EXTRAPOLATABLE RELATIONAL REASONING WITH COMPARATORS IN LOW-DIMENSIONAL MANIFOLDS

Anonymous authors
Paper under double-blind review

ABSTRACT

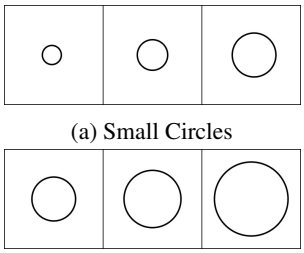
While modern deep neural architectures generalise well when test data is sampled from the same distribution as training data, they fail badly for cases when the test data distribution differs from the training distribution even along a few dimensions. This lack of out-of-distribution generalisation is increasingly manifested when the tasks become more abstract and complex, such as in relational reasoning. In this paper we propose a neuroscience-inspired inductively biased module that can be readily amalgamated with current neural network architectures to improve out-of-distribution (o.o.d) generalisation performance on relational reasoning tasks. This module learns to project high-dimensional object representations to low-dimensional manifolds for more efficient and generalisable relational comparisons. We show that neural nets with this inductive bias achieve considerably better o.o.d generalisation performance for a range of relational reasoning tasks, thus more closely models human ability to generalise even when no previous examples from that domain exist. Finally, we analyse the proposed inductive bias module to understand the importance of lower dimensional projection, and propose an augmentation to the algorithmic alignment theory to better measure algorithmic alignment with generalisation.

1 INTRODUCTION

The goal of Artificial Intelligence research, first proposed in the 1950s and reiterated many times, is to create machine intelligence comparable to that of a human being. While today's deep learning based systems achieve human-comparable performances in specific tasks such as object classification and natural language understanding, they often fail to generalise in out-of-distribution (**o.o.d**) scenarios, where the test data distribution differs from the training data distribution (Recht et al., 2019; Trask et al., 2018; Barrett et al., 2018; Belinkov & Bisk, 2018). Moreover, it is observed that the generalisation error increases as the tasks become more abstract and require more reasoning than perception. This ranges from small drops (3% to 15%) in classification accuracy on ImageNet (Recht et al., 2019) to accuracy only slightly better than random chance for the Raven Progressive Matrices (RPM) test (a popular Human IQ test), when testing data are sampled completely out of the training distribution (Barrett et al., 2018).

In contrast, human brain is observed to generalise better to unseen inputs (Geirhos et al., 2018), and typically requires only a small number of training samples. For example, a human, when trained to recognise that there is a progression relation of circle sizes in Figure 1a, can easily recognise that the same progression relation exists for larger circles in Figure 1b, even though such size comparison has not been done between larger circles. However, today's state-of-the-art neural networks (Barrett et al., 2018; Wang et al., 2020) are not able to achieve the same. Researchers (Spelke & Kinzler, 2007; Chollet, 2019; Battaglia et al., 2018; Xu et al., 2020) argue that the human brain developed special inductive biases that adapt to the form of information processing needed for humans, thereby improving generalisation. Examples include convolution-like cells in the visual cortex (Hubel & Wiesel, 1959; Güçlü & van Gerven, 2015) for visual information processing, and grid cells (Hafting et al., 2005) for spatial information processing and relational comparison between objects (Battaglia et al., 2018).

In this work, we propose a simple yet effective inductive bias which improves **o.o.d** generalisation for relational reasoning. We specifically focus on a the type of **o.o.d** called 'extrapolation'. For extrapolation tasks, the range of one or more data attributes (e.g., object size) from training and test



(a) Small Circles

(b) Big Circles

Figure 1: Size Progression Relations for circles of different sizes.

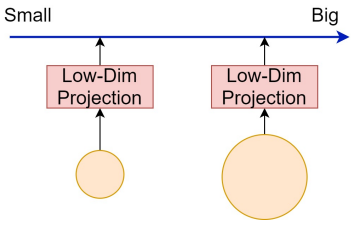


Figure 2: Illustration of projecting object representations onto a 1-dimensional manifold in which size comparison can be achieved by simply measuring the difference between two projections.

datasets are completely non-overlapping. The proposed inductive bias is inspired by neuroscience and psychology research (Fitzgerald et al., 2013; Chafee, 2013; Summerfield et al., 2020) showing that in primate brain there are neurons in the Parietal Cortex which only responds to different specific attributes of perceived entities. For examples, certain LIP neurons fire at higher rate for larger objects, while the firing rate of other neurons correlates with the horizontal position of objects in the scene (left vs right) (Gong & Liu, 2019). From a computational perspective, this can be viewed as projecting object representations to low-dimensional manifolds. Based on these observations (Summerfield et al., 2020), we hypothesise that these neurons are developed to learn low-dimensional representations of relational structure that are optimised for abstraction and generalisation, and the same inductive bias can be readily adapted for artificial neural networks to achieve similar optimisation for abstraction and generalisation.

We test this hypothesis by designing an inductive bias module which projects high-dimensional object representations into low-dimensional manifolds, and make comparisons between different objects in these manifolds. We show that this module can be readily amalgamated with existing architectures to improve extrapolation performance for different relational reasoning tasks. Specifically, we performed experiments on three different extrapolation tasks, including maximum of a set, visual object comparison on dSprites dataset (Higgins et al., 2017) and extrapolation on RPM style tasks (Barrett et al., 2018). We show that models with the proposed low-dimensional comparators perform considerably better than baseline models on all three tasks. In order to understand the effectiveness of comparing in low-dimensional manifolds, we analyse the projection space and corresponding function space of the comparator to show the importance of projection to low-dimensional manifolds in improving generalisation. Finally, we perform the analysis relating to algorithmic alignment theory (Xu et al., 2020), and propose an augmentation to the sample complexity criteria used by this theory to better measure algorithmic alignment with generalisation.

2 METHOD

Here, we describe the inductive bias module we developed to test our hypothesis that the same inductive bias of low-dimensional representation observed in Parietal Cortex can be readily adapted for artificial neural network, thus enabling it to achieve similar optimisation for abstraction and generalisation. The proposed module learns to project object representations into low-dimensional manifolds and make comparisons in these manifolds. In Section 2.1 we describe the module in detail. In Section 2.2, 2.3 and 2.4 we discuss how this module can be utilised for three different relational reasoning tasks, which are: finding the maximum of a set, visual object comparisons and Raven Progressive Matrics (RPM) reasoning.

2.1 COMPARATOR IN LOW-DIMENSIONAL MANIFOLDS

The inductive bias module is comprised of low-dim projection functions p and comparators c. Let $\{o_i; i \in 1 \dots N\}$ be the set of object representations, obtained by extracting features from raw inputs such as applying Convolutional Neural Networks (CNN) on images. Pairwise comparison between object pair o_i and o_j can be achieved with a function f expressed as:

$$f(o_i, o_j) = g(\mathop{||}_{k=1}^{K} c_k(p_k(o_i), p_k(o_j))). \tag{1}$$

Here p_k is the kth projection function that projects object representation o into the kth low dimensional manifold, c_k is the kth comparator function that compares the projected representations, $||$ is the

concatenation symbol and g is a function that combines the K comparison results to make a prediction. Having K parallel projection functions p_k and comparators c_k enables a simultaneous comparison between objects with respect to their different attributes. Figure 2 shows an example of comparing sizes of circles by projection onto a 1-dimensional manifold. Both p and c can be implemented as feed forward neural networks. While the comparator c, implemented as a neural network, can theoretically learn a rich range of comparison metrics, we found that adding to c an additional inductive bias of distance measure for the projection, such as vector distance $p(o_i) - p(o_j)$ or absolute distance $|p(o_i) - p(o_j)|$, improves the generalisation performance.

