Slot Abstractors: Toward Scalable Abstract Visual Reasoning

Shanka Subhra Mondal [1] **Jonathan D. Cohen** [2] **Taylor W. Webb** [3]

Abstract

Abstract visual reasoning is a characteristically human ability, allowing the identification of relational patterns that are abstracted away from object features, and the systematic generalization of those patterns to unseen problems. Recent work has demonstrated strong systematic generalization in visual reasoning tasks involving multi-object inputs, through the integration of slot-based methods used for extracting object-centric representations coupled with strong inductive biases for relational abstraction. However, this approach was limited to problems containing a single rule, and was not scalable to visual reasoning problems containing a large number of objects. Other recent work proposed Abstractors, an extension of Transformers that incorporates strong relational inductive biases, thereby inheriting the Transformer's scalability and multi-head architecture, but it has yet to be demonstrated how this approach might be applied to multi-object visual inputs. Here we combine the strengths of the above approaches and propose Slot Abstractors, an approach to abstract visual reasoning that can be scaled to problems involving a large number of objects and multiple relations among them. The approach displays state-of-the-art performance across four abstract visual reasoning tasks, as well as an abstract reasoning task involving real-world images.

1. Introduction

Abstract visual reasoning problems contain visual objects, the features of which collectively exemplify an abstract pattern or rule. These are a common testbed of human intelligence. Given a few example demonstrations of such problems, humans can easily identify the abstract pattern or rule, and then systematically generalize it to novel stimuli (Raven, 1938; Kotovsky & Gentner, 1996; Fleuret et al., 2011; Lake et al., 2015). Neural networks, on the other hand, often struggle to perform this type of abstract visual reasoning, tending to overfit to the concrete details of problems in the training data, and failing to extract the underlying abstract pattern or rule (Lake & Baroni, 2018; Barrett et al., 2018; Ricci et al., 2018).

To build models that can demonstrate human-like systematic generalization, several methods have recently been proposed (Webb et al., 2020; 2021; Kerg et al., 2022; Altabaa et al., 2023) to promote inference of relations among objects in the internal representations of neural networks. These methods implement a simple architectural inductive bias for relational abstraction, called the "relational bottleneck" (Webb et al., 2023a), that drives the network to abstract over the features of objects, and identify relations among them that are necessary to perform the task. These methods demonstrated the learning and strong systematic generalization of abstract patterns from a few training examples, but were limited to inputs with pre-segmented visual objects. To address this, subsequent work proposed Object-Centric Relational Abstraction (OCRA) (Webb et al., 2023b), an approach that integrates a relational bottleneck with object-centric representation learning methods (Greff et al., 2019; Burgess et al., 2019; Locatello et al., 2020; Engelcke et al., 2021). Though OCRA extends the capacity for strong abstract visual reasoning to images with more than one object, it is not scalable to problems with a large number of objects (example problem in Figure 1). This is because OCRA computes relational embeddings for all pairs of N objects in a given scene, which are then further processed by a Transformer (Vaswani et al., 2017), resulting in a combined complexity of $O(N^4)$. Moreover, OCRA's reasoning capacity was limited to problems involving a single relation.

Recently, Abstractors (Altabaa et al., 2023) were proposed as an extension of Transformers (Vaswani et al., 2017) for modeling relations between objects disentangled from the object features. The Abstractor implements a relational bottleneck using *relational cross-attention*, a variant of the attention mechanisms in Transformers. Though this ap-

[1]Department of Electrical and Computer Engineering, Princeton University, Princeton, NJ, US [2]Princeton Neuroscience Institute, Princeton University, Princeton, NJ, US [3]Department of Psychology, University of California Los Angeles, Los Angeles, CA, US. Correspondence to: Shanka Subhra Mondal <smondal@princeton.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

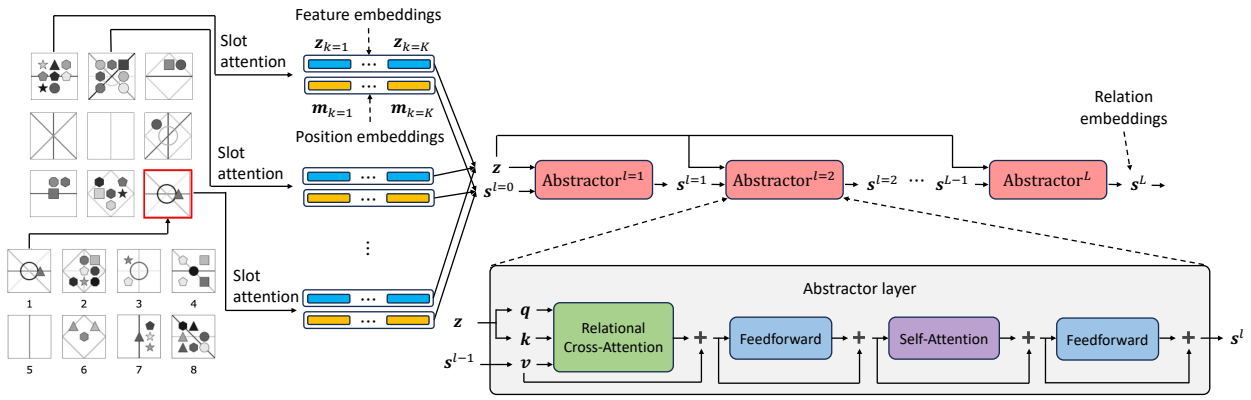


Figure 1. **Slot Abstractor**. The Slot Abstractor consists of two major components. First, object-centric representations are extracted using Slot Attention (Locatello et al., 2020). Relation embeddings are then computed using a series of Abstractor layers (Altabaa et al., 2023). Example problem on left is from the PGM dataset (Barrett et al., 2018), consisting of a 3×3 matrix of image panels populated with objects. The task is to identify the abstract pattern among the image panels, and use this pattern to fill in the missing panel (bottom right), selecting from a set of eight choices. To generate scores for each answer choice, the corresponding image panel is inserted into the problem. Slot attention is then used to extract feature embeddings $\boldsymbol{z}_{k=1\ldots K}$, and position embeddings $\boldsymbol{m}_{k=1\ldots K}$ for each panel. Relation embeddings $\boldsymbol{s}$ are then computed through a series of Abstractor layers. Each layer consists of relational cross-attention, self-attention, and feedforward layers, with residual connections after each of these. Relational cross-attention uses feature embeddings to generate keys k and queries q, and the relation embeddings from the previous layer to generate values v. Relation embeddings are initialized using position embeddings. After L Abstractor layers, relation embeddings are averaged and passed through a linear layer to generate a score y.

proach exhibits superior sample efficiency and improved generalization performance compared to other relational architectures (Shanahan et al., 2020; Kerg et al., 2022), it has yet to be applied to visual problems involving multi-object inputs (Fleuret et al., 2011; Barrett et al., 2018; Webb et al., 2023b). Here, we address this challenge, taking a step toward the development of scalable abstract visual reasoning algorithms that can be effectively applied to problems with a large number of objects and multiple relations among them.

Specifically, we combine Slot Attention (Locatello et al., 2020), a method for unsupervised object-centric representation learning, with Abstractors (Altabaa et al., 2023). The combined approach, Slot Abstractors (Figure 1), inherits the $O(N^2)$ complexity of Transformers, allowing it to be scaled to much more complex problems than previous abstract visual reasoning methods. The approach also inherits the Transformer's multi-head architecture, enabling it to solve problems involving multiple relations. We evaluate the Slot Abstractor on five abstract visual reasoning tasks (including a task involving real-world images), with a diverse range of visual and rule complexity. We find that the Slot Abstractor is capable of strong systematic generalization of learned abstract rules, and can be scaled to problems with multiple rule types and more than 100 objects, significantly improving over a number of competitive baselines in most settings.

2. Approach

Figure 1 shows a schematic description of our approach (described in detail in Algorithm 1). The Slot Abstractor consists of two major components: 1) extraction of object-centric representations; and 2) extraction of relational representations abstracted away from the object-level representations. We describe these two components in detail in the following sections.

2.1. Object-Centric Representation Learning

The Slot Abstractor employs slot attention (Locatello et al., 2020) in the same way as OCRA (Webb et al., 2023b), to extract object-centric representations from multi-object visual inputs.

Given an image $\boldsymbol{x}$, slot attention learns to extract a set of latent embeddings (i.e., *slots*), each of which captures a focused representation of a single visual object, when trained to reconstruct the image through decoding the object representations. The entire process is done without ground truth segmentation data and hence is completely unsupervised.

To accomplish this, the image is first passed through a convolutional encoder, which generates a feature map $\boldsymbol{feat} \in \mathbb{R}^{H \times W \times D}$. A position code (encoding the four cardinal directions) is passed through a linear layer, which generates positional embeddings $\boldsymbol{pos} \in \mathbb{R}^{H \times W \times D}$. Feature and position embeddings are additively combined, and then passed

through a series of 1x1 convolutions, followed by flattening to generate $\boldsymbol{inputs} \in \mathbb{R}^{N \times D}$ (where $N = H \times W$).

