A Unified View of Abstract Visual Reasoning Problems

Mikołaj Małkiński[1][0000−0002−5214−5928] (✉)
and Jacek Mańdziuk[1,2][0000−0003−0947−028X]

[1] Warsaw University of Technology, Warsaw, Poland
[2] AGH University of Krakow, Krakow, Poland
mikolaj.malkinski.dokt@pw.edu.pl ◇ jacek.mandziuk@pw.edu.pl

Abstract. The field of Abstract Visual Reasoning (AVR) encompasses a wide range of problems, many of which are inspired by human IQ tests. The variety of AVR tasks has resulted in state-of-the-art AVR methods being task-specific approaches. Furthermore, contemporary methods consider each AVR problem instance not as a whole, but in the form of a set of individual panels with particular locations and roles (context vs. answer panels) pre-assigned according to the task-specific arrangements. While these highly specialized approaches have recently led to significant progress in solving particular AVR tasks, considering each task in isolation hinders the development of universal learning systems in this domain. In this paper, we introduce a unified view of AVR tasks, where each problem instance is rendered as a single image, with no a priori assumptions about the number of panels, their location, or role. The main advantage of the proposed unified view is the ability to develop universal learning models applicable to various AVR tasks. What is more, the proposed approach inherently facilitates transfer learning in the AVR domain, as various types of problems share a common representation. The experiments conducted on four AVR datasets with Raven's Progressive Matrices and Visual Analogy Problems, and one real-world visual analogy dataset show that the proposed unified representation of AVR tasks poses a challenge to state-of-the-art Deep Learning (DL) AVR models and, more broadly, contemporary DL image recognition methods. In order to address this challenge, we introduce the *Unified Model for Abstract Visual Reasoning (UMAVR)* capable of dealing with various types of AVR problems in a unified manner. UMAVR outperforms existing AVR methods in selected single-task learning experiments, and demonstrates effective knowledge reuse in transfer learning and curriculum learning setups. Code is available at: `https://github.com/mikomel/avr-unified-view`

Keywords: Abstract Visual Reasoning · Deep Learning · Transfer Learning · Curriculum Learning

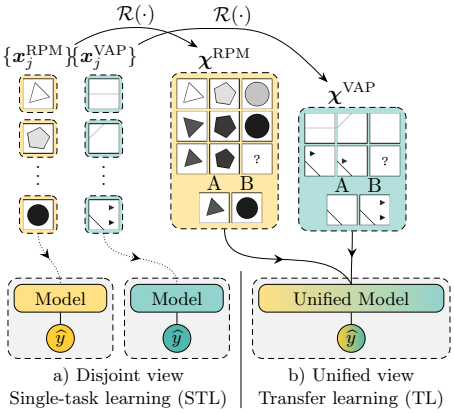


Fig. 1: **Disjoint vs. unified perspective.** Contemporary literature considers each AVR problem instance as a set of separate images (a), which leads to task-specific methods with limited applicability to other, even similar, tasks. In contrast, we propose the unified view (b), in which the problem instance is rendered as a single image (Fig. 2). This viewpoint facilitates the development of general AVR models inherently capable of incorporating advances from a broader CV field.

1 Introduction

Recent years have brought dynamic progress in the application of neural networks to computer vision (CV) problems. This increasing interest has led to the development of a broad family of effective vision models based on convolutional networks [6, 11], transformers [5, 10], and multi-layer perceptrons (MLPs) [23, 8], that often generalize well to other vision tasks. The universality of these models should be primarily attributed to the simplicity and wide applicability of the typical problem representation in the CV domain, in the form of a single image.

Abstract Visual Reasoning (AVR) is one of the CV subdomains gaining momentum in recent years. AVR encompasses problems that resemble tests used for measuring human abstract intelligence (IQ). A classical example are Raven's Progressive Matrices (RPMs) [21] that consist of simple 2D shapes (e.g., circles, hexagons, triangles) characterized by several attributes (e.g., rotation, colour, size). In most cases, an RPM problem instance is in the form of a 3×3 grid of panels, with the bottom-right panel missing (see Appendix D for typical examples). The test-taker has to complete the matrix by selecting one of the provided answer panels. In RPM datasets, there are usually up to 8 answer panels to choose from. In order to select the correct one, the subject has to identify various underlying abstract rules (e.g., progression, constancy, conjunction) that govern the location and attributes of RPM shapes. The selected answer has to conform to all these rules after being placed in the bottom-right corner of the matrix.

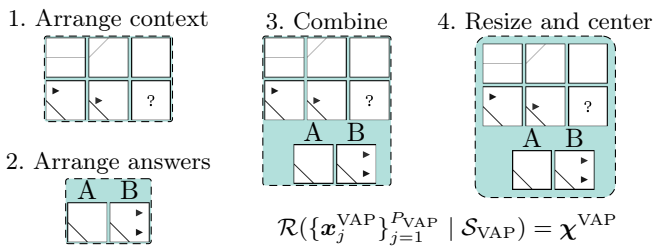


Fig. 2: **Rendering algorithm.** In the unified view a problem instance is rendered as a single image and without explicit division into context and answer panels.

RPMs are considered to be highly indicative for human intelligence [4], as they allow evaluating relational and abstract reasoning skills [22], and test one's ability to apply previously gained knowledge in new settings (new problem instances). Inspired by this crucial role of RPMs in designing IQ tests, recent streams of research have focused on evaluating the capacity of modern learning systems in solving RPM instances [17]. Motivated by the early successes [19], many approaches have been subsequently proposed that gradually improved the state-of-the-art (SOTA).

Despite impressive results, contemporary methods are built on a strict assumption that RPMs (or AVR tasks in general) are split beforehand into a set of individual matrix panels. In stark contrast, vision datasets typically render each problem instance as a single image. In this work, we coin these two perspectives as *disjoint* and *unified* representations, respectively (see Fig. 1). Due to inherent differences, models developed for one representation are not directly applicable to the other one. This, in turn, limits applicability of methods developed for solving AVR tasks to other vision problems and, *vice versa*, prevents evaluation of modern vision models on AVR tasks.

