \in \mathcal{D}} ||\text{Linear}(\mathcal{N}_q^a \circ \mathcal{N}_p^o(X); W, b) - f_i^a \circ f_j^o(X)||^2.$$

Table 4: Test Compositional Structure Loss and the symbolic representation prediction accuracy for the model trained on joint tasks.

	Center	2Grid	3Grid	L-R	U-D	O-IC	O-IG
Loss	0.02	0.27	0.35	0.04	0.05	0.04	0.26
Acc	99.4	85.8	79.8	98.4	96.6	97.9	80.5

We took the model we trained in Section 4.3, and recorded the average of compositional learning losses for attributes "color", "shape", and "type" of every object in the test data. In addition to the loss, we also measured the accuracy of predicting the symbolic representation by the linear transformation followed by rounding. The loss as well as the accuracy for each configuration is reported in Table 4. We can see that in configurations `Center`, `L-R`, `U-D`, and `O-IC`, the model learned the attributes almost perfectly, with close to full predictability. It learned slightly worse in the other three configurations, echoing the degradation of test accuracy for these tasks.

We also tracked the compositional loss along training to show the evolution of compositional structure learning. We used the single training setting of `L-R` for illustration. We plotted the average loss of $\mathcal{L}_{comp}$ over all six compositions (3 attributes $\times$ 2 objects) along with training, as well as the test accuracy in Figure 3a. We observed that the test accuracy and $\mathcal{L}_{comp}$ evolved exactly in the opposite trends of each other at every epoch. This strong correlation supports the hypothesis that the success of generalization was largely due to learning compositional structures.

Table 5: Generalization to unseen attribute-relationship pairs for joint training data.

	Test Accuracy (Difference to Validation Accuracy)					
	Type		Size		Color	
	CoPINet (Zhang et al.)	*SCL* (ours)	*CoPINet* (Zhang et al.)	*SCL* (ours)	*CoPINet* (Zhang et al.)	*SCL* (ours)
Constant	21.7 (-25.5)	**90.2 (-0.2)**	39.5 (-10.2)	**92.1 (-0.3)**	37.0 (-6.6)	**83.8 (-8.0)**
Progression	33.6 (-20.7)	**90.1 (-2.1)**	42.9 (-5.1)	**90.3 (-0.7)**	40.8 (-7.0)	**93.3 (+0.1)**
Union	32.9 (-22.5)	**85.4 (-6.9)**	34.1 (-17.0)	**92.0 (-1.0)**	29.8 (-17.1)	**92.5 (+0.1)**

4.4.2 LEARNING RELATIONSHIPS

Lastly, we investigated if the model learned to extract meaningful relationships. Given the ground truth relationship defined on each attribute, we performed a t-SNE (van der Maaten and Hinton, 2008) for the output vectors of $\mathcal{N}^r$ over 1024 examples, each containing 4 relationships. An illustration with configuration L-R is shown in Figure 3b. We observed that each cluster is formed by a single relationship, with very little overlap, indicating the model was able to identify each relationship concept. We also visualized the relationship learning in all configurations via t-SNE in Appendix F, where we observed clusters were well-separated in 4 out of 7 tasks.

4.5 GENERALIZATION TO UNSEEN ANALOGIES

If the model understands attribute-relationship pair ("color", "constant"), and ("size", "progression"), can the model generalize to ("color", "progression")? We hypothesize that if the model uncovers the compositional structure of attributes and relationships, new combinations of those known concepts can be grasped without further training. For a given relationship-attribute pair, we created a dataset where the pair appear in every test examples but not in the training/validation set. We examined the joint training setting, and took 9 pairs that exist in all configurations and created a new dataset for each of them.

We compared SCL to CoPINet, the previous state-of-the-art on RAVEN. The test accuracy and the generalization gaps (i.e., the difference between the test accuracy and the validation accuracy) are shown in Table 5. SCL achieved an average test accuracy of 90.0% over 9 novel attribute-relationship pairs, whereas the CoPINet only achieved an average of 34.7%. We also observe SCL had an average degradation of 2% from validation to test accuracy over nine pairs, whereas CoPINet heavily suffered from distribution shift, with a degradation of 14.6% on average. We also examined the single training setting with the configuration L-R, where results are reported in the Appendix G.2.