Then, a set of K $\boldsymbol{slots} \in \mathbb{R}^{K \times D}$ is randomly initialized (from a normal distribution with learnable shared mean and variance parameters). The image is then presented, and the slots attend over the pixels of the feature map, through a transformer-style cross-attention. Specifically, each slot emits a query $\mathrm{q}(\boldsymbol{slots}) \in \mathbb{R}^{K \times D}$ (through a linear projection), and each location in the feature map emits a key $\mathrm{k}(\boldsymbol{inputs}) \in \mathbb{R}^{N \times D}$ and a value $\mathrm{v}(\boldsymbol{inputs}) \in \mathbb{R}^{N \times D}$. An attention distribution over the feature map $\boldsymbol{attn} = \mathrm{softmax}(\frac{1}{\sqrt{D}} \mathrm{k}(\boldsymbol{inputs}) \cdot \mathrm{q}(\boldsymbol{slots})^\top)$ is generated for each slot, and a weighted mean of the values $\boldsymbol{updates} = \boldsymbol{attn} \cdot \mathrm{v}(\boldsymbol{inputs})$ is used to update the slot representations using a Gated Recurrent Unit (Cho et al., 2014), followed by a feedforward network with residual connection. After T iterations of slot attention, the updated slot representations are passed through a spatial broadcast decoder (Watters et al., 2019), which generates both a reconstructed image and a mask for each slot. The masks are normalized using a softmax across the slots, and the normalized masks are used to compute a weighted average of the slot-specific reconstructions, which generates a combined reconstruction of the input image.

To compute relational representations, it is crucial to represent position distinctly from other object features. For a given input image, we used the final attention map $\boldsymbol{attn}_T \in \mathbb{R}^{K \times N}$ after T iterations of slot attention, to compute feature- and position-specific embeddings for each slot:

$$\boldsymbol{z}_k = \boldsymbol{attn}_T \, \mathrm{flatten}(\boldsymbol{feat})[k] \tag{1}$$

$$\boldsymbol{m}_k = \boldsymbol{attn}_T \, \mathrm{flatten}(\boldsymbol{pos})[k] \tag{2}$$

where $\boldsymbol{feat}$ and $\boldsymbol{pos}$ represent the feature- and position-embeddings before being additively combined. The position embeddings help to keep track of which relations correspond to which objects.

Although slot attention works well for synthetic images, it is not as effective for real-world images (Seitzer et al., 2022). To extend our approach to problems involving real-world images, we used DINOSAUR (Seitzer et al., 2022), an approach in which slot attention is applied to the features obtained from DINO (Caron et al., 2021) (a large-scale pre-trained vision transformer (ViT) (Dosovitskiy et al., 2020)). This approach has been shown to be an effective method for unsupervised instance segmentation of real-world images. The final attention map is used with the ViT's positional embeddings to compute position-specific embeddings for each slot as described in Equation (2). The slot embeddings obtained after performing slot attention over the ViT's features constituted the feature-specific embeddings for each slot.

We assume that human visual representations are generally not learned from scratch when performing each new task. Instead, visual object representations are shaped by a wide variety of non-task-specific prior experiences. To model the effects of this prior experience, we pre-trained slot attention on a reconstruction objective, using a slot-based autoencoder framework (more details in Section 4.3).

2.2. Relational Representation Learning

The Slot Abstractor employs the Abstractor module (Altabaa et al., 2023) to process the object-centric representations extracted by the Slot Attention mechanism. The core operation of the Abstractor module is relational cross-attention (RCA), a variant on the standard form of cross-attention found in Transformers (Vaswani et al., 2017). Like Transformers, the Abstractor also employs a multi-head architecture, enabling it to model problems with multiple distinct relations.

The Abstractor module computes abstract relational representations $\boldsymbol{s}$ over a series of L layers. The initial relational representations $\boldsymbol{s}^{l=0}$ are formed using the position-specific embeddings $\boldsymbol{m}$ obtained from slot attention. Then, at each layer l, these representations are updated using multi-head relational cross-attention between the feature-specific embeddings $\boldsymbol{z}$:

$$\boldsymbol{s}^l = \mathrm{RCA}(\boldsymbol{z}, \boldsymbol{s}^{l-1}) \tag{3}$$

$$\mathrm{RCA}(\boldsymbol{z}, \boldsymbol{s}^{l-1}) = \mathrm{concat}(\tilde{\boldsymbol{s}}^{\boldsymbol{h=1}}, .., \tilde{\boldsymbol{s}}^{\boldsymbol{h=H}})\boldsymbol{W_o} \tag{4}$$

$$\tilde{\boldsymbol{s}}^h = \mathrm{softmax}\left(\frac{(\boldsymbol{z}\boldsymbol{W_q}^h)^T(\boldsymbol{z}\boldsymbol{W_k}^h)}{\sqrt{D}}\right)\boldsymbol{s}^{l-1}\boldsymbol{W_v}^h \tag{5}$$

where $\boldsymbol{W_q}^h \in \mathbb{R}^{D \times D}, \boldsymbol{W_k}^h \in \mathbb{R}^{D \times D}, \boldsymbol{W_v}^h \in \mathbb{R}^{D \times D}$ are the linear projection matrices used by the hth head to generate queries, keys, and values respectively; $\tilde{\boldsymbol{s}}^h$ is the result of relational cross-attention in the hth head; and $\boldsymbol{W_o}$ are the output weights through which the concatenated outputs of all H heads are passed. The key difference relative to standard cross-attention is that queries and keys are generated from the input features, enabling the relations between those features to be modeled through inner products, and the values are generated from a separate set of (position-specific) embeddings. This relational variant on cross-attention implements a relational bottleneck (Webb et al., 2023a), in that downstream processing (which only has access to $\boldsymbol{s}$) is

driven only by the relations between objects, as represented by inner products (see Altabaa and Lafferty (2024) for theoretical considerations regarding the use of inner products to represent relations), and thus is disentangled from the individual features of those objects. We hypothesized that this relational bottleneck would enable systematic generalization to problems with previously unseen object features.

In each layer, multi-head relational cross-attention was composed together with feedforward networks and standard multi-head self-attention, with residual connections after each of these, as depicted in Figure 1, and described in detail in Algorithm 1. The application of multiple layers of relational and self-attention enabled the Slot Abstractor to flexibly model higher-order relations (relations between relations), which play an important role in abstract visual reasoning. Crucially, this approach retains the quadratic complexity of standard Transformers (given K slots, Slot Abstractors have $O(K^2)$ complexity), making it feasible to scale the approach to visual inputs containing a large number of objects. After L layers, the relational representations were averaged over the K slots, and passed through a task-specific output layer to generate a score y. Depending on the task, a score y was generated for each answer choice, and the abstractor module was trained using cross entropy loss. More details can be found in the last paragraph of Section 4.3.

3. Related Work

A number of neural network models have been proposed (Barrett et al., 2018; Steenbrugge et al., 2018; Zhang et al., 2019b; Zheng et al., 2019; Spratley et al., 2020; Jahrens & Martinetz, 2020; Wang et al., 2020; Wu et al., 2020; Benny et al., 2021; Hu et al., 2021; Zhuo & Kankanhalli, 2022; Zhang et al., 2022; Yang et al., 2023) to solve abstract visual reasoning problems (Barrett et al., 2018; Zhang et al., 2019a; Teney et al., 2020) based on Raven's Progressive Matrix (RPM) (Raven, 1938). Many of these models incorporate problem-specific inductive biases, and therefore cannot be applied to other types of visual reasoning problems. To address this, the Slot Transformer Scoring Network (STSN) was proposed (Mondal et al., 2023), combining a transformer reasoning module with a general-purpose inductive bias for object-centric visual processing (Greff et al., 2019; Burgess et al., 2019; Locatello et al., 2020; Engelcke et al., 2021; Dittadi et al., 2021; Jiang et al., 2023). This approach achieved state-of-the-art performance on RPM-based abstract visual reasoning tasks, consistent with results from object-centric models in other visual reasoning tasks (Ding et al., 2021; Wu et al., 2022). However, unlike human abstract reasoning, these approaches generally require very large training sets, and tend to generalize poorly out-of-distribution.

To address these limitations, neural network architectures have recently been proposed that incorporate strong relational inductive biases (Webb et al., 2021; Kerg et al., 2022). By constraining downstream processing to focus only on the relations between inputs (and to ignore the concrete features of those inputs), these architectures demonstrated human-like systematic generalization of learned abstract rules from a few training examples. This approach has more generally been referred to as the "relational bottleneck" principle (Webb et al., 2023a). Some of these architectures, however, have been limited by their dependence on inputs consisting of pre-segmented visual objects.

One recent approach, Object-Centric Relational Abstraction (OCRA) (Webb et al., 2023b), combined both strong relational and object-centric inductive biases (specifically Slot Attention (Locatello et al., 2020)) to demonstrate systematic generalization for abstract visual reasoning problems based on multi-object inputs. This approach was limited, however, both by its inability to process problems with multiple relations, and by its computational complexity ($O(N^4)$), preventing it from being scaled to problems containing a large number of objects.

