

Multi-Label Contrastive Learning for Abstract Visual Reasoning

Mikołaj Małkiński, Jacek Mańdziuk

Faculty of Mathematics and Information Science
Warsaw University of Technology, Warsaw, Poland
mikolaj.malkinski@gmail.com, mandziuk@mini.pw.edu.pl

Abstract

For a long time the ability to solve abstract reasoning tasks was considered one of the hallmarks of human intelligence. Recent advances in application of deep learning (DL) methods led, as in many other domains, to surpassing human abstract reasoning performance, specifically in the most popular type of such problems - the Raven’s Progressive Matrices (RPMs). While the efficacy of DL systems is indeed impressive, the way they approach the RPMs is very different from that of humans. State-of-the-art systems solving RPMs rely on massive pattern-based training and sometimes on exploiting biases in the dataset, whereas humans concentrate on identification of the rules / concepts underlying the RPM (or generally a visual reasoning task) to be solved. Motivated by this cognitive difference, this work aims at combining DL with human way of solving RPMs and getting the best of both worlds. Specifically, we cast the problem of solving RPMs into multi-label classification framework where each RPM is viewed as a multi-label data point, with labels determined by the set of abstract rules underlying the RPM. For efficient training of the system we introduce a generalisation of the Noise Contrastive Estimation algorithm to the case of multi-label samples. Furthermore, we propose a new sparse rule encoding scheme for RPMs which, besides the new training algorithm, is the key factor contributing to the state-of-the-art performance. The proposed approach is evaluated on two most popular benchmark datasets (Balanced-RAVEN and PGM) and on both of them demonstrates an advantage over the current state-of-the-art results. Contrary to applications of contrastive learning methods reported in other domains, the state-of-the-art performance reported in the paper is achieved with no need for large batch sizes or strong data augmentation.

1 Introduction

Abstract visual reasoning tasks are considered a widely-accepted way of measuring human intelligence. The most popular example of such task are Raven’s Progressive Matrices (RPMs) [25, 26], where one is required to identify abstract relations between visually simple objects and their attributes (see Figure 1). The importance of RPMs in measuring human intelligence is justified by the fact that their solving requires an incremental strategy for inducing regularities in each problem [3]. Previous works identified a major gap between the performance of Machine Learning (ML) algorithms and humans [28, 37], which sparked interest in these problems within the deep learning (DL) community. In effect, human performance advantage has quickly vanished along with development of recent DL models [35].

However, the way ML algorithms solve RPMs leaves something to be desired. In order to solve these tasks, humans are required to come up with a strategy which correctly identifies all the underlying rules and differentiates them from distracting features, while the goal of the vast majority of ML approaches is to focus on selecting a correct answer, without explicit explanation for the output.

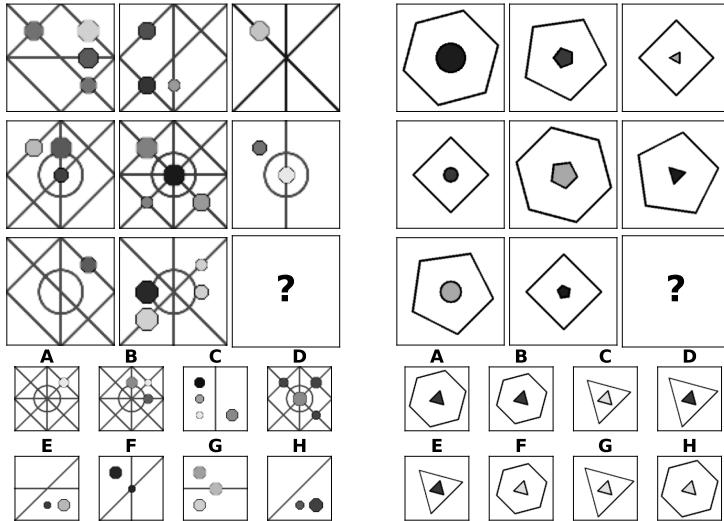


Figure 1: Solving RPMs requires to identify abstract relationships hidden behind random visual distractors and contrast possible answers to select the one which fits best. Left RPM, although has more visual details, is governed by only a single rule (row-wise AND applied to shape position), whereas perceptually simpler right RPM contains 8 distinct relations applied to both outer and inner structures. The examples come from the PGM and Balanced-RAVEN datasets, respectively and in both cases the correct answer, which is to be placed in the bottom-right panel, is A.

Such a direct optimization problem formulation encourages neural models to simplify the solution process and rely on biases instead of understanding the internal problem structure. Indeed, such pathology was identified for the RAVEN dataset [37], where networks were able to arrive at a correct answer just by comparing the set of possible choices [14]. Similar problems were noted throughout the visual reasoning literature [39, 16].

A large body of cognitive literature identified contrastive mechanisms and the ability to create analogies as key ingredients for adaptive problem solving [7, 13, 29], which allow to apply previous experiences to novel domains. Naturally, there have been various approaches to replicate such behaviour in neural models either in the form of an explicit contrastive module [38] or through the way the data is presented during training [11], which in both cases resulted in notable improvements. Moreover, it was demonstrated that when models were additionally trained to predict a symbolic explanation for their answers (so-called *auxiliary training*), their generalisation capabilities increased substantially [28]. Nonetheless, despite notable positive impact of the auxiliary training, the topic remains underexplored.

Motivation and Contribution. In order to build models which *understand* how to solve RPMs, we look for alternative training approaches. Encouraged by the effectiveness of both contrastive learning methods and symbolic explanations, we seek to develop a novel auxiliary training method for abstract visual reasoning tasks. We aim to exploit two recurring themes present in human approaches: the contrastive mechanism which differentiates between correct and wrong answers to the RPMs and the inherent ability to first identify all the abstract relations defining a given RPM and then use them to select the answer. The main contribution of this work is four-fold:

- Approaching the problem of solving RPMs by casting it into a multi-label classification framework, where labels are determined by the underlying abstract rules. This viewpoint allows us to explore a novel training method for abstract visual reasoning tasks.
- Devising a new formulation of the Noise Contrastive Estimation (NCE) learning algorithm for the case of multi-label samples, whose application imitates human approach to solving RPMs.

- Proposition of a new sparse rule encoding scheme for RPMs which provides a more explicit rule representation compared to the method used in prior works.
- Integration of both contrastive and auxiliary training into a novel ML approach to solving abstract visual reasoning tasks, which sets new state-of-the-art results on two major benchmark datasets – Balanced-RAVEN [14] and PGM [28].

2 Related work

Raven’s Progressive Matrices. Although RPMs are characterised by rather simple visual representation, solving them is often a challenging task, as it requires to correctly identify all abstract relations between the component RPM images. In order to measure generalisation ability of neural modules in relational reasoning problems, Santoro et al. (2018) introduced the dataset of Procedurally Generated Matrices (PGM), which contains RPM problems divided by the authors into train and test sets. An important part of the PGM dataset are meta-target annotations, which encode the relations between objects and their attributes in a given RPM. Models trained to additionally predict these meta-targets by means of an auxiliary training, were shown to posses stronger generalisation capabilities compared to those that did not employ such an auxiliary training.

The topic of meta-annotations was further extended in the RAVEN dataset [37], which contains additional structural annotations and RPMs with highly compositional structure. However, as reported in previous works, in the case of RAVEN data the auxiliary training seems to have little to no positive impact [37, 38]. Our investigations on Balanced-RAVEN dataset show that a more explicit rule encoding scheme can mitigate this problem.

A recent paper [14] put in question the abstract visual reasoning capabilities of reported models, by highlighting a major flaw in the process of generating candidate answers in RAVEN. Due to an unintended bias in the generated set of possible choices, models trained purely on answers were still able to select a correct solution. To mitigate this problem, the dataset Balanced-RAVEN was proposed, in which the answers are generated in an unbiased way [14].

Initial reports of machine RPM solving demonstrated a notable gap between humans and ML algorithms, which stimulated active research in this area. A number of methods were designed for solving RPMs, which aim to identify relations between sets of objects [28, 40] using a Relation Network [27], reason with multi-layer multiplex graph neural network [33] or discover compositional representations using a scattering transformation [35]. Other works investigate human-inspired approaches, by measuring feature differences [21] or exploiting the hierarchical structure of RPMs [14].

Contrastive representation learning. Previous studies shown that the ability of making analogies and contrasting experiences is a key ingredient of human intelligence [7, 13, 15, 20]. Consequently, these concepts were also implemented in ML systems solving abstract reasoning problems. Hill et al. (2019) discussed how to make analogies by contrasting abstract relational structure, while Zhang et al. (2019) incorporated contrast directly in the model architecture. Motivated by these findings, we propose another approach to abstract reasoning, by incorporating contrastive mechanism directly in the objective function, similarly to contrastive learning implementations in other domains.

Among contrastive learning algorithms, the most related to our work is the Noise Contrastive Estimation (NCE) method [8, 23], which was successfully utilized across various domains, such as image recognition [4], natural language processing [22] or reinforcement learning [19, 31]. NCE aims at building congruent representations for semantically related samples (positive pairs) and dissimilar representations for unrelated observations (negative pairs). It was shown that the concept of learning representations with NCE (as well as with other contrastive learning methods) is related to mutual information (MI) maximization principles [24, 12, 32].