We additionally created another task for testing out-of-distribution generalization, where the model is asked to generalize to a different number of attribute-relationship pairs (see Appendix G.1 for details). Similarly to our previous findings, SCL models were able to generalize but the baseline models failed badly.

5 CONCLUSION

In this work, we introduced a neural architecture SCL for discovering the underlying compositional structure of the data. We applied the proposed method to an analogical reasoning task, Raven's Progressive Matrices, which exhibits strong compositional structures. SCL achieved state-of-the-art performance in two datasets, with a 48.7% relative improvement on Balanced-RAVEN and 26.4% on PGM. We validated the learned compositional structure by showing that the neural representation matches the symbolic representation of attribute concepts up to a linear transformation, and well-separated relationship clusters in a t-SNE plot. With learned compositional structures, SCL significantly outperforms other methods under test-time distribution shifts. Our work provides a promising research direction in designing compositional neural architectures to achieve stronger generalization.

REFERENCES

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 39–48, 2015.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

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. *CoRR*, abs/1912.05783, 2019a. URL http://arxiv.org/abs/1912.05783.

Dzmitry Bahdanau, Shikhar Murty, Michael Noukhovitch, Thien Huu Nguyen, Harm de Vries, and Aaron C. Courville. Systematic generalization: What is required and can it be learned? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019b. URL https://openreview.net/forum?id=HkezXnA9YX.

Mikhail Moiseevich. Bongard. *Pattern recognition*. Spartan Books New York, 1970. ISBN 0876711182.

Joan Bruna and Stéphane Mallat. Classification with invariant scattering representations. In *IEEE 10th Image, Video, and Multidimensional Signal Processing Workshop: Perception and Visual Signal Analysis, IVMSP 2011, Ithaca, NY, USA, 16-17 June 2011*, pages 99–104. IEEE, 2011. doi: 10.1109/IVMSPW.2011.5970362. URL https://doi.org/10.1109/IVMSPW.2011.5970362.

Patricia Carpenter, Marcel Adam Just, and Peter Shell. What one intelligence test measures: a theoretical account of the processing in the Raven Progressive Matrices Test. *Psychological review*, 97 3:404–31, 1990.

Michael Chang, Abhishek Gupta, Sergey Levine, and Thomas L. Griffiths. Automatically composing representation transformations as a means for generalization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=B1ffQnRcKX.

Noam Chomsky. *Aspects of the Theory of Syntax*. The MIT Press, Cambridge, 1965.

J. A. Fodor and Z. W. Pylyshyn. Connectionism and cognitive architecture - a critical analysis. *Cognition*, 28(1-2):3–71, 1988.

Jerry A. Fodor and Ernest Lepore. *The Compositionality Papers*. Oxford University Press, 2002.

Harry E. Foundalis. *Phaeaco : a cognitive architecture inspired by Bongard's problems*. Indiana University, 2006. ISBN 0876711182.

Alexander L. Gaunt, Marc Brockschmidt, Nate Kushman, and Daniel Tarlow. Differentiable programs with neural libraries. In *ICML*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016a. doi: 10.1109/CVPR.2016.90. URL https://doi.org/10.1109/CVPR.2016.90.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016b.

Felix Hill, Adam Santoro, David G. T. Barrett, Ari S. Morcos, and Timothy P. Lillicrap. Learning to make analogies by contrasting abstract relational structure. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=SylLYsCcFm.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

Douglas R. Hofstadter and Melanie Mitchell. *Fluid Concepts and Creative Analogies*, chapter The copycat project: A model of mental fluidity and analogy-making, pages 205–267. Basic Books, 1995.

