CS 231N Project Report
Unsupervised learning of Visual Object Relations with Graph-Level Analogy

Daniel Zeng
Stanford University
dazeng@stanford.edu

Abstract

Visual relations form the basis of understanding our compositional world, as relationships between visual objects capture key information in a scene. It is then advantageous to learn relations automatically from the data, as learning with predefined labels cannot capture all possible relations. However, current relation learning methods typically require supervision, and are not designed to generalize to scenes with more complicated relational structures than those seen during training. Here, we introduce a method for unsupervised discovery and learning of Visual Relations with graph-level analogy. In a setting where scenes within a task share the same underlying relational subgraph structure, our learning method of contrasting isomorphic and non-isomorphic graphs discovers the relations across tasks in an unsupervised manner. Once the relations are learned, our method can then retrieve the shared relational graph structure for each task by parsing the predicted relational structure. Using a dataset based on grid-world and the Abstract Reasoning Corpus, we show that our method achieves above 95% accuracy in relation classification, discovers the relation graph structure for most tasks, and further generalizes to unseen tasks with more complicated relational structures.

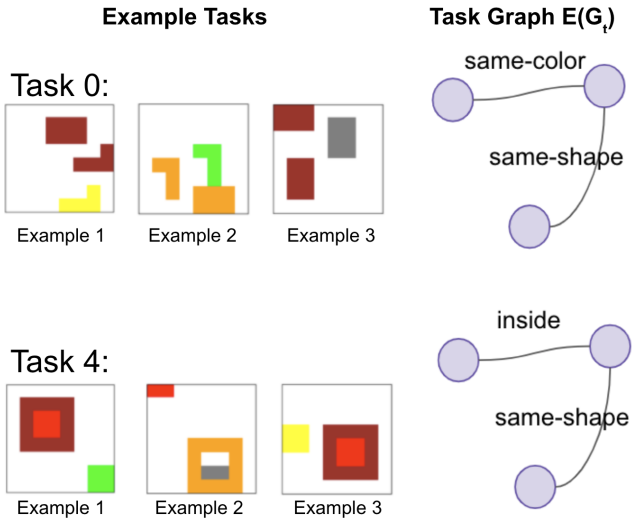


Figure 1. Two example BabyARC tasks and respective graphs. Each task t contains images with a shared relation subgraph $E(G_t)$. The input are image examples under "Example Tasks". Both the relation types and the graph of each task are unknown. The goal is to infer the relation graph $E(g_i)$, the corresponding relations, and the shared relation graph $E(G_t)$ for each task t. More example inputs are shown in Fig. 7.

1. Introduction

Our world is naturally compositional: where concepts, whether abstract or physical, are hierarchically composed of constituent concepts and their relations. In parallel, human intelligence has evolved to understand and reason with compositional structure. In the visual domain, this ability has endowed humans to quickly understand visual scenes and generalize to previously unseen, complex scenes. A key factor in compositionality is understanding visual relations, for example, that one object has the same shape as another.

In this work, we tackle a novel problem in visual relations, which is learning and discovering relations in the unsupervised setting, where relation types and labels are not known *a priori*. In the unsupervised setting, the relevant relations are learned from the data, removing the dependence on predefined relation types. This unlocks greater generalization capacity for learning, reasoning and compositionality. In contrast, when relations are learned with predefined labels in a supervised context, this limits us to settings which depend on those seen relations.

Our key insight is that the emergence of relations comes from graph-level analogy, where scenes within a task share

a common relational subgraph consisting of concepts as nodes and relations as edges. We introduce a graph-isomorphism based objective to learn a distinct graph embedding (computed from relation embeddings with a GNN) for each task, which then encourages distinct clusters of relation embeddings to form. Once our method has learned the relations, our method retrieves the shared relational graph structure for each task by parsing the predicted relational structures.

Specifically, our input is a image of the scene, where the dimensions of the image are [width, height, # colors, # objects]. The width and height are 16, and there are 9 colors, and number of objects depend on the scene. Images are padded to the maximum number of objects to handle varying object count. We use a convolutional neural net (CNN) $\rightarrow$ multi-layer perceptron (MLP) $\rightarrow$ graph neural network (GNN) to process the input, and an output is a final latent embedding of the image representing the scene. We try out two different loss functions, a contrastive loss function and a classification loss function, using the task label against the prediction.

While the types and appearances of concepts within a task may vary, our method is able to infer the global relation types, achieve above 95% accuracy in relation classification in all our 6 dataset configurations, and retrieve the common relational graph structure in seen tasks and further generalize to unseen, more complex tasks.

2. Related Work

A number of recent works on compositionality have focused on the concepts, towards learning unsupervised object discovery from visual data. These approaches include decomposing an image into object slot representations [21], using iterative variational inference [8], or encoder-decoder autoregressive inference [6]. Furthermore, the importance of object centric representations and decomposition have arisen in visual reasoning tasks, as have been explored in [5, 27, 30].

From the other angle of compositionality is visual relations, which interconnect the properties between visual objects. A number of works have explored visual relations, for scene graph generation or image reconstruction [9, 11, 17], localization and grounding of subject-predicate-object triplets [18], dynamics prediction [4, 15], visual relation detection [22], and few-shot classification [26]. Liu et al. [20] uses a pretrained text-encoder to encode relation tokens in a natural language setting for scene generation. Shanahan et al. [25] proposes PrediNet for compound relation classification. These works have focused on applying visual relations to downstream higher-level tasks with known relation labels, or learning relations in a supervised setting. For our specific task setting of unsupervised visual relation learning, there are no prior state of the art due to that we have created our own task formulation and approach.

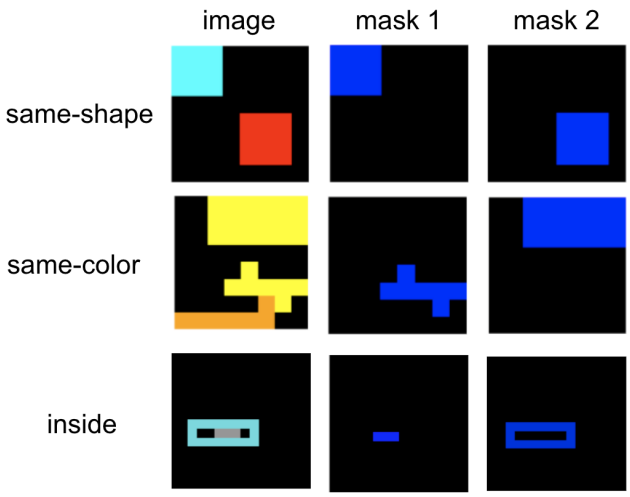


Figure 2. Example of each of the three global relation types of the BabyARC dataset.

3. Task and Dataset

Our setup is inspired by the Abstraction and Reasoning Corpus (ARC) dataset [7], where the tasks provided in the dataset aim to serve as a benchmark for human-level intelligence and reasoning. One of the inductive priors in many ARC tasks is that the training examples are graph-isomorphic, where there exists a common relational structure between the examples. A motivating ARC task example is shown in Fig. 8 and more in Fig. 9 and 10. This follows our motivation of applying graph-level analogy, for the shared relational structure between tasks and to discover the relations between objects.