In addition to RPMs, other kinds of AVR benchmarks have recently been proposed [12], that specifically focus on conceptual abstraction, extrapolation, or arithmetic reasoning. This variety of AVR problems further exacerbates the issues arising from the disjoint perspective, as AVR tasks often differ w.r.t. the number and arrangement of panels the matrix is composed of. Consequently, a model constructed to handle tasks with a fixed number of panels arranged in a fixed configuration isn't directly capable of handling novel problem configurations, or problems with different numbers of panels. In effect, SOTA solutions in the AVR literature are task-specific, which limits progress towards general AVR solvers.

Contribution. In this work we pose and address the challenge of building AVR models capable of solving diverse AVR problems. To this end, we:

- Formulate a *unified view* of AVR tasks where a problem instance is represented as a single image (as opposed to a set of pre-defined panels), thus constituting a challenge for modern AVR/CV methods;
- Evaluate different CV models: convolutional networks, transformers, and MLPs on four AVR datasets with RPMs and Visual Analogy Problems (VAPs), and one real-world visual analogy dataset, represented in the unified manner, and demonstrate their limitations in this unified problem setup;
- Introduce the *Unified Model for Abstract Visual Reasoning (UMAVR)*, capable of effectively dealing with the unified problem representation, and outperforming strong baselines in this arrangement;
- Show the benefits of transfer learning (TL) and curriculum learning (CL) within the proposed unified AVR problem formulation.

On a general note, we postulate shifting the main focus of AVR research from developing dedicated task-specific models to general AVR models, capable of solving a variety of AVR problems.

2 Related Work

Tasks. Recently AVR has seen rapid expansion in terms of available benchmarks. In particular, several RPM datasets, such as PGM [1], I-RAVEN [33, 9], A-I-RAVEN [15] or G-set [19, 24] have been proposed. While the benchmarks differ in the number of rendered matrices, types and attributes of the objects, degree of compositional complexity, or the number of available answers, they all present matrices with the context panels arranged in the form of a 3×3 grid. VAPs [7] form a related challenge that focuses on conceptual abstraction, and are composed of a 2×3 grid of panels (see Fig. 1b and Appendix D). Other AVR problems further differ from the above two tasks in the number of context panels and their structure. For instance, the Odd One Out tests (O3) [19] present images arranged in a single row, while panels in Bongard Problems [3] are divided into left and right ones. Common to all AVR problems is their input representation in Machine Learning (ML) models, as an explicit set of distinct panels. This way of representing input data is in stark contrast to typical image recognition tasks, where each input is simply presented in the form of a single image, without further division into sub-images (though, such a division is often induced as part of a training process). In effect, existing AVR datasets cannot be easily considered when evaluating contemporary image recognition methods and are largely omitted in multi-task vision datasets. The proposed unified representation of AVR tasks enables to utilize existing AVR datasets for measuring abstract reasoning skills of current (and future) CV models.

AVR models. Due to inherent differences in the number of panels and the task structure across AVR datasets, as well as their explicit panel-based arrangement, main AVR research lines focus on designing task-specific architectures that operate on matrices pre-segmented into individual panels. Specifically, Wild Relation Network [1] employs a convolutional encoder to generate embeddings of

individual panels, which are later processed by a Relation Network. Stratified Rule-Aware Network [9] considers 3 different RPM hierarchies: single images, rows/columns, and pairs of rows/columns. For each hierarchy, a separate convolutional network is used, and a set of MLPs is employed to gradually merge the obtained embeddings. SCL [30] introduces the scattering transformation that splits panel representations, processes them in parallel with an MLP, and merges the results. Slot Transformer Scoring Network (STSN) [20] employs Slot Attention to discover objects in matrix panels and reasons about them with a transformer-based module [26]. SCAR [14] introduces the structure-aware dynamic layer that adapts its computation to the considered problem instance enabling the processing of AVR tasks with diverse structure. PoNG [16] integrates group convolution, normalization, and a parallel design.

The underlying assumption of the above approaches is that a problem instance is already pre-segmented into individual panels. Consequently, direct application of these methods to other image recognition tasks (where the input is formed by a single image), or other AVR tasks that differ in the number of panels or their structure is significantly hindered. In contrast, the proposed unified perspective facilitates and encourages the development of universal ML image recognition models that can be applied to solving diverse AVR tasks and, furthermore, other vision problems.

3 Method

We start with introducing the unified perspective that can be applied to virtually all AVR tasks, and then propose a novel image recognition model, well-suited to the proposed challenge.

3.1 Current Perspective (*Disjoint*)

In general, each AVR task $t \in \mathcal{T}$, where $\mathcal{T}$ is the family of all AVR tasks, can be defined as $t = (\{\mathcal{M}_i^t\}_{i=1}^{N_t}, \mathcal{S}_t)$, where $\{\mathcal{M}_i^t\}_{i=1}^{N_t}$ is a set of N_t different matrices (problem instances) and $\mathcal{S}_t$ is a task's structure (common for all instances). In this work, we treat each AVR dataset as a separate task. While matrices don't repeat across tasks, some tasks may have a common structure.

For each t, a single matrix (problem instance) from t can be defined as $\mathcal{M}_i^t = (X_i^t, y_i^t)$, where $X_i^t = \{x_{i,j}^t\}_{j=1}^{P_t}$ is composed of P_t panels. Each panel $x_{i,j}^t$ is a greyscale image with height h and width w, i.e. $x_{i,j}^t \in [0,1]^{h \times w}$, and y_i^t is the answer to the problem. For single-choice tasks, such as RPMs or VAPs, y_i^t is an index of the correct answer.

The above definition of a problem instance is rather general and doesn't include any information about the problem's specificity. Such problem-specific metadata is expressed by the task's structure $\mathcal{S}_t$, which defines how the images should be interpreted and arranged to form a 2D problem instance. For example, the structure of RPMs specifies that the set of images should be split into two sets: a set of 8 context panels arranged in a 3×3 grid, with a missing panel in

the bottom-right corner, and a set of up to 8 answer panels also arranged in a grid.