Contrastive learning is particularly useful in semi-supervised pre-training methods, where the availability of labeled data in downstream tasks is scarce [4, 5, 6, 10]. A recent work [17] has shown that contrastive learning can outperform classical approaches also in fully supervised setting (with cross-entropy) and is characterised by greater robustness and stability to hyperparameter selection.

This superior performance is a result of employing multiple positive pairs, which was further shown to increase MI lower bound by considering MI estimation as multi-label classification [30]. Despite this multi-label viewpoint, the framework proposed by Song and Ermon, as well as the vast majority of prior works, focus on multi-class classification and don't discuss applicability of contrastive methods to multi-label problems. Our work aims at bridging this gap by defining a general contrastive learning framework which supports multi-label samples.

3 Multi-Label Contrastive Learning

Motivated by successful applications of contrastive learning methods in other domains, we aim to extend it for multi-label setting and investigate its usage in learning representations of abstract visual reasoning problems. We start by proposing a general multi-label learning framework and then discuss its applicability to solving RPMs.

3.1 Preliminaries

Our method builds on the foundations of Supervised Contrastive Learning proposed by Khosla et al. (2020) and extends it to support multi-label data. Given a randomly sampled batch $\{x_i, y_i\}_{i=1\dots N} \in \{\mathcal{X} \times \mathcal{Y}\}$ of size N , the base method consists of the following integral components:

- A data augmentation module, which transforms image x_i into two randomly augmented views \tilde{x}_{2i} and \tilde{x}_{2i-1} , leading to an extended batch $\{\tilde{x}_i, \tilde{y}_i\}_{i=1\dots 2N}$ such that $\tilde{y}_{2i} = \tilde{y}_{2i-1} = y_i$. It should be noted that while this step is critical for Supervised Contrastive Learning its usage in our framework is optional.
- An encoder network f_θ which forms latent representations of the augmented views, defined as $h_i = f_\theta(\tilde{x}_i)$. The representations obtained from the encoder are ℓ_2 -normalized, which encourages learning from hard negatives and hard positives [17] and simplifies the final linear classification task by aligning the features from positive pairs and uniformly distributing them on the hypersphere [34].
- A projection network g_ϕ , which maps feature representation into a lower-dimensional vector $z_i = g_\phi(h_i)$ suitable for computation of the contrastive loss. The representations obtained from the projection network are ℓ_2 -normalized, which this time allows to measure similarity between two vectors based on their dot-product. The projection network is realized by a non-linear function, an MLP with a single hidden layer, which was shown to be of critical importance [4, 5, 6].

Both f_θ and g_ϕ are optimized jointly with respect to a Supervised Contrastive Loss, defined as follows:

$$\mathcal{L}^{\text{sup}} = \sum_{i=1}^{2N} \mathcal{L}_i^{\text{sup}} \quad (1)$$

$$\mathcal{L}_i^{\text{sup}} = \frac{1}{2N_{\tilde{y}_i} - 1} \sum_{j=1}^{2N} \mathbb{1}_{i \neq j} \cdot \mathbb{1}_{\tilde{y}_i = \tilde{y}_j} \cdot \mathcal{L}_{i,j}^{\text{sup}} \quad (2)$$

$$\mathcal{L}_{i,j}^{\text{sup}} = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{i \neq k} \cdot \exp(\text{sim}(z_i, z_k)/\tau)} \quad (3)$$

where $N_{\tilde{y}_i}$ is the number of samples in a given mini-batch with the same label as the anchor i , $\mathbb{1}_B \in \{0, 1\}$ is an indicator which evaluates to 1 iff B is true, $\text{sim}(z_i, z_j) = z_i \cdot z_j$ is a similarity measure defined as dot-product and $\tau > 0$ is a constant temperature parameter.

After this pre-training stage, the weights of the encoder f_θ are frozen and the projection network g_ϕ is replaced with a randomly initialized linear classification head, which is trained with cross entropy on the downstream task. This procedure, commonly referred to as the linear evaluation protocol [24, 1], provides a simple way to measure the quality of learned representations.

3.2 Multi-Label Contrastive Loss

In the default setting, the Supervised Contrastive Loss supports multi-class samples, i.e. $y_i \in \mathcal{Y}$. In order to extend it to multi-label samples $\{x_i, Y_i\}$ such that $Y_i \subset \mathcal{Y}$, we propose a novel objective function, the Multi-Label Contrastive Loss, which is defined as follows:

$$\mathcal{L}^{\text{mlc}} = \sum_{i=1}^{2N} \mathcal{L}_i^{\text{mlc}} \quad (4)$$

$$\mathcal{L}_i^{\text{mlc}} = \frac{1}{2N_{\tilde{Y}_i} - 1} \sum_{j=1}^N \mathbb{1}_{i \neq j} \cdot \mathbb{1}_{\tilde{Y}_i \cap \tilde{Y}_j \neq \emptyset} \cdot \mathcal{L}_{i,j}^{\text{mlc}} \quad (5)$$

where $N_{\tilde{Y}_i} = |\{\tilde{Y}_j \mid \tilde{Y}_i \cap \tilde{Y}_j \neq \emptyset\}_{j=1 \dots 2N}|$ is the number of samples in a given mini-batch which share at least one label with the anchor i and $\mathcal{L}_{i,j}^{\text{mlc}} = \mathcal{L}_{i,j}^{\text{sup}}$.

The difference when compared to eq. (2) lies in the definition of positive pairs for an anchor i . While the base formulation defines the set of positives as those samples with exactly the same label, eq. (5) defines it as those samples which share at least one label. Our formulation preserves key properties of the base objective, i.e. 1) aggregates an arbitrary number of positive samples in the numerator and 2) increases contrastive strength by using all negative samples in the denominator. At the same time, the modified definition of positive pairs allows to handle samples with multiple labels. Analogously to the base method, our framework relies on the inner product of ℓ_2 -normalized vectors as a measure of similarity.

3.3 Adaptation to RPMs

In order to utilise the above-proposed Multi-Label Contrastive Learning (MLCL) framework for solving RPMs, let us first observe that each RPM can be naturally viewed as a multi-label sample, where labels correspond to the rules governing the RPM. Let us consider a mini-batch $\{\mathcal{M}_i\}_{i=1 \dots N}$ of size N , where $\mathcal{M}_i = \{X_i, Y_i, k_i\}$ represents the whole RPM instance composed of a set of 16 images X_i , i.e. 8 context panels and 8 choice panels (out of which only one correctly completes the matrix). $Y_i \subset \mathcal{Y}$ is a set of associated rules such that $1 \leq |Y_i| \leq N_{\mathcal{Y}}$ and $k_i \in \{1 \dots 8\}$ is an index of the correct answer. Here, \mathcal{Y} is the set of all possible rules and $N_{\mathcal{Y}}$ is a dataset-dependent maximal number of rules for a single RPM.

Data augmentation. Previous works on solving RPMs do not report the usage of data augmentation. On the other hand, augmentation was highlighted as a fundamental component of NCE-based learning in other domains, e.g. by Chen et al. (2020). In order to conduct a meaningful comparison with prior literature, in the experiments both setups *with* and *without* augmentation are considered. In the former case (with augmentation), for each RPM in the mini-batch we apply two randomly selected augmentations, which gives a mini-batch of $2N$ RPMs $\tilde{\mathcal{M}}_i = \{\tilde{X}_i, \tilde{Y}_i, \tilde{k}_i\}_{i=1 \dots 2N}$, where \tilde{X}_{2i} and \tilde{X}_{2i-1} are both obtained by augmenting X_i . All augmentations preserve both the underlying rules and index of the correct answer, hence: $\tilde{Y}_{2i-1} = \tilde{Y}_{2i} = Y_i$ and $\tilde{k}_{2i-1} = \tilde{k}_{2i} = k_i$. Since RPMs consist of greyscale images, we cannot rely on augmentation methods popular in image recognition. Instead, simple transformations which rearrange or rotate images are used, as showcased in Figure 2. If augmentation is not applied a base mini-batch $\{\mathcal{M}_i\}_{i=1 \dots N}$ is used.

Contrastive pre-training. For each RPM, we construct 8 different matrices by filling in the remaining context panel with each of the choice panels. For each sample $\tilde{\mathcal{M}}_i$, this gives us a single RPM which satisfies the rules \tilde{Y}_i and 7 RPMs which satisfy fewer rules, because of the incorrectly chosen answer. We arrive at a batch divided into two components, one containing correct RPMs with their rules $\{\tilde{x}_i, \tilde{Y}_i\}_{i=1 \dots 2N}$ and the other one composed of incorrect RPMs $\{\tilde{x}'_{i,l}\}_{i=1 \dots 2N, l=1 \dots 7}$. This division allows us to consider the incorrectly completed RPMs as additional negative samples,

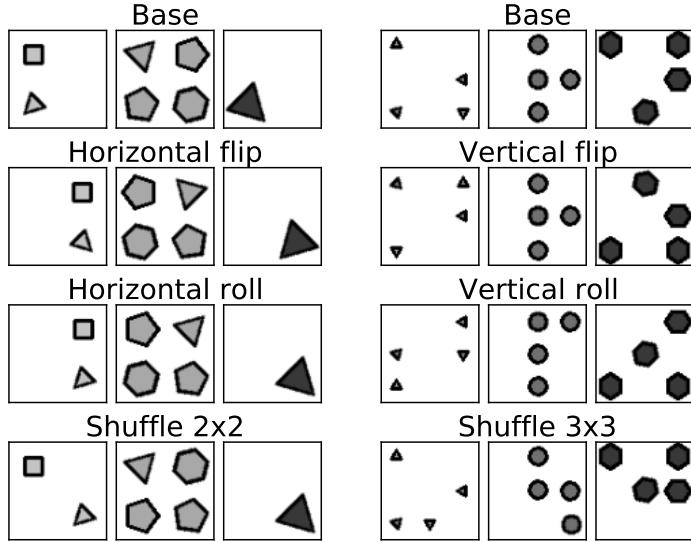


Figure 2: Augmentation of RPMs from the Balanced-RAVEN dataset. Selected transformation is applied in the same way to all images in a given RPM. The data augmentation module applies a randomly selected combination of presented methods with additional random rotation and transposition, using implementation from [2]. For clarity, we only depict different views of a single row for two RPMs belonging to configurations **2x2Grid** (left part) and **3x3Grid** (right part), respectively.

for which:

$$\mathcal{L}_{i,j}^{\text{mlc}} = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{i \neq k} \cdot \exp(\text{sim}(z_i, z_k)/\tau) + \Sigma_{i,k}} \quad (6)$$

$$\Sigma_{i,k} = \sum_{l=1}^7 \exp(\text{sim}(z_i, z'_{k,l})/\tau) \quad (7)$$

We optimize both f_θ and g_ϕ with respect to the Multi-Label Contrastive Loss defined in eqs. (4)-(7).