However, due to the highly challenging and low data nature of ARC, we evaluate our method on BabyARC [28], a dataset generator which captures the graph isomorphic essence of ARC tasks. Moreover, BabyARC provides the underlying metadata of each generated example, allowing evaluation of our method, e.g., the accuracy of the predicted relations.

Datasets for visual reasoning such as CLEVR [12], SHAPES [2], or Visual Question Answering [3, 10, 16, 19] exist, but their setup differs greatly such as due to lack of configuration to specify shared graph structure, use of language query, their own additional task-specific complexities, or emphasis on different learning objectives.

3.1. BabyARC Setup

In the following, we introduce definitions for BabyARC:
Definition 1. Observation: represents a single image, which

contains a collection of n concepts, where n is known.

Definition 2. Concept: represents an *object* in the given observation. Concepts may be rectangles, lines, etc.

Definition 3. Relation: represents the relationship between two objects, specifically a visual relation in the BabyARC setting. For example, if two objects share the same color, the relation between these two objects would be referred to as "same-color". Similarly, "same-shape" represents that two objects have the same shape, and "inside" represents that one object is inside another object. If two objects do not have any relations, the relation label is "none".

Our task definition on the BabyARC is the following: We are given a collection of observations (images) $x_i, i = 1, 2, \ldots, N$, where each observation belongs to some known task $t \in T$. Each task t has an unknown unique task graph $G_t = (V, E)$ with the nodes $V = \{\text{objects}\}$ and $E = \{\text{relations between objects}\}$, and $e_{k,l} \in E$ represents the relation type between the k^{th} and l^{th} object. All of its corresponding observations share this common relational subgraph $E(G_t)$.

In addition, *only the observations* of each task are provided to the model, *without* knowledge of the objects, the global relation types, or the underlying graph of each observation. The goal is to infer the global relation types, the graph $E(g_i)$ of each observation x_i, and the relational graph structure $E(G_t)$ belonging to each task t. To accurately infer the underlying graph $E(g_i)$, this requires the model to identify the correct relation type $e_{k,l}$ between every object pair.

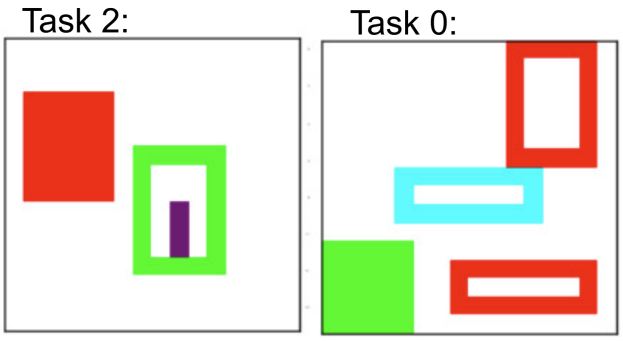


Figure 3. Example of Task 2 (2 core objects), with red rectangle distractor object, and Task 0 (3 core objects), with green distractor object. The core objects are part of the relational subgraph.

Two example BabyARC tasks are shown in Fig. 1. These examples show that the graph of each observation $E(g_i)$ is isomorphic to the task graph $E(G_t)$. However, it may be that only a subgraph of $E(g_i)$ is isomorphic to task graph $E(G_t)$. We define objects which are part of the relational subgraph as "core" objects, and ones which are not as "distractor" objects. Distractor objects can be viewed as random objects added to the observation and irrelevant to task graph $E(G_t)$. The distinction between distractor and core objects is not given to the model. Examples of tasks with distractor objects are shown in Fig. 3.

Since we use a dataset generator, we are able to generate as much training examples as needed. Specifically for training, we use 235 image examples per task for datasets with up to 3 core objects (6 tasks total), and 350 image examples per task for datasets with up to 4 core objects (13 tasks total). For validation dataset, we use a total of 1200 image examples for validation, irrespectable of task count. As mentioned, the width and height of the image is 16, and there are 9 colors, and number of objects depend on the scene (which is given). Images are padded to the maximum number of objects to handle varying object count. There is no data processing since we directly generate our examples, but standard data augmentation is used, where each image is augmented with probability 0.9, and the possible augmentations are random flipping, rotating, resizing, and color. More examples are shown in Appendix A.5.

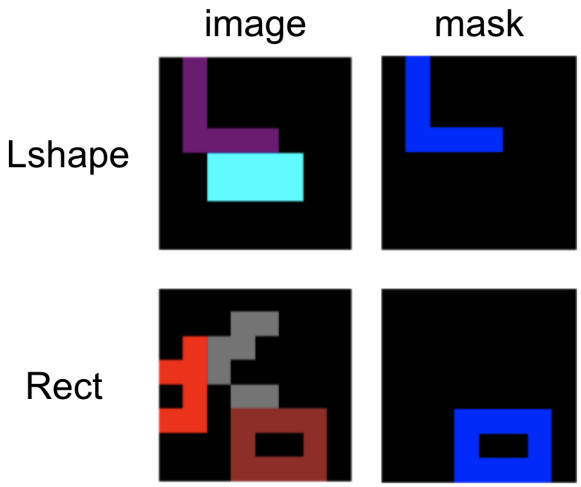


Figure 4. Concepts: Examples of "Lshape" and "Rect".

4. Method

4.1. Concept-Relation Graph Neural Network (CR-GNN) Architecture

We propose the Concept-Relation Graph Neural Network (CR-GNN) architecture, which takes advantage of the subgraph isomorphic properties of each task. Our CR-GNN architecture consists of a CNN object encoder f_o to encode each object, and outputs object embedding $z_{i,k}^{(o)}$ for the k^{th} object from image x_i, a MLP relation encoder f_r to encode the relation $z_{i;k,l}^{(r)}$ between the k^{th} and l^{th} object.

Encoding the input image through the object and relation encoders, the image x_i is represented as a latent graph g_i,

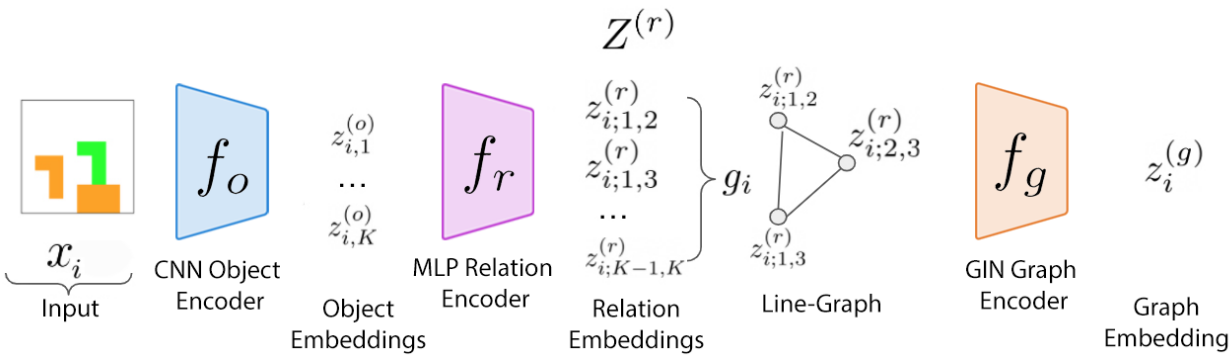


Figure 5. CR-GNN architecture, consisting of a CNN object encoder f_o, a MLP relation encoder f_r, and a GIN f_g.

with $z^{(r)}_{i;k,l}$ being its node features (note that here we use the line-graph representation, where the node features are the relation embedding), and two nodes are connected if two relations shares the same object. After, a 2 layer f_g graph isomorphism network (GIN) [29] is applied to encode the learned line-graph. An illustration of the CR-GNN architecture is shown in Fig. 5.