Sheng Hu, Yuqing Ma, Xianglong Liu, Yanlu Wei, and Shihao Bai. Hierarchical rule induction network for abstract visual reasoning. *arXiv preprint arXiv:2002.06838*, 2020.

Drew A Hudson and Christopher D Manning. Compositional attention networks for machine reasoning. In *International Conference on Learning Representations (ICLR)*, 2018.

Susanne M. Jaeggi, Martin Buschkuehl, John Jonides, and Walter J. Perrig. Improving fluid intelligence with training on working memory. *Proceedings of the National Academy of Sciences*, 105(19):6829–6833, 2008. ISSN 0027-8424. doi: 10.1073/pnas.0801268105. URL https://www.pnas.org/content/105/19/6829.

John and Jean Raven. *Raven Progressive Matrices*, pages 223–237. Springer US, Boston, MA, 2003. ISBN 978-1-4615-0153-4. doi: 10.1007/978-1-4615-0153-4_11. URL https://doi.org/10.1007/978-1-4615-0153-4_11.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. Measuring compositional generalization: A comprehensive method on realistic data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=SygcCnNKwr.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114, 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.

Brenden M. Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *ICML*, 2018.

Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel Gershman. Building machines that learn and think like people. *The Behavioral and brain sciences*, 40:e253, 2018.

João Loula, Marco Baroni, and Brenden M. Lake. Rearranging the familiar: Testing compositional generalization in recurrent networks. In *BlackboxNLP@EMNLP*, 2018.

Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B. Tenenbaum, and Jiajun Wu. The neurosymbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJgMlhRctm.

Gary F. Marcus. Rethinking eliminative connectionism. *Cognitive Psychology*, 37:243–282, 1998.

Gary F. Marcus. *The Algebraic Mind*. MIT Press, 2001.

J.C. Raven. *The Performances of Related Individuals in Tests Mainly Educative and Mainly Reproductive Mental Tests Used in Genetic Studies*. University of London (King's College), 1936. URL https://books.google.ca/books?id=G-6hswEACAAJ.

Adam Santoro, David Raposo, David G. T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter W. Battaglia, and Timothy P. Lillicrap. A simple neural network module for relational reasoning. In *NIPS*, 2017.

Adam Santoro, Felix Hill, David G. T. Barrett, Ari S. Morcos, and Timothy P. Lillicrap. Measuring abstract reasoning in neural networks. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4477–4486. PMLR, 2018. URL `http://proceedings.mlr.press/v80/santoro18a.html`.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. URL `http://www.jmlr.org/papers/v9/vandermaaten08a.html`.

Duo Wang, Mateja Jamnik, and Pietro Liò. Abstract diagrammatic reasoning with multiplex graph networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL `https://openreview.net/forum?id=ByxQB1BKwH`.

Ke Wang and Zhendong Su. Automatic generation of raven's progressive matrices. In *IJCAI*, 2015.

Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5987–5995. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.634. URL `https://doi.org/10.1109/CVPR.2017.634`.

Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. RAVEN: A dataset for relational and analogical visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019a.

Chi Zhang, Baoxiong Jia, Feng Gao, Yixin Zhu, Hongjing Lu, and Song-Chun Zhu. Learning perceptual inference by contrasting. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 1073–1085, 2019b. URL `http://papers.nips.cc/paper/8392-learning-perceptual-inference-by-contrasting`.

Kecheng Zheng, Zheng-Jun Zha, and Wei Wei. Abstract reasoning with distracting features. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 5834–5845, 2019. URL `http://papers.nips.cc/paper/8819-abstract-reasoning-with-distracting-features`.

A   ARCHITECTURE DETAILS

The architectural hyperparameter is the following. In our architecture, we use a feedforward residual block, and hence we define it upfront: a feedforward residual block (motivated by He et al. (2016b)) denoted by $FR(x) = x + \text{Lin}_2(\text{LayerNorm}(\text{ReLU}(\text{Lin}_1(x))))$, where LayerNorm denotes Layer Normalization Ba et al. (2016), and Lin_1 and Lin_2 are two linear layers with the same input output size.