Another recent work proposed Abstractors (Altabaa et al., 2023), an extension of Transformers (Vaswani et al., 2017) that implemented the relational bottleneck using a relational variant of cross-attention. This approach demonstrated strong sample efficiency and generalization performance, but has yet to be explored for abstract visual reasoning problems with multi-object inputs (Fleuret et al., 2011; Barrett et al., 2018; Webb et al., 2023b). Abstractors inherit the quadratic complexity of Transformers, making it feasible to scale them to problems with a large number of objects. Abstractors also employ a multi-head architecture, similar to Transformers, allowing them to model multiple distinct relations. Here we combine an unsupervised object-centric encoding mechanism, Slot Attention (Locatello et al., 2020) with Abstractors, enabling abstract visual reasoning to be scaled to problems involving a large number of objects and multiple relations.

Though our proposed model employed previously designed components (slot attention and relational cross-attention), combining these components into an effective architecture necessitated many novel design choices. We have proposed a factorized slot representation, in which the position embeddings are used for the initial relational representations $\boldsymbol{s}^{l=0}$, whereas the feature embeddings are used to compute the keys and queries for the relational cross attention. We also find that relational cross attention was most effective when it was interleaved with self attention and feedforward layers in a particular manner. In Section 5.1, we show through ablation experiments targeted toward the use of the factorized slot representation (Row 4 of Table 6), and the

interleaved relational-cross-attention/self-attention (Row 3 of Table 6), that these design choices are essential for obtaining strong performance. Thus, the novel contribution of our work is to identify an effective way for combining object-centric and relational components, yielding an architecture that achieves state-of-the-art performance on challenging abstract reasoning tasks.

4. Experiments

4.1. Datasets

We evaluated the Slot Abstractor on five challenging abstract visual reasoning datasets, ART (Webb et al., 2021), SVRT (Fleuret et al., 2011), CLEVR-ART (Webb et al., 2023b), PGM (Barrett et al., 2018), and V-PROM (Teney et al., 2020). Problems in ART and SVRT datasets consist of simple 2D shapes, whereas CLEVR-ART consists of more complicated and realistic 3D shapes. Each problem of ART, SVRT, and CLEVR-ART contains a small number of objects and is governed by a single rule among them. Problems from the PGM dataset contain a much larger number of objects, with multiple rules among them. V-PROM is a matrix reasoning task (similar to PGM) involving real-world images. The datasets are described in more detail below.

4.1.1. ART

The Abstract Reasoning Tasks (ART) dataset was proposed by Webb et al. (2021). It consists of four visual reasoning tasks ('same/different', 'relational-match-to-sample', 'distribution-of-3', 'identity rules'), each defined by a different abstract rule (Figure 3). This dataset was created using 100 unicode character objects, and generalization regimes of varying difficulty were created defined by the number of unique objects used to instantiate the rules during training. We focused on the most difficult generalization regime, in which the training set consists of problems that are created using 5 out of the 100 possible objects, and the test problems are created using the remaining 95 objects. This is a difficult test of systematic generalization, as it requires learning of an abstract rule from a very small set of examples (Table 7), with little perceptual overlap between training and test sets. Consistent with previous work (Webb et al., 2023b) we investigated a version of the tasks with multi-object displays and random spatial jitter (random translation of up to 5 pixels in any direction) applied to each object (which has previously been shown to increase task difficulty for some relational architectures (Vaishnav & Serre, 2023)).

4.1.2. SVRT

The Synthetic Visual Reasoning Test (SVRT) dataset (Fleuret et al., 2011) consists of 23 binary classification tasks. Each task consists of synthetic 2D shapes, with an underlying abstract pattern among the shapes, and can be broadly grouped into two categories: those that are defined by same/different relations (Figure 4(a)), and those that are defined by spatial relations (Figure 4(b)). We used a small training set of 500 or 1000 examples per task, consistent with previous work (Vaishnav & Serre, 2023).

4.1.3. CLEVR-ART

This dataset (Figure 5) was proposed by Webb et al. (2023b), and created using photorealistic synthetic 3D shapes from CLEVR (Johnson et al., 2017). It consists of two visual reasoning tasks from ART: relational-match-to-sample and identity rules. The training set consists of images created using small and medium-sized rubber cubes in four colors (cyan, brown, green, and gray). The test set consists of images created using large-sized metal spheres and cylinders in four other colors (yellow, purple, blue, and red). The features of objects in the training and test set are completely different, which tests the systematic generalization of learned abstract rules to previously unseen object features.

4.1.4. PGM

The Procedurally Generated Matrices (PGM) dataset was proposed by Barrett et al. (2018), and is also based on Raven's Progressive Matrices problem sets (Raven, 1938). Each problem consists of a 3×3 matrix of image panels populated with objects of varying shape, size, and color (Figure 6). The task is to identify the abstract pattern among the image panels in the first two rows and/or columns of the matrix, and use that pattern to fill the panel in the third row and column from a set of eight choices. The maximum number of objects possible in an image panel is 16, with 9 panels per problem, yielding a total maximum of 144 objects per problem (compared to a maximum of 6 objects per problem in the other three datasets). Each matrix problem in PGM is defined by the abstract structure $\mathcal{S} = \{[r, o, a] : r \in \mathcal{R}, o \in \mathcal{O}, a \in \mathcal{A}\}$, where $\mathcal{R} = \{$progression, XOR, AND, OR, consistent union$\}$ are the set of rules, $\mathcal{O} = \{$shape, line$\}$ are the set of objects, and $\mathcal{A} = \{$size, type, position, color, number$\}$ are the set of attributes. Problems consist of multiple rules (up to 4) governed by triples $[r, o, a]$.

The PGM specifies eight different generalization regimes of varying difficulty. Each regime consists of 1.2M training problems, 20K validation problems, and 200K testing problems. The neutral regime is the easiest generalization regime, with training and test sets sampled from the same distribution. The other seven regimes — interpolation (Intp.), extrapolation (Extp.), heldout attribute shape color (H.O.S-C), heldout attribute line type (H.O.L-T), heldout triples (H.O.Triples), heldout triple pairs (H.O.T.P.), heldout arribute pairs (H.O.A.P.) — test out-of-distribution (OOD) generalization (see Appendix A.4.1 for more details).

4.1.5. V-PROM

The Visual Progressive Matrices (V-PROM) dataset was proposed by Teney et al. (2020). V-PROM is a matrix reasoning dataset (similar to PGM), but unlike standard matrix reasoning tasks, the problems in V-PROM are constructed from real-world images (Figure 7). The maximum number of objects possible in an image is 10, and since there are 9 panels per problem, the total maximum number of objects is 90. Like PGM, each problem consists of multiple abstract rules and the dataset consists of generalization regimes of varying difficulty. In this work, we focused only on the Neutral regime, with around 139K training problems, 8K validation problems, and 73K test problems.

4.2. Baselines

For the ART, SVRT, and CLEVR-ART datasets, we compared the Slot Abstractor to baseline methods reported in Webb et al. (2023b), which includes the OCRA model, the GAMR architecture proposed by Vaishnav and Serre (2023), ResNet-50 (He et al., 2016), a version of ResNet that uses self-attention (Attn-ResNet) (Vaishnav et al., 2022), and a set of baselines that combined pre-trained slot attention with various reasoning architectures (GAMR, Transformer, ESBN (Webb et al., 2021), CorelNet (Kerg et al., 2022), Relation Net (RN) (Santoro et al., 2017), and Interaction Net (IN) (Watters et al., 2017)).

For the PGM dataset, we compared the Slot Abstractor to the PredRNet model (Yang et al., 2023) and several state-of-the-art methods reported in Zhang et al. (2022), including their proposed ARII model. Since our main focus was on systematic generalization, for comparison we only included models that were evaluated on at least one OOD regime. For the V-PROM dataset, we compared the Slot Abstractor to the best performing model (RN) reported in Teney et al. (2020). We didn't use any auxiliary information (i.e., rule labels), and hence for a fair comparison we only compared to baselines that were trained without auxiliary loss.

4.3. Experimental Details

We pre-trained slot attention using a reconstruction (i.e., autoencoding) objective. For ART, SVRT, and CLEVR-ART datasets we used the pre-trained model from Webb et al. (2023b)[1]. For PGM and V-PROM, we pre-trained slot attention on the neutral regime training set. Table 8 and Table 9 describe the hyperparameters for the convolutional encoder and the slot-based spatial broadcast decoder respectively, whereas the pre-training hyperparameters are described in Table 12 for the PGM dataset. For the V-PROM dataset, Table 10 describes the hyperparameters for the convolutional encoder used on top of the pre-trained ViT [2] features, Table 11 describes the hyperparameters for the decoder, and Table 13 describes the pre-training hyperparameters. After pre-training, we selected the model from the epoch with the lowest mean squared error on the validation set. The slot attention parameters were frozen after pre-training, except for the V-PROM dataset where they were also finetuned on the reasoning task.