The CNN is a specific type of a neural network, where at each layer, a set of filters slide over the input to produce the output. At each filter location, the current filter takes a dot product of its weights against the input, which lead to a single output in the respective output channel. A MLP is a standard vanilla multi-layer neural network, with activation functions between each layer. A graph neural network operates on a graph data structure, where the input is a graph consisting of nodes and edges. The GNN processes each node by feeding it through the MLP or single layer, depending on implementation, and uses message passing to aggregate messages from nodes which share a same edge. This is done repeatedly until we arrive at our final representation.

The model design choices are guided by the goal of the task objective. The CNN object encoder embeds each visual object into an embedding vector, and the MLP relation encoder maps each pairwise concatenation of objects into a relation embedding vector. We output an explicit representation for the relation vector encoding as this forces the GNN encoder to depend only on the relational properties of the input, rather than other properties of the objects themselves. We apply a GIN encoder on top of the line-graph constructed with the relation latent vectors to take advantage that all observations of the same task are subgraph-isomorphic.

4.2. Loss function objective

We introduce and explore two main loss objectives to learn the CR-GNN model. The first objective is a contrastive learning based objective which minimizes the distance of the graph representation of each example of the same task (intra-task), while maximizing the distance with respect to examples not in the same task (inter-task). The mathematical formulation of the contrastive objective is:

$$\mathcal{L}_{\text{contrastive}} = \sum_{i,j \in \text{same-task}} ||f_g(g_i) - f_g(g_j)||_2 + \sum_{k,m \in \text{diff-task}} \max(0, \eta - ||f_g(g_k) - f_g(g_m)||_2 \quad (1)$$

where the first summation defines the intra-task loss, and the second summation defines the inter-task loss. Given that each task shares a common subgraph, its graph representation $f_g(g_i)$ should be similar within the task (intra-task loss), and should be different between different tasks (inter-task loss). η here is a margin hyperparameter.

The second objective is a classification (cross-entropy) based objective. Each example is classified as one of task graphs, using the true task label as ground truth. The mathematical formulation of the classification objective is:

$$\mathcal{L}_{\text{classify}} = \sum_{\forall i \in n} \mathcal{L}_{CE}(\text{Linear}(f_g(g_i)), y) \quad (2)$$

where $\mathcal{L}_{CE}$ is the standard cross-entropy loss, between the true task ID y against the predicted task ID. The predictions logits are obtained via a linear layer following the graph representation learned from the GIN f_g. This linear layer is trained along-side the other model components.

An additional loss term, which is the information bottleneck loss, allows constraining the information between the observation X and the relation embedding $Z^{(r)}$, which forces the relation latent dimension to use more clustered embedding. This can be seen as a regularization on $Z^{(r)}$ to capture only the most relevant information about X, which is the relation types in our case. The mathematical formulation of the information bottleneck (IB) is:

$$\mathcal{I}(X; Z^{(r)}) \tag{3}$$

where $\mathcal{I}$ is the mutual information function (Eq. 4), $Z^{(r)}$ is the random variable concatenating the relation embedding $z^{(r)}_{i;k,l}$ for an observation $x_i \in X$. This implementation follows from Alemi et al. [1] which uses a variational upper bound [14] for tractable computation.

5. Experiments

5.1. Setup

In the following experiments, the global relation types for BabyARC dataset are defined to be "none", "same-shape", "same-color", and "inside". Visual examples of these relations are shown in Fig. 2. The object shapes are a rectangle ("rect-solid"), hollow rectangle ("rect"), "Lshape", and line "line", and see Fig. 4 for some examples.

Our dataset specifications also allow us to generate datasets of varying numbers of core objects and distractor objects. We investigate two categories of datasets: tasks containing 2-3 core objects, and tasks containing 2-4 core objects. We also vary the number of distractor objects, with three configurations: no distractor objects, 1 distractor object, and 0-2 distractor objects. The relation specification for each task are described in Appendix A.2.

Examples of two tasks of the (2-3 core, 0 distractor) generated dataset is shown in Fig. 1. In the first image of Task 0, the brown rectangle has the relation "same-color" with the brown "Lshape", and the brown "Lshape" has the relation "same-shape" as the yellow "Lshape".

We train our CR-GNN model with the two loss objectives as previously defined, and on datasets with 2-3 core objects or 2-4 core objects, and varying number of distractor objects depending on the configuration. We also observed the effect of adding the IB loss.

5.2. Hyperparameters: Model and Training

The CNN object encoder is a 4 layer CNN, MLP relation encoder is a 3 layer MLP, and GIN graph encoder is 2 layer GNN with 3 layer MLP for each GNN layer. All the activation functions used are the LeakyRelu function. The model was optimized using the Adam [13] optimizer with 1×10^{-4} learning rate, 0.9 momentum, and no weight decay. 1×10^{-4} learning rate as it was the largest learning rate without the loss diverging, and momentum is a standard value to use with Adam, which is also a standard optimizer which has worked well. Mini-batch size was 168 image examples, which worked well so it was not changed. The latent dimension for objects is 100, and the latent dimension for relations is 20. The input with is an image of $16 \times 16 \times 9 \times n$, where 16×16 is width by height, 9 is the number of total colors, and n is the maximum number of possible objects in the dataset. The ground truth object masks are provided as input, as our work is focused on relation discovery, and not on object discovery. The margin hyperparameter η for the contrastive loss is $20 \times \frac{2}{3}$, where 20 is the relation latent dimension. Each image is augmented with probability 0.9, where the possible augmentations are random flipping, rotating, resizing, and color. The weighting for each respective losses are 1.0 for contrastive, if used, 1.0 for classify, if used, and 0.1 for IB, if used.

Table 1. Sanity check: Relation classification validation accuracy with supervision, 2-3 core objects.