To apply the proposed architecture SCL to the RAVEN task, we first make eight copies of the 3×3 matrix with the missing panel and fill in a different candidate in each of them. Given one of the eight matrices, we pass every panel into $\mathcal{N}^o$ to extract object representation. $\mathcal{N}^o$ consists of a convolutional neural network of four layers, with channel sizes 16, 16, 32, 32, kernel size 3×3 and padding 1, followed by a linear layer with a Relu activation with an output dimension of 80, followed by a feedforward residual block. We then perform scattering with $\mathcal{N}^a$: 1. "Split" the output of $\mathcal{N}^o$ into 10 heads. 2. "Share" the neural network transformation $\mathcal{N}^a$ for each head. $\mathcal{N}^a$ is a one-layer MLP with hidden size 128 with output size 8. 3. "Merge": the output of 10 heads is then concatenated as a vector of 80 dimension, followed by a feedforward residual block. Note that the last feedforward residual block is applied to the output of all objects' attributes as a whole. The reason is to extract some potentially useful global information (such as "number" of objects). Since the transformation is a residual block $x + f(x)$, it also will maintain useful original information x for independent attributes if it needs to. We then perform the second scattering transformation with $\mathcal{N}^r$: 1."Split": we split the 80 dimensional output vector from the previous scattering transformation into 80 groups. We concatenate 1 scalar of each panel from the entire matrix to form a vector of 9 dimensions. 2. "Share": We take each 9 dimensional vector as the input to $\mathcal{N}^r$, which is a two-layer MLP with hidden sizes 64 and 32, with an output size of 5. 3."Merge": concatenate 80 groups of 5 neurons into a vector of 400. Lastly, each 400 dimensional vector is fed into an MLP of hidden size 128 to obtain a score for each matrix. An illustration of the diagram is found in Figure 2.

B   DETAILS ON THE RAVEN DATASET

The configuration `Center` is the simplest one, where there is a single object in the center of each panel. The analogies are constructed by defining one relationship on every object attribute, totaling to three analogies in each problem. In configurations `L-F`, `U-D`, `O-IC`, each image contains two objects of various placements. The analogy is defined on every attribute of each object in the panel, in a total of six analogies. When there are more objects, in configurations such as `2x2Grid`, `3x3Grid`, more sophisticated analogies are introduced, such as "progression" over "number" and "arithmetic" over "position". An illustration of each configuration is show in Figure 4.
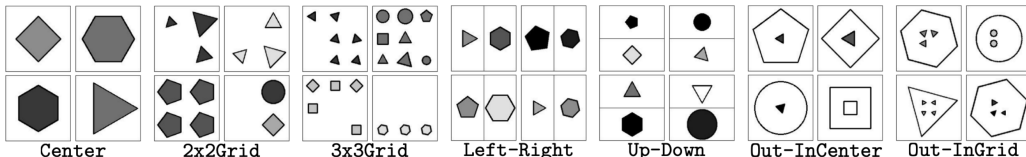
Figure 4: Examples of 7 different figure configurations in the RAVEN dataset (taken from Zhang et al. (2019a)).

C   FURTHER RESULTS ON RAVEN/BALANCED-RAVEN

C.1   RESULTS ON BALANCED-RAVEN

We also introduce single-task training settings, where the model was trained and evaluated on each configuration separately. Due to the lack of space, the results are shown in Table 6. SCL can achieve nearly perfect test accuracy on 4 out of 7 configurations.

Table 6: Single Task Training results on Balanced-RAVEN dataset.