In summary, while each AVR task t is defined as: $t = (\{(\{\boldsymbol{x}_{i,j}^t\}_{j=1}^{P_t}, \boldsymbol{y}_i^t)\}_{i=1}^{N_t}, \mathcal{S}_t)$ its actual representation (i.e. the respective AVR dataset) is a collection of images: $t = (\{(\{\boldsymbol{x}_{i,j}^t\}_{j=1}^{P_t}, \boldsymbol{y}_i^t)\}_{i=1}^{N_t})$ with *implicitly* defined structure. Consequently, each AVR model $\mathcal{F}_t$ for solving matrices from t embeds the problem structure directly in its architecture. A construction of such a model using a building process $\mathcal{B}$ may be defined as a composition of functions f, given the task's structure $\mathcal{S}_t$:

$$\mathcal{B}(f_* \circ f_\# \circ \ldots \circ f_\$ \mid \mathcal{S}_t) = \mathcal{F}_t \tag{1}$$

In the AVR literature, $f_*, f_\#, \ldots, f_\$$ are most often implemented as neural network components. This leads to the following general form of a model for solving problems from t:

$$\mathcal{F}_t(\{\boldsymbol{x}_{i,j}^t\}_{j=1}^{P_t}) = \widehat{\boldsymbol{y}}_i^t \tag{2}$$

Due to the dependence on both $\mathcal{S}_t$ (implicit) and P_t (explicit), $\mathcal{F}_t$ cannot be directly applied to solving any task $t' \in \mathcal{T}_t^{'}$ with a different structure or number of panels: $\mathcal{T}_t^{'} = \{t' \in \mathcal{T} : \mathcal{S}_t \neq \mathcal{S}_{t'} \vee P_t \neq P_{t'}\}$

3.2 Proposed New Perspective (*Unified*)

Instead of treating an AVR problem instance as a set of images with an associated structure, we propose to employ a rendering algorithm $\mathcal{R}$ that merges separate images, given the task's structure, into a single image $\chi \in [0,1]^{h' \times w'}$ with new height h' and width w' (see Figs. 1 and 2):

$$\mathcal{R}(\{\boldsymbol{x}_{i,j}^t\}_{j=1}^{P_t} \mid \mathcal{S}_t) = \boldsymbol{\chi}_i^t \tag{3}$$

In practice, $\mathcal{R}$ defines how instances of a given task should be presented. To implement the algorithm for RPMs and VAPs considered in this work, for a given instance we first arrange the context panels, together with the missing panel (which is presented as a blank image with a centred question mark) into a grid with a small margin separating the panels. This gives a 3×3 grid accommodating 8 context panels for RPMs, and a 2×3 grid containing 5 context panels for VAPs. Next, we arrange the answer panels in another grid and position it below the context grid. Depending on the task, the number of available answers (n_a) differs, e.g. matrices from VAP, G-set, and I-RAVEN datasets, have $n_a = 4, 5, 8$, resp. To accommodate this variability, we render the answer grid with up to 4 panels in each row, resulting in up to 2 rows. A text label is placed above each answer panel. Next, we initialize a blank canvas and resize the constructed image to fit the canvas. While resizing, we keep the height to width ratio of the image in order to preserve the relative size of panel dimensions and to not distort the encompassed objects. We fix the width of the canvas to 416, and depending on the considered problem setting, set its height to 384 for VAPs, 448 for RPMs with $n_a \leq 4$, and 544 for RPMs with $4 < n_a \leq 8$, which gives enough space to clearly render the panels. All dimensions are divisible by 16 to ensure that

patch-based methods (with patch size $p = 16z$, for $z \in \mathbb{N}$) can be directly applied without the need to resize the underlying image.

$\mathcal{R}$ converts a considered AVR task into a common, unified representation that (a) facilitates construction of new universal AVR models capable of solving diverse AVR tasks, and (b) enables application of TL in existing SOTA AVR models. To the best of our knowledge, neither (a) nor (b) have ever been considered in the AVR literature. Within the above unified perspective, one can view any task t as: $t = \{(\boldsymbol{\chi}_i^t, \boldsymbol{y}_i^t)\}_{i=1}^{N_t}$.

3.3 General AVR Solver

Using the proposed unified perspective, it is possible to construct a general AVR model:

$$\mathcal{B}(f_* \circ f_\# \circ \ldots \circ f_\$) = \mathcal{F} \quad (4) \qquad \mathcal{F}(\boldsymbol{\chi}_i^t) = \widehat{\boldsymbol{y}}_i \quad (5)$$

Since $\mathcal{B}$ no longer depends on the task's structure (Eq. 1 vs Eq. 4) and the model doesn't depend on the number of panels in the matrix (Eq. 2 vs Eq. 5), the proposed unified perspective allows building a general model $\mathcal{F}$ applicable to solving diverse AVR tasks. At the same time, $\mathcal{F}$ has to support input images χ_i^t that may vary in size. Also, since $\mathcal{F}$ is no longer aware of which task it operates on, it has to provide its prediction in a common (fixed) format $\widehat{\boldsymbol{y}}$. We assume that $\widehat{y} \in \mathbb{N}$ is an index of the correct answer, which is the case of RPMs, VAPs, and many other AVR tasks [12]. This assumption may not be valid in some tasks, such as the original Bongard Problems [3], where an answer has to be provided in natural language. An extension of the unified perspective to problems with specific output formats is left for future work. Lastly, to be applicable to diverse AVR tasks, $\mathcal{F}$ has to be flexible enough to handle various structures of the tasks of interest.

3.4 Proposed Unified AVR Model

In the initial experiments, we've discovered that SOTA CV baseline models struggle to deal with the above structural diversity. Consequently, we propose UMAVR (Unified Model for Abstract Visual Reasoning) a neural architecture well-suited for the introduced unified view that takes rectangular images as input. To build *local* representations of low-level features, a convolutional backbone with depth D_L is employed. Each layer is composed of a convolution layer with kernel size 3×3, and 2×2 stride for dimensionality reduction, followed by Batch Normalization layer with ReLU activation. In the default setting, we use $D_L = 4$, and the layers have 16, 16, 32, and 128 output channels, resp. The size of the last channel determines the embedding size of a token at a given spatial location and is further referred to as d. Applying this perception backbone yields a latent representation $z_0 \in \mathbb{R}^{d \times r \times c}$, where r and c are the numbers of rows and columns in the resultant token embedding matrix, resp. For an image of size 544×416, this gives $z_0 \in \mathbb{R}^{128 \times 27 \times 25}$.