Let $a_t(o_i)$ be the ground truth mapping function from i^{th} object's representation o_i to its tth attributes (such as colour and size for a visual object). If such ground truth labels of object attributes exist, $f(o_i, o_j)$ can be trained to directly predict the differences in attributes by minimising the loss $\mathcal{L}(d(a_t(o_i), a_t(o_j)), f(o_i, o_j))$, where d is a distance function (*e.g.*, $a_t(o_i) - a_t(o_j)$ for continuous attributes or $\mathbb{1}_{a_t(o_i)=a_t(o_j)}$ for categorical attributes). However, in real-world datasets, such ground truth attribute labels seldom exist. Instead, in many relational reasoning tasks, learning signals for attribute comparison are only provided implicitly in the training objective. For example, in Visual Question Answering task, an example question might be 'Is the object behind Object A smaller?'. The learning signals for the required size and spatial position comparator are provided only through correctness of the answers to the given questions. Thus, the proposed module is only useful and scalable if it can be integrated into neural architectures for relational reasoning and still learn to compare attributes with the weaker, implicit learning signal. Next, we describe 3 examples of such integrations for different relational reasoning tasks, and show in Section 3 that the proposed module can learn relational reasoning tasks with better generalisation capability.

2.2 Architecture: Maximum of a Set

The first task we consider is finding the maximum of a set of real numbers. Formally, given a set $\{x_i; i \in 1 \dots N\}$ where x_i is a real number represented as a scalar value, we want to train a function $h_{max}(\{x_i, \dots, x_N\})$ that gives the maximum value in the set. Many neural architectures have been applied on this task, including Deep Sets (Zaheer et al., 2017) and Set Transformer (Lee et al., 2019), but none of them test (**o.o.d**) generalisation capability. In order to test **o.o.d** generalisation in the extrapolation scenario, we create the training and test sets such that their ranges do not overlap. We sample from the range $(V_{low}^{train}, V_{high}^{train})$ for the training set, and from the range $(V_{low}^{test}, V_{high}^{test})$ for the test set, and restrict that $V_{high}^{train} < V_{low}^{test}$.

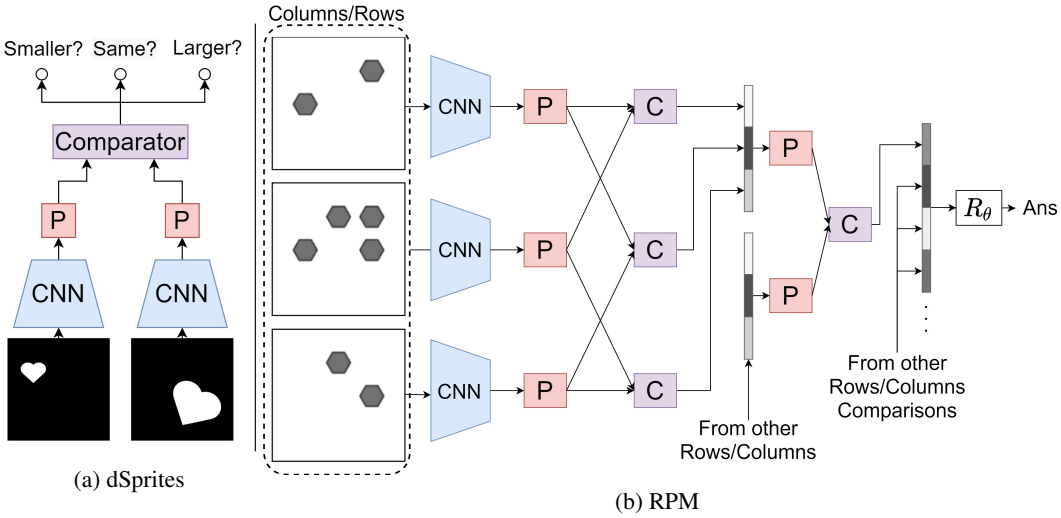
We integrate the proposed low-dim comparator module with Set Transformer (Lee et al., 2019), a state-of-the-art neural architecture for sets. Set Transformer first encodes each element in the set with respect to all other elements with a Multihead Attention Block (MAB), an attention module modified from self-attention used in language tasks (Vaswani et al., 2017): $e_i = encode(x_i) = \mathsf{MAB}(x_i, x_j)$. The Set Transformer then uses Pooling with Multihead Attention (PMA) to combine all encoded elements of the set as $\mathsf{PMA}(e_1, \dots, e_N)$. While MAB uses query and key embeddings to generate attention variables, which are then used as weights in the weight sum of value embeddings of elements, we swap the query-key attention mechanism with our low-dim comparator as:

$$e_i = MLP(\sum_{j=1}^{N} f(x_i, x_j)) \tag{2}$$

Here f is the low-dim comparator and MLP is a standard Multi-Layer Perceptron. Note that we directly use the scalar input x_i as object representation o_i in Equation (1) as no feature extraction is needed. We then use attention-based pooling to combine projection of x_i as $\sum_{i=1}^{N} a(e_i)p(x_i)$, where a outputs attention values while p is the 1-dim projection function. For detailed architecture configuration, please refer to Appendix A.

2.3 Architecture: Visual Object Comparison

The second task we consider is comparing visual objects for different attributes such as size and spatial position. For this task, two images x_1 and x_2 containing single objects of randomly sampled attributes are given, and one is asked if a specific attribute of the second object is larger than, equal to, or smaller than the attribute of the first object. Figure 3a shows an example of this task for comparing sizes between two heart-shaped objects. To sample images of objects in the implementation, we used the dSprites dataset (Higgins et al., 2017), a widely used dataset for studying latent space



(a) dSprites

(b) RPM

Figure 3: Figure (a) illustrates the architecture for comparing sizes of objects sampled from dSprites dataset. "P" is the projection function. Figure (b) illustrates the architecture for logical reasoning on RPM-style tasks. "P" is the projection function and "C" is the comparator (g in Equation 1).

disentangling. To test extrapolation capability, we sample the training set and the test set such that for the compared attribute, the training attribute range has no overlap with the test attributes. We leave details of the dataset construction to Appendix B.

Figure 3a shows an overview of the architecture integrated with the proposed low-dimensional comparator. The image pair x_1 and x_2 is first passed through a CNN to extract feature embeddings e_1 and e_2. The feature embeddings are then projected to low-dim manifold and compared as $c(p(e_1), p(e_2))$, where p is the projector and c is the comparator. The comparator has 3 output units with softmax to predict probabilities that an attribute of the second object $a(x_2)$ is smaller than, equal to, or larger than the attribute of the first object $a(x_1)$. The architecture is trained with cross entropy loss with respect to ground truth labels. While we are testing the **o.o.d** generalisation of relational reasoning, it is reasonable to expect that the visual perception module is exposed to all possible scenarios of the input distribution in an unsupervised way. The same assumption also holds for humans, whose vision system has to be exposed to the world inputs sufficiently after birth before they can associate objects with semantic meaning and perform relational reasoning (Maurer, 2016). Thus, we initialise the CNN with the pretrained encoder weights of Beta-VAE (Higgins et al., 2017), a disentangled VAE model trained in the unsupervised setup on dSprites dataset. For detailed configuration of the architecture please refer to Appendix C.

2.4 Architecture: Visual Reasoning for Raven Progressive Matrices

The third task we consider is a more complex visual reasoning task named 'Raven Progressive Matrices' (RPM), which is a popular human IQ test. In this task one is given 8 context diagrams with logic relations present in them, and is asked to pick an answer that best fits with the context diagrams. In our experiment we use the PGM dataset (Barrett et al., 2018), the largest RPM-style dataset available. In the PGM dataset, there is a special data split called 'extrapolation' which is designed to test for the extrapolation scenario of **o.o.d** generalisation.

In this split, colour and size values of objects in the training set are sampled from the lower half of the range, while the same attributes in the test set are sampled from the upper half of the range. Thus the attribute ranges of training and test sets are non-overlapping. For details and examples of the PGM dataset, please refer to Appendix D and Barrett et al. (2018).