Model	Test Accuracy(%)							
	Avergae	Center	2Grid	3Grid	L-R	U-D	O-IC	O-IG
LSTM(Zhang et al.)	12.5	12.3	13.3	12.8	12.7	10.3	12.9	13.1
WReN	17.8	23.3	18.1	17.4	16.5	15.2	16.8	17.3
CNN+MLP	12.9	12.9	13.2	12.7	11.5	13.5	12.9	13.7
Resnet-18	14.5	20.8	12.9	14.3	13.2	13.4	13.8	12.9
LEN	28.4	42.5	21.1	19.9	27.6	28.1	32.9	27.0
CoPINet	38.6	50.4	30.9	28.5	40.0	40.8	42.7	36.9
SCL(ours)	**85.4**	**99.8**	**72.4**	**64.2**	**99.5**	**99.4**	**98.6**	**64.2**

C.2 SANITY CHECK ON RAVEN

We show further results on RAVEN for completeness. The results of the joint training setting are shown in Table 7. By comparing the results on RAVEN to Balanced-RAVEN, we conclude that most of the existing baselines have the issues discovered by Hu et al. (2020), namely, they exploited the short-cut solution, whereas SCL was robust to the change in candidate generation. Notably, the previous state-of-the-art method CoPINet Zhang et al. (2019b), achieving a test accuracy of 91.4% on RAVEN, only was able to obtain a test accuracy of 46.3% on Balanced-RAVEN, strongly suggesting that this model exploited the cheating solution to obtain good performances on this task.

Table 7: Joint Training results on RAVEN dataset.

Model	Test Accuracy							
	Avergae	Center	2Grid	3Grid	L-R	U-D	O-IC	O-IG
LSTM (Zhang et al.)	13.1	13.2	14.1	13.7	12.8	12.5	12.5	12.9
WReN (Zhang et al.)	14.7	13.1	28.6	28.3	7.5	6.3	8.4	10.6
CNN+MLP (Zhang et al.)	37.0	33.6	30.3	33.5	39.4	41.3	43.2	37.5
Resnet-18 (Zhang et al.)	53.4	52.8	41.9	44.2	58.8	60.2	63.2	53.1
LEN (Zheng et al.)	72.9	80.2	57.5	62.1	73.5	81.2	84.4	71.5
CoPINet (Zhang et al.)	91.4	95.1	77.5	78.6	99.1	99.7	98.5	91.4
SCL(ours)	91.6	98.1	91.0	82.5	96.8	96.5	96.0	80.1
Human (Zhang et al.)	84.4	95.5	81.8	79.6	86.4	81.8	86.4	81.8

D COMPARISONS TO PREVIOUS MODULAR NEURAL NETWORKS

There has been an old debate over the limits of connectionism for compositional generalization. In a set of influential and controversial papers, Jerry Fodor and other researchers criticized that neural networks fail to model the intricacy of mind because they cannot capture systematic compositionality Fodor and Pylyshyn (1988); Marcus (1998; 2001); Fodor and Lepore (2002). Such voices have died down since deep learning revolutionized various domains. However, there have been a few papers recently bringing up these issues again, after observing non-robustness in the systematic generalization of neural networks Lake and Baroni (2018); Loula et al. (2018); Bahdanau et al. (2019b;a); Keysers et al. (2020). Our work is an attempt to address the issues of systematic generalization with an architectural bias that respects compositions. We advocate for the same principle as those methods in the past that add modularity for better generalization, such as Neural Module Networks(NMN) Andreas et al. (2015), NTPT Gaunt et al. (2017), MAC Hudson and Manning (2018), Neural Symbolic Concept Learner Mao et al. (2019), CRL Chang et al. (2019). We detail what each model does and compare it to our models.

Neural Module Networks NMN is a neural network that is assembled from what are known as neural modules. Each neural module is a neural network that is specialized in some particular subtask. Hence, by composing different neural modules, NMN can perform various tasks, analogous to how a computer program composed of functions. However, how to connect different modules requires domain knowledge. In the original NMN, the layout was set in an ad-hoc manner for each question by analyzing a dependency parse on the language query. Similar to NMN, each neural network of each type can be seen as a neural module. However, our approach is specifically designed for composing a sequence of neural networks, without any help from symbolic hints.