Next, we attach a component with a *global* receptive field to facilitate the discovery of patterns that span multiple panels. As a base layout, we adopt the MetaFormer architecture [31], which follows a layer-wise design, where the same layer is repeated $D_G = 4$ times, and the weights are not shared across layers. The operation performed in layer $l \in [1, D_G]$ can be formalized as:

$$z_l^* = \text{TokenMixer}\left(\text{Norm}(z_{l-1})\right) + z_{l-1} \tag{6}$$

$$z_l = \text{ChannelMixer}\left(\text{Norm}(z_l^*)\right) + z_l^* \tag{7}$$

where z_l is the output of the l'th layer, z_l^* is an intermediate representation in layer l, and Norm is the Layer Normalization. After the last layer, the token embedding matrix is passed through Layer Normalization, averaged along the width and height dimensions, and projected with a linear layer into n_a-dimensional vector. The output is passed through the softmax function, and the model is optimized end-to-end with cross-entropy. The model architecture diagram is presented in Appendix B.

TokenMixer. The input z_{l-1} is normalized and passed through a residual 2D convolution layer with a kernel of size 5×5, and 2×2 padding to enrich tokens with the context from their spatial proximity. Next, three parallel pathways are employed as introduced in the Vision Permutator (ViP) [8], which process the token matrix along the width, height, and channel dimensions, resp. In each branch, $S = 8$ segments are used. Outputs of the pathways are concatenated depthwise and a 2D convolution with 1×1 kernel and d output channels is applied to fuse information from the separate paths, in contrast to ViP which merges the branches with sum or Split Attention [34].

ChannelMixer. The above global reasoning step is followed by local processing using a two-layer non-linear feed-forward block, input normalization, and a residual connection, as popularized by the Transformer [26]: $z_l = \sigma(\text{Norm}(z_l^*)W_1)W_2 + z_l^*$, where $W_1 \in \mathbb{R}^{d \times kd}$ and $W_2 \in \mathbb{R}^{kd \times d}$ are learnable weights with an expansion factor k, and σ is a non-linearity. In the default setting, we use $k = 4$ and GELU.

4 Experiments

We conduct experiments in three learning settings: STL, TL and CL. In each case, the performance of UMAVR is compared with the baseline models belonging to distinct model families. All models are designed to return a logit vector $v \in \mathbb{R}^{n_a}$ representing a score for each answer, and the softmax function is used to compute the probability distribution $\widehat{p}$ over the set of answers. The index corresponding to the highest probability is considered the predicted answer.

Baselines. The set of benchmark models includes convolutional networks, represented by ResNet [6] and ConvNext [11], Vision Transformer (ViT) [5], MaxViT

Table 1: **Model size.** The number of parameters of each model in millions (M).

Model	# Parameters	Model	# Parameters
ResNet-18	11.2M	Mixer S/16	20.0M
ResNet-50	23.8M	Mixer S/32	18.2M
ConvNext-Pico	8.7M	ViP-Nano	4.0M
ConvNext-Nano	15.1M	ViP-Tiny	7.9M
MaxViT-Pico	7.3M	UMAVR	3.5M
MaxViT-Nano	15.0M		
TinyViT-5M	5.1M		
TinyViT-11M	10.6M		

Table 2: **Dataset details.** The benchmarks differ w.r.t. their size, allocation into train / val / test splits, and the maximal number of available answers n_a^{max}.

Dataset	Size	Train	Val	Test	n_a^{max}
G-set	49K	34.4K	9.8K	4.8K	5
I-RAVEN	70K	42K	14K	14K	8
PGM	1.42M	1.2M	20K	200K	8
VAP	710K	600K	10K	100K	4
VASR	154.8K	150K	2.25K	2.55K	4

[25], TinyViT [29] and Swin Transformer [10] as representatives of the Transformer family, adapted to vision tasks, and MLP-based models such as MLP-Mixer [23] and Vision Permutator (ViP) [8]. Table 1 compares the numbers of model parameters.

Tasks. The models are evaluated on three challenging AVR problems. Firstly, we consider the problem of solving RPMs from three datasets: G-set [19, 24] with visually simple matrices following experiment 1 from [24]; I-RAVEN [33, 9] with a much richer set of available objects, attributes, and abstract rules; and the Neutral regime of PGM [1] which is of much bigger size. Secondly, we consider the VAP dataset [7] which presents a conceptual abstraction challenge with matrices that are structurally different from RPMs. Thirdly, we employ the Visual Analogies of Situation Recognition dataset (VASR) [2] that presents visual analogies comprising real-world images. We employ the dataset variant with random distractors.

Overall, the datasets vary in size, the content of the matrices (geometric shapes, real-world images), and the number of available answers (see Table 2). This diversity allows gaining insights into models' performance in different axes. To deeper analyse the limitations of the discussed models, in some experiments we reduced the number of available answers in the matrices, expecting that this simplification would make the task more comprehensible.

Experimental setting. The models are trained with batches of 256 matrices for PGM and VAP datasets, and of 128 matrices in all the remaining cases. Model parameters are optimized with Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$, until the validation loss doesn't improve for 10 consecutive epochs. We tuned learning rate λ of each model separately with 3 randomly initialized runs on G-set and I-RAVEN with $n_a = 2$ and selected λ that worked best on average. We applied linear learning rate warmup starting from $\lambda = 10^{-6}$ over 500 iterations and cosine decay with $\lambda_{\text{min}} = 10^{-6}$. When learning to solve matrices from PGM, I-RAVEN and VAP, we make use of a supplementary training signal in the form of an auxiliary loss [1, 33] with sparse encoding [13], where the model has to additionally predict the hidden rules that govern the matrix construction. To this end, in parallel to the answer prediction layer, a shallow rule classifier is applied. The classifier operates on the token embedding matrix z_{D_G} passed through the Layer Normalization and averaged along the width and height dimensions. The classifier is composed of a linear layer with 128 units, followed by GELU and another linear layer with $|r|$ output neurons, where $|r|$ is the size of the one-hot encoded rule vector. Depending on the dataset, this gives 50, 40 and 28 units for PGM, I-RAVEN and VAP, respectively, which corresponds to the number of unique rules in each dataset. Each training run is performed on a node with a single NVIDIA DGX A100 GPU.