We resized the images to 128×128 for ART, SVRT, and CLEVR-ART, and resized the image panels (of which there are 9 per problem) to 80×80 for PGM, and 224×224 for V-PROM. Pixels were normalized to the range $[0, 1]$ for ART, and to $[-1, 1]$ for SVRT, CLEVR-ART, and PGM datasets. For V-PROM, we applied a channel-wise mean and standard deviation normalization with values of $[0.485, 0.456, 0.406]$ for the mean, and $[0.229, 0.224, 0.225]$ for the standard deviation corresponding to the RGB channels, after converting the pixel values to the range $[0, 1]$. For SVRT, we also applied random horizontal and vertical flips, consistent with prior work (Vaishnav & Serre, 2023; Webb et al., 2023b).

The hyperparameters for the Abstractor module of the Slot Abstractor are described in Table 14. We initialized $\boldsymbol{W_q}$ and $\boldsymbol{W_k}$ (the linear projection matrices used to generate queries and keys) with the same values, but allowed them to vary during training. To generate the output, we took the mean of the K final relational representations $\boldsymbol{s}^L$, and passed this through a task-specific output layer. For the same/different ART task and SVRT tasks, this layer had a single unit and a sigmoid activation, and the model was trained with binary cross entropy loss. For the other ART tasks, PGM, and the V-PROM datasets, each candidate answer choice was inserted into the problem before being fed as input to the model, and the output was a linear layer with a single unit. This output was treated as a score for the answer choice in each input. A softmax activation was then applied over the scores for all answer choices, and the model was trained with cross-entropy loss. More experimental details can be found at Appendix A.2. The code is available at https://github.com/Shanka123/Slot-Abstractor.

5. Results

Table 1 shows the results on the four ART tasks. The Slot Abstractor achieved state-of-the-art accuracy on three tasks (same/different, relational-match-to-sample, and distribution-of-3), beating the next best baseline (OCRA) by as much as 9%. It also demonstrated an average improvement of 3% over all tasks compared to previous state-of-the-art model (OCRA).

[1] https://github.com/Shanka123/OCRA

[2] we used the features of ViT (token dimensionality 768, 12 heads, patch size 16) pre-trained using DINO, and provided by the timm library (Wightman et al., 2019), with the model name *vit_base_patch16_224.dino*.

Table 1. Results on the four tasks of the ART dataset. Results reflect test accuracy averaged over 10 trained networks(± standard error).

MODEL	SAME/DIFFERENT	RELATIONAL-MATCH-TO-SAMPLE	DISTRIBUTION-OF-3	IDENTITY RULES
RESNET	66.60±1.5	49.89±0.2	50.07±1.3	54.84±2.4
SLOT-CORELNET	50.50±0.2	49.82±0.2	26.80±0.8	43.50±5.2
SLOT-ESBN	50.02±0.2	49.99±0.2	25.56±0.1	50.33±2.8
SLOT-GAMR	62.98±1.4	59.55±2.7	32.77±1.0	61.92±0.9
SLOT-RN	77.26±1.9	61.62±1.1	52.10±0.7	65.96±1.1
SLOT-IN	59.23±2.3	56.93±0.8	49.48±1.8	72.82±1.6
SLOT-TRANSFORMER	68.46± 2.0	73.99± 3.0	60.61± 1.9	78.32± 1.8
GAMR	83.49± 1.4	72.20± 3.0	68.62± 1.8	66.23± 4.8
OCRA	87.95± 1.3	85.31± 2.0	86.42± 1.3	92.8± 0.3
SLOT-ABSTRACTOR	**96.36± 0.4**	**91.64± 1.6**	**95.22± 0.4**	**96.41±0.4**

Table 2. Results on the two task categories of the SVRT dataset. Results reflect test accuracy averaged over different tasks from each category (± standard error), for 1 trained network for each task.

MODEL	SAME/DIFFERENT		SPATIAL RELATIONS	
	DATASET SIZE =0.5K	DATASET SIZE =1K	DATASET SIZE =0.5K	DATASET SIZE = 1K
RESNET	54.97±2.2	56.88±2.5	85.18±4.3	94.87±1.6
ATTN-RESNET	62.30±3.5	68.83±4.4	94.80±1.4	97.66±0.7
SLOT-CORELNET	52.95±1.4	57.13±2.6	60.95±3.7	74.59±3.7
SLOT-ESBN	53.83±1.1	51.67±1.1	61.30±2.3	62.69±2.4
SLOT-GAMR	63.06±3.7	66.87±3.2	84.90±2.4	86.99±2.2
SLOT-RN	71.48±4.8	81.79±4.4	91.73±1.8	96.20±1.4
SLOT-IN	68.23±4.8	74.99±4.9	90.23±2.0	94.86±1.4
SLOT-TRANSFORMER	**76.54±5.1**	**89.85±4.2**	94.06±1.6	**97.86±0.9**
GAMR	**76.80±4.9**	82.05±4.4	**97.40±0.7**	**98.74±0.3**
OCRA	**79.89±4.5**	**90.30±4.1**	89.25±2.5	95.02±2.4
SLOT-ABSTRACTOR	**82.20±4.7**	**91.86±4.0**	91.74±2.2	97.26±1.1

Table 3. Results on the two tasks (relational-match-to-sample (RMTS) and identity rules (ID)) of the CLEVR-ART dataset. Results reflect test accuracy averaged over 5 trained networks (± standard error).

MODEL	RMTS	ID
SLOT-CORELNET	49.87±0.2	24.80±0.3
SLOT-ESBN	62.53±0.1	28.87±0.7
SLOT-GAMR	52.56±0.5	39.83±0.9
SLOT-RN	64.79±0.5	60.27±0.6
SLOT-IN	66.72±3.7	67.22±1.7
SLOT-TRANSFORMER	87.54±0.7	78.81±1.6
GAMR	70.40±5.8	74.15±4.0
OCRA	93.34±1.0	77.06±0.7
SLOT-ABSTRACTOR	**96.34±0.5**	**91.61±0.2**

Table 2 shows the results on the two task categories for the SVRT dataset. The Slot Abstractor displayed comparable overall performance with previous state-of-the-art models (GAMR, OCRA), demonstrating a marginal improvement on tasks defined by same/different relations, whereas GAMR performed slightly better on tasks defined by spatial relations.

Table 3 shows the results on the two CLEVR-ART tasks. The Slot Abstractor achieved state-of-the-art accuracy beating previous state-of-the-art model (OCRA) by as much as 14%. This demonstrates that the Slot Abstractor's capacity for systematic generalization can also be extended to abstract reasoning problems with more complicated and realistic visual inputs.

Table 4 shows the results on the PGM generalization regimes. The Slot Abstractor demonstrated notable improvements on many of the OOD generalization regimes (Intp., H.O.T.P., Extp.) with as much as 21% compared to the previous state-of-the-art model (PredRNet), and an average improvement of around 5% over all the regimes. It is also worth noting that OCRA, previously the state-of-the-art model for ART, SVRT, and CLEVR-ART datasets, could not be trained on the PGM dataset due to memory issues. It was not possible to train the model on an A100 GPU with 80GB memory, even when using a batch size of 1 (given 144 slots, OCRA would need to process $144^2 \approx 20$k relational embeddings). This highlights the benefits of the Slot Abstractor's improved complexity ($O(N^2)$ vs. $O(N^4)$ in OCRA), allowing it to be effectively applied to problems involving a much larger number of objects.

Table 4. Results on different generalization regimes of the PGM dataset. Results reflect test accuracy for 1 trained model for each regime.

MODEL	NEUTRAL	INTP.	H.O.A.P.	H.O.T.P.	H.O.TRIPLES	H.O.L-T	H.O.S-C	EXTP.
WREN $\beta = 0$	62.6	64.4	27.2	41.9	19.0	14.4	12.5	17.2
VAE-WREN	64.2	-	36.8	43.6	24.6	-	-	-
MXGNET $\beta = 0$	66.7	65.4	33.6	43.3	19.9	16.7	16.6	18.9
DCNET	68.6	59.7	-	-	-	-	-	17.8
REL-BASE	85.5	-	-	-	-	-	-	22.1
MRNET	93.4	68.1	38.4	55.3	25.9	**30.1**	**16.9**	19.2
MLRN	98.0	57.8	-	-	-	-	-	14.9
STSN	**98.2**	78.5	-	-	-	-	-	20.4
ARII	88.0	72.0	50.0	64.1	**32.1**	16.0	12.7	29.0
PREDRNET	97.4	70.5	**63.4**	67.8	23.4	27.3	13.1	19.7
SLOT-ABSTRACTOR	91.5	**91.6**	**63.3**	**78.3**	20.4	16.7	14.3	**39.3**

Table 5. Results on the PGM Neutral regime when trained on a subset (20% vs. 100%) of the training data. Results reflect test accuracy for 1 trained model.