Our architecture integrates the low-dim comparator with a Multi-Layer Relation Network (Jahrens & Martinetz, 2020). Figure 3b shows an overview of the architecture developed for PGM tasks. We use a 2-layer relation network with the first layer encoding pairs of diagrams within a row/column and the second layer encoding pairs of rows/columns. Applying such prior knowledge that rules only exist in rows and columns has been standard practice in state-of-the-art methods for RPM reasoning (Wang et al., 2020; Zhang et al., 2019). Following Wang et al. (2020) we fill each of the 8 candidate answers

to the third row and column to obtain in total of 16 answer rows and columns. At each layer of the relation network, we use the low-dimensional comparator instead of the MLP in the original relation network (Santoro et al., 2017) for diagram comparison. Diagram x_i is first passed through a CNN to produce the embedding o_i. embedding pairs (o_i, o_j) are then compared as $e_{ij} = f(o_i, o_j)$, where f is the low-dimensional comparator described in Equation (1). The comparison results from the same rows/columns are then concatenated to form row/column embedding $r_{ijk} = e_{ij}||e_{ik}||e_{jk}$. The row/column embeddings are compared with the second layer comparator. The comparison results are then concatenated and input into a reasoning network R_θ to predict the correct answer. Similar to the dSprites comparison task, as discussed in Section 2.3, we pre-train CNN as encoders of VAE, a technique that has also been previously explored for the PGM dataset (Steenbrugge et al., 2018). For detailed configuration of the architecture, please refer to Appendix E.

2.5 Algorithmic Alignment and o.o.d Generalisation

Xu et al. (2020) proposed to measure algorithmic alignment of neural networks to a specific task with sample complexity $\mathcal{C}_\mathcal{A}(g, \epsilon, \delta)$, which is the minimum sample size M so that g, the ground truth label mapping function, is (M, ϵ, δ)-learnable with a learning algorithm $\mathcal{A}$. This essentially says that a model is more algorithmically aligned with a task if it can learn the task more easily with fewer samples. However, in the original definition, both training and test data are independently and identically distributed (i.i.d) samples drawn from the same data distribution. Thus, the algorithmic alignment theory measures how well can a NN fit to a particular data distribution, but does not measure how well can a NN model perform in the **o.o.d** scenario. For example, for the visual object comparison task, an over-parameterised MLP can learn the following two algorithms with low complexity: (1) $m(hash(o_i), hash(o_j))$ where $hash$ is a hashing function and m is a memory read/write function based on the hash index; and (2) $c(p(o_i), p(o_j))$ which is our proposed comparator function. While both algorithms can fit well for the training data, the first algorithm clearly does not **o.o.d** generalise as the memory function does not store unseen samples. In Section 3.5 we confirm experimentally that algorithmic alignment is not indicative of **o.o.d** generalisation.

Intuitively, a more algorithmically aligned model should generalise better as it captures better the underlying algorithm of label generation. Here we propose an augmentation to sample complexity metric (Definition 3.3 in Xu et al. (2020)) in order to measure for algorithmic alignment with generalisation (specifically extrapolation).

Definition 2.1. o.o.d metric. Fix an error parameter $\epsilon > 0$ and failure probability $\delta \in (0, 1)$. Suppose $\{x_i^s, y_i^s\}_{i=1}^M$ are i.i.d samples from distribution $\mathcal{D}_s = \mathcal{T}(\mathcal{D}, \beta, \mathbf{u})$, where $\mathcal{D}$ is the full data distribution, $\mathcal{T}$ is a truncating function, $\beta \in (0, 1)$ is the truncation ratio, and $\mathbf{u}$ is the set of dimensions for truncation. Let g be the underlying data function $y_i = g(x_i)$, and $f = \mathcal{A}(\{x_i, y_i\}_{i=1}^M)$ be the function learnt with the learning algorithm $\mathcal{A}$. Then g is $(M, \epsilon, \delta, V, \beta, \mathbf{u}) - learnable$ with $\mathcal{A}$ if:

$$\mathbb{P}_{x \sim \mathcal{D}}[||f(x) - g(x)|| < \epsilon] \geq 1 - \delta \tag{3}$$

The sample complexity is the minimum M for g to be $(M, \epsilon, \delta, V, \beta, \mathbf{u}) - learnable$ with $\mathcal{A}$. In Section 3.5 we show experimentally that this metric measures better a NN's ability to generalise.

3 Evaluation

3.1 Maximum of a Set

For the task of finding the maximum number in a set, we randomly sample number sets of cardinality ranging from 2 to 20 for training, and 2 to 40 for testing. For number sets for training, we uniformly sample numbers in the real value range $[0, 100)$. For testing we sample numbers in the range $[100, 200]$. In this way we test if the model can generalise both, for sets of larger cardinality and for numbers sampled from an unseen range. We sampled 10000 sets for training and 2000 sets for testing. For hyper-parameters of this and subsequent experiments please refer to Appendix F. Table 1 shows the test error of our model compared to Deep Sets (Zaheer et al., 2017) and Set Transformer (Lee et al., 2019), two previous state-of-the-art architectures for sets. Our model achieves a much lower extrapolation error than other methods, even lower than Deep Sets with a built-in Max-Pooling function.

3.2 Visual Object Comparison

For visual object comparison task, we set three sub-tasks for comparing different attributes of the object, including size, horizontal position and colour intensity. For each task we sample visual objects

Table 1: Extrapolation test error for learning to find the maximum number in a set of numbers ($mean \pm std$ for 10 runs). M.S.E means Mean Squared Error.

Model	Deep Sets(Mean) Zaheer et al. (2017)	Deep Sets(Max) Zaheer et al. (2017)	Set Transformer Lee et al. (2019)	OURS
M.S.E	73.22 ± 17.11	0.51 ± 0.29	1.62 ± 0.76	$\mathbf{0.0015 \pm 0.0008}$

Table 2: Extrapolation test accuracies of baseline and our proposed model for dSprites attribute comparison task (10 runs). X-Coord is horizontal position.

Model \ Attributes	Size	X-Coord	Colour
Baseline	$79.52 \pm 6.71\%$	$66.14 \pm 5.03\%$	$78.45 \pm 3.34\%$
OURS	$\mathbf{94.05 \pm 3.03\%}$	$\mathbf{79.11 \pm 1.92\%}$	$\mathbf{97.71 \pm 2.81\%}$

Table 3: Extrapolation test accuracies for the extrapolation split of PGM dataset.

Model	WReN Barrett et al. (2018)	MXGNet Wang et al. (2020)	MLRN Jahrens & Martinetz (2020)	MLRN-P	OURS
Accuracy	17.2%	18.9%	14.9%	18.1%	**25.9%**

with different range for the compared attributes from the dSprites dataset (Higgins et al., 2017). Given the compared attribute range $[V_{low}, V_{high}]$, we sample training data from range $[V_{low}, \frac{2}{3}V_{high})$ and test data from range $[\frac{2}{3}V_{high}, V_{high}]$. As ground truth attribute value is provided in the dSprites dataset, we can build the comparison label as $(\mathbb{1}_{a_1<a_2}, \mathbb{1}_{a_1=a_2}, \mathbb{1}_{a_1>a_2})$. For all experiments we sample 60000 training pairs and 20000 test pairs. We test our proposed model against an MLP baseline, which directly processes the object representations o_i and o_j extracted by CNN as $MLP(o_i, o_j)$. We select the best MLP by hyper-parameter search over the number of layers and layer sizes. We use a 1-dimensional projection as we find this gives the highest accuracy. Table 2 shows the extrapolation test accuracies of our model compared to the baseline. Our model with low-dimensional comparator significantly outperforms baselines for all three compared attributes.