Auxiliary training. We support the contrastive pre-training with an auxiliary loss \mathcal{L}^{aux} introduced in [28]. For this purpose, we employ a rule discovery network ρ (implemented as an MLP with a single hidden layer), which transforms outputs of the encoder f_θ into a meta-target prediction $\pi_i = \rho(\{h_i\} \cup \{h'_{i,l}\}_{l=1 \dots 7}) \in \mathbb{R}^d$, where d is the dimension of meta-target encoding. We compare two setups for the rule encoding:

- *Dense* - multi-hot encoding scheme introduced in [28] and [37]. It encodes each rule as a binary string of fixed length ($d = 12$ for PGM and $d = 9$ for Balanced-RAVEN) and performs logical OR operation on the whole set of rules.
- *Sparse* - our proposed scheme, which encodes each rule as a one-hot vector of fixed length equal to the number of unique relations in the dataset ($d = 50$ for PGM and $d = 38$ for Balanced-RAVEN). Similarly to dense encoding it performs logical OR operation on the whole set of rules.

Intuitively, the advantage of proposed sparse encoding over the dense encoding used in prior works stems from information lossless OR operation, i.e. from sparse representation one can recover which rules were encoded, which is not always the case for dense encoding (e.g. when a single object is governed by multiple rules, or the same relation is applied to different objects). The difference between the two encodings is further discussed in supplementary material. Since sparse encoding provides a more explicit training signal it is the default encoding method for MLCL. Rule predictions

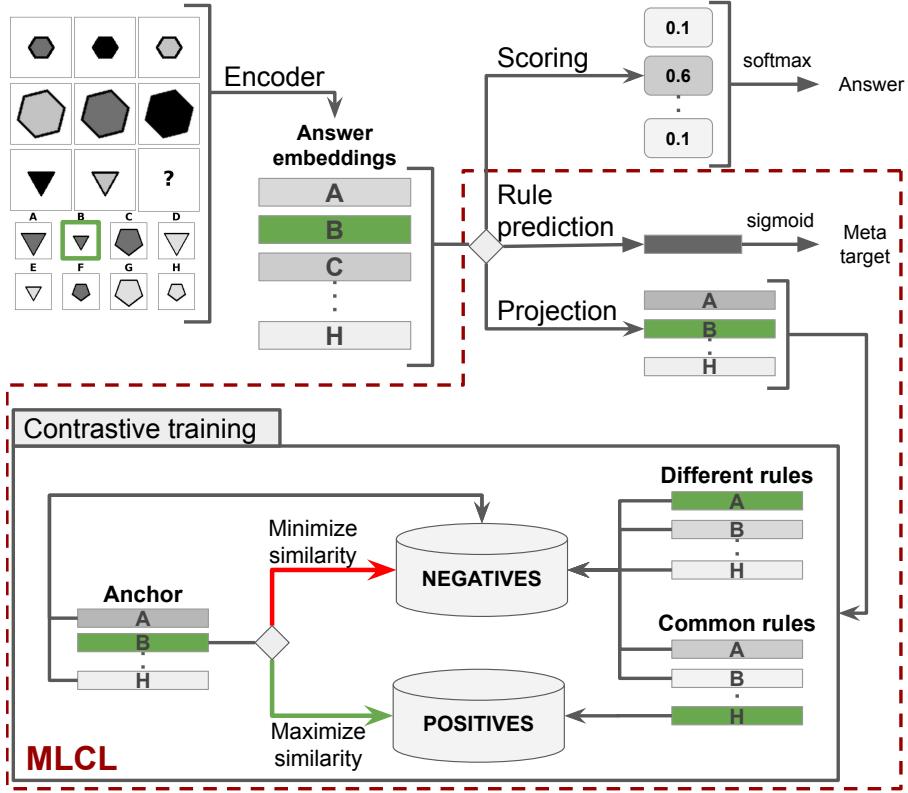


Figure 3: In all training setups, we start by filling in the context panels with each choice panel. In each case, we generate an embedding with an abstract reasoning encoder network (SCL, HriNet or CoPINet). The embeddings are used as an input to 1) a scoring module which predicts the answer (supervised training, denoted as CE), 2) a rule prediction module which predicts rules as encoded meta-target (auxiliary training, denoted as AUX) and 3) a projection network which maps them into a lower-dimensional representation suitable for computing contrastive loss. The proposed approach combines both auxiliary and contrastive training into a joint learning framework – MLCL.

are activated using a sigmoid unit and the loss is calculated using binary cross-entropy. We define the Multi-Label Contrastive Loss for solving RPMs as:

$$\mathcal{L} = \gamma \mathcal{L}^{\text{mlc}} + \beta \mathcal{L}^{\text{aux}} \quad (8)$$

where γ and β are balancing factors. For simplicity, we set $\gamma = 1$ and $\beta = 10$ in all main experiments. Additional results are presented in supplementary material.

Linear evaluation. After the contrastive pre-training step, we discard the projection head g_ϕ , freeze the parameters of the encoder f_θ and attach a simple linear scoring head s_ψ with a single output neuron. For each RPM problem \mathcal{M}_i , we employ f_θ to generate the matrix representations $\{h_i\} \cup \{h_{i,l}\}_{l=1\dots 7}$, calculate a score using the scoring head s_ψ and apply softmax to produce a probability distribution over the set of possible answers. Using the estimated probability, we optimize the scoring head with a standard cross-entropy loss and keep weights of the encoder network frozen.

4 Experiments

We compare our method (MLCL) with a fully supervised framework for solving RPMs used throughout the literature [28, 37], referred to as CE. In CE both the encoder network f_θ and s_ψ are used

and optimized in an end-to-end manner with respect to the cross-entropy objective function. Since our contrastive training method is inherently based on the description of the RPM rules, we support the supervised training with the same auxiliary loss \mathcal{L}^{aux} and rule discovery network ρ , obtaining:

$$\mathcal{L} = \mathcal{L}^{\text{ce}} + \beta \mathcal{L}^{\text{aux}} \quad (9)$$

where β is a balancing coefficient. This enhanced training is referred to as CE+AUX. Following the setup from [28], we set $\beta = 10$ in all experiments. In order to comprehensively compare our method with the baseline, in the experimental evaluation we consider three different state-of-the-art models for abstract visual reasoning and use two abstract reasoning RPM benchmarks.

Datasets. Our main findings are demonstrated on the newest RPM benchmark set – Balanced-RAVEN [14], which contains visually diverse RPMs with highly compositional structure. Balanced-RAVEN is a modification of the original RAVEN dataset [37] which fixes the defect of biased choice panels. It contains novel relation types which are not present in PGM. Each RPM in Balanced-RAVEN can be governed by up to 8 different rules. Balanced-RAVEN is divided into seven distinct visual configurations. In the default setting each of them contains 10K problem instances.

Additionally, we support our studies with the experiments on large-scale PGM dataset [28], containing RPM problems with between 1 and 4 abstract rules. The main purpose of PGM is to test generalisation skills of ML models across various regimes. Each regime is characterised by explicitly defined differences between training and test data, which address various types of generalisation, e.g. interpolation or ability to reason about relations not seen in the training set. The dataset contains 1.42M RPM problems per regime.

Models. We use three different state-of-the-art models for solving RPMs as the encoder network f_θ : SCL [35], HriNet [14] and CoPINet [38]. In order to reduce the training time and memory consumption of HriNet, we replaced its ResNet backbone [9] in all three hierarchies with a simpler CNN architecture, analogous to the one used in the Wild Relation Network [28]. The details are presented in supplementary material.

Implementation details. We devoted a maximum of 50 training epochs with batch size of 256 and learning rate 0.003 for each model on the PGM dataset and 100 epochs with batch size of 128 and learning rate 0.002 for Balanced-RAVEN. Model parameters were optimized using the ADAM optimizer [18] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ until the maximal number of epochs was reached or the loss stopped improving on a validation set. All experiments were conducted using mixed-precision training on a single worker with 4 NVIDIA Tesla P100 GPUs, each with 16Gb of memory.