MODEL	% TRAINING DATA	
	20%	100%
STSN	53.35	**98.2**
SLOT-ABSTRACTOR	**71.25**	91.5

On V-PROM, the Slot Abstractor displayed state-of-the-art performance achieving 67.7% test accuracy, compared to 51.2% by the previous best model, Relation Net. This demonstrates that the Slot Abstractor's systematic generalization capacity can also be extended to abstract visual reasoning problems involving real-world images.

To evaluate the Slot Abstractor's sample efficiency, we trained it on 20% of the training data in the PGM neutral regime. We also compared it to the state-of-the-art model in the neutral regime, STSN (Mondal et al., 2023), using the publicly available code [3]. As done in the original implementation, slot attention parameters were initialized using the same pre-trained slot attention as used in the Slot Abstractor, but then fine-tuned while training on the downstream task. Table 5 shows the results. When trained on only 20% of the training data, the Slot Abstractor significantly outperforms STSN, demonstrating the Slot Abstractor's superior sample efficiency.

5.1. Ablation Study

To understand the importance of each of the Slot Abstractor's major components, we performed an ablation study using the ART dataset (Table 6). First, we removed Slot Attention, instead dividing the feature map $\boldsymbol{feat}$ into a 4×4 grid (treating each location within the feature map as an 'object'), and trained the model end to end. We noticed a significant drop in performance for the relational-match-to-sample and same/different tasks. This demonstrates the importance of using object-centric representations (even when pre-trained DINO ViT features are used, as described in A.3). Second, we removed different components from the Abstractor module. In one case, we removed the self-attention and the subsequent feedforward layer. This impaired performance on the relational-match-to-sample, distribution-of-3, and identity rules tasks, where the ablation of self-attention resulted in 4-7% drop in test accuracy. This demonstrates the importance of modeling higher-order relations through self-attention. In another case, we ablated the factorized slot representations, using the standard slot representations instead of separate feature and position embeddings. The relation embeddings $\boldsymbol{s}^{l=0}$ were initialized using learned parameters. We observed a significant drop in performance (as much as 40%) for all tasks except identity rules. This demonstrates the importance of using factorized feature and position embeddings, which allow the Slot Abstractor to keep track of the correspondence between relations and objects. Finally, we replaced the relational cross-attention (RCA) with standard cross-attention (CA), where keys and values are formed from feature embeddings, and queries are formed from position embeddings, thereby removing the relational bottleneck. This significantly impaired performance for all tasks (by as much as 52%). This reflects the centrally important role of the relational bottleneck as an inductive bias underlying the Slot Abstractor's capacity for generalization.

The removal of slot attention (Row 2 of Table 6) and the replacement of relational cross-attention with standard cross-attention (Row 5 of Table 6), resulted in a significant drop in performance, which demonstrates the importance of combining slot attention with relational cross-attention in Abstractors. The removal of self-attention and the subsequent feedforward layer after relational cross-attention (Row 3 of Table 6) and the use of standard slot representations without factorizing into position and feature specific embeddings

[3] https://github.com/Shanka123/STSN

Table 6. Ablation study on tasks of the ART dataset. Results reflect test accuracy averaged over 10 trained networks(± standard error).

MODEL	SAME/DIFFERENT	RELATIONAL-MATCH-TO-SAMPLE	DISTRIBUTION-OF-3	IDENTITY RULES
SLOT-ABSTRACTOR	**96.36± 0.4**	**91.64± 1.6**	**95.22± 0.4**	**96.41±0.4**
W/O SLOT ATTENTION	89.29±0.8	49.85±0.2	89.10±1.2	86.09±1.3
W/O SELF ATTENTION	**95.26±0.9**	85.41±2.0	93.83±0.6	92.58±0.7
W/O FACTORIZED REPS.	77.24±2.8	50.02±0.1	64.4±1.5	85.08±1.5
REPLACE RCA WITH CA	80.04±1.4	58.24±1.5	43.90±1.3	60.02±1.0

(Row 4 of Table 6) explored different design choices of combining slot attention with the relational cross-attention in Abstractors, both of which significantly underperformed compared to the proposed method, thereby demonstrating that these design choices are essential for obtaining strong performance.

5.2. Visualization of Abstractor's Relation Embeddings

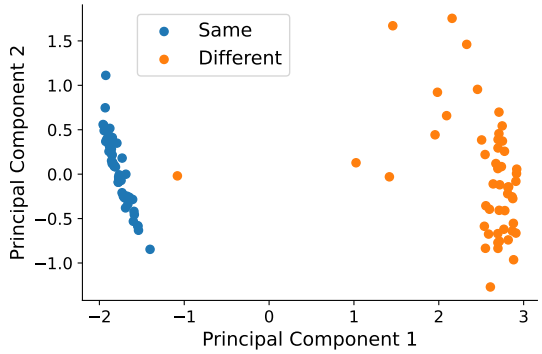


Figure 2. **Visualization of Abstractor's Relation Embeddings.** The output of the first head of relational cross attention, after projecting to the first two principal components for 100 examples from the test set of same/different ART task. Two different clusters are formed corresponding to problems with the same and different relation among the objects.

To demonstrate that the Abstractor can capture the different relations present in the tasks, we performed a qualitative analysis of the relational cross-attention head. Specifically, we visualized the output of the relational cross-attention corresponding to the first head ($\tilde{s}^1$ in Equation (5)) for 100 examples from the test set of same/different ART task, with equal number of examples belonging to same and different category. We applied principal component analysis (PCA) to the output features after taking the mean over the K slots, to reduce it to 2 dimensions. Figure 2 shows that the two-dimensional output form two different clusters corresponding to images belonging to the same and different categories, indicating that the first relational cross-attention head was able to capture the same/different relation.

6. Conclusion and Future Directions

In this work, we developed Slot Abstractors, by combining slot-based object-centric encoding mechanisms with Abstractors, a recently proposed approach for implementing the relational bottleneck using a relational variant of cross-attention. We evaluated the Slot Abstractor on five abstract visual reasoning tasks including one involving real-world images, finding that it achieves overall state-of-the-art systematic generalization of learned abstract rules, and can also be scaled to problems containing a large number of objects and multiple rules.

Finally, we discuss some limitations of our work and potentially interesting directions for future work. First, although Slot Abstractors demonstrated state-of-the-art performance in many of the OOD generalization regimes of PGM (and achieved state-of-the-art average performance across all regimes), there is still significant room for improvement in some of these regimes. Even for some regimes on which the Slot Abstractor achieved state-of-the-art performance, that performance was sometimes well below what would be expected of human reasoners (e.g., especially for the 'Extrapolation' regime). Second, the number of slots in slot-based models is fixed, which may pose a challenge for settings in which there are large variations in the number of objects that must be processed. Recent work (Löwe et al., 2023; Stanić et al., 2023) has built methods addressing this limitation by doing away with slots entirely, and future work might explore how such non-slot-based methods can be integrated with relational inductive biases. Finally, future work could also improve the quadratic complexity of Slot Abstractors by using more efficient attention mechanisms (Dao et al., 2022; Dao, 2023).

Acknowledgements

Shanka Subhra Mondal was supported by Office of Naval Research grant N00014-22-1-2002 during the duration of this work. We would like to thank the reviewers for their valuable feedback, and the Princeton Research Computing, especially William G. Wischer and Josko Plazonic, for their help with scheduling training jobs on the Princeton University Della cluster.

Impact Statement

This paper presents work the goal of which is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Altabaa, A. and Lafferty, J. Approximation of relation functions and attention mechanisms. *arXiv preprint arXiv:2402.08856*, 2024.

Altabaa, A., Webb, T., Cohen, J., and Lafferty, J. Abstractors: Transformer modules for symbolic message passing and relational reasoning. *arXiv preprint arXiv:2304.00195*, 2023.

Barrett, D., Hill, F., Santoro, A., Morcos, A., and Lillicrap, T. Measuring abstract reasoning in neural networks. In *International conference on machine learning*, pp. 511–520. PMLR, 2018.

Benny, Y., Pekar, N., and Wolf, L. Scale-localized abstract reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12557–12565, 2021.

Burgess, C. P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M., and Lerchner, A. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Dao, T. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

Dao, T., Fu, D., Ermon, S., Rudra, A., and Ré, C. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

Ding, D., Hill, F., Santoro, A., Reynolds, M., and Botvinick, M. Attention over learned object embeddings enables complex visual reasoning. *Advances in neural information processing systems*, 34:9112–9124, 2021.

Dittadi, A., Papa, S., De Vita, M., Schölkopf, B., Winther, O., and Locatello, F. Generalization and robustness implications in object-centric learning. *arXiv preprint arXiv:2107.00637*, 2021.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Engelcke, M., Parker Jones, O., and Posner, I. Genesis-v2: Inferring unordered object representations without iterative refinement. *Advances in Neural Information Processing Systems*, 34:8085–8094, 2021.

Fleuret, F., Li, T., Dubout, C., Wampler, E. K., Yantis, S., and Geman, D. Comparing machines and humans on a visual categorization test. *Proceedings of the National Academy of Sciences*, 108(43):17621–17625, 2011.