METHOD	ACCURACY
SUPERVISED TRAINING	0.990

Table 2. Relation classification accuracy for 2-3 core objects

METHOD	# DISTRACTORS		
	0	1	0-2
CLASSIFY	0.923	0.926	0.946
CLASSIFY + IB	0.919	0.918	0.901
CONTRASTIVE	**0.959**	0.961	0.954
CONTRASTIVE + IB	0.952	**0.963**	**0.957**
BEST	**0.959**	**0.963**	**0.957**

5.3. Sanity Check

To first verify and sanity check the representational capacity of the model architecture, I first trained the model in a supervised setting, directly training the model with the relation labels. The supervised validation accuracy of the model relation classification is shown in Table 1. This result demonstrates that model is currently of sufficient capacity to learn the relation representations.

5.4. Results

The model is evaluated on two main aspects: the accuracy of the model with predicting the correct relation type between two objects, and inferring the relational graph structure belonging to each task. These evaluations are based off the dataset task goals we have defined in Section 2. Our primary metric for accuracy is the standard accuracy calculation, (# number of correct predictions) \(# number of predictions).

We evaluate the relation prediction accuracy as follows: we apply k-means clustering to assign cluster labels to each of the learned relation embeddings $z^{(r)}_{i;k,l}$. We permute globally how each cluster label is assigned to the ground-truth

Table 3. Relation classification accuracy for 2-4 core objects

METHOD	# DISTRACTORS		
	0	1	0-2
CLASSIFY	0.956	0.955	0.965
CLASSIFY + IB	0.960	0.962	0.959
CONTRASTIVE	**0.965**	0.971	0.965
CONTRASTIVE + IB	0.960	**0.973**	**0.971**
BEST	**0.965**	**0.973**	**0.971**

relation label. The maximum accuracy with respect to the ground truth label $e_{k,l}$ is then taken. We only compute the relation accuracy between objects which contain a relation label, as we do not know the underlying relation of object pairs without labels. Model accuracy evaluation is done on a validation dataset with the same parameters used to generate the training dataset, but with different random seed.

We did not overfit to the training set, as since this is an unsupervised learning objective, we do not provide the ground truth relation labels. We also observe that our validation losses do not go up as training loss decreases, which also indicate that the model is not overfitting.

Table 2 & 3 show the relation classification accuracy for our configurations. The model performance is similar between both objectives, with the contrastive objective performing slightly better. In varying the number of introduced distractor objects in both cases, there is not any accuracy degradation due to this introduction.

Comparing the 2-3 core objects dataset against the 2-4 core objects dataset, there is an overall slight improvement with training on the 2-4 core objects dataset. This is because the 2-4 core objects dataset includes a greater number of task examples (13 vs 6), which allows learning better relation representations in order for the model to distinguish between different graphs.

Our model is able to infer the global relation types, as shown in Fig. 6, a t-SNE visualization of the learned relation embeddings. The model clusters relation embeddings of the same relation label close to each other, even when not given the ground truth relation labels.

To infer the relational graph structure belonging to each task, *k*-means evaluation method is used to predict the cluster labels of the learned relation. The graph for each observation is then constructed with these predicted labels. Since the number of objects in the observations of each task may be different due to distractors, we take the maximum common subgraph (MCS) of all the same-task constructed graphs, which allows identifying the shared relational graph belonging to each task.

Our MCS retrievals for the 2-4 core object, 0 distractor dataset with the contrastive + IB objective are in Appendix A.3, with evaluation method details described. our method is trained and evaluated on the same 2-4 core object, 0 distractor dataset. The top 3 most frequently obtained maximum common subgraphs for each task are shown, and the ground truth relational graph occurs in the top 3 retrievals for most tasks. This demonstrates that our method is able to uncover the ground truth shared relational graph structure for most tasks.

Furthermore, we demonstrate the ability of our model to generalize to tasks unseen during training. We train our method on the 2-3 core object, 1 distractor dataset with the contrastive + IB objective, and at inference time, evaluate MCS retrievals on the 2-4 core object, 0 distractor dataset. The relational structure of Tasks 6 to 12 are not seen during training, but obtains the correct MCS in the top 3 retrievals for most tasks. The full retrievals are shown in Appendix A.4.

One current limitation is that the model only learns the necessary relation representations needed to distinguish between the given tasks. We observed this similarly in our earlier experiments, where we evaluated our method in a setting with a limited number of tasks, and our method only learned two relation representations as it was sufficient to separate the tasks: “inside“ and “same-shape\same-color“.

Our method currently does not distinguish between relations without a label “none“, and the “same-color“ relation. This can be visualized in the t-SNE visualization in Fig. 6, where the “none“ and “same-color“ relation distributions overlap with each other. In the MCS retrievals, extraneous predictions of “same-color“ are found due to this reason.

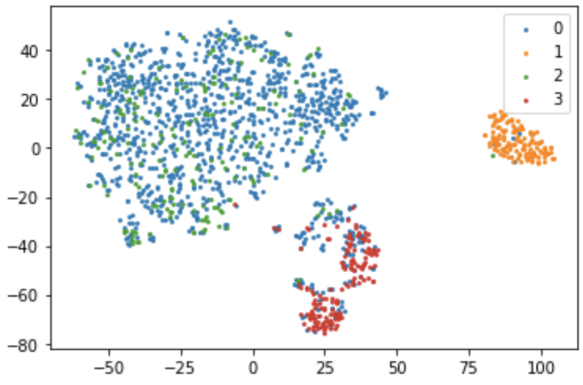


Figure 6. t-SNE visualization of learned relation embeddings, for 2-3 core, 0-2 distractor dataset with contrastive + IB objective. The colors represent ground truth labels, where label 0 represents no relationship label “none”, 1 is “inside”, 2 is “same-color”, 3 is “same-shape”.

6. Conclusion

In this work, we propose our method for unsupervised learning and discovery of visual relations with graph-level analogy. We show that our method is able to infer the global relation types, achieve above 95% accuracy in relation classification, and retrieve the shared relational graph structure for most tasks.

We found that using the contrastive method worked best for learning the graph embedding for each input, as this gave us best results in each scenario for relation classification. The reason is likely due to that the relation embeddings are better clustered when using this loss, thus using k-means is able to achieve a higher accuracy. We also found that the information bottleneck, while designed to encourage the relation embedding to only focus on the relevant information in the input, did not change model performance by much. We also see that introducing more tasks helped improve relation classification, which generally aligns with our intuition that introducing more tasks will result in a better learning signal.

Per our discussion on limitations, future work would be understanding how to separate "none" and "same-color" representations, with a promising approach being introducing more diverse tasks. I would also expand towards more datasets, such as potentially trying to allow generation of CLEVR with specified graph structure.

7. Contributions & Acknowledgements

My non CS231N collaborator is Tailin Wu, a Stanford CS postdoc. Wu served as an advisor for the project, and helped guide ideas and approach throughout the project. SNAP group hosted GPU resources.