4.1 Results

We start experimental evaluation by comparing the proposed MLCL method with other training setups for three different ML abstract reasoning models on the Balanced-RAVEN dataset (see Table 1). For all three models MLCL with data augmentation excels the best results from the literature, and in the case of CoPINet by a significant margin. Furthermore, for all three models MLCL significantly outperforms the base setup (CE), which does not utilize the rule-related information in the training signal. This observation confirms that MLCL is able to absorb and efficiently utilize this additional structural information, which is not a common property, as shown by the CoPINet’s performance deterioration when trained with dense encoding.

For SCL and HriNet, MLCL (without augmentation) matches the performance of supervised training supported by auxiliary loss with sparse encoding and excels it by above 10 p.p. for CoPINet. We hypothesize that contrastive nature of our objective function (eqs. (4)-(7)) amplifies the benefits of architectural contrastive mechanisms in CoPINet.

Moreover, auxiliary training with sparse encoding (CE+AUX+sparse) always improves performance over the baseline CE setup (with $\beta = 0$ in eq. (9)). At the same time, CE+AUX+dense

Method	Average \pm Std	Test accuracy (%)							
		Center	2x2Grid	3x3Grid	L-R	U-D	O-IC	O-IG	
SCL	CE	82.8 \pm 0.7	99.0	80.3	74.7	83.9	84.4	87.3	69.8
	CE+AUX-dense	86.9 \pm 3.5	98.9	86.9	80.6	88.4	88.8	92.0	72.7
	CE+AUX-sparse	95.9 \pm 0.3	99.8	97.0	91.8	99.1	99.1	98.7	85.8
	MLCL	95.7 \pm 0.5	99.6	96.4	90.7	99.2	99.1	99.1	85.9
	MLCL+AUG	96.8 \pm 0.4	99.8	97.6	93.8	99.3	99.4	99.1	88.9
	[35]	95.0	99.0	96.2	89.5	97.9	97.1	97.6	87.7
HriNet	CE	50.6 \pm 1.3	66.3	41.0	35.6	56.0	54.9	58.5	41.8
	CE+AUX-dense	51.6 \pm 1.5	75.7	41.2	36.5	55.0	55.4	57.0	40.2
	CE+AUX-sparse	57.0 \pm 1.6	84.5	43.2	36.9	65.4	66.1	60.5	42.6
	MLCL	56.2 \pm 1.5	78.9	44.5	38.2	64.4	64.1	60.5	42.9
	MLCL+AUG	65.7 \pm 2.4	94.4	57.3	46.8	75.3	76.0	64.2	45.8
	[14]	63.9	80.1	53.3	46.0	72.8	74.5	71.0	49.6
CoPINet	CE	44.8 \pm 0.8	53.6	35.9	30.2	51.0	52.1	51.1	39.9
	CE+AUX-dense	32.2 \pm 1.9	37.1	30.1	27.6	33.4	35.8	33.3	28.0
	CE+AUX-sparse	46.8 \pm 2.3	52.7	37.6	34.2	52.6	55.7	55.5	39.0
	MLCL	57.1 \pm 0.4	68.5	42.8	36.3	68.8	71.4	68.7	42.9
	MLCL+AUG	58.0 \pm 0.4	68.3	44.6	38.9	69.0	71.7	69.0	44.4
	[35]	46.3	54.4	33.4	30.1	56.8	55.6	54.3	39.0

Table 1: Test accuracy on the Balanced-RAVEN dataset averaged across 4 random seeds. Results are reported for three different encoder networks and the following training setups: supervised with cross-entropy (CE), supervised augmented with auxiliary training with *dense* (CE+AUX-dense) or *sparse* (CE+AUX-sparse) encoding and Multi-Label Contrastive Learning *without* (MLCL) or *with* (MLCL+AUG) data augmentation. Average - denotes the mean accuracy for all configurations, L-R denotes (Left-Right), U-D (Up-Down), O-IC (Out-InCenter), O-IG (Out-InGrid). For each model, the last row presents the best result reported in the literature.

results in worse than CE performance for CoPINet, which aligns with the outcomes reported in [38], and yields only a slight improvement for both SCL and HriNet. Hence, for the Balanced-RAVEN dataset, the more explicit, sparse rule encoding scheme is overall beneficial, regardless of a particular ML model.

Table 2 presents results on the PGM dataset. Due to huge number of RPM instances, we compare results of the best-performing encoder network – SCL, with the best MLCL configuration – with augmentation. In the most demanding regime (H.O. Shape-Colour) all training methods achieve close-to-random results. MLCL+AUG achieves superior results in 2 regimes and in the remaining 5 cases best outcomes are accomplished by SCL supervised training supported with an auxiliary loss with either dense or sparse encoding. MLCL outperforms the base setup (CE) in 5 regimes. The results show that for the PGM dataset, none of the evaluated training setups clearly stands out. On the contrary – each method seems to present different kind of generalisation abilities. It should be noted, however, that MLCL is tested under linear evaluation protocol, without fine-tuning of the encoder network f_θ and therefore reaching the results comparable to fully supervised setups (or even outperforming them in 2 regimes) proves its strength and potential.

4.2 Ablation study

In the ablation study we further validate the role of MLCL framework as an auxiliary training method and analyze its contrastive properties on the Balanced-RAVEN dataset.

Data augmentation. Depending on the choice of the encoder network MLCL either achieves comparable performance (SCL and HriNet) or outperforms (CoPINet) supervised approaches even without augmentations (cf. Table 1). However, since most of the contrastive methods heavily rely on data augmentation, we analyze its influence on MLCL for solving abstract visual problems. The

Regime	Accuracy (%)											
	CE			CE+AUX-dense			CE+AUX-sparse			MLCL+AUG		
Val.	Test.	Diff.	Val.	Test.	Diff.	Val.	Test.	Diff.	Val.	Test.	Diff.	
Neutral	86.2	85.6	-0.6	87.6	87.1	-0.5	87.4	87.1	-0.3	71.0	71.1	+0.1
Interpolation	91.2	55.8	-35.4	97.9	56.0	-41.9	88.1	54.1	-34.0	93.2	70.9	-22.3
H.O. Attribute Pairs	56.4	40.8	-15.6	88.6	79.6	-9.0	80.8	63.6	-17.2	79.7	66.0	-13.7
H.O. Triple Pairs	78.2	64.5	-13.7	88.7	76.6	-12.1	77.5	64.0	-13.5	86.1	71.7	-14.4
H.O. Triples	78.6	27.0	-51.6	88.1	23.0	-65.1	86.0	30.8	-55.2	84.0	22.1	-61.9
H.O. Line-Type	87.6	15.1	-72.5	87.9	14.1	-73.8	93.5	17.0	-76.5	86.1	16.1	-70.0
H.O. Shape-Colour	96.9	12.7	-84.2	98.3	12.6	-85.7	98.0	12.7	-85.3	94.1	12.8	-81.3
Extrapolation	96.3	17.3	-79.0	99.1	19.8	-79.3	96.9	17.5	-79.4	83.0	21.9	-61.1

Table 2: Accuracy in all regimes of the PGM dataset with SCL as the encoder network. We use the same notation for training methods as in Table 1. For each regime, we report results on the validation set (Val.), test set (Test.) and their difference (Diff.). Training and validation sets have the same distribution, whereas the test set has different distribution, specific to a given regime.

results of using data augmentation reported in Table 1 (denoted as MLCL+AUG) show consistent improvement for all encoders. This aligns with observations reported throughout the contrastive representation learning literature. Most notably, combining our contrastive framework for solving RPMs with data augmentation reduces the error rate on Balanced-RAVEN to 3.2%, which sets the new state-of-the-art result. Additionally, we see a large performance gain for HriNet, which surpasses the original result reported in [14] even with a much weaker perceptual backbone.

Joint optimization. MLCL combines both contrastive and auxiliary losses, as shown in eq. (8). We have verified that using either one of its individual components alone is not sufficient for building strong representations. When using contrastive loss only, that is with $\beta = 0$, we observed a serious performance downgrade for all models: the accuracy on SCL dropped to 35.3%, on HriNet to 20.7% and on CoPINet to 19.0%. We hypothesize that the lack of auxiliary training information makes it difficult to extract abstract rules and encourages to focus too much on the visual similarity of RPMs, which is unprofitable for the final downstream task. Similarly, setting $\gamma = 0$, decreased the accuracy on SCL, HriNet and CoPINet to 46.7%, 31.9% and 24.2%, respectively. This suggests that with purely auxiliary training, models are unable to relate the same abstract relationships to different visual figure configurations. Moreover, the absence of contrastive loss significantly hinders the ability to discriminate between correctly and incorrectly completed RPMs. These observations stress the importance of using joint loss, which is realized by setting both $\beta > 0$ and $\gamma > 0$.

Contrast strength. The quality of representations learned with contrastive frameworks benefits from applying bigger contrast to positive samples, which is realized by including additional negative pairs while calculating loss [4]. Analogously, methods which support multiple positive samples tend to benefit from higher number of positive pairs in a given batch [17]. These two properties of contrastive approaches result in the reliance on large batch sizes, which some works tried to tackle by storing additional negative samples in a memory bank [36] or using dynamically updated queue with a moving-average encoder [10, 6]. Contrary to the above findings, empirical evaluation of MLCL shows that the method does not require large batch sizes and surpasses the performance of traditional supervised training under the same experimental protocol. In fact, for larger batches we observed a slight drop in the final performance across all models, which suggests their negative influence on the auxiliary training. A further analysis of this phenomenon is presented in supplementary material.