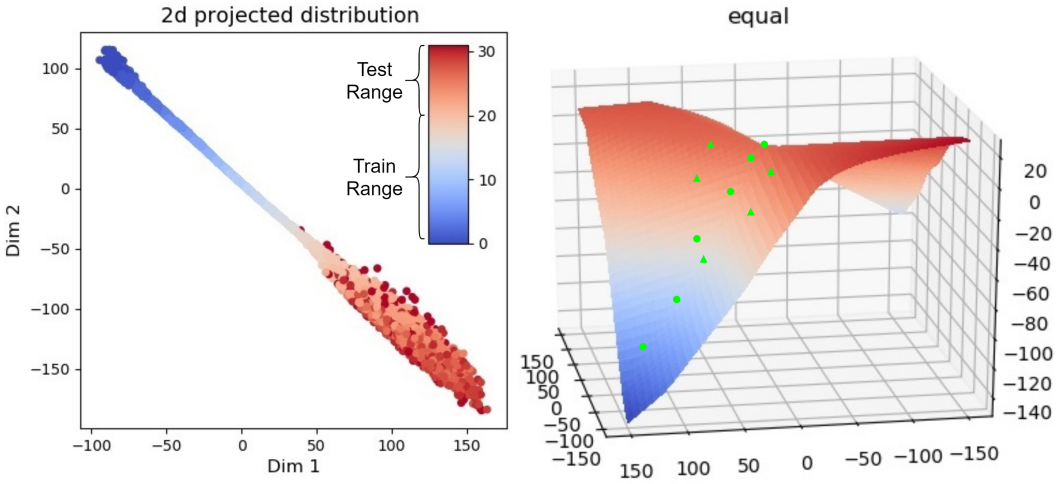
3.3 Visual Reasoning for Raven Progressive Matrices

For RPM-style tasks we use the extrapolation split of the PGM dataset (Barrett et al., 2018), which is already a well-defined extrapolation type of generalisation task. We compare our proposed model against all previous methods (to the best of our knowledge) that have reported results on the extrapolation data split. We additionally include a baseline model named "MLRN-P", which is a 2-layer MLRN (Jahrens & Martinetz, 2020) with prior knowledge of only the relations present in rows/columns and with pre-training. Table 3 shows the test accuracy comparison. Our proposed model outperforms all other baselines. We note that we used a vanilla CNN as the perception module, same as most previous methods (Barrett et al., 2018; Zhang et al., 2019; Jahrens & Martinetz, 2020) on RPM tasks. Multiple-object representation learning methods (Greff et al., 2019; Kosiorek et al., 2019; Wang et al., 2019a), which achieve better results for multi-object scene learning than CNN, can be investigated for potential improvement of generalisation performance. We leave this for future work.

3.4 Why Low Dimension?

While we show that the comparators in low-dimensional manifolds improve **o.o.d** generalisation for a range of relational reasoning tasks, the reason behind it is still not clear. In this section we analyse the projection space and the comparator function landscape of the visual object comparison task to shed light on this. We first state 3 observations invariant across different sub-tasks comparing different attributes:

1. *When the ground truth attribute can be represented in a 1-dimensional manifold (such as vertical position), comparators in higher dimensions learn to project the object representation into the 1-dimensional manifold.* Figure 4a illustrates this with a plot of the projection distribution for the task of comparing vertical positions. It can be observed that even though the projection space is 2-dimensional, the projected points cluster around a line.



(a) 2D projection space

(b) 2D comparator function

Figure 4: (a) shows a scatter plot of the 2D projected distribution of objects in the task of comparing vertical positions of objects in the image. X-axis and Y-axis are 2 dimensions of the projection manifold. Latent vertical position (ranging from 0 to 32) is indicated by colour. (b) plots the comparator's function landscape for "equal" output unit (before softmax) in the space of vector differences between 2D projections of objects. Green circles represent the vector difference sampled from the training set while triangles represent the vector difference sampled from the test set.

2. *The projection of the test data in the manifold is less clustered around the sub-manifold of attributes than that of training data.* This can be observed from Figure 4a, where the points projected from the test set are more spread out than from the training set. There is also less order in the distribution of test points, where points of noticeably different intensity appear next to each other.
3. *The function landscape becomes less defined outside of the sub-manifold.* Figure 4b plots the comparator function landscape for the "equal" ($\mathbb{1}_{a_1=a_2}$) output unit over the space of vector differences between 2D projected representation $p(o_2) - p(o_1)$. The equality function is well defined in the sub-manifold in which training points (circles) lie, peaking close to the $(0, 0)$ point. However, outside of the training points' sub-manifold, the function is more random, with a significant region with higher function value than at $(0, 0)$ point. Note that the vector differences of test points (triangles) may be in this region.

From the above observations, we conclude that when comparators are of higher dimension than the intrinsic dimension of compared attributes, the projection tends to lie in a sub-manifold of the same dimension as for the attributes, resulting in the comparator function to be only well defined in that manifold. However, the projection of test data tends to escape from this sub-manifold into the region where the comparator function is never trained on, resulting in incorrect prediction.

In addition to the observations above, we also use Hausdorff distance as a quantitative measure of discrepancies between projected repesentations from the training and test sets. Due to space limitations, this is discussed in Appendix H.

3.5 Algorithmic Alignment

Figure 5a shows the **o.o.d** and **i.d** (identically distributed) test accuracies for size comparison task of baseline and our proposed comparator model for different training sample sizes. It can be observed that **i.d** test accuracy's sample complexity (training samples needed to achieve the same accuracy), which measures algorithmic alignment, is not indicative of **o.o.d** test accuracies.

Figure 5b shows the size comparison of the test accuracies of baseline and comparator for different training sample sizes with different β rates for truncating the training distribution along the latent dimension 'size'. This corresponds to our proposed metric in Section 2.5. The new metric reflects that the model which learns better with truncated training distribution is the one with better **o.o.d** generalisation.

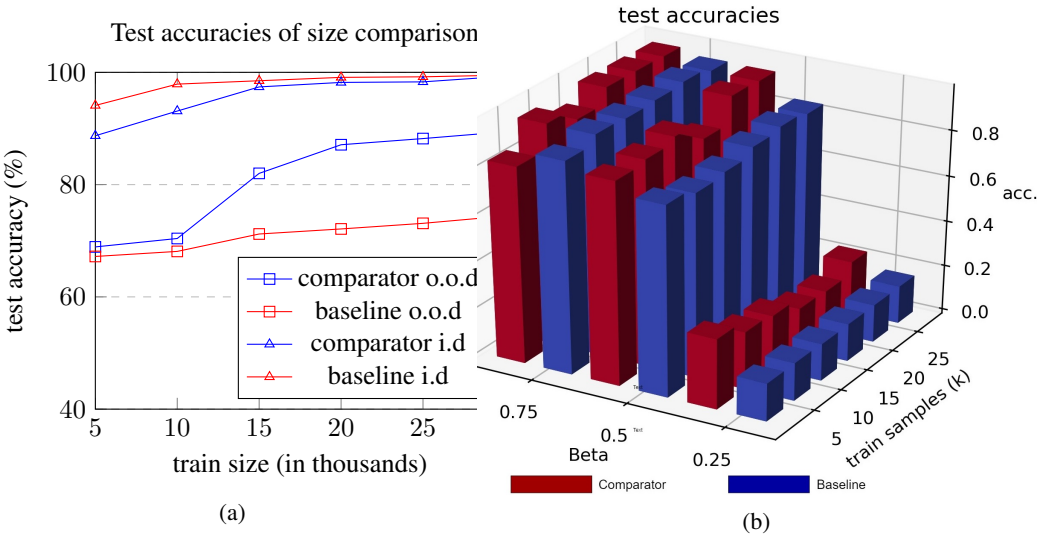


Figure 5: Figure (a) shows o.o.d and i.d. (identically distributed) test accuracy of baseline and comparator for different training sample sizes. (b) shows test accuracies of baseline and comparator for differnt training sample sizes and with different β-rate for truncating training distribution.