References

[1] Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep variational information bottleneck. *CoRR*, abs/1612.00410, 2016. 5

[2] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Deep compositional question answering with neural module networks. *CoRR*, abs/1511.02799, 2015. 2

[3] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. VQA: visual question answering. *CoRR*, abs/1505.00468, 2015. 2

[4] Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. *CoRR*, abs/1612.00222, 2016. 2

[5] Shyamal Buch, Li Fei-Fei, and Noah D. Goodman. Neural event semantics for grounded language understanding. *Transactions of the Association for Computational Linguistics*, 9:875–890, 2021. 2

[6] Christopher P. Burgess, Loïc Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matthew M. Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *CoRR*, abs/1901.11390, 2019. 2

[7] François Chollet. On the measure of intelligence. *CoRR*, abs/1911.01547, 2019. 2

[8] Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Chris Burgess, Daniel Zoran, Loïc Matthey, Matthew M. Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. *CoRR*, abs/1903.00450, 2019. 2

[9] Jiuxiang Gu, Handong Zhao, Zhe Lin, Sheng Li, Jianfei Cai, and Mingyang Ling. Scene graph generation with external knowledge and image reconstruction. *CoRR*, abs/1904.00560, 2019. 2

[10] Drew A. Hudson and Christopher D. Manning. GQA: a new dataset for compositional question answering over real-world images. *CoRR*, abs/1902.09506, 2019. 2

[11] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. *CoRR*, abs/1804.01622, 2018. 2

[12] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross B. Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. *CoRR*, abs/1612.06890, 2016. 2

[13] 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. 5

[14] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013. 5

[15] Thomas Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. 2019. 2

[16] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *CoRR*, abs/1602.07332, 2016. 2

[17] Yikang Li, Tao Ma, Yeqi Bai, Nan Duan, Sining Wei, and Xiaogang Wang. Pastegan: A semi-parametric method to generate image from scene graph. *CoRR*, abs/1905.01608, 2019. 2

[18] Yikang Li, Wanli Ouyang, and Xiaogang Wang. Vip-cnn: A visual phrase reasoning convolutional neural network for visual relationship detection. *CoRR*, abs/1702.07191, 2017. 2

[19] Fangyu Liu, Guy Emerson, and Nigel Collier. Visual spatial reasoning, 2022. 2

[20] Nan Liu, Shuang Li, Yilun Du, Joshua B. Tenenbaum, and Antonio Torralba. Learning to compose visual relations. *CoRR*, abs/2111.09297, 2021. 2

[21] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *CoRR*, abs/2006.15055, 2020. 2

[22] Cewu Lu, Ranjay Krishna, Michael S. Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. *CoRR*, abs/1608.00187, 2016. 2

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 7

[24] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009. 7

[25] Murray Shanahan, Kyriacos Nikiforou, Antonia Creswell, Christos Kaplanis, David Barrett, and Marta Garnelo. An explicitly relational neural network architecture. *CoRR*, abs/1905.10307, 2019. 2

[26] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. *CoRR*, abs/1711.06025, 2017. 2

[27] Ruocheng Wang, Jiayuan Mao, Samuel J. Gershman, and Jiajun Wu. Language-mediated, object-centric representation learning. *CoRR*, abs/2012.15814, 2020. 2

[28] Tailin Wu, Megan Tjandrasuwita, Zhengxuan Wu, Xuelin Yang, Kevin Liu, Rok Sosič, and Jure Leskovec. Zeroc: A neuro-symbolic model for zero-shot concept recognition and acquisition at inference time. *Under review*, 2022. 2

[29] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018. 4

[30] Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Joshua B. Tenenbaum. Neural-symbolic VQA: disentangling reasoning from vision and language understanding. *CoRR*, abs/1810.02338, 2018. 2

A. Appendix

A.1. Mutual Information Function

The mutual information for PDFs for continuous distributions is defined as the following:

$$\mathcal{I}(X,Y) = \int_y \int_x p(y,x) \log \frac{p(y,x)}{p(y)p(x)} dx\, dy \tag{4}$$

A.2. Tasks in each Dataset

The notation is the following: [(0, 1), 'same-color'] represents that the relation 'same-color' holds between the 0th and the 1st object. Only core objects are defined in the relations specification, not distractor objects. Visual examples for each task are shown in Fig. 7.

For datasets containing 2-3 core objects, the tasks in the dataset are the following:

Task 0 (3 objects): [(0, 1), 'same-color'], [(1, 2), 'same-shape']
Task 1 (2 objects): [(0, 1), 'same-color']
Task 2 (2 objects): [(0, 1), 'inside']
Task 3 (3 objects): [(0, 1), 'inside'], [(1, 2), 'same-color']
Task 4 (3 objects): [(0, 1), 'inside'], [(1, 2), 'same-shape']
Task 5 (3 objects): [(0, 1), 'same-color'], [(1, 2), 'same-color']

For datasets containing 2-4 core objects, the tasks in the dataset are the following:
Task 0 (2 objects): [(0, 1), 'inside']
Task 1 (3 objects): [(0, 1), 'inside'], [(1, 2), 'same-color']
Task 2 (3 objects): [(0, 1), 'inside'], [(1, 2), 'same-shape']
Task 3 (2 objects): [(0, 1), 'same-color']
Task 4 (3 objects): [(0, 1), 'same-color'], [(1, 2), 'same-color']
Task 5 (3 objects): [(0, 1), 'same-color'], [(1, 2), 'same-shape']
Task 6 (4 objects): [(0, 2), 'same-color'], [(1, 2), 'same-color'], [(2, 3), 'same-shape']
Task 7 (4 objects): [(0, 1), 'inside'], [(1, 2), 'same-color'], [(2, 3), 'same-color']
Task 8 (4 objects): [(0, 2), 'same-shape'], [(1, 2), 'same-color'], [(2, 3), 'same-shape']
Task 9 (4 objects): [(0, 1), 'inside'], [(1, 2), 'same-color'], [(2, 3), 'same-shape']
Task 10 (4 objects): [(0, 1), 'inside'], [(1, 2), 'same-shape'], [(2, 3), 'same-shape']
Task 11 (4 objects): [(0, 1), 'inside'], [(1, 2), 'same-shape'], [(2, 3), 'same-color']
Task 12 (4 objects): [(0, 2), 'same-color'], [(1, 2), 'same-color'], [(2, 3), 'same-color']

A.3. Task Maximum Common Subgraph Retrieval, with all seen tasks

The following show the Maximum Common Subgraph Retrievals for our method trained on 2-4 core object, 0 distractor dataset with the contrastive + IB objective. The evaluation dataset is the same, on 2-4 core object, 0 distractor dataset. Thus, relational structure of all the tasks shown is seen by the model during training.

To obtain the shared relational graph structure for each task, we would ideally take the maximum common subgraph of all the constructed graphs on each task. In practice, however, the relation predictions are not perfect and one incorrect relation prediction would then change the whole maximum common subgraph. To mitigate this, we compute maximum common subgraph of subgroups instead of all the observations at once, and then use a counting mechanism to identify the most frequently retrieved maximum common subgraph. The size of these subgroups is referred as the "group size". In the following result, the group size used is 5. The notation for interpreting the result is same as in Section A.2.