In the final linear classification stage, the main goal is to select a correct answer, which requires to discriminate between correctly and incorrectly completed RPMs. We additionally analyze the importance of using RPMs with incorrect answers as additional negative samples, by removing the term $\Sigma_{i,k}$ from the denominator of eq. (6). Unavailability of these RPMs results in a notable drop of performance on the downstream task to 58.0% for SCL, 37.2% for HriNet and 26.5% for CoPINet. In fact, when this additional signal is ignored, the encoder network only learns how to

differentiate between correctly completed RPMs governed by different sets of rules, whereas the final classification task requires to differentiate between a correct RPM and a set of incorrectly completed RPMs. We conclude that this additional training signal obtained from RPMs with incorrect answers is mandatory for high downstream task performance.

5 Conclusion

In this work we propose a novel NCE algorithm suitable for multi-label samples and integrate it with an auxiliary training to devise a new ML approach (MLCL) to abstract visual reasoning tasks. The efficacy of MLCL is tested on a challenging task of solving RPMs which is formulated in this paper as a multi-label classification problem with the 1-1 correspondence between labels and abstract rules underlying a given RPM. The proposed approach establishes new state-of-the-art results on the Balanced-RAVEN dataset and demonstrates superior performance in 2 regimes from PGM. The MLCL framework is additionally supported by a sparse rule encoding scheme for RPMs introduced in the paper, which consistently outperforms the encoding method used in prior works on the Balanced-RAVEN dataset and is the preferred scheme for half of the PGM regimes.

References

- [1] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*, pages 15535–15545, 2019.
- [2] Alexander Buslaev, Vladimir I Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A Kalinin. Albumentations: fast and flexible image augmentations. *Information*, 11(2):125, 2020.
- [3] Patricia A Carpenter, Marcel A Just, and Peter Shell. What one intelligence test measures: a theoretical account of the processing in the raven progressive matrices test. *Psychological review*, 97(3):404, 1990.
- [4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.
- [5] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*, 2020.
- [6] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.
- [7] Dedre Gentner. Structure-mapping: A theoretical framework for analogy. *Cognitive science*, 7(2):155–170, 1983.
- [8] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304, 2010.
- [9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [10] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

- [11] Felix Hill, Adam Santoro, David Barrett, Ari Morcos, and Timothy Lillicrap. Learning to make analogies by contrasting abstract relational structure. In *International Conference on Learning Representations*, 2019.
- [12] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2019.
- [13] Douglas R Hofstadter. *Fluid concepts and creative analogies: Computer models of the fundamental mechanisms of thought*. Basic books, 1995.
- [14] Sheng Hu, Yuqing Ma, Xianglong Liu, Yanlu Wei, and Shihao Bai. Hierarchical rule induction network for abstract visual reasoning. *arXiv preprint arXiv:2002.06838*, 2020.
- [15] John E Hummel and Keith J Holyoak. Distributed representations of structure: A theory of analogical access and mapping. *Psychological review*, 104(3):427, 1997.
- [16] 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*, pages 2901–2910, 2017.
- [17] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362v1*, 2020.
- [18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2014.
- [19] Thomas Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. In *International Conference on Learning Representations*, 2020.
- [20] Andrew Lovett and Kenneth Forbus. Modeling visual problem solving as analogical reasoning. *Psychological review*, 124(1):60–90, 2017.
- [21] Jacek Mańdziuk and Adam Żychowski. Deepiq: A human-inspired ai system for solving iq test problems. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.
- [22] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
- [23] Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in neural information processing systems*, pages 2265–2273, 2013.
- [24] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
- [25] James C Raven. Mental tests used in genetic studies: The performance of related individuals on tests mainly educative and mainly reproductive. *Master’s thesis, University of London*, 1936.
- [26] John C Raven and John Hugh Court. *Raven’s progressive matrices and vocabulary scales*. Oxford psychologists Press Oxford, England, 1998.
- [27] 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*, pages 4967–4976, 2017.

- [28] Adam Santoro, Felix Hill, David Barrett, Ari Morcos, and Timothy Lillicrap. Measuring abstract reasoning in neural networks. In *International Conference on Machine Learning*, pages 4477–4486, 2018.
- [29] Linsey Smith and Dedre Gentner. The role of difference-detection in learning contrastive categories. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 36, 2014.
- [30] Jiaming Song and Stefano Ermon. Multi-label contrastive predictive coding. *arXiv preprint arXiv:2007.09852*, 2020.
- [31] Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*, 2020.
- [32] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning. *arXiv preprint arXiv:2005.10243*, 2020.
- [33] Duo Wang, Mateja Jamnik, and Pietro Lio. Abstract diagrammatic reasoning with multiplex graph networks. In *International Conference on Learning Representations*, 2020.
- [34] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. *arXiv preprint arXiv:2005.10242*, 2020.
- [35] Yuhuai Wu, Honghua Dong, Roger Grosse, and Jimmy Ba. The scattering compositional learner: Discovering objects, attributes, relationships in analogical reasoning. *arXiv preprint arXiv:2007.04212*, 2020.
- [36] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.
- [37] Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. Raven: A dataset for relational and analogical visual reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [38] 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*, pages 1075–1087, 2019.
- [39] Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Yin and yang: Balancing and answering binary visual questions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5014–5022, 2016.
- [40] Kecheng Zheng, Zheng-Jun Zha, and Wei Wei. Abstract reasoning with distracting features. In *Advances in Neural Information Processing Systems*, pages 5842–5853, 2019.

Supplementary Material

A Extended problem description

In the main body of the paper we have introduced the problem of solving RPMs only briefly. Here we provide additional details and in particular discuss the differences between rule encoding schemes used in the two considered datasets (Balanced-RAVEN and PGM). In both of them the goal is to complete a 3×3 matrix with a missing panel in the bottom-right corner. The answer has to be chosen from a set of 8 candidates, such that the chosen image satisfies *all* abstract rules defining a given RPM. In each problem instance there is only one answer which correctly completes the matrix, while the remaining answers usually satisfy some (but not all) of the underlying rules.

The sets of possible rules differ between the datasets. In Balanced-RAVEN a set of rules associated to a given RPM (called an *abstract structure*) is defined as a set of pairs $\mathcal{S} = \{[r, a] | r \in \mathcal{R}, a \in \mathcal{A}\}$, where $\mathcal{R} = \{\text{Constant, Progression, Arithmetic, Distribute Three}\}$ is the set of relation types and $\mathcal{A} = \{\text{Number, Position, Type, Size, Color}\}$ is the set of attributes.

The abstract structure of RPMs from the PGM dataset is defined as a set of triples $\mathcal{S} = \{[r, o, a] | r \in \mathcal{R}, o \in \mathcal{O}, a \in \mathcal{A}\}$, where $\mathcal{R} = \{\text{progression, XOR, OR, AND, consistent union}\}$ is the set of relation types, $\mathcal{O} = \{\text{shape, line}\}$ is the set of object types and $\mathcal{A} = \{\text{size, type, color, position, number}\}$ is the set of attribute types.

In order to better present the problem, let us consider the first example of an RPM from the main text (Fig. A.1). Searching for common patterns among the images leads us to the following observation: in both the first (top) and the second (middle) row images, the octagon shapes in the last (right) column images appear only in positions where there are octagons in both the first (left) and the second (middle) column images. Consequently, the respective rule ($s = [\text{shape, position, AND}]$) is potentially a part of the underlying abstract structure associated with this RPM, i.e. one may suppose that $s \in \mathcal{S}$. After applying this rule to the third (bottom) row, one would expect the missing image to have only a single octagon, located in the top-right corner.

In fact, only a single image (with label A) from the set of potential candidates matches this description. Furthermore, the remaining shape attributes (*size, color, number*), as well as various combinations of lines in this RPM do not satisfy any of the allowed rules from \mathcal{S} (actually their role is to *distract* the solver). The above observations lead us to a conclusion that for the considered RPM $\mathcal{S} = \{[\text{shape, position, AND}]\}$ and that the correct answer is A.

Observe that in the described example, the rule was applied row-wise, however, in general the RPMs from the PGM dataset may have rules applied either row- or column-wise, whereas rules from the Balanced-RAVEN dataset are only applied row-wise.

Further examples of RPMs together with explanation of their solutions are presented in Figs. F.1-F.9. We refer the readers interested in a more detailed description of both datasets to the original works of Santoro et al. (2018) and Zhang et al. (2019).

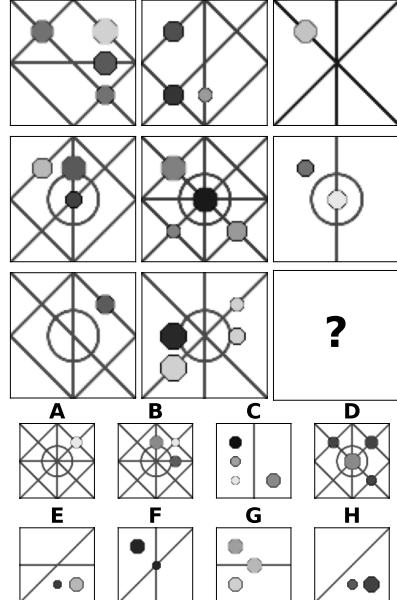


Figure A.1: RPM example from the PGM dataset presented in the main paper. The matrix is governed by a single rule – AND applied row-wise to octagon shape position. The correct answer is A.