4 Related Work and Conclusion

o.o.d Generalisation: The deep neural network's lack of **o.o.d** generalisation (sometimes termed domain generalisation or extrapolation) capability has recently come under scrutiny. Different types of approaches have been proposed to improve **o.o.d** generalisation, such as reducing superficial domain specific statistics of training data (Wang et al., 2019b; Carlucci et al., 2019), adversarially learn representations that are domain-invariant (Li et al., 2017; Albuquerque et al., 2019), disentangling representations to separate functional variables with spurious correlations (Heinze-Deml & Meinshausen, 2017; Gowal et al., 2019), and constructing models with innate causal inference graphs to reduce the dependence on spurious correlations (Arjovsky et al., 2019; Bengio et al., 2020). Our work aligns more with the line of work on discovering inductive bias that improves generalisation. Arguably CNN (LeCun et al., 1995) is such an inductive bias that improves generalisation on visual input, and Graph Neural Network an inductive bias on graph-structured data (Battaglia et al., 2018). Trask et al. (2018) proposed Neural Arithmetic Logic Units (NALU), an inductive bias that allows neural networks to learn simple arithmetic with improved **o.o.d** generalisation.

Relational Reasoning: There is a wide range of relational reasoning tasks such as Visual Question Answering (Antol et al., 2015; Johnson et al., 2017), Raven Progressive Matrices (Barrett et al., 2018; Zhang et al., 2019), and Inferring Physical interactions (Kipf et al., 2018; Sanchez-Gonzalez et al., 2018). These tasks involve comparing entities such as visual objects to infer relations between them. A large proportion of models proposed to solve relational reasoning tasks fall into the broader range of graph neural networks and relational networks (Battaglia et al., 2018). Our work is mostly orthogonal to these works, and may be viewed as a special type of edge layer that can be integrated into most of these models. There is another line of research on neural-symbolic models (Yi et al., 2018; Mao et al., 2019). Our work differs from these approaches in not using pre-defined programs, but learnable modules for comparing representations. PrediNet (Shanahan et al., 2019) use self-attention to extract features from the input, and then compare extracted features with a comparator. While our work is similar to PrediNet in that we also use a comparator module to compare extracted features, PrediNet uses a different comparator module, different ways to extract features, and does not explicitly investigate how low-dimensional projections help with **o.o.d** generalisation.

Low-Dimensional Embedding: While there is a rich history of producing low-dimensional embeddings for machine learning (see Appendix J for a brief discussion), to the best of our knowledge, little prior work has investigated how low-dimensional embedding improves **o.o.d** generalisation for relational reasoning tasks. van Steenkiste et al. (2019) is the most related work, but focuses on the disentanglement property rather than on low-dimensional property for relational reasoning tasks.

Conclusion: We proposed a neuroscience-inspired inductive bias for improving the generalisation ability of neural networks for relational reasoning tasks. We showcased its effectiveness on three selected tasks, but the method can readily be adapted to any other relational reasoning task.

REFERENCES

Isabela Albuquerque, João Monteiro, Tiago H Falk, and Ioannis Mitliagkas. Adversarial target-invariant representation learning for domain generalization. *arXiv preprint arXiv:1911.00804*, 2019.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pp. 2425–2433, 2015.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

David Barrett, Felix Hill, Adam Santoro, Ari Morcos, and Timothy Lillicrap. Measuring abstract reasoning in neural networks. In *International Conference on Machine Learning*, pp. 511–520, 2018.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Yonatan Belinkov and Yonatan Bisk. Synthetic and natural noise both break neural machine translation. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BJ8vJebC-.

Yoshua Bengio, Tristan Deleu, Nasim Rahaman, Nan Rosemary Ke, Sebastien Lachapelle, Olexa Bilaniuk, Anirudh Goyal, and Christopher Pal. A meta-transfer objective for learning to disentangle causal mechanisms. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ryxWIgBFPS.

Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2229–2238, 2019.

Matthew V Chafee. A scalar neural code for categories in parietal cortex: Representing cognitive variables as "more" or "less". *Neuron*, 77(1):7–9, 2013.

François Chollet. On the measure of intelligence, 2019.

Pierre Comon. Independent component analysis, a new concept? *Signal processing*, 36(3):287–314, 1994.

Jamie K Fitzgerald, David J Freedman, Alessandra Fanini, Sharath Bennur, Joshua I Gold, and John A Assad. Biased associative representations in parietal cortex. *Neuron*, 77(1):180–191, 2013.

Robert Geirhos, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 7538–7550, 2018.

Mengyuan Gong and Taosheng Liu. Biased neural coding of feature-based attention in human brain. *bioRxiv*, pp. 688226, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 27*, pp. 2672–2680. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

Sven Gowal, Chongli Qin, Po-Sen Huang, Taylan Cemgil, Krishnamurthy Dvijotham, Timothy Mann, and Pushmeet Kohli. Achieving robustness in the wild via adversarial mixing with disentangled representations. *arXiv preprint arXiv:1912.03192*, 2019.

Klaus Greff, Raphaël Lopez Kaufmann, Rishab Kabra, Nick Watters, Chris Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. *arXiv preprint arXiv:1903.00450*, 2019.

Umut Güçlü and Marcel AJ van Gerven. Deep neural networks reveal a gradient in the complexity of neural representations across the ventral stream. *Journal of Neuroscience*, 35(27):10005–10014, 2015.

Torkel Hafting, Marianne Fyhn, Sturla Molden, May-Britt Moser, and Edvard I Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806, 2005.

Christina Heinze-Deml and Nicolai Meinshausen. Conditional variance penalties and domain shift robustness. *arXiv preprint arXiv:1710.11469*, 2017.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *Iclr*, 2(5):6, 2017.

David H Hubel and Torsten N Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574–591, 1959.

Marius Jahrens and Thomas Martinetz. Solving raven's progressive matrices with multi-layer relation networks. *arXiv preprint arXiv:2003.11608*, 2020.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 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.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. *arXiv preprint arXiv:1802.04687*, 2018.

Adam Kosiorek, Sara Sabour, Yee Whye Teh, and Geoffrey E Hinton. Stacked capsule autoencoders. In *Advances in Neural Information Processing Systems*, pp. 15486–15496, 2019.

Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, pp. 3744–3753, 2019.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pp. 5542–5550, 2017.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*, 2019.

Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. *arXiv preprint arXiv:1904.12584*, 2019.

Daphne Maurer. How the baby learns to see: Donald o. hebb award lecture, canadian society for brain, behaviour, and cognitive science, ottawa, june 2015. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, 70(3):195, 2016.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pp. 5389–5400, 2019.

Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. *arXiv preprint arXiv:1806.01242*, 2018.

Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pp. 4967–4976, 2017.

Murray Shanahan, Kyriacos Nikiforou, Antonia Creswell, Christos Kaplanis, David Barrett, and Marta Garnelo. An explicitly relational neural network architecture. *arXiv preprint arXiv:1905.10307*, 2019.

Elizabeth S Spelke and Katherine D Kinzler. Core knowledge. *Developmental science*, 10(1):89–96, 2007.

Xander Steenbrugge, Sam Leroux, Tim Verbelen, and Bart Dhoedt. Improving generalization for abstract reasoning tasks using disentangled feature representations. *arXiv preprint arXiv:1811.04784*, 2018.

Christopher Summerfield, Fabrice Luyckx, and Hannah Sheahan. Structure learning and the posterior parietal cortex. *Progress in neurobiology*, 184:101717, 2020.

Andrew Trask, Felix Hill, Scott E Reed, Jack Rae, Chris Dyer, and Phil Blunsom. Neural arithmetic logic units. In *Advances in Neural Information Processing Systems*, pp. 8035–8044, 2018.

Sjoerd van Steenkiste, Francesco Locatello, Jürgen Schmidhuber, and Olivier Bachem. Are disentangled representations helpful for abstract visual reasoning? In *Advances in Neural Information Processing Systems*, pp. 14245–14258, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

D. Wang, M. Jamnik, and P. Lio'. Unsupervised and interpretable scene discovery with discrete-attend-infer-repeat. In *ICML workshop on Self-Supervised Learning*, pp. 9pp, 2019a. URL https://arxiv.org/abs/1903.06581.