In the experiments with $n_a < n_a^{max}$, in the original problem instance (with n_a^{max} answer panels) $n_a^{max} - n_a$ randomly sampled incorrect answers are deleted.

4.1 Single-Task Learning

STL experiments assess the ability of modern CV models to solve uniformly viewed AVR tasks. In preliminary experiments conducted on 4 AVR tasks (G-set, I-RAVEN, PGM, VAP) we discovered that large CV models typically struggled to perform better than chance, irrespectively of the dataset and the number of possible answers. These models included ConvNext (Tiny, Small and Base), ViT-B/16, ViT-B/32, Swin (Tiny and Small), ViP-S/7 and ViP-M/7. To overcome their limitations, in subsequent experiments we employed their smaller parameter-efficient variants including ConvNext Pico and Nano [28], MaxViT (Pico and Nano) [25] and TinyViT (5M and 11M) [29]. For ViP, variants smaller than Small weren't defined in the original paper, which lead us to construct two new variants coined Nano and Tiny. Their detailed description is provided in Appendix A.

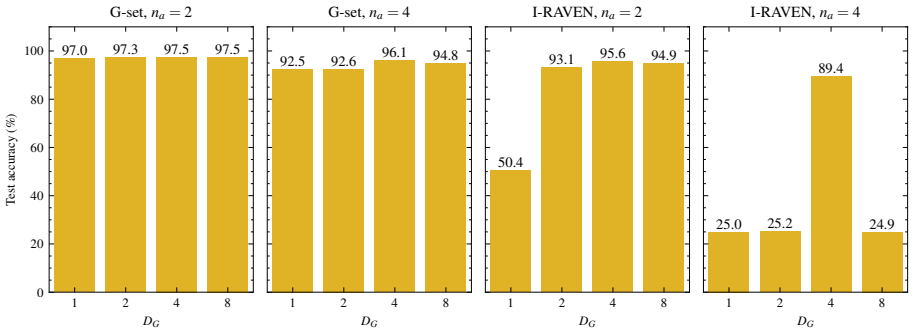
Table 3 compares test accuracy of the models on 4 datasets with variable numbers of answers. In G-set, where the amount of available data is scarce, ConvNext and ViP models struggle to perform better than chance, while remaining models achieve high performance, typically above 90%. ResNet-50 and Mixer S/16 scored slightly above 80% on the most challenging dataset configuration with $n_a = 5$. In I-RAVEN, which contains visually richer matrices with a hierarchical structure, both ResNet variants and MaxViT-Pico demonstrate performance at the random guess level across all n_a, other baseline models achieve non-random results only for $n_a = 2$, while UMAVR significantly outcompetes

Table 3: **Single-task learning.** Test accuracy of three baseline families of models (convolutional networks, vision transformers, MLP models for vision) and UMAVR in solving matrices from five datasets with variable numbers of possible answers n_a. Results higher than a random guess by more than 0.5 p.p are highlighted with a blue background. Best results are marked in bold and the second best are underlined. P, N and T denote Pico, Nano and Tiny, resp.

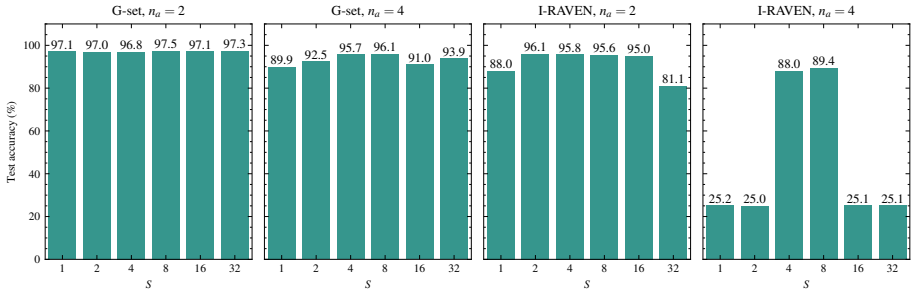
Model	G-set, $n_a =$			I-RAVEN, $n_a =$			PGM, $n_a =$			VAP, $n_a =$		VASR
	2	4	5	2	4	8	2	4	8	2	4	4
ResNet-18	97.0	91.5	91.1	50.0	25.0	12.5	73.8	59.0	41.2	98.3	95.4	25.0
ResNet-50	96.9	94.1	81.7	50.2	25.0	12.5	73.8	56.4	32.7	98.4	96.1	43.7
ConvNext-P	50.0	25.0	20.0	67.0	<u>29.4</u>	12.7	50.0	25.0	12.5	96.0	90.4	25.0
ConvNext-N	50.0	25.0	20.0	69.8	25.1	12.5	50.0	25.0	12.5	95.9	91.7	25.0
MaxViT-P	97.1	<u>95.8</u>	**95.6**	50.1	25.2	12.6	85.7	**97.1**	<u>92.0</u>	**99.4**	**98.4**	**62.3**
MaxViT-N	<u>97.3</u>	**96.1**	<u>95.5</u>	81.0	25.0	12.7	85.2	<u>94.6</u>	**97.3**	<u>99.3</u>	<u>98.2</u>	24.0
TinyViT-5M	97.2	95.6	95.4	83.5	25.0	12.5	72.7	41.1	34.3	99.0	97.1	55.4
TinyViT-11M	97.0	94.7	95.4	71.2	25.5	12.7	84.3	46.6	34.0	98.3	96.3	54.2
Mixer S/16	96.3	92.2	82.8	62.3	25.5	<u>12.9</u>	<u>88.8</u>	74.5	61.1	97.5	88.4	54.5
Mixer S/32	97.0	95.6	94.9	74.9	25.5	12.8	**90.1**	76.4	73.4	97.2	93.0	54.1
ViP-N	50.0	25.0	20.0	<u>88.3</u>	25.1	12.5	77.6	46.3	81.7	97.6	95.2	25.0
ViP-T	50.0	25.0	20.0	75.6	25.0	12.5	85.2	73.8	49.0	97.3	95.5	25.0
UMAVR	**97.5**	**96.1**	95.1	**95.6**	**89.4**	**13.1**	76.9	63.3	52.3	93.9	97.3	<u>59.8</u>

all methods for $n_a \in \{2, 4\}$. On PGM, all models but ConvNext learned to solve some matrices, though notably the best results were achieved by MaxViT ones. On VAP, all models present satisfactory results, commonly exceeding 90%. UMAVR demonstrates consistent and strong performance across nearly all considered settings, showing its general strength in visual reasoning, rather than overfitting to a particular task. The only exception is I-RAVEN with $n_a = 8$, in which all tested models performed at the random guess level. We conclude that UMAVR is a versatile method that in spite of its simplicity outcompetes other mainstream CV models in certain settings (specifically I-RAVEN with $n_a = 4$).