B Sparse vs dense rule encoding

Let us now revisit the two rule encoding schemes discussed in the paper. Santoro et al. (2018) proposed to encode the abstract rules as binary strings of length 12 (called *meta-targets*) according to the following syntax: (`shape`, `line`, `color`, `number`, `position`, `size`, `type`, `progression`, `XOR`, `OR`, `AND`, `consistent union`). In order to support RPMs with multiple rules, the meta-target for a given RPM is obtained by performing an `OR` operation on the set of encoded individual rules.

For example, for an RPM instance with rules $\mathcal{S} = \{\text{[OR, shape, type]}, \text{[AND, line, color]}\}$, the method, referred to as *dense* encoding in the paper, yields the following meta-target:

$$\text{OR}([100000100100], [011000000010]) = [111000100110]$$

Based on the resultant string, one is able to conclude that the underlying structure consists of `OR` and `AND` relations, `shape` and `line` objects, `type` and `color` attributes. However, recovering the exact relations governing the considered RPM is not possible.

The new rule encoding scheme proposed in the paper, referred to as *sparse* encoding, is more explicit and allows to unambiguously retrieve the encoded relations, with the aim of providing a more accurate training signal. Since the set of all possible *abstract structures* in PGM is composed of 50 elements, i.e. there are $|\mathcal{R}| \times |\mathcal{O}| \times |\mathcal{A}| = 2 \times 5 \times 5 = 50$ unique rules, our method encodes them as a one-hot vector of length 50. Similarly to the dense encoding, we perform an `OR` operation on the set of all rules for a given RPM. However, due to the nature of one-hot encoding, all information about the underlying abstract structure is preserved.

Analogously to PGM, the rule encoding method for RPMs from the Balanced-RAVEN dataset, proposed by Zhang et al. (2019), represents each rule as a multi-hot vector of length $|\mathcal{R}| + |\mathcal{A}| = 9$ and combines the set of individual rule encodings by means of an `OR` operation. In effect, similarly to the case of PGM, recovering individual rules constituting the final encoded representation is not possible. This problem is further exacerbated by the generally higher numbers of rules in Balanced-RAVEN instances. RPMs in the Balanced-RAVEN dataset contain 6.29 rules on average, compared to only 1.37 rules (on average) in RPMs from the PGM dataset [37]. Consequently, for RPMs from the Balanced-RAVEN dataset, the resultant representation obtained with dense encoding is highly ambiguous.

Sparse rule representations of Balanced-RAVEN RPMs are obtained analogously to the PGM ones. In this dataset there are $|\mathcal{R}| \times |\mathcal{A}| = 4 \times 5 = 20$ unique configurations of rules and attributes. However, Zhang et al. (2019) pointed out that applying `Arithmetic` to `Type` is counterintuitive. Additionally, in some configurations the rules can be applied to both component structures. Namely, each of the configurations `L-R`, `U-D`, `O-IC`, `O-IG` consists of two distinct substructures: left/right, up/down, outer/inner, outer/inner, respectively. This gives a total of 38 unique rules, which is the length of our one-hot vector representation. Again, due to the nature of one-hot encoding, all information related to the abstract structure is preserved after applying the `OR` operation.

In summary, although the advantage of sparse encoding for PGM data is demonstrated only in certain regimes, it consistently outperforms the previous (dense) encoding method on Balanced-RAVEN, due to the significantly higher average number of rules per RPM instance in this dataset.

C Data augmentation

To the best of our knowledge this work is the first to investigate the use of data augmentation strategies in solving RPMs. For each sample, our data augmentation module chooses a random subset from available transformations and applies them in the same way to all RPM panels (images). We consider only image-level augmentations, which do not change the order of RPM panels. The following augmentation methods are used: *vertical flip*, *horizontal flip*, *rotation* by a random angle, *transposition*, *grid shuffle* and *roll*. Grid shuffle transformation splits an image into a 2x2 or 3x3 grid and randomly shuffles its components, whereas the roll operation, rolls an image along the vertical axis, horizontal axis or both axes simultaneously.

Examples of augmented RPMs are presented in Figs. F.1-F.9. Each figure, from left to right, presents (a) the base RPM from either Balanced-RAVEN or PGM, (b) the RPM after applying the

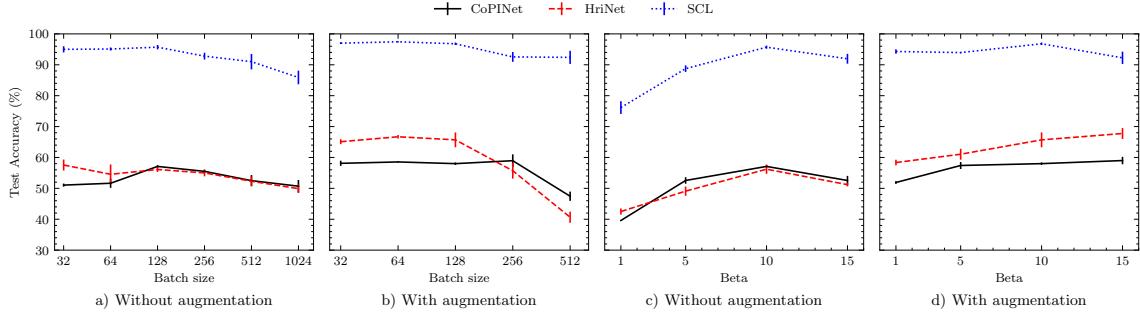


Figure B.1: Additional ablation studies on the Balanced-RAVEN dataset. The figures present variations in the final classification performance averaged across 4 random seeds depending on: 1) a batch size used for the contrastive pre-training stage of MLCL a) *without* and b) *with* data augmentation; 2) balancing factors in the definition of Multi-Label Contrastive Loss c) *without* and d) *with* data augmentation.

horizontal/vertical flip augmentation, (c) the RPM after horizontal/vertical roll, and (d) the RPM after 2x2 or 3x3 grid shuffle. The biggest structural changes are introduced to the images after applying the roll operation (the 3rd column in each figure) and the grid shuffle transformation (the last column in each figure). Even though the images become more difficult to decipher for human solvers, they greatly diversify the training set and increase the overall number of training instances.

D Implementation details

For the sake of speeding-up the experiments, in each hierarchy of HriNet we have replaced the ResNet backbone with a simpler CNN. This CNN was composed of 4 layers, each with 32 convolutional kernels of size 3x3 and stride equal to 2. Each layer was followed by batch normalization and ReLU activation. All hyperparameters reported in the paper were chosen without any extensive search, based on manual convergence analysis of a limited number of runs.

In all experiments on both Balanced-RAVEN and PGM datasets, we have trained models on images rescaled to the size 80x80, following the setup from [28]. This allowed to overcome hardware memory limitations, train with larger batches and conduct extensive comparisons. For the ease of reproducibility, in the source code attachment the experiments are structured as PyTorch Lightning modules.

E Batch size

We have further validated the influence of batch sizes on the results of MLCL training *with* and *without* data augmentation (see Figure B.1). We have varied batch sizes in the contrastive pre-training stage and used a constant size of 128 for linear evaluation.

The best performance for all models was achieved with small and middle-size batches, ranging from 32 to 128. We hypothesize that although larger batches allow for increasing the contrast strength of the proposed Multi-Label Contrastive Loss (eq. (8) in the paper), at the same time they hinder the convergence ability of the auxiliary loss.

This empirical evaluation stresses the importance of balanced interplay between the auxiliary and contrastive losses in our learning framework.

F Loss coefficients

In all experiments reported in the paper we set $\gamma = 1$ and $\beta = 10$ as the balancing factors used for the Multi-Label Contrastive Loss (eq. (8)). In order to verify the accuracy of this choice, we

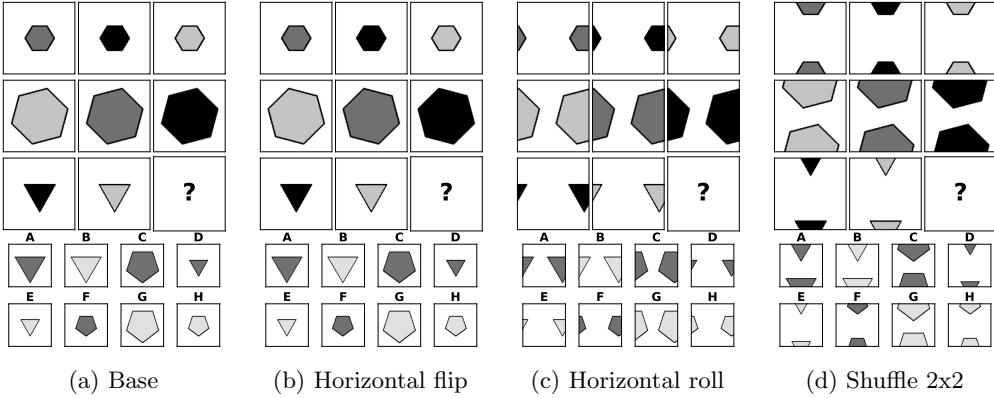


Figure F.1: Augmented RPM from the Balanced-RAVEN dataset with configuration **Center**. In each of two top rows, there is a single shape in each image ([Constant, Number]), located in the same position ([Constant, Position]), the shapes are of the same type [Constant, Type] and size [Constant, Size]. In each row there are shapes in 3 different colors ([Distribute_Three, Color]). This leads to an underlying abstract structure $\mathcal{S} = \{\text{[Constant, Number]}, \text{[Constant, Position]}, \text{[Constant, Type]}, \text{[Constant, Size]}, \text{[Distribute_Three, Color]}\}$, which is realized by completing the matrix with answer D.

run additional experiments with γ fixed to 1 and various values of β . As presented in Figure B.1, a correct choice of these hyperparameters is critical for the final performance. In the experiments without data augmentation, too small, as well as too big values of β have a negative impact on the accuracy of all models and the best results are achieved with $\beta = 10$. When data augmentation is applied, small values of β once again hinder the performance, however, the results for bigger values of β are less diverse than in the former case, and depending on the model the best results are accomplished with either $\beta = 10$ or $\beta = 15$.