Duo Wang, Mateja Jamnik, and Pietro Lio. Abstract diagrammatic reasoning with multiplex graph networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ByxQB1BKwH.

Haohan Wang, Zexue He, Zachary C Lipton, and Eric P Xing. Learning robust representations by projecting superficial statistics out. *arXiv preprint arXiv:1903.06256*, 2019b.

Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

Keyulu Xu, Jingling Li, Mozhi Zhang, Simon S. Du, Ken ichi Kawarabayashi, and Stefanie Jegelka. What can neural networks reason about? In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rJxbJeHFPS.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. In *Advances in Neural Information Processing Systems*, pp. 1031–1042, 2018.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in neural information processing systems*, pp. 3391–3401, 2017.

Chi Zhang, Baoxiong Jia, Feng Gao, Yixin Zhu, Hongjing Lu, and Song-Chun Zhu. Learning perceptual inference by contrasting. In *Advances in Neural Information Processing Systems*, pp. 1073–1085, 2019.

Shengjia Zhao, Hongyu Ren, Arianna Yuan, Jiaming Song, Noah Goodman, and Stefano Ermon. Bias and generalization in deep generative models: An empirical study. In *Advances in Neural Information Processing Systems*, pp. 10792–10801, 2018.

A MAXIMUM OF A SET: ARCHITECTURE CONFIGURATIONS

The architecture for the maximum of a set task has three sub-modules, namely a comparator $f(x_i, x_j)$, a comparison summariser $e_i = MLP(\sum_{j=1}^{N} f(x_i, x_j))$ and a pooling function $\sum_{i=1}^{N} a(e_i)p(x_i)$. For comparator f, we set K, the number of parallel comparisons (Equantion 1 in the main body of the paper) to be 1 because the scalar valued real numbers do not have multiple parallel attributes. We implement the projection function p as a single feed forward layer. We choose 1-dimensional comparison space as this gives the best result. The comparison function c takes the projected difference $p(x_i) - p(x_j)$ as input, and is implemented as a single feed forward layer with 1 output unit. The MLP in the comparison summariser is implemented as a 2-layer MLP of hidden-size $16 - 1$. In the pooling function, the attention function a is implemented as a softmax layer, which normalises e_i across $i \in 1 \dots N$.

B VISUAL OBJECT COMPARISON: DATASET GENERATION

In this section we describe details of the visual object comparison dataset. We sample images from the dSprites dataset Higgins et al. (2017) and generate comparison labels (categories include smaller than, equal to, greater than) from ground truth latent values provided in the dataset. For each image in the dSprites dataset, 5 ground truth attribute values are provided, which are shape "category", "size", "rotation angle", "horizontal position" and "vertical position". We add "colour" as the 6th attribute by randomly generating the colour intensity value from the set [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0] for each image. We multiply image pixel values with the colour intensity, and add the colour intensity value to the ground truth latent values.

Algorithm 1 shows the pseudo-code for generating the visual object comparison dataset. compare_attr indicates the object attribute to be compare for the task. We pick three different attributes for experiments, which are "size", "horizontal position" and "colour intensity". We set training attribute range to be the lower 60% while test attribute range to be the upper 40%.

```
Input: train_size, test_size, compare_attr
train_data = EmptyList()
test_data = EmptyList()
for split in {train,test} do
  for i = 1 to train_size do
    attr_range, attr_idx = attr_stats(compare_attr)
    if split == train then
      LO, HI = 0, 0.6
    else
      LO, HI = 0.6, 1.0
    end if
    sample_range = Truncate(attr_range, LO, HI)
    latent_values_A = SampleLatent(attr_idx, sample_range)
    latent_values_B = SampleLatent(attr_idx, sample_range)
    image_A = SampleImage(latent_values_A)
    image_B = SampleImage(latent_values_B)
    less_than = latent_values_A < latent_values_B
    equal = latent_values_A == latent_values_B
    greater_than = latent_values_A > latent_values_B
    label = Concat(less_than, equal, greater_than)
    if split == train then
      train_data ← (image_A,image_B,label)
    else
      test_data ← (image_A,image_B,label)
    end if
  end for
end for
```

Algorithm 1: Visual Object Comparison Dataset Generation

C VISUAL OBJECT COMPARISON: ARCHITECTURE CONFIGURATIONS

In this section we list detailed configuration of the architecture with low-dim comparator and the baseline architecture. The architecture with low-dim comparator is illustrated in Figure 3a. The

CNN module is a 4-layer CNN of filter number $32 - 32 - 64 - 64$ followed by a 2-layer MLP of hidden size $256 - 10$. Each CNN layer has stride value 2 and padding value 1. The output of each CNN is the object representation o_i of raw image input x_i. The projection module p is a single feed forward layer projecting to a space with dimension d. We use $d = 1$ for reporting results except for the manifold analysis experiments in Section 3.4. The comparator takes projected vector difference $p(o_i) - p(o_j)$ as input, and is implemented as a 2-layer MLP of size $h - 3$, where h is the hidden size and 3 is the output size (corresponding to 3 different output categories). We found that varying h in the range 4 to 32 has little effect on the performance. Hence we report performance with the $h = 4$ to reduce computational costs.

The baseline architecture uses the same CNN module as the architecture with low-dim comparator. The baseline model concatenates the CNN output o_i and o_j and feed the concatenated vector into a MLP. We performed hyper-parameter search of the MLP with number of layers ranging from 1 to 4, and with hidden unit sizes from 32 to 64. We found that $64 - 64 - 3$ is the best performing architecture.

D PGM DATASET

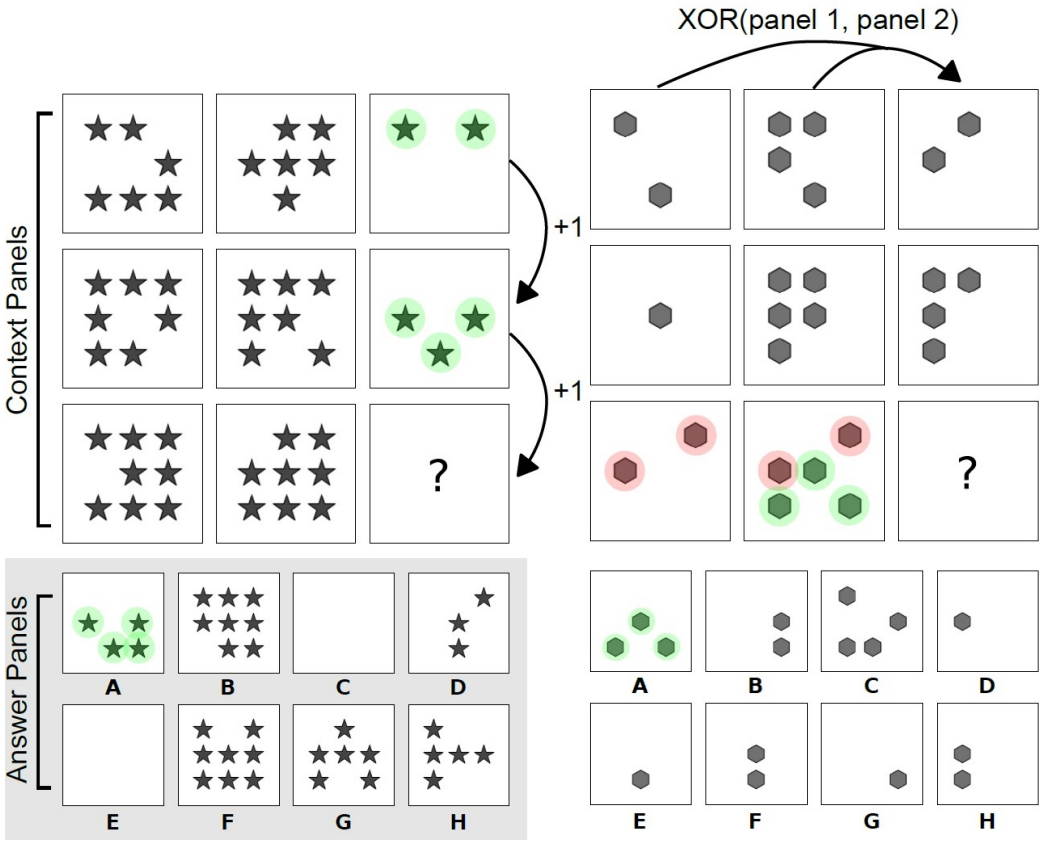
In this section we give a brief description of PGM dataset. For more details please refer to Barrett et al. (2018). PGM contains 8 context panels and 8 answer panels. The 8 context panels for a 3×3 diagram matrix. One is asked to pick the answer the logically fit with the context panels. In PGM, logic relations can exist in both rows and columns of the diagram matrix. Figure 6a and 6b show two examples from the PGM dataset(Image courtesy Barrett et al. (2018)). The first example contains a 'Progression' relation of the number of objects across diagrams in columns. The second examples contains a 'XOR' relation of position of objects across diagrams in rows. The objects in PGM datasets have different attributes such as colour and size. In total five types of relations can be present in the task: $\{Progression, AND, OR, XOR, ConsistentUnion\}$.