Task 0:
Count: 420, **MCS:** [(0, 1), 'inside']

Task 1:
Count: 287, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(1, 2), 'same-color']
Count: 133, **MCS:** [(0, 1), 'inside'], [(1, 2), 'same-color']
Task 2:
Count: 337, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(1, 2), 'same-shape']
Count: 83, **MCS:** [(0, 1), 'inside'], [(1, 2), 'same-shape']
Task 3:
Count: 420, **MCS:** [(0, 1), 'same-color']
Task 4:
Count: 270, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(1, 2), 'same-color']
Count: 150, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color']
Task 5:
Count: 305, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-shape'], [(1, 2), 'same-color']
Count: 115, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-shape']
Task 6:
Count: 159, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-shape']
Count: 144, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(2, 3), 'same-shape']
Count: 78, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(1, 2), 'same-color'], [(2, 3), 'same-shape']
Task 7:
Count: 230, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-color']
Count: 101, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-color']
Count: 69, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color']
Task 8:
Count: 242, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-shape'], [(0, 3), 'same-shape'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-shape']
Count: 110, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-shape'], [(0, 3), 'same-shape'], [(1, 2), 'same-color'], [(2, 3), 'same-shape']
Count: 39, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-shape'], [(0, 3), 'same-shape'], [(2, 3), 'same-shape']
Task 9:
Count: 195, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-shape']
Count: 112, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(2, 3), 'same-shape']
Count: 97, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(2, 3), 'same-shape']
Task 10:
Count: 253, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-shape'], [(1, 3), 'same-shape'], [(2, 3), 'same-shape']
Count: 117, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-shape'], [(1, 3), 'same-shape']
Count: 37, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-shape']
Task 11:
Count: 196, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-shape'], [(1, 3), 'same-color'], [(2, 3), 'same-color']
Count: 117, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-shape'], [(1, 3), 'same-color']
Count: 82, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-shape']
Task 12:
Count: 15, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-color']
Count: 9, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-color']
Count: 2, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color']

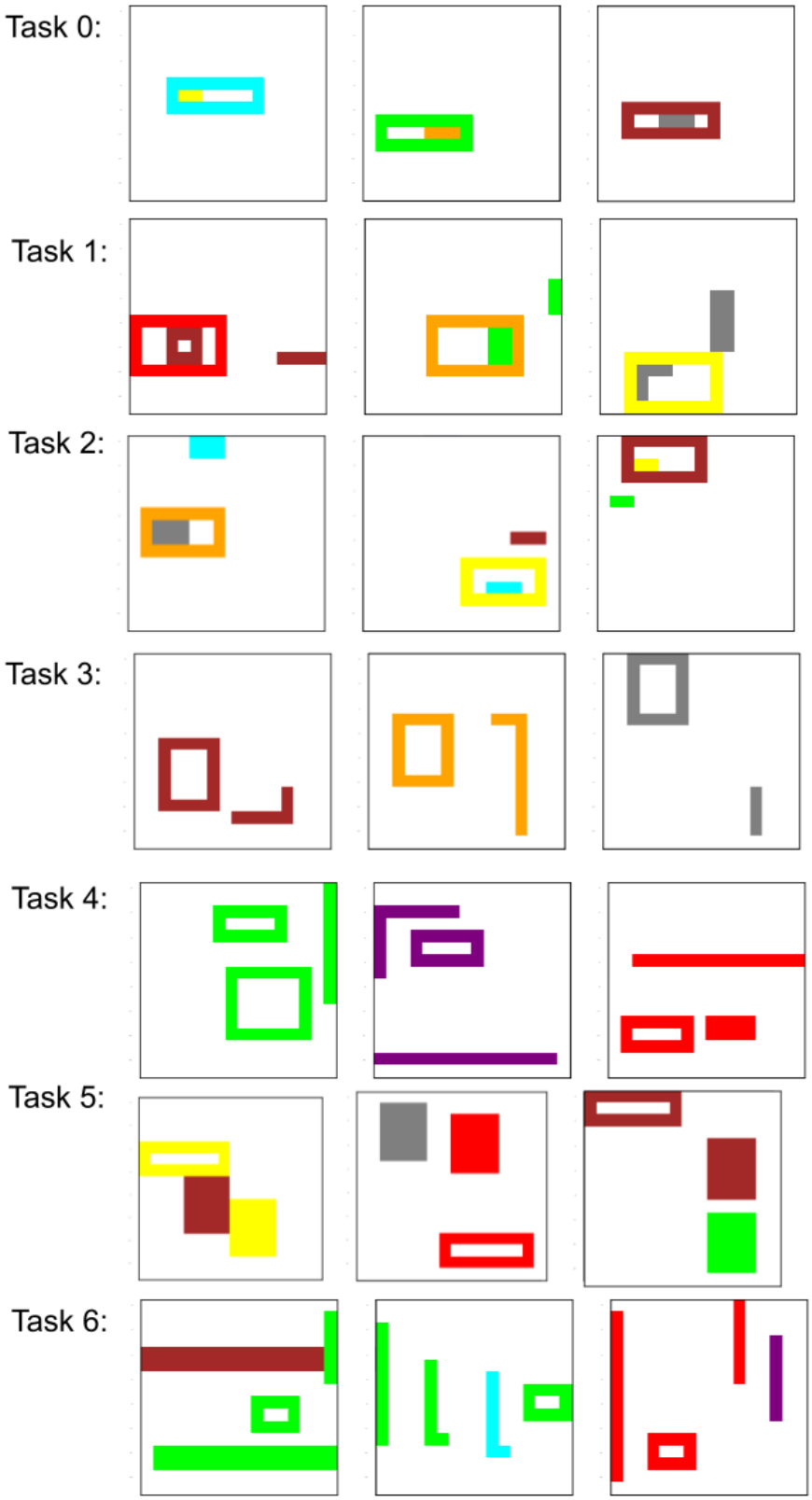
A.4. Task Maximum Common Subgraph Retrieval, with unseen tasks

The following show the Maximum Common Subgraph Retrievals for our method trained on 2-3 core object, 1 distractor dataset with the contrastive + IB objective. The evaluation dataset is different, on the 2-4 core object, 0 distractor dataset. As a result, the relational structure of Tasks 6 to 12 are not seen by the model during training.