Neural Symbolic Concept Learner NSCL extends the NMN work by training the dependency parser jointly with the neural modules, without any auxiliary labels on parsing. Namely, it is a method

of learning how to connect neural modules along with training neural modules themselves. As the parser operates on discrete tokens hence indifferentiable, they used REINFORCE to estimate the gradient. Despite learning the layout of NMN in an unsupervised manner, NSCL still makes use of language data to guide the construction of the computational graph. In contrast, our model learned compositional structures without any symbolic hints.

MAC   MAC is a recurrent cell composed of control, read and write units that maintain a separation between control and memory. The control units attend to natural language questions and employ the read and write units to gather information and update memory for performing reasoning. By stacking together multiple recurrent MAC cells, the MAC network performs an explicit multi-step reasoning process. However, MAC could suffer from handling the long reasoning chain without language guidance in the case of RAVEN. Instead, our SCL learn to reason about the correct answer without any auxiliary labels or hints, solely based on the compositional inductive bias.

NEURAL TERPRET   NTPT provides a system for constructing differentiable program interpreters. In their work, the benefits of knowledge transfer brought by modularity were extensively studied. Similar to the prior work, it requires additional symbolic information (source code) for composing neural modules.

Compositional Recursive Learner   CRL builds on the prior work and study how to connect neural modules via reinforcement learning (RL). The method was demonstrated on a few simple tasks, but also admitted issues with scalability to more challenging problems due to large search space with RL.

In summary, our work differentiates from the past work, in that our architecture targets at composing a sequence of neural modules. The past work relies on domain knowledge or ad-hoc design (e.g., NMN), and symbolic information (e.g., language in NSCL, MAC; source code in NTPT), or non-differentiable search(e.g., NSCL, CRL) for composing neural modules in task-specific layout.

E   ABLATION STUDIES

To understand the way how our model works, we perform the following ablation studies, each starts with a question. For all the experiments run in this section, we follow the experimental protocol described in Section 4.2. The results reported in this section used an average of 5 runs.

How important is sharing?   Compared to ResNeXt, a crucial difference is that we used a shared transformation in the *scattering transformation*. To validate the importance of this design choice, we show the average results on the joint training task for sharing versus non-sharing models in Table 8.

Table 8: How important is sharing?

Test Accuracy	
Sharing	Non-sharing
95.0	12.3

How to choose the number of heads of $\mathcal{N}^a$?   We took the single training setting of the configuration `Center` for illustration. Since there are in total 3 attributes in this task (color, size, type), it may suggest using 3 heads for $\mathcal{N}^a$ is a good choice on this task. However, this turned out to be a poor choice. The average test accuracy over 5 runs were shown in Table 9. We found that as we increase the number of heads, it is easier for the model to learn the ground truth solution. We believe this is because of an ensemble effect, where more number of neurons provide more opportunities for learning the correct attribute concepts.

How important is to use scattering transformation with $\mathcal{N}^o$?   It may seem plausible that if the convolutional neural network is strong enough, it can extract the attributes of all objects directly without the first scattering transformation of $\mathcal{N}^o$ (i.e., when $\mathcal{N}^o$ is a regular single-head CNN). We showed in Table 10 that compares the number of heads of $\mathcal{N}^o$ in the joint training setting. We observe that having more than 1 head was essential to our model.

Table 9: How to choose the number of heads of $\mathcal{N}^a$ for `Center`?

	Test Accuracy	
4	8	80
57.3	43.2	**100.0**

Table 10: Test accuracy with various number of heads of $\mathcal{N}^o$ in joint training setting.