E PGM ARCHITECTURE CONFIGURATIONS

In this section we describe detailed configuration of the PGM architecture with low-dim comparators, and the baseline model MLRN-P, which is an augmented version of MLRN Jahrens & Martinetz (2020). The descriptions here is supplementary to descriptions in section 2.4 of the main paper and figure 3b.

The CNN module is a 4-layer CNN of filter number $32 - 32 - 64 - 64$ followed by a 2-layer MLP of hidden size $256 - 128$. Each CNN layer has stride value 2 and padding value 1. The output of each CNN is the object representation o_i of raw image input x_i. Following Barrett et al. (2018) we attach to o_i a position tag to indicate its position in the diagram matrix. The tagged object representation is then projected onto $K = 512$ 1-dimensional manifold for parallel attribute comparison. Next we describe the comparator module f (equation 1. As shown in figure 3b, there two hierarchical projection comparison. For the first level, we found that implementing c as a simple vector difference module achieve best results, which means $c = p(o_i) - p(o_j)$. This reflects the fact that comparison between diagrams is directional, such as increase in number of objects from one diagram to the other. We implement g in equation 1 as a residual MLP of 4 layers with hidden size $2048 - 2048 - 2048 - 796$. The output from each pairwise comparison of diagrams in a row/column are concatenated to form the relation embedding for that row/column. For the second level we implement c as absolute difference, which means $c = |p(o_i) - p(o_j)|$. This works better because relation comparison is less directional. For example the difference between relation of increasing number of objects and relation of increasing sizes of objects should be the same when the compared diagram is swapped. For the second level we implement g as 3 layer MLP of hidden sizes $1024 - 512 - 1$, which directly output the predicted similarity score between two rows/columns. For predicting the correct answer candidate we follow Barrett et al. (2018) by applying a softmax function to scores produced by comparing each answer rows/columns with context row/columns to produce scores for each answer candidates. For meta-target prediction we sum all context row/column embedding and process it with a 3-layer MLP of hidden size $1024 - 512 - 12$, where 12 is the meta-target label size. We have attached source code in the submission for clarity.

The baseline model MLRN-P is modified from MLRN Jahrens & Martinetz (2020). We have performed three architecture modifications for fair comparison with our model. Firstly we inject the prior knowledge of relations existing only in rows/columns into the model. The first level of Relation Networks compare diagrams within row/columns and the second level of Relation Networks compare



(a) (b)

Figure 6: Two examples in PGM dataset. (a) task contains a 'Progression' relation of the number of objects across diagrams in columns while (b) contains a 'XOR' relation of position of objects across diagrams in rows. For both task "A" is the correct answer.

row/colum embeddings. Secondly we swap MLP in the original MLRN with residual MLP, which is shown to improve performance slightly in our model. Thirdly we pretrained CNN module with Beta-VAE Higgins et al. (2017) for fair comparison with our model.

F Training Details

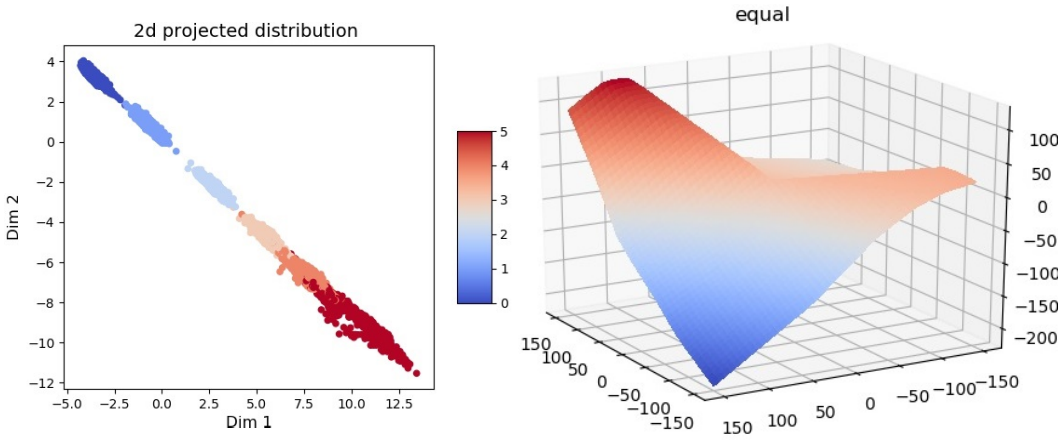
In this section we describe the training details for all three experiments. We use PyTorch[1] for implementation. For gradient descent optimiser, we use RAdam Liu et al. (2019), an improved version of the Adam optimiser. For all 3 experiments we use learning rate 0.001 and betas (0.9,0.999). We used 2 Nvidia Geforce Titan Xp GPUs for training all models. For Maximum of a set and visual object comparison, we set batch size to be 64. For PGM we found a larger batch size of 512 slightly improves result. For Maximum of the set and visual object comparison we set training epochs to be 20. For PGM we trained for 50 epochs. For visual object comparison and RPM tasks we pre-trained CNN as the encoder of a Beta-VAE Higgins et al. (2017). We follow standard training procedures of Beta-VAE as described in the paper, and set the β value to be 1.

G Additional Plots

In section 3.4 of the main paper we show plots of projected distribution and comparator's function landscape for position comparison task. In this section we show the same plots for comparison tasks for other attributes, which are sizes and colour intensity. Figure 7 shows the plot for size comparison while figure 8 shows the plot for colour intensity comparison. Training range is the lower 60% of the

[1] https://pytorch.org/

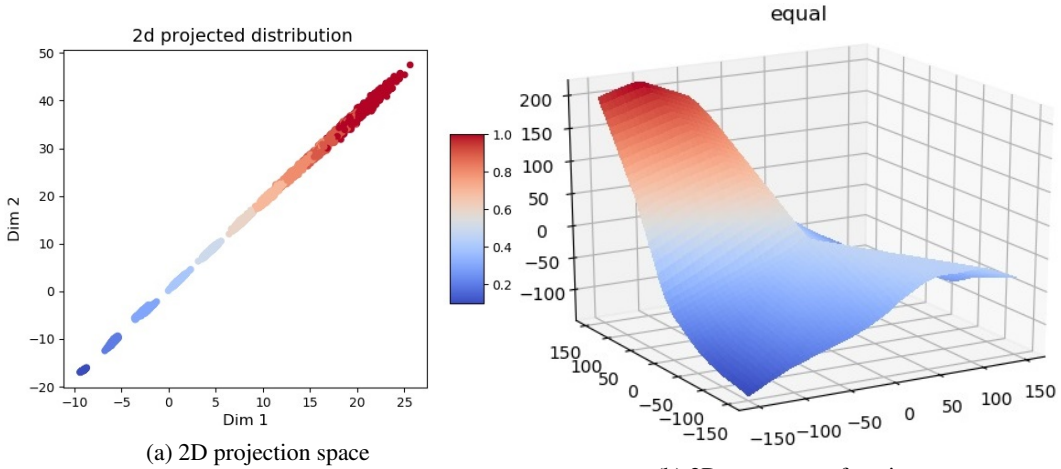
colour bar while test range is the upper 40%. The observations stated in section 3.4 also holds true for these attributes. For size and colour intensity comparison tasks, the projected distribution plots show that the test data is more clustered than that of spatial position comparison. This shows that size and colour intensity attributes can be learnt better with a bespoke CNN perception module than spatial position attributes. This is supported by that models trained for these two attributes achieved higher **o.o.d** test accuracies.