Task 0:
Count: 420, **MCS:** [(0, 1), 'inside']
Task 1:
Count: 270, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(1, 2), 'same-color']
Count: 150, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color']
Task 2:
Count: 360, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(1, 2), 'same-shape']
Count: 60, **MCS:** [(0, 1), 'inside'], [(1, 2), 'same-shape']
Task 3:
Count: 420, **MCS:** [(0, 1), 'same-color']
Task 4:
Count: 255, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(1, 2), 'same-color']
Count: 165, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color']
Task 5:
Count: 214, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-shape'], [(1, 2), 'same-color']
Count: 206, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-shape']
Task 6:
Count: 148, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-shape']
Count: 146, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(1, 3), 'same-color']
Count: 79, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-shape'], [(1, 2), 'same-color'], [(1, 3), 'same-color']
Task 7:
Count: 220, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(2, 3), 'same-color']
Count: 116, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-color']
Count: 67, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color']
Task 8:
Count: 135, **MCS:** [(0, 1), 'same-color'], [(0, 3), 'same-shape'], [(1, 2), 'same-color'], [(1, 3), 'same-color']
Count: 120, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-shape'], [(0, 3), 'same-shape'], [(1, 2), 'same-color'], [(1, 3), 'same-color']
Count: 107, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-shape'], [(0, 3), 'same-shape'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-shape']
Task 9:
Count: 137, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(2, 3), 'same-shape']
Count: 128, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-shape']
Count: 108, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(2, 3), 'same-shape']
Task 10:
Count: 258, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-shape'], [(1, 3), 'same-shape'], [(2, 3), 'same-shape']
Count: 112, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(1, 2), 'same-shape'], [(1, 3), 'same-shape'], [(2, 3), 'same-shape']
Count: 46, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-shape']
Task 11:
Count: 201, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-shape'], [(1, 3), 'same-color'], [(2, 3), 'same-color']
Count: 98, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-shape'], [(1, 3), 'same-color']

Count: 93, **MCS:** [(0, 1), 'inside'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-shape']
Task 12:
Count: 16, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-color']
Count: 9, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(1, 2), 'same-color'], [(1, 3), 'same-color'], [(2, 3), 'same-color']
Count: 2, **MCS:** [(0, 1), 'same-color'], [(0, 2), 'same-color'], [(0, 3), 'same-color'], [(2, 3), 'same-color']

A.5. Additional BabyARC examples



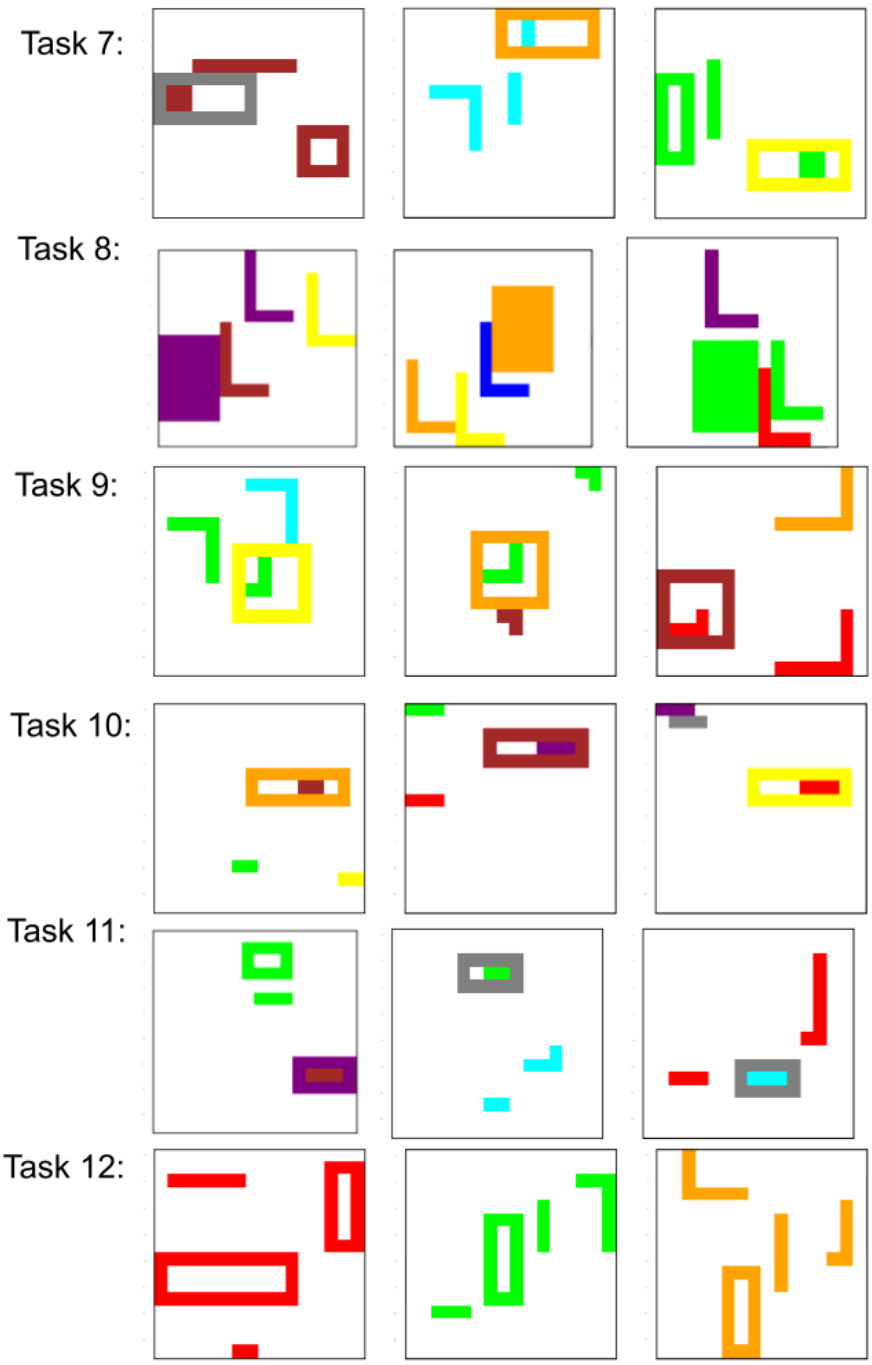


Figure 7. BabyARC: The images above depict 3 examples for each task. While only 3 are shown for each task, we use 200-300 examples per task during training.