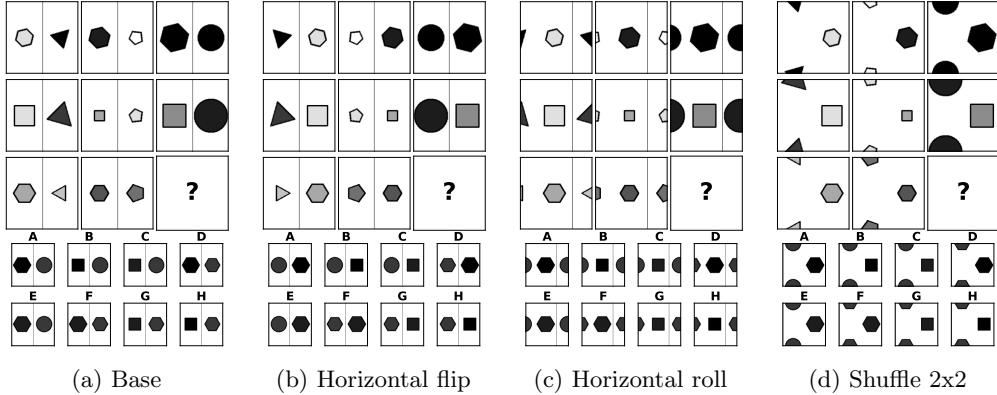


Figure F.2: Augmented RPM from the Balanced-RAVEN dataset with configuration L-R. In this configuration, both left and right parts of the images can be governed by different sets of rules. Starting from the left parts of the panels, one can see that each part is composed of a single object ([Constant, Number]), located in the centre ([Constant, Position]), objects in each row are of the same type – hexagons in the top row and squares in the middle row ([Constant, Type]), the sizes and the colors of objects in the third column are equal to the sum of object sizes (respectively colors) from the first and the second column ([Arithmetic, Size]) and ([Arithmetic, Color]). These rules constitute the left structure $\mathcal{S}_{\text{left}} = \{[\text{Constant}, \text{Number}], [\text{Constant}, \text{Position}], [\text{Constant}, \text{Type}], [\text{Arithmetic}, \text{Size}], [\text{Arithmetic}, \text{Color}]\}$. Next, looking at the right subpanels of each image, it can be observed that each part contains a single object ([Constant, Number]), located in the center ([Constant, Position]). In each row, types of objects (from right subpanels) follow progression ([Progression, Type]). Similarly to the left subpanels, sizes and colors of object in the third column can be calculated by summing up the respective attribute values of objects in the two preceding columns ([Arithmetic, Size] and [Arithmetic, Color]). This leads to the right structure defined as $\mathcal{S}_{\text{right}} = \{[\text{Constant}, \text{Number}], [\text{Constant}, \text{Position}], [\text{Progression}, \text{Type}], [\text{Arithmetic}, \text{Size}], [\text{Arithmetic}, \text{Color}]\}$. The final underlying abstract structure of the whole RPM is the sum of both left and right structures $\mathcal{S} = \mathcal{S}_{\text{left}} \cup \mathcal{S}_{\text{right}}$. A is the only answer which satisfies all 10 of the above-discussed rules.

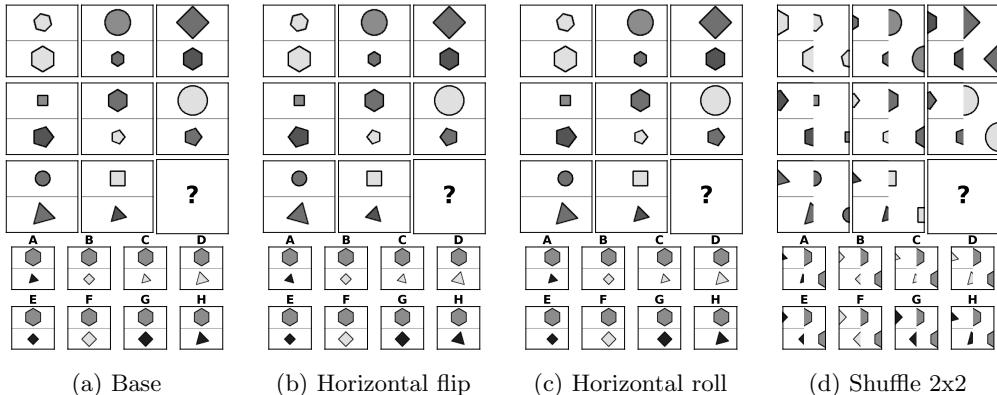


Figure F.3: Augmented RPM from the Balanced-RAVEN dataset with configuration U-D. Similarly to configuration L-R, both upper and lower parts can be governed by different sets of rules. The abstract structure of upper image parts is $\mathcal{S}_{\text{upper}} = \{[\text{Constant}, \text{Number}], [\text{Constant}, \text{Position}], [\text{Distribute_Three}, \text{Type}], [\text{Progression}, \text{Size}], [\text{Distribute_Three}, \text{Color}]\}$ and the set of rules of the lower parts is $\mathcal{S}_{\text{lower}} = \{[\text{Constant}, \text{Number}], [\text{Constant}, \text{Position}], [\text{Constant}, \text{Type}], [\text{Arithmetic}, \text{Size}], [\text{Distribute_Three}, \text{Color}]\}$. The final set of rules \mathcal{S} is defined as $\mathcal{S} = \mathcal{S}_{\text{upper}} + \mathcal{S}_{\text{lower}}$. The correct answer is C.

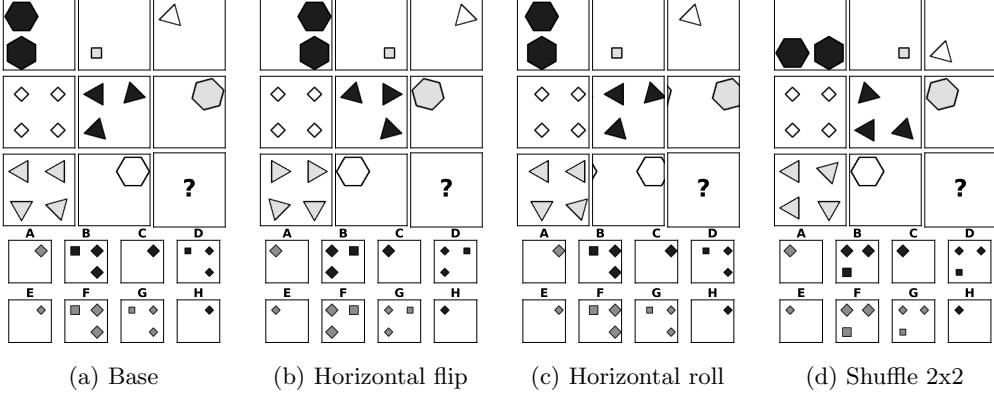


Figure F.4: Augmented RPM from the Balanced-RAVEN dataset with configuration 2x2Grid. The number of objects in the third column can be calculated by subtracting the number of objects in the second column from the number of objects present in the first column ([Arithmetic, Number]). In rows each panel contains objects with one out of three unique types, sizes and colors ([Distribute_Three, Type], [Distribute_Three, Size], [Distribute_Three, Color]). These relations lead to an underlying abstract structure defined as $\mathcal{S} = \{\text{[Arithmetic, Number]}, \text{[Distribute_Three, Type]}, \text{[Distribute_Three, Size]}, \text{[Distribute_Three, Color]}\}$. The only correct answer which satisfies all the above rules is D.

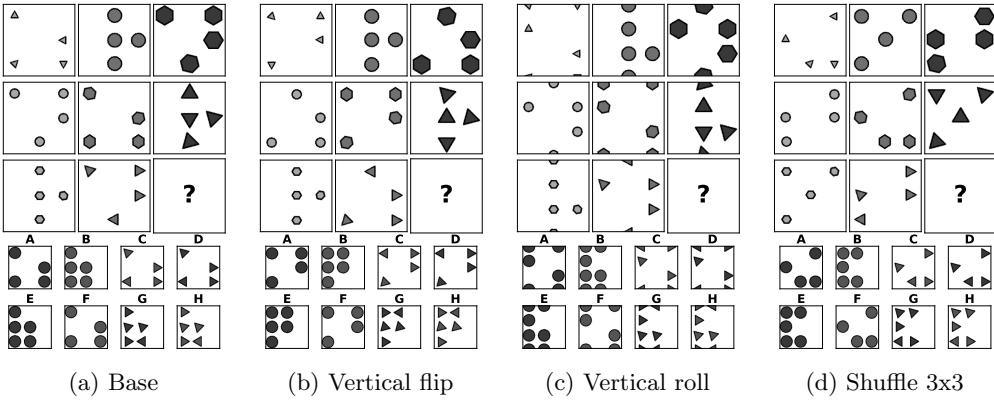


Figure F.5: Augmented RPM from the Balanced-RAVEN dataset with configuration 3x3Grid. In each row, the objects are arranged in one out of three available positions and each image contains a unique type of objects ([Distribute_Three, Position] and [Distribute_Three, Type]). Moreover, in each row, from left to right, the objects both increase in size and become darker ([Progression, Size] and [Progression, Color]). Therefore, the underlying abstract structure is of the form $\mathcal{S} = \{\text{[Distribute_Three, Position]}, \text{[Distribute_Three, Type]}, \text{[Progression, Size]}, \text{[Progression, Color]}\}$ and the correct answer is A.