(a) 2D projection space

(b) 2D comparator function

Figure 7: (a) shows a scatter plot of the 2D projected distribution of objects in the task of comparing sizes of objects in the image. X-axis and Y-axis are 2 dimensions of the projection manifold. Latent size variable (possible values are [0,1,2,3,4,5]) is indicated by colour. (b) plots the comparator's function landscape for "equal" output unit (before softmax) in the space of vector differences between 2D projections of objects.



(a) 2D projection space

(b) 2D comparator function

Figure 8: (a) shows a scatter plot of the 2D projected distribution of objects in the task of comparing colour intensity of objects in the image. X-axis and Y-axis are 2 dimensions of the projection manifold. Latent colour intensity variable (possible values are [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0]) is indicated by colour. (b) plots the comparator's function landscape for "equal" output unit (before softmax) in the space of vector differences between 2D projections of objects.

H QUANTITATIVE EVIDENCE FOR LOW DIMENSION

In section 3.4 and G we showed visual illustration for 1-dim and 2-dim comparator's function landscape for the visual object comparison task to illustrate how out-of-distribution test data lies outside of the training function landscape. It is difficult to visualise function landscape of higher dimensions. In this section we discuss how to use Hausdorff Distance, which measures distance between sets, to measure the differences between projected distance from training and **o.o.d** test set.

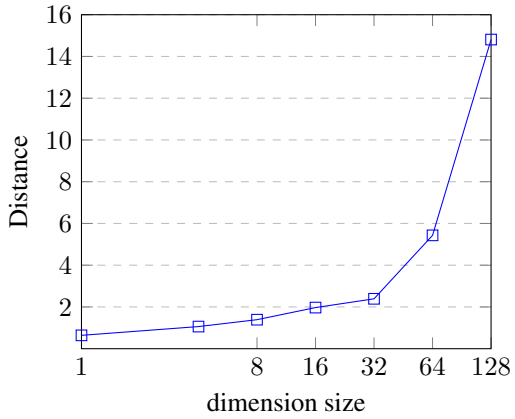


Figure 9: Hausdorff distance between training set and test set of vector differences between projected object representations (normalised) for the visual object comparison task.

Hausdorff distance measures how far subsets of a metric space are from each other. Given two sets $A = \{a_1, a_2, \ldots, a_N\}$ and $B = \{b_1, b_2, \ldots, b_N\}$ of the same metric space. The Hausdorff distance of A and B is computed as:

$$d_H(A, B) = \max \left\{ \sup_{a \in A} \inf_{b \in B} \mathrm{d}(a, b), \sup_{b \in B} \inf_{a \in A} \mathrm{d}(a, b) \right\} \tag{4}$$

where d is a distance measure of the metric space. Informally Hausdorff distance measures the greatest of all the distances from a point in one set to the closest point in the other set.

We apply Hausdorff distance to measure distances between sets of vector differences from training and test set. The training set S_{train} contains vector difference $p(e_i) - p(e_j)$ for all pairs of inputs $e_i, e_j \in \mathcal{D}_{train}$ from the training distribution $\mathcal{D}_{train}$. Recall that p is the projection module. The test set S_{test} contains vector differences of pairs from the test data distribution $\mathcal{D}_{test}$. We measure the Hausdorff distance between training and test set $d_H(S_{train}, S_{test})$ for different projection dimensions, and show the result in Figure 9. To compare in the same scale, we normalise vector differences with the mean and standard deviation of the training set. It can be observed that as the dimensionality of the projection space increase, there is increased distance between S_{train} and S_{test}. This quantitatively shows that the observations in Section 3.4 extends beyond 2-dimensional cases.

I Ablation Studies

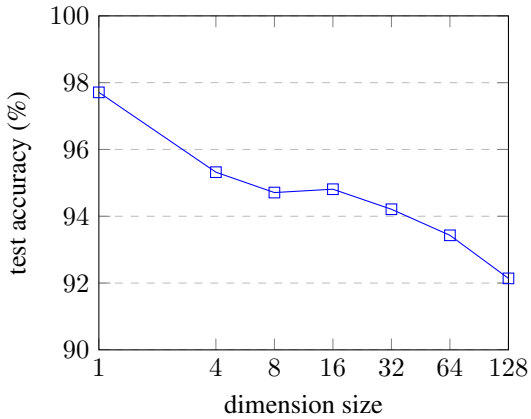


Figure 10: Projection dimension ablation study for visual object comparison tasks.

We performed ablation studies of different hyper-parameters in our experiments. and show the results in Figure 10 and Table 4. Figure 10 shows the plot of test accuracy against different dimension size

of the projection functions for the visual object comparison tasks. It can be observed that increasing dimension sizes (x-axis) reduce the test performance. This further validate the claim in our paper. Table 4 shows the test accuracy of architectures with different number of projection functions for PGM task. The number of projection functions is indeed another hyper-parameter to tune, but we think the improved performance is definitely worth it. Table 5 shows the test accuracy on the extrapolation split of PGM dataset with different pre-training for the perception module. We picked the hyper-parameters giving the best extrapolation test accuracy.

Number of Projector Functions	Accuracy
128	20.1
256	24.2
512	25.9

Table 4: Ablation Study on Number of Projector/Comparator functions for PGM task.

Pre-trained Encoder	Accuracy
β-VAE (β=4, dim=64) Steenbrugge et al. (2018)	20.1
β-VAE (β=4, dim=128)	24.2
β-VAE (β=1, dim=128) (OURS)	25.9

Table 5: Ablation study on different pre-trained encoders for PGM task.

J A BRIEF DISCUSSION ON LOW-DIMENSIONAL EMBEDDING

In machine learning there is a rich history of learning low-dimensional embedding, ranging from more classic methods like Principal Component Analysis(PCA) (Wold et al., 1987) and Independent Component Analysis(ICA) (Comon, 1994) to recent neural-network-based methods like Variational Auto-Encoders(VAE) (Kingma & Welling, 2013) and Generative Adversarial Networks(GAN) (Goodfellow et al., 2014). VAE and GAN have been frequently used to unsupervisedly learn low-dimensional representations from an unlabelled dataset, and use the learnt representations to boost performance of supervised downstream tasks, such as image classification and natural language processing. However, most researches on VAE and GANs put more emphasis on understanding how disentanglement affect downstream task performance, but rather than low-dimensions, such as Beta-VAE (Higgins et al., 2017) and its variants. Arguably disentanglement can be considered as splitting the latent distribution into multiple orthogonal distributions of lower dimensions, but to our best knowledge this view has not been discussed in the literature. Moreover, there is limited work on how low-dimension representations learnt by neural generative models help with **o.o.d** generalisation. To our best knowledge, the only work disucssing this is by Zhao et al. (2018), who systematically study the generalisation capability of GANs and VAEs using experimental methods from cognitive psychology.