A.6. Additional ARC example tasks

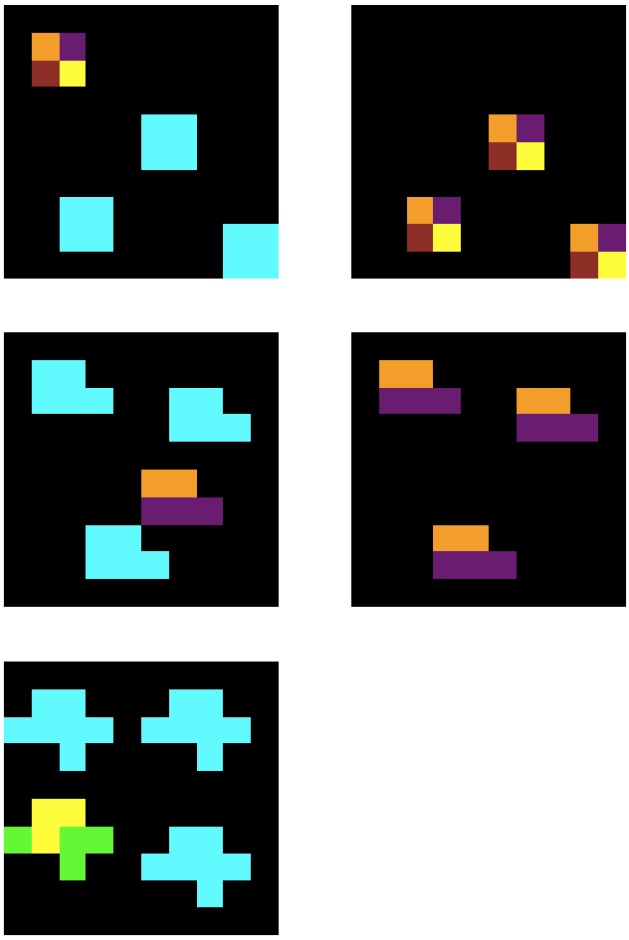


Figure 8. Example of an ARC task. The first column represents the input, and the second column represents the output, and the first two rows are independent training examples. The last row is the test example for this task, where only the input image is given. In this task, the objective is to color all the cyan objects with the same pattern as the non-cyan object, and then remove the original non-cyan object. This procedure explains obtaining the output from the input. All the training examples share a common relational graph structure, between the inputs and between the outputs. In the inputs, all the objects hold the relation "same-shape", and three of the objects hold the relation "same-color". In the outputs, all the three objects hold the relation "same-shape" and "same-color". The ability to learn and identify relations in unsupervised manner is essential, as this task requires both understanding of "same-shape" and "same-color" relations.

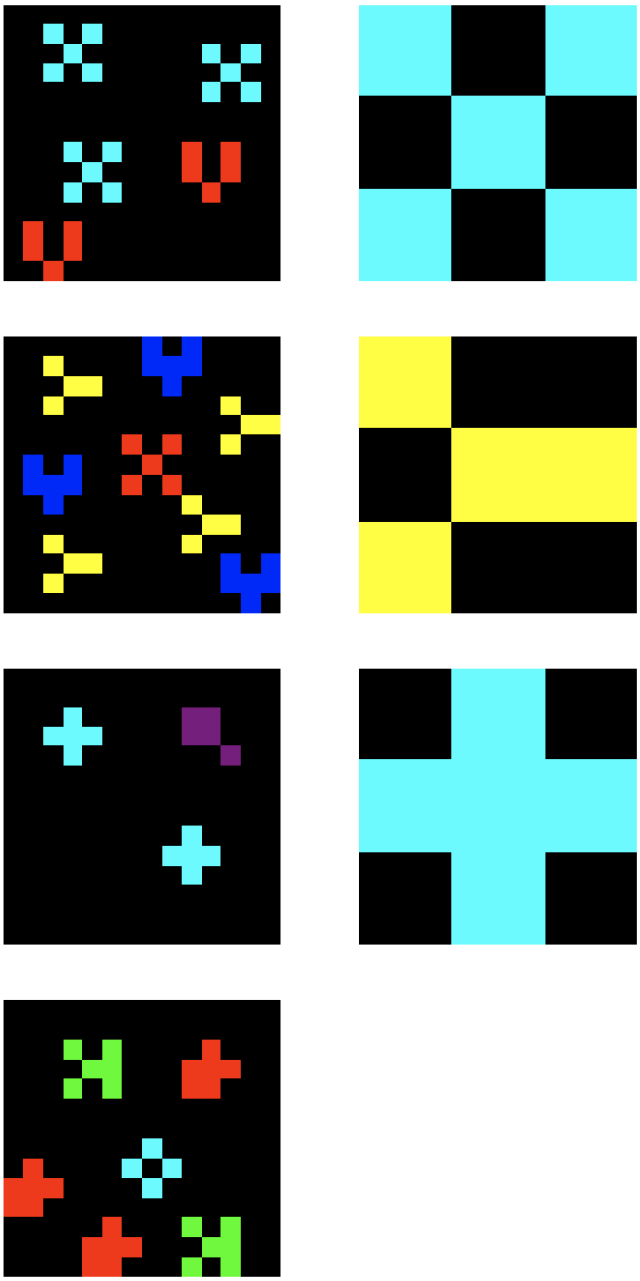


Figure 9. Another example of an ARC task. See caption in Fig. 8 for general task description. In this task, the objective is to count the unique distinct objects by shape and color, and present the object with the largest count. In the inputs, the relational graph is the following: various groups of objects share the relation of "same-color" and "same-shape" with each other.

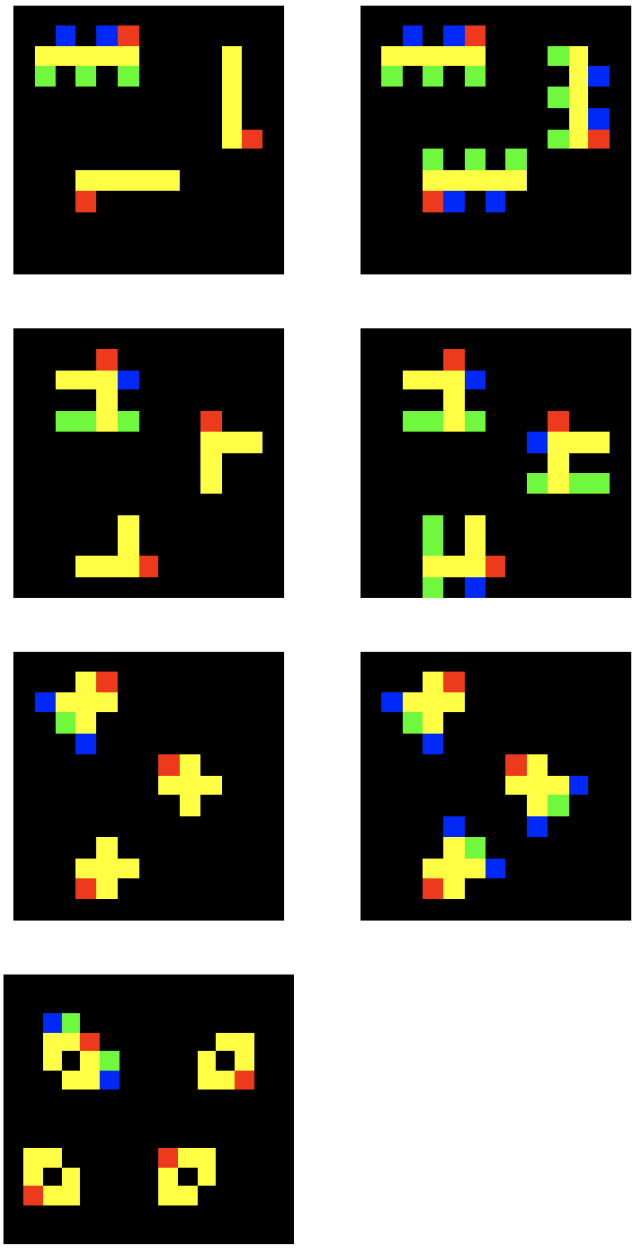


Figure 10. Another example of an ARC task. See caption in Fig. 8 for general task description. In this task, the objective is identify the object with blue and green colors, and copy the object to the other shapes by aligning the red and yellow pixels. In the inputs, the relational graph is the following: all but one of the objects hold the relation “same-shape“ and “same-color“. In the outputs, all of the objects hold the relation “same-shape“ and “same-color“.