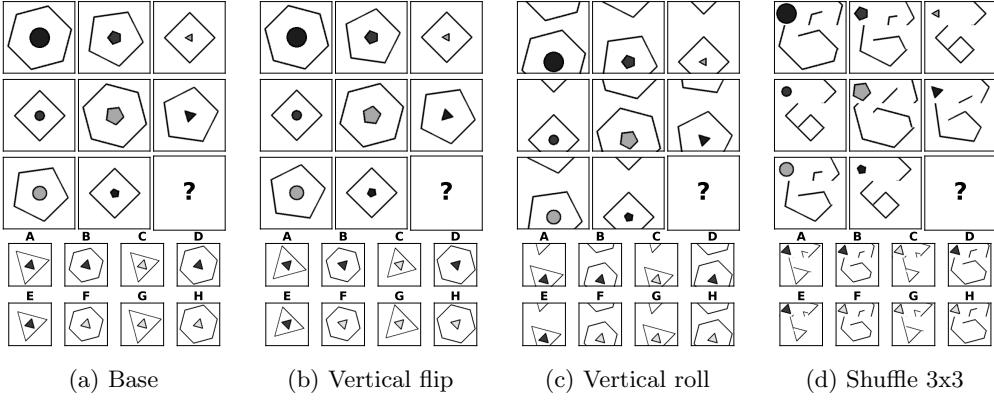


Figure F.6: Augmented RPM from the Balanced-RAVEN dataset with configuration 0-IC. In this configuration, both inner and outer structures can be governed by different sets of rules. Looking at the outer structures, one can see that each panel is composed of a single object ([Constant, Number]) at the central position ([Constant, Position]) of the same color ([Constant, Color]). In each row, the outer structures have one out of three possible sizes and types – hexagon, pentagon or a square ([Distribute_Three, Size] and [Distribute_Three, Type]). This gives an outer structure defined as $S_{\text{outer}} = \{\text{[Constant, Number]}, \text{[Constant, Position]}, \text{[Distribute_Three, Type]}, \text{[Distribute_Three, Size]}, \text{[Constant, Color]}\}$. Similarly, the inner structure of each image consists of single objects ([Constant, Number]) located in the centre ([Constant, Position]). The objects differ in type, with circles in the left column, pentagons in the middle one and triangles in the right column ([Progression, Type]). The objects in inner structures of each row have 3 different sizes and colors ([Distribute_Three, Size] and [Distribute_Three, Color]), which leads to a final outer structure $S_{\text{outer}} = \{\text{[Constant, Number]}, \text{[Constant, Position]}, \text{[Progression, Type]}, \text{[Distribute_Three, Size]}, \text{[Distribute_Three, Color]}\}$. The only answer which matches all rules for both inner and outer structures is D.

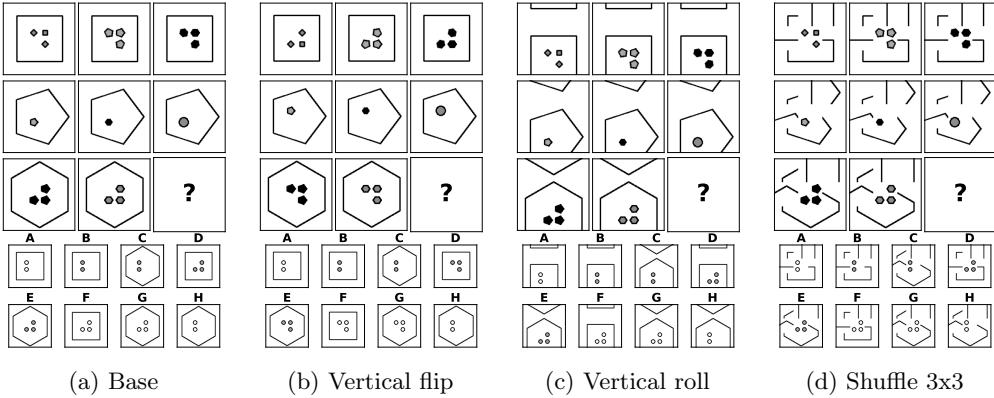


Figure F.7: Augmented RPM from the Balanced-RAVEN dataset with configuration 0-IG. Similarly to 0-IC configuration, the rules of 0-IG are applied to both inner and outer structures. Here, the outer structure is defined as $S_{\text{outer}} = \{\text{[Constant, Number]}, \text{[Constant, Position]}, \text{[Constant, Type]}, \text{[Constant, Size]}, \text{[Constant, Color]}\}$ and the inner structure as $S_{\text{inner}} = \{\text{[Constant, Number]}, \text{[Constant, Position]}, \text{[Progression, Type]}, \text{[Distribute_Three, Size]}, \text{[Distribute_Three, Color]}\}$. E is the correct answer.

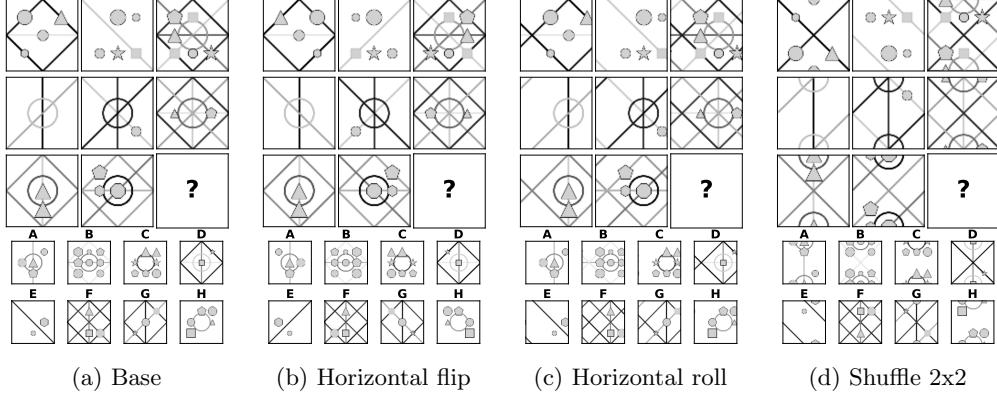


Figure F.8: Augmented RPM from the PGM dataset. In each row, the number of shapes increases by one in each consecutive panel from left to right. Namely, the first row contains images with 4, 5 and 6 shapes, respectively, whereas the second row with 0, 1 and 2 shapes, respectively. Following this pattern, we expect the missing image in the bottom row to be composed of four shapes, which is realised by choosing the answer A. The underlying abstract structure is defined as $\mathcal{S} = \{[\text{shape}, \text{number}, \text{progression}]\}$.

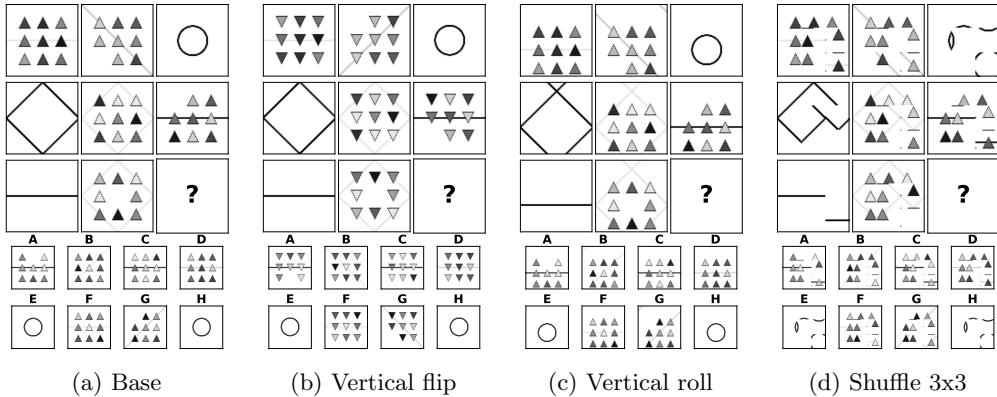


Figure F.9: Augmented RPM from the PGM dataset, with two rules applied to two different object types – shape and line. Firstly, note that the first two rows have consistent unions of shape numbers. That is, the first row contains panels with 9, 8 and 0 shapes, whereas the second row with 0, 9 and 8 shapes. This suggests that the missing panel should contain 9 shapes. However, there are multiple choices which satisfy this condition (B, C, D and F). Therefore, it is necessary to further notice the consistent union of line colors in both completed rows. This results in an abstract structure for this RPM defined as $\mathcal{S} = \{[\text{shape}, \text{number}, \text{consistent union}], [\text{line}, \text{color}, \text{consistent union}]\}$ and D being the correct answer.