Pre-trained checkpoints. To better understand the abstract reasoning capacity of large vision models, we repeated the STL experiments for ConvNext (Tiny, Small, Base), ViT (B/16, B/32) and Swin (Tiny, Small), starting from checkpoints pre-trained on ImageNet available in the TorchVision package. However, the only setting where any of these models performed better than chance was I-RAVEN with $n_a = 2$, where ConvNext-T and ConvNext-S achieved test accuracy of 75.6% and 83.6%, resp. This shows that despite using large pre-training datasets, the contemporary large vision models are generally incapable of solving AVR tasks represented in a unified manner proposed in this paper. Since the problem of classifying real-world images formulated in ImageNet is fundamentally different from AVR tasks considered in this work, we hypothesize that



(a) Depth D_G



(b) Number of segments S

Fig. 3: **Ablation study.** UMAVR test accuracy with various D_G and S settings on G-set and I-RAVEN with $n_a \in \{2, 4\}$. In the default setting, we used $D_G = 4$ and $S = 8$.

pre-training of large vision models on AVR data of sufficient scale might potentially boost their performance. Until this hypothesis is validated in future work, relatively smaller, parameter-efficient supervised models remain SOTA in the domain.

Real-world analogies. The VASR dataset [2] presents visual analogies formed from real-world images. Methods used in [2] employ pre-trained popular CV models (e.g. ViT or ConvNext) to embed each matrix image separately. The answer is predicted by applying vector arithmetics or training a shallow supervised classifier on frozen image embeddings. A limitation of these approaches is the lack of ability to reason about the relations between matrix panels in early layers of the model. Instead, the reasoning is only framed as a post-processing step. Differently, we applied the proposed unified view to VASR, which enables application of CV models in an entirely different setup, in which the models reason over the whole matrix starting already from early layers of the model. The results are displayed in Table 3 (the rightmost column). Certain models, including ResNet-18, both ConvNext variants, MaxVit-Nano and both ViP variants

Table 4: **Transfer learning.** Test accuracy of models that performed better than chance with STL on PGM (cf. Table 3). The models are first pre-trained on PGM with $n_a = 2, 4, 8$, resp., and then fine-tuned on G-set with $n_a = 2, 4, 5$ and I-RAVEN with $n_a = 2, 4, 8$, resp. TL and STL scores are shown on left/right resp.

Model	G-set test accuracy (%)			I-RAVEN test accuracy (%)		
	$n_a = 2$	$n_a = 4$	$n_a = 5$	$n_a = 2$	$n_a = 4$	$n_a = 8$
ResNet-18	96.7/97.0	92.4/91.5	93.0/91.1	78.7/50.0	40.2/25.0	12.5/12.5
ResNet-50	96.7/96.9	92.9/94.1	93.8/81.7	71.2/50.2	44.6/25.0	12.5/12.5
MaxViT-P	97.5/97.1	96.0/95.8	95.6/95.6	83.7/50.1	67.5/25.2	41.6/12.6
MaxViT-N	97.2/97.3	**96.2**/96.1	**95.9**/95.5	86.4/81.0	61.3/25.0	38.5/12.7
TinyViT-5M	95.7/97.2	92.7/95.6	92.9/95.4	56.9/83.5	29.8/25.0	26.0/12.5
TinyViT-11M	97.4/97.0	90.6/94.7	95.6/95.4	82.7/71.2	46.6/25.5	27.3/12.7
Mixer S/16	96.9/96.3	93.6/92.2	91.6/82.8	78.3/62.3	46.7/25.5	20.1/12.9
Mixer S/32	96.5/97.0	93.4/95.6	94.9/94.9	79.5/74.9	45.4/25.5	36.2/12.8
ViP-N	96.3/50.0	91.8/25.0	95.0/20.0	74.0/88.3	46.4/25.1	**43.6**/12.5
ViP-T	**97.6**/50.0	95.2/25.0	94.9/20.0	81.2/75.6	60.1/25.0	28.4/12.5
UMAVR	97.3/97.5	95.9/96.1	95.4/95.1	**95.5**/95.6	**89.7**/89.4	42.8/13.1

performed indistinguishably from random guessing. Other models present performance typically exceeding 50%, with the best results achieved by MaxViT-Pico and UMAVR. We conclude that VASR matrices presented in the unified view pose a significant challenge for the contemporary vision models.

Ablation study. We conducted additional experiments with several model variants to understand the contribution of respective components: 1) UMAVR with the convolutional backbone replaced with MetaFormer's patch emebdding layer; 2) UMAVR with TokenMixer replaced with the identity layer; 3) MetaFormer. We also considered PoolFormer-S12 [31], PoolFormerV2-S12, ConvFormer-S18, and CAFormer-S18 [32] as baselines related to UMAVR. The models were evaluated in STL on G-set and I-RAVEN with $n_a \in \{2, 4\}$. In all cases, the results were indistinguishable from random guessing, which shows the relevance of all specific UMAVR components for effective learning and reasoning. Further, we explored various model depths $D_G \in \{1, 2, 4, 8\}$ and numbers of segments $S \in \{1, 2, 4, 8, 16, 32\}$ to identify the optimal configuration. As shown in Fig. 3, $D_G = 4$ and $S = 8$ lead to the best performance, justifying our choice of these values as the default setting for these hyperparameters.

4.2 Transfer Learning

Next, for the models that performed better than chance in at least one of the STL experiments, we explore a TL scenario, where models pre-trained on the largest dataset (PGM) are fine-tuned on the two smallest ones (G-set or I-RAVEN).

Algorithm 1 Curriculum learning. Solving matrices of gradually increasing difficulty.