	Test Accuracy		
1	4	10	16
12.4	34.5	**95.0**	88.2

F FURTHER VISUALIZATIONS OF RELATIONSHIP LEARNING

We visualized the relationship learning in all configurations via t-SNE in Fig 5, where we observed clusters were well separated in `Center`, `L-R`, `U-D`, and `O-IC`, but became less distinguishable in the other three tasks.
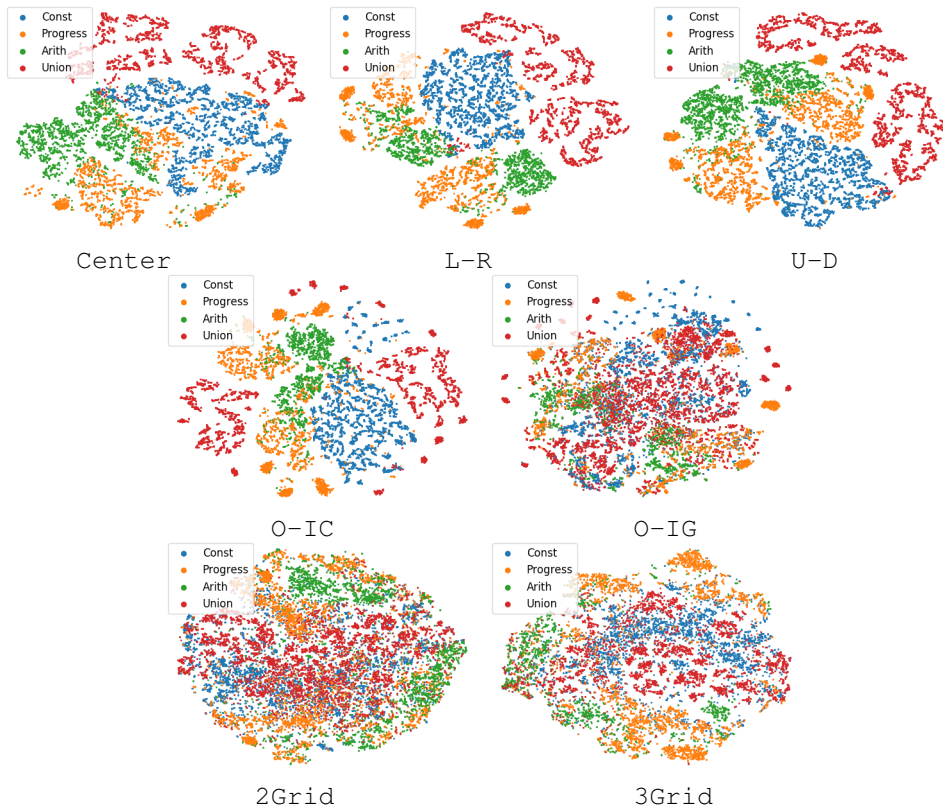

Figure 5: t-SNE plots on the output of the relationship neural network $\mathcal{N}^r$, in the order: `Center`, `L-R`, `U-D`, `O-IC`, `O-IG`, `2Grid`, `3Grid`.

G   FURTHER RESULTS ON OUT-OF-DISTRIBUTION GENERALIZATION

G.1   GENERALIZATION TO A DIFFERENT NUMBER OF ATTRIBUTE RELATIONSHIP PAIRS

In this section, we ask if the model can generalize to a problem that consists of a different number of attribute-relationship pairs. In Balanced-RAVEN, within each configuration, the number of attribute-relationship pairs in every problem is fixed. For example, the configuration `Center` contains 3 pairs, i.e., one needs to discover 3 underlying patterns to solve the problem. A seemingly simple generalization task to humans but can be difficult for learned models is to generalize to a problem that only consists of 2 attribute-relationship pairs.

To evaluate this kind of generalization, we created two new datasets for each configuration of `Center`, `L-R`, `U-D`, and `O-IC`. The first dataset, denoted as "Rel-1", defines 1 relationship on each object; the second dataset, denoted as "Rel-2", defines 2 relationships on each object. For those attributes that are not associated with any relationships, we randomly sample them. We denote the original dataset, in which there are 3 relationships defined on each object as "Rel-3". We performed training and validation on each of the datasets while using the remaining 2 datasets as the test set. We present the results in Table 11-14, with comparisons to CoPINet. SCL was able to generalize within 1% accuracy drop from training dataset to test dataset in 22 out of 24 tasks. In contrast, the baseline model LEN severely suffered from a distribution shift from training to test. The accuracy went down from 46.0% to 34.1% when training on "Rel-3" and testing on "Rel-1'' on `Center`, whereas SCL only had a slight degradation of 0.7%.