Greff, K., Kaufman, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pp. 2424–2433. PMLR, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hu, S., Ma, Y., Liu, X., Wei, Y., and Bai, S. Stratified rule-aware network for abstract visual reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 1567–1574, 2021.

Jahrens, M. and Martinetz, T. Solving raven's progressive matrices with multi-layer relation networks. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–6. IEEE, 2020.

Jiang, J., Deng, F., Singh, G., and Ahn, S. Object-centric slot diffusion. *arXiv preprint arXiv:2303.10834*, 2023.

Johnson, J., Hariharan, B., Van Der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., and Girshick, R. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2901–2910, 2017.

Kerg, G., Mittal, S., Rolnick, D., Bengio, Y., Richards, B., and Lajoie, G. On neural architecture inductive biases for relational tasks. *arXiv preprint arXiv:2206.05056*, 2022.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kotovsky, L. and Gentner, D. Comparison and categorization in the development of relational similarity. *Child Development*, 67(6):2797–2822, 1996.

Lake, B. and Baroni, M. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International conference on machine learning*, pp. 2873–2882. PMLR, 2018.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention. *Advances in Neural Information Processing Systems*, 33:11525–11538, 2020.

Löwe, S., Lippe, P., Locatello, F., and Welling, M. Rotating features for object discovery. *arXiv preprint arXiv:2306.00600*, 2023.

Mondal, S. S., Webb, T. W., and Cohen, J. D. Learning to reason over visual objects. In *11th International Conference on Learning Representations, ICLR*, 2023.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. 2017.

Raven, J. C. *Progressive matrices: A perceptual test of intelligence, individual form*. London: Lewis, 1938.

Ricci, M., Kim, J., and Serre, T. Same-different problems strain convolutional neural networks. *arXiv preprint arXiv:1802.03390*, 2018.

Santoro, A., Raposo, D., Barrett, D. G., Malinowski, M., Pascanu, R., Battaglia, P., and Lillicrap, T. A simple neural network module for relational reasoning. *Advances in neural information processing systems*, 30, 2017.

Seitzer, M., Horn, M., Zadaianchuk, A., Zietlow, D., Xiao, T., Simon-Gabriel, C.-J., He, T., Zhang, Z., Schölkopf, B., Brox, T., et al. Bridging the gap to real-world object-centric learning. *arXiv preprint arXiv:2209.14860*, 2022.

Shanahan, M., Nikiforou, K., Creswell, A., Kaplanis, C., Barrett, D., and Garnelo, M. An explicitly relational neural network architecture. In *International Conference on Machine Learning*, pp. 8593–8603. PMLR, 2020.

Spratley, S., Ehinger, K., and Miller, T. A closer look at generalisation in raven. In *European Conference on Computer Vision*, pp. 601–616. Springer, 2020.

Stanić, A., Gopalakrishnan, A., Irie, K., and Schmidhuber, J. Contrastive training of complex-valued autoencoders for object discovery. *arXiv preprint arXiv:2305.15001*, 2023.

Steenbrugge, X., Leroux, S., Verbelen, T., and Dhoedt, B. Improving generalization for abstract reasoning tasks using disentangled feature representations. *arXiv preprint arXiv:1811.04784*, 2018.

Teney, D., Wang, P., Cao, J., Liu, L., Shen, C., and van den Hengel, A. V-prom: A benchmark for visual reasoning using visual progressive matrices. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 12071–12078, 2020.

Vaishnav, M. and Serre, T. Gamr: A guided attention model for (visual) reasoning. In *11th International Conference on Learning Representations, ICLR*, 2023.

Vaishnav, M., Cadene, R., Alamia, A., Linsley, D., VanRullen, R., and Serre, T. Understanding the computational demands underlying visual reasoning. *Neural Computation*, 34(5):1075–1099, 2022.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, D., Jamnik, M., and Lio, P. Abstract diagrammatic reasoning with multiplex graph networks. *arXiv preprint arXiv:2006.11197*, 2020.

Watters, N., Zoran, D., Weber, T., Battaglia, P., Pascanu, R., and Tacchetti, A. Visual interaction networks: Learning a physics simulator from video. *Advances in neural information processing systems*, 30, 2017.

Watters, N., Matthey, L., Burgess, C. P., and Lerchner, A. Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes. *arXiv preprint arXiv:1901.07017*, 2019.

Webb, T., Dulberg, Z., Frankland, S., Petrov, A., O'Reilly, R., and Cohen, J. Learning representations that support extrapolation. In *International conference on machine learning*, pp. 10136–10146. PMLR, 2020.

Webb, T. W., Sinha, I., and Cohen, J. D. Emergent symbols through binding in external memory. In *9th International Conference on Learning Representations, ICLR*, 2021.

Webb, T. W., Frankland, S. M., Altabaa, A., Krishnamurthy, K., Campbell, D., Russin, J., O'Reilly, R., Lafferty, J., and Cohen, J. D. The relational bottleneck as an inductive bias for efficient abstraction. *arXiv preprint arXiv:2309.06629*, 2023a.

Webb, T. W., Mondal, S. S., and Cohen, J. D. Systematic visual reasoning through object-centric relational abstraction. *arXiv preprint arXiv:2306.02500*, 2023b.

Wightman, R. et al. Pytorch image models, 2019. URL https://github.com/huggingface/pytorch-image-models.

Wu, Y., Dong, H., Grosse, R., and Ba, J. The scattering compositional learner: Discovering objects, attributes, relationships in analogical reasoning. *arXiv preprint arXiv:2007.04212*, 2020.

Wu, Z., Dvornik, N., Greff, K., Kipf, T., and Garg, A. Slotformer: Unsupervised visual dynamics simulation with object-centric models. *arXiv preprint arXiv:2210.05861*, 2022.

Yang, L., You, H., Zhen, Z., Wang, D., Wan, X., Xie, X., and Zhang, R.-Y. Neural prediction errors enable analogical visual reasoning in human standard intelligence tests. In *International Conference on Machine Learning*, pp. 39572–39583. PMLR, 2023.

Zhang, C., Gao, F., Jia, B., Zhu, Y., and Zhu, S.-C. Raven: A dataset for relational and analogical visual reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5317–5327, 2019a.

Zhang, C., Jia, B., Gao, F., Zhu, Y., Lu, H., and Zhu, S.-C. Learning perceptual inference by contrasting. *Advances in neural information processing systems*, 32, 2019b.

Zhang, W., Mo, S., Liu, X., Song, S., et al. Learning robust rule representations for abstract reasoning via internal inferences. *Advances in Neural Information Processing Systems*, 35:33550–33562, 2022.

Zheng, K., Zha, Z.-J., and Wei, W. Abstract reasoning with distracting features. *Advances in Neural Information Processing Systems*, 32, 2019.

Zhuo, T. and Kankanhalli, M. Effective abstract reasoning with dual-contrast network. *arXiv preprint arXiv:2205.13720*, 2022.

A. Appendix

A.1. Algorithm for Slot Abstractors

Algorithm 1 Slot Abstractors: The inputs are the feature map obtained by passing an image $\boldsymbol{x}$ through a convolutional encoder, a position code (encoding the four cardinal directions) which is passed through a linear layer to generate positional embeddings, and K slots which are initialized from shared mean and variance parameters.

Inputs: feature map $\boldsymbol{feat} \in \mathbb{R}^{N \times D}$; positional embeddings $\boldsymbol{pos} \in \mathbb{R}^{N \times D}$; $\boldsymbol{slots} \sim Normal(\mu, \sigma) \in \mathbb{R}^{K \times D}, \mu \in \mathbb{R}^{D}, \sigma \in \mathbb{R}^{D}$

Parameters: q, k, v linear projection matrices for slot attention, relational cross-attention, self-attention; GRU; FeedForward; slot attention iterations T; attention heads H; Abstractor layers L

$\boldsymbol{inputs} =$ FeedForward($\boldsymbol{feat} + \boldsymbol{pos}$)

for $t = 1$ **to** T **do**

  $\boldsymbol{attn}_t = \mathrm{softmax}(\frac{1}{\sqrt{D}}\,\mathrm{k}(\boldsymbol{inputs}) \cdot \mathrm{q}(\boldsymbol{slots})^{\top}, axis = \text{`}slots\text{'})$

  $\boldsymbol{updates} = \boldsymbol{attn}_t \cdot \mathrm{v}(\boldsymbol{inputs})$

  $\boldsymbol{slots}$ = GRU(inputs = $\boldsymbol{updates}$, hidden state = $\boldsymbol{slots}$)

  $\boldsymbol{slots} =$ FeedForward($\boldsymbol{slots}$) $+ \boldsymbol{slots}$

end for

$\boldsymbol{z}_k = \boldsymbol{attn}_T\, \mathrm{flatten}(\boldsymbol{feat})[k]$