Input: randomly initialized model f_θ, $n_a^{\max}$, λ_0
Output: trained model f_θ
1: $n_a = 2$
2: **while** $n_a \leq n_a^{max}$ **do**
3: $\lambda \leftarrow \lambda_0$ # reset the learning rate
4: $f_\theta \leftarrow \text{train}(f_\theta, n_a, \lambda)$ # until convergence with early stopping
5: $n_a \leftarrow n_a + 1$
6: **end while**

For both the pre-training and fine-tuning datasets the same value of n_a is used, except for G-set with $n_a = 5$, for which pre-training on PGM is performed with $n_a = 8$. Table 4 presents the results. Application of TL leads to significant gains in multiple considered settings. Specifically, the performance of ResNet-50 and Mixer S/16 improved respectively from 81.7% and 82.8% to 93.8% and 91.6% on G-set with $n_a = 5$, while the performance of ViP Nano and Tiny improved from random guessing level to being on-par with other top performers across all n_a configurations. On I-RAVEN, significant improvement across most considered settings is observed, as after TL only ResNets struggle in the most demanding setting ($n_a = 8$).

The results signify the importance of utilizing a shared problem representation in the AVR domain, as pre-training the models on a large dataset can lead to notable performance improvements on the tasks, for which the available data is scarce. In certain cases, however, we observe the impact of the negative transfer effect, which brings attention to the need of designing robust TL techniques.

4.3 Curriculum Learning

We evaluate the CL approach, in which the model is iteratively trained on gradually more demanding matrices, starting from $n_a = 2$ to $n_a = n_a^{max}$ with step 1, with the reuse of previously gathered knowledge (see Algorithm 1).

We considered all models listed in Table 3 and evaluated them with CL on G-set and I-RAVEN. On the former dataset, CL improved the performance of ResNet-50 from 81.7% to 90.4% and Mixer S/16 from 82.8% to 95.2%. On I-RAVEN, CL improved the results of MaxViT Pico to 75.7% (+63.1 p.p.), TinyViT-5M to 28.3% (+15.8 p.p.), TinyViT-11M to 31.4% (+18.7 p.p.), Mixer S/16 to 31.7% (+18.8 p.p.), Mixer S/32 to 37.7% (+24.9 p.p.), ViP Tiny to 55.7% (+43.2 p.p.), and UMAVR to 86.9% (+73.8 p.p.). Performance in the remaining settings stayed at the STL level (±0.3 p.p.). Overall, for certain models, including the best-performing model – UMAVR, application of CL raised the STL results significantly, showing the potential of effective knowledge reuse within the unified view framework.

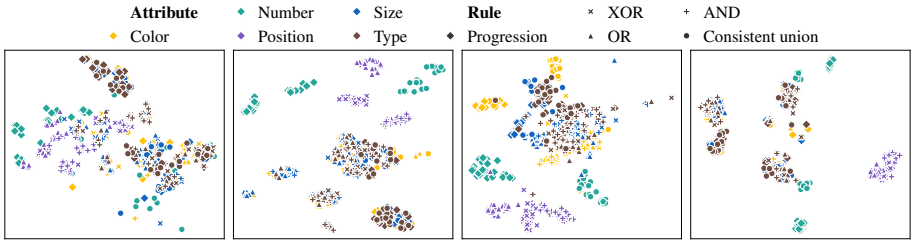


Fig. 4: **PGM embeddings.** The embeddings of PGM matrices ($n_a = 2$) from the test split of the `Neutral` regime, visualized with t-SNE. For the sake of interpretability, the figure considers matrices with a single rule applied to Shape objects.

4.4 Qualitative Analysis

Figure 4 compares UMAVR matrix embeddings ($n_a = 2$) with selected representative models on PGM. Overall, the clustering quality correlates with the model performance on the target task (cf. Table 3). Across all visualized models, the embeddings of matrices with rules applied to Number and Position attributes (green and purple, resp.) cluster into distinct groups. In addition, the embeddings of Mixer S/32 start to form distinguishable clusters for the remaining attributes as well, which aligns with the model leading performance in this setting. The visualization confirms that the models learn to identify the underlying abstract rules instead of relying on visual shortcuts or dataset biases. Appendix C extends this analysis to other relevant models and datasets.

4.5 Unified vs. Disjoint Representation

For the most demanding setting of $n_a = n_a^{\max}$, the best-performing unified approaches outperformed the state-of-the-art disjoint result on G-set (97.6% vs. 82.8% [24]), were on-par on PGM (97.3% vs. 98.2% [20]) and VAP (98.4% vs. 98.5% [20]), and were inferior on I-RAVEN (86.9% vs. 95.7% [20]) and VASR (62.3% vs. 86.0% [2]).

While the comparison shows a slight advantage of disjoint representation, it is important to note that the unified view opens several research avenues and poses challenges that extend beyond the sole performance improvement. Specifically, the use of the unified representation allows bridging the domains of broad CV and AVR by means of enabling the development of methods that could be seamlessly applied to both areas. Furthermore, the unified representation allows employing the current CV methods operating on single images to solve AVR tasks, and to use pre-trained checkpoints for their initialization. Finally, universal methods developed for solving uniformly viewed AVR tasks may accelerate progress in other domains that require relational reasoning, via knowledge reuse.

5 Conclusions and Future Work

Existing CV models that excel in solving AVR problems are generally task-specific, which prevents their application to other, even similar problems. In this work, we have formulated a unified view on AVR tasks in which an AVR instance is regarded as a single image, with no indication about the location or role of individual panels (context panels vs. answer panels). Apparently, even the SOTA CV models struggle to efficiently perform in this new setting, and in certain cases are unable to exceed a random guess level. To address this new challenge, we propose the UMAVR model, applicable to solving diverse AVR tasks within the above unified perspective. UMAVR shows its strength in the STL setup, and also demonstrates effective knowledge reuse in TL and CL setups, surpassing the performance of strong baselines.

The development of universal AVR methods has potential to foster progress in related areas via knowledge transfer. One possible target domain is document understanding that requires a high degree of relational reasoning. Large-scale datasets, with uniformly viewed AVR tasks, could be used to pre-train effective reasoning models and facilitate the development of new solutions in this area.

Moreover, AVR tasks were employed to analyze the reasoning capabilities of large language models (LLMs) using text-based task representations [27]. Recent studies extended this line of research to encompass multi-modal LLMs [18]. We believe that the proposed unified view of AVR tasks can support these advancements by providing a general problem representation, reducing the reliance on task-specific solution strategies.