Table 11: Generalization to various number of relationships on `Center`.

Train\Test	Rel-1		Rel-2		Rel-3	
	CoPI	SCL	CoPI	SCL	CoPI	SCL
Rel-1	42.7	99.9	39.8	99.9	37.8	99.7
Rel-2	41.3	99.9	40.5	99.9	41.7	99.8
Rel-3	34.1	99.0	41.1	99.2	46.0	99.7

Table 12: Generalization to various number of relationships on `L-R`.

Train\Test	Rel-1		Rel-2		Rel-3	
	CoPI	SCL	CoPI	SCL	CoPI	SCL
Rel-1	36.8	99.9	41.0	99.9	44.8	99.9
Rel-2	39.1	99.9	44.3	99.8	44.9	99.6
Rel-3	35.1	99.5	43.7	99.3	47.6	99.6

Table 13: Generalization to various number of relationships on `U-D`.

Train\Test	Rel-1		Rel-2		Rel-3	
	CoPI	SCL	CoPI	SCL	CoPI	SCL
Rel-1	37.3	99.9	39.7	100.0	42.1	100.0
Rel-2	35.7	99.9	42.6	99.9	46.3	99.8
Rel-3	34.1	99.1	42.7	99.3	46.9	99.6

Table 14: Generalization to various number of relationships on `O-IC`.

Train\Test	Rel-1		Rel-2		Rel-3	
	CoPI	SCL	CoPI	SCL	CoPI	SCL
Rel-1	42.6	98.9	44.1	99.1	43.8	99.1
Rel-2	37.4	99.3	43.7	99.6	45.6	99.5
Rel-3	34.5	77.9	45.4	90.4	50.0	99.2

G.2 GENERALIZATION TO UNSEEN ANALOGIES IN `L-R`

In addition to the unseen analogy experiments for the joint-training setting (Section 4.5), we also performed further experiments on the single training setting of `L-R`.

Similarly done as in Section 4.5, we created nine datasets for nine attribute-relationship pairs with 60%/20%/20% train/valid/test split for `L-R`. We reported both the test accuracy as well as the performance drop from the validation accuracy in Table 15. We observed that our model could generalize almost perfectly on 8 out of 9 tasks, with an average test accuracy of 97.8%. SCL achieves similar validation and test performance with an average gap of 2.2%. We compared our model to the existing baselines LEN and CoPINet, and observed large gaps from validation accuracy to test accuracy, an average of 14.1% for CoPINet and 5.3% for LEN, showing their poor generalization to unseen analogies.

Table 15: Generalization to unseen attribute-relationship pairs.

	Test Accuracy (Difference to Validation Accuracy)								
	Type			Size			Color		
	LEN (Zheng et al.)	*CoPINet*(Zhang et al.)	*SCL* (ours)	*LEN* (Zheng et al.)	*CoPINet* (Zhang et al.)	*SCL* (ours)	*LEN* (Zheng et al.)	*CoPINet* (Zhang et al.)	*SCL* (ours)
Constant	28.0 (-0.8)	25.1 (-25.9)	**100.0 (-0)**	24.4 (-1.1)	37.2 (-6.1)	**99.9 (-0)**	25.3 (-1.4)	38.8 (-4.9)	**100.0 (-0)**
Progression	24.0 (-1.3)	36.2 (-19.2)	**98.1 (-0.7)**	27.9 (-2.5)	36.2 (-7.9)	**100.0 (+0.1)**	25.3 (-3.2)	35.8 (-5.9)	**99.8 (-0.1)**
Union	29.4 (-11.2)	32.9 (-24.7)	**83.3 (-16.4)**	27.6 (-12.3)	36.4 (-18.1)	**98.8 (-1.1)**	22.0 (-13.6)	29.2 (-14.6)	**100.0 (-0)**