$\boldsymbol{m}_k = \boldsymbol{attn}_T\, \mathrm{flatten}(\boldsymbol{pos})[k]$

$\boldsymbol{z} = \{\boldsymbol{z}_1, ...\boldsymbol{z}_K\}$

$\boldsymbol{m} = \{\boldsymbol{m}_1, ...\boldsymbol{m}_K\}$

Initialize $\boldsymbol{s}^0 = \boldsymbol{m}$

for $l = 1$ **to** L **do**

  $\boldsymbol{s}^l$ = RelationalCrossAttention($\boldsymbol{z}$, $\boldsymbol{s}^{l-1}$) + $\boldsymbol{s}^{l-1}$

  $\boldsymbol{s}^l$ = FeedForward($\boldsymbol{s}^l$) + $\boldsymbol{s}^l$

  $\boldsymbol{s}^l$ = SelfAttention($\boldsymbol{s}^l$) + $\boldsymbol{s}^l$

  $\boldsymbol{s}^l$ = FeedForward($\boldsymbol{s}^l$) + $\boldsymbol{s}^l$

end for

Output: $\boldsymbol{s}^L$

A.2. More Experimental Details

For ART and CLEVR-ART tasks, we used a learning rate of $8e-5$, batch size of 16, and the number of training epochs are described in Table 15. For SVRT, we used a learning rate of $4e-5$, a batch size of 32, and trained for 2000 epochs on each task. We used a single A100 GPU with 80GB memory for training on ART, SVRT, and CLEVR-ART tasks. For training on PGM regimes, we used 4 A100 GPUs with 80GB memory each, a learning rate of $8e-5$, and a batch size of 24 per GPU. The number of training epochs for each PGM generalization regime is displayed in Table 16. For training on the V-PROM neutral regime, we also used 4 A100 GPUs with 80GB memory each, a learning rate of $8e-5$, and a batch size of 12 per GPU, and trained for 30 epochs. For optimization, we used the ADAM (Kingma & Ba, 2014) optimizer, and for implementation, we used the PyTorch library (Paszke et al., 2017).

A.3. Importance of Slot Attention in DINOSAUR

To justify the importance of the slot attention component, when the DINOSAUR model is used with the Abstractor, we conducted an ablation experiment on the V-PROM neutral regime, by removing the slot attention component from the DINOSAUR model, and used the pre-trained DINO ViT features with the Abstractor. Since, it wasn't possible to train the ablation model using the patchwise DINO ViT features on an A100 GPU of 80GB memory even with a batch size of 1 (given 9 image panels and 196 patches for each panel, the size of the matrix which models relations between the features would be 1764×1764), and hence we used the CLS output of the DINO ViT features, as feature embeddings $\boldsymbol{z}$ to Abstractor. On V-PROM, the ablation model achieved a test accuracy of 30.01%, compared to 67.7% by our proposed method. This result suggests the importance of slot attention as an effective way of finding a low-dimensional representation, with slots bound to objects in the image.

A.4. Datasets

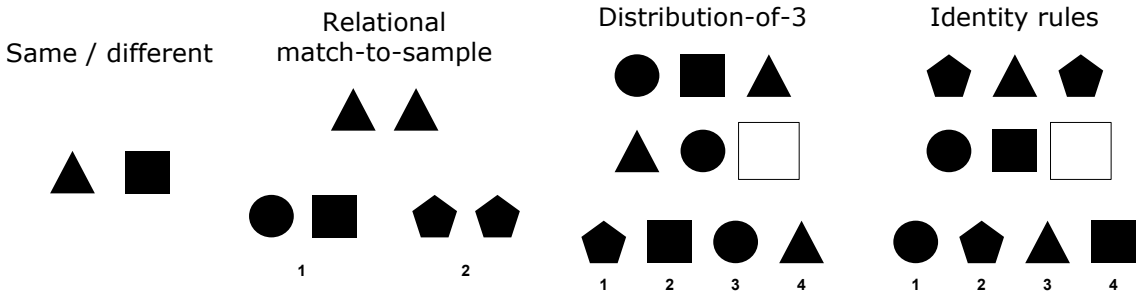
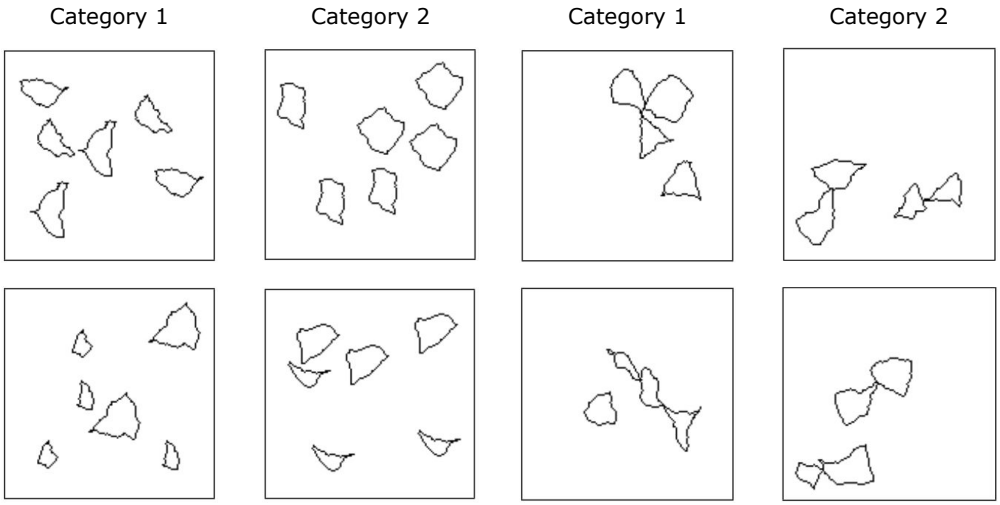


Figure 3. **Abstract Reasoning Tasks (ART) Dataset**. The 'same/different' task, requires identifying whether two objects are the same or different. The 'relational-match-to-sample' task requires selecting a pair of objects, out of two pairs called the target objects that has the same relation, as the relation ('same' or 'different') among a source pair of objects. We presented the problem as a 2×2 array format, with the source pair of objects in the top row, and a target pair in the bottom row (separate images for each target pair). In the 'distribution-of-3' task, the first row contains a set of three objects, and the second row contains an incomplete set. The task is to select the missing object from a set of four choices. We presented the problem as a 2×3 array format, with one of the choices inserted in the bottom right cell (separate images for each choice). In the 'identity rules' task, the first row contains three objects that follow an abstract pattern (ABA, ABB, or AAA), and the task is to select the choice that would result in the same relation being instantiated in the second row. We presented the problems in the same format as the 'distribution of-3' task.

Table 7. Number of training examples for ART tasks.

SAME/DIFFERENT	RELATIONAL-MATCH-TO-SAMPLE	DISTRIBUTION-OF-3	IDENTITY RULES
40	480	360	8640



(a) Examples of same/different task. (b) Examples of spatial relation task.

Figure 4. **Synthetic Visual Reasoning Test (SVRT) Dataset. (a)** Examples of task depicting same/different relation. Each row shows an example from each of the two categories. In category 1 there are three sets of two identical shapes. In category 2 there are two sets of three identical shapes. **(b)** Examples of task depicting spatial relation. Each row shows an example from each of the two categories. In category 1 three out of four shapes are touching each other. In category 2 there are two sets of two shapes touching each other.

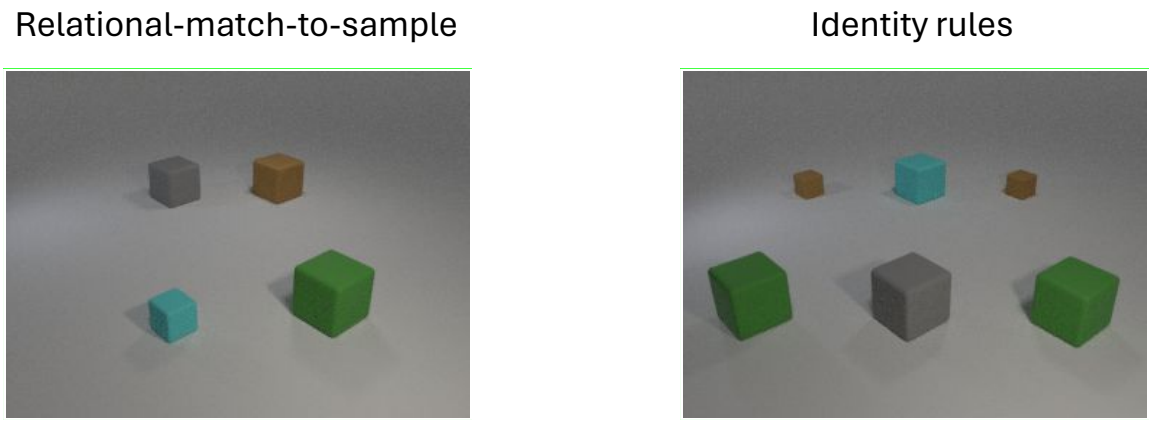


Figure 5. **CLEVR-ART Dataset**. **Relational-match-to-sample:** An example depicting 'different' relation between the source pair (back row) of objects and the target pair (front row) of objects. Problems were presented in the same format as the 'relational match-to-sample' task of the ART dataset. **Identity rules:** An example problem depicting ABA rule among the back row and front row of objects. Problems were presented in the same format as the 'identity rules' task of the ART dataset.