Acknowledgments. This research was carried out with the support of the Laboratory of Bioinformatics and Computational Genomics and the High Performance Computing Center of the Faculty of Mathematics and Information Science Warsaw University of Technology. Mikołaj Małkiński was funded by the Warsaw University of Technology within the Excellence Initiative: Research University (IDUB) programme.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Barrett, D., Hill, F., Santoro, A., Morcos, A., Lillicrap, T.: Measuring abstract reasoning in neural networks. In: International Conference on Machine Learning. pp. 511–520. PMLR (2018)
2. Bitton, Y., Yosef, R., Strugo, E., Shahaf, D., Schwartz, R., Stanovsky, G.: VASR: Visual analogies of situation recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 241–249 (2023)
3. Bongard, M.M.: The recognition problem. Tech. rep., Foreign Technology Div Wright-Patterson AFB Ohio (1968)
4. Carpenter, P.A., Just, M.A., Shell, P.: What one intelligence test measures: a theoretical account of the processing in the Raven Progressive Matrices Test. Psychological Review **97**(3), 404 (1990)

5. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2021)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
7. Hill, F., Santoro, A., Barrett, D., Morcos, A., Lillicrap, T.: Learning to make analogies by contrasting abstract relational structure. In: International Conference on Learning Representations (2019)
8. Hou, Q., Jiang, Z., Yuan, L., Cheng, M.M., Yan, S., Feng, J.: Vision Permutator: A permutable MLP-like architecture for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**(1), 1328–1334 (2023)
9. Hu, S., Ma, Y., Liu, X., Wei, Y., Bai, S.: Stratified rule-aware network for abstract visual reasoning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 1567–1574 (2021)
10. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin Transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10012–10022 (2021)
11. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A ConvNet for the 2020s. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11976–11986 (2022)
12. Małkiński, M., Mańdziuk, J.: A review of emerging research directions in abstract visual reasoning. Information Fusion **91**, 713–736 (2023)
13. Małkiński, M., Mańdziuk, J.: Multi-label contrastive learning for abstract visual reasoning. IEEE Transactions on Neural Networks and Learning Systems **35**(2), 1941–1953 (2024)
14. Małkiński, M., Mańdziuk, J.: One self-configurable model to solve many abstract visual reasoning problems. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 14297–14305 (2024)
15. Małkiński, M., Mańdziuk, J.: A-I-RAVEN and I-RAVEN-Mesh: Two new benchmarks for abstract visual reasoning. In: Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25 (2025), (Accepted)
16. Małkiński, M., Mańdziuk, J.: Advancing generalization across a variety of abstract visual reasoning tasks. In: Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25 (2025), (Accepted)
17. Małkiński, M., Mańdziuk, J.: Deep learning methods for abstract visual reasoning: A survey on Raven's Progressive Matrices. ACM Computing Surveys **57**(7), 1–36 (2025)
18. Małkiński, M., Pawlonka, S., Mańdziuk, J.: Reasoning limitations of multimodal large language models. A case study of Bongard Problems. In: International Conference on Machine Learning. PMLR (2025), (Accepted)
19. Mańdziuk, J., Żychowski, A.: DeepIQ: A human-inspired AI system for solving IQ test problems. In: 2019 International Joint Conference on Neural Networks. pp. 1–8. IEEE (2019)
20. Mondal, S.S., Webb, T.W., Cohen, J.: Learning to reason over visual objects. In: International Conference on Learning Representations (2023)
21. Raven, J.C., Court, J.H.: Raven's progressive matrices and vocabulary scales. Oxford pyschologists Press Oxford, England (1998)

22. Snow, R.E., Kyllonen, P.C., Marshalek, B.: The topography of ability and learning correlations. Advances in the Psychology of Human Intelligence **2**(S 47), 103 (1984)
23. Tolstikhin, I.O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al.: MLP-mixer: an all-MLP architecture for vision. Advances in Neural Information Processing Systems **34**, 24261–24272 (2021)
24. Tomaszewska, P., Żychowski, A., Mańdziuk, J.: Duel-based deep learning system for solving IQ tests. In: International Conference on Artificial Intelligence and Statistics. pp. 10483–10492. PMLR (2022)
25. Tu, Z., Talebi, H., Zhang, H., Yang, F., Milanfar, P., Bovik, A., Li, Y.: MaxViT: Multi-axis vision transformer. In: European Conference on Computer Vision. pp. 459–479. Springer (2022)
26. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in Neural Information Processing Systems **30**, 5998–6008 (2017)
27. Webb, T., Holyoak, K.J., Lu, H.: Emergent analogical reasoning in large language models. Nature Human Behaviour **7**(9), 1526–1541 (2023)
28. Wightman, R.: PyTorch image models. `https://github.com/rwightman/pytorch-image-models` (2019). https://doi.org/10.5281/zenodo.4414861
29. Wu, K., Zhang, J., Peng, H., Liu, M., Xiao, B., Fu, J., Yuan, L.: TinyViT: Fast pretraining distillation for small vision transformers. In: European Conference on Computer Vision. pp. 68–85. Springer (2022)
30. Wu, Y., Dong, H., Grosse, R., Ba, J.: The scattering compositional learner: Discovering objects, attributes, relationships in analogical reasoning. arXiv:2007.04212 (2020)
31. Yu, W., Luo, M., Zhou, P., Si, C., Zhou, Y., Wang, X., Feng, J., Yan, S.: MetaFormer is actually what you need for vision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10819–10829 (2022)
32. Yu, W., Si, C., Zhou, P., Luo, M., Zhou, Y., Feng, J., Yan, S., Wang, X.: MetaFormer baselines for vision. IEEE Transactions on Pattern Analysis and Machine Intelligence **46**(2), 896–912 (2024)
33. Zhang, C., Gao, F., Jia, B., Zhu, Y., Zhu, S.C.: RAVEN: A dataset for relational and analogical visual reasoning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5317–5327 (2019)
34. Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Lin, H., Zhang, Z., Sun, Y., He, T., Mueller, J., Manmatha, R., et al.: ResNeSt: Split-attention networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2736–2746 (2022)