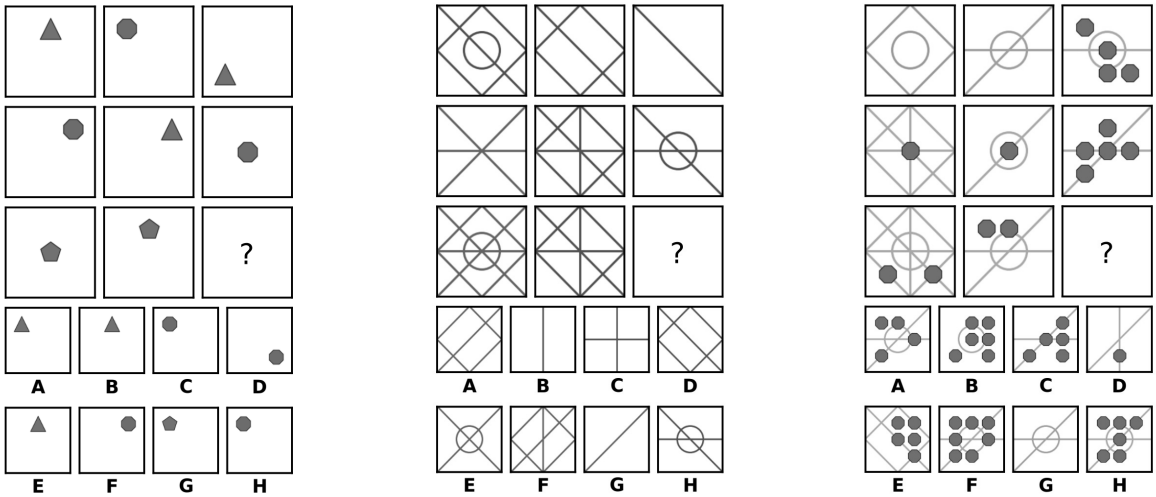


Figure 6. **Procedurally Generated Matrices (PGM) Dataset**. Examples of problems with only shapes (left), only lines (middle), and both shapes and lines (right). In each problem, the task is to identify the abstract pattern among the image panels in the first two rows and/or columns of the matrix, and use that pattern to fill the panel in the third row and column from a set of eight choices.

A.4.1. PGM OOD Regimes

Each matrix problem in PGM is defined by the abstract structure $\mathcal{S} = \{[r, o, a] : r \in \mathcal{R}, o \in \mathcal{O}, a \in \mathcal{A}\}$, where $\mathcal{R} = \{\text{progression, XOR, AND, OR, consistent union}\}$ are the set of rules, $\mathcal{O} = \{\text{shape, line}\}$ are the set of objects, and $\mathcal{A} = \{\text{size, type, position, color, number}\}$ are the set of attributes. Problems consist of multiple rules (upto 4) governed by triples $[r, o, a]$. The PGM dataset contains seven OOD regimes of varying difficulty. The interpolation regime involves training on even-indexed feature values for color and size attributes, and testing on all odd-indexed values, whereas the extrapolation regime involves training on the lower half of feature values and testing on the upper half of feature values. In the heldout attribute shape color (H.O.S-C) regime, the training set problems doesn't contain shape objects and color attribute, whereas all problems in the test set contain at least shape objects and color attribute. Similarly in the heldout attribute line type (H.O.L-T) regime, the training set problems doesn't contain line objects and type attribute. The heldout triples (H.O.Triples) regime, involves randomly holding out seven triples out of 29 possible unique triples $[r, o, a]$ for test set problems. In heldout triple pairs (H.O.T.P.) regime, the problems in both the training and test set contain at least two triples, and out of 400 possible pairs of triples, randomly chosen 40 pairs of triples are used only for test set problems. The heldout arribute pairs

(H.O.A.P.) regime, involves holding out 4 attribute pairs out of possible 20 pairs for test set problems.

Figure 7. **Visual Progressive Matrices (V-PROM) Dataset**. Example problems of Raven's Progressive Matrices involving real-world images. Similar to PGM dataset, the task is to identify the abstract pattern among the images in the first two rows and/or columns of the 3×3 matrix, and use that pattern to fill the missing image in the third row, by selecting one among a set of eight choices.

A.5. Visualization of Slot-wise Reconstruction

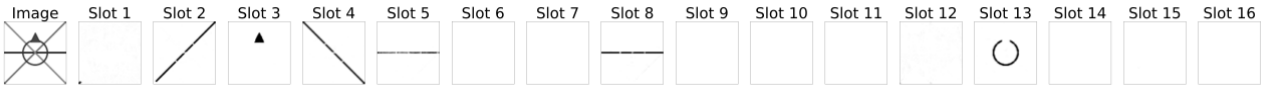


Figure 8. Visualization of the individual slot-wise reconstruction for all the 16 slots for an image panel from the PGM dataset.

Figure 9. Visualization of the individual slot-wise reconstruction for all the 16 slots for an image panel from the PGM dataset.

A.6. Training Details and Hyperparameters

Table 8. CNN encoder hyperparameters for PGM dataset.

Type	Channels	Activation	Kernel size	Stride	Padding
2D Conv	32	ReLU	5×5	1	2
2D Conv	32	ReLU	5×5	1	2
2D Conv	32	ReLU	5×5	1	2
2D Conv	32	ReLU	5×5	1	2
Position Embedding	-	-	-	-	-
Flatten	-	-	-	-	-
Layer Norm	-	-	-	-	-
1D Conv	32	ReLU	1	1	0
1D Conv	32	-	1	1	0

Table 9. Slot decoder hyperparameters for PGM dataset.

TYPE	CHANNELS	ACTIVATION	KERNEL SIZE	STRIDE	PADDING
SPATIAL BROADCAST	-	-	-	-	-
POSITION EMBEDDING	-	-	-	-	-
2D CONV	32	RELU	5×5	1	2
2D CONV	32	RELU	5×5	1	2
2D CONV	32	RELU	5×5	1	2
2D CONV	2	-	3×3	1	1

Table 10. CNN encoder hyperparameters for V-PROM dataset.

TYPE	CHANNELS	ACTIVATION	KERNEL SIZE	STRIDE	PADDING
LAYER NORM	-	-	-	-	-
1D CONV	768	RELU	1	1	0
1D CONV	256	-	1	1	0

Table 11. Slot decoder hyperparameters for V-PROM dataset.

TYPE	CHANNELS	ACTIVATION	KERNEL SIZE	STRIDE	PADDING
SPATIAL BROADCAST	-	-	-	-	-
POSITION EMBEDDING	-	-	-	-	-
1D CONV	2048	RELU	1	1	0
1D CONV	2048	RELU	1	1	0
1D CONV	2048	RELU	1	1	0
1D CONV	769	-	1	1	0

Table 12. Slot attention pretraining hyperparameters for PGM dataset.

SLOT DIMENSIONALITY D	32
NO OF ITERATIONS T	3
NO OF SLOTS K	16
BATCH SIZE	16
LEARNING RATE	$4e-4$
LEARNING RATE WARMUP STEPS	75000
EPOCHS BEFORE LEARNING RATE DECAY	5
LEARNING RATE DECAY RATE	0.5
LEARNING RATE DECAY STEPS	100000
EPOCHS	8

Table 13. Slot attention pretraining hyperparameters for V-PROM dataset.

SLOT DIMENSIONALITY D	256
NO OF ITERATIONS T	3
NO OF SLOTS K	11
BATCH SIZE	48
LEARNING RATE	$4e-4$
LEARNING RATE WARMUP STEPS	10000
EPOCHS	80

Table 14. Hyperparameters of the Abstractor module of the Slot Abstractor. For both multi-head relational cross-attention and multi-head self-attention, H is the number of heads, L is the number of layers, D_{head} is the dimensionality of each head. D_{MLP} is the dimensionality of the MLP hidden layer in the feedforward network.

	ART	SVRT	CLEVR-ART	PGM	V-PROM
H	8	8	8	8	8
L	6	24	24	24	24
D_{head}	64	64	64	32	256
D_{MLP}	512	512	512	512	512
DROPOUT	0	0	0	0	0.1

Table 15. Number of training epochs for ART and CLEVR-ART tasks.

	SAME/DIFFERENT	RELATIONAL-MATCH-TO-SAMPLE	DISTRIBUTION-OF-3	IDENTITY RULES
ART	600	400	400	100
CLEVR-ART	-	50	-	200

Table 16. Number of training epochs for PGM generalization regimes.

	NEUTRAL	INTERPOLATION	H.O.A.P.	H.O.T.P.	H.O.TRIPLES	H.O.L-T	H.O.S-C	EXTRAPOLATION
PGM	73	50	90	90	90	65	65	115