



OPEN Untrained neural networks can demonstrate memorization-independent abstract reasoning

Tomer Barak^{1✉} & Yonatan Loewenstein^{1,2}

The nature of abstract reasoning is a matter of debate. Modern artificial neural network (ANN) models, like large language models, demonstrate impressive success when tested on abstract reasoning problems. However, it has been argued that their success reflects some form of memorization of similar problems (data contamination) rather than a general-purpose abstract reasoning capability. This concern is supported by evidence of brittleness, and the requirement of extensive training. In our study, we explored whether abstract reasoning can be achieved using the toolbox of ANNs, without prior training. Specifically, we studied an ANN model in which the weights of a naive network are optimized during the solution of the problem, using the problem data itself, rather than any prior knowledge. We tested this modeling approach on visual reasoning problems and found that it performs relatively well. Crucially, this success does not rely on memorization of similar problems. We further suggest an explanation of how it works. Finally, as problem solving is performed by changing the ANN weights, we explored the connection between problem solving and the accumulation of knowledge in the ANNs.

The topic of this paper is abstract reasoning, sometimes referred to as “fluid intelligence”¹. Abstract reasoning is, broadly speaking, the ability to solve complex problems by identifying regularities and relations in the problem being solved and utilizing them for deducing the solution^{2,3}. It is often studied using intelligence tests that comprise word analogy tests (e.g., infer that the relationship between “cow” and “milk” is the same as between “chicken” and “egg”) and visual reasoning tests (e.g., Raven Progression Matrices)^{4,5}. As artificial intelligence continues to advance, understanding the nature of abstract reasoning in both humans and machines is becoming a central question in cognitive science and AI research⁶.

Abstract reasoning in artificial neural networks (ANNs) appears to be closely tied to training. While deep ANNs have shown impressive performance on various intelligence tests^{7–12}, their success relied heavily on extensive prior training. Additionally, questions have been raised about the nature of this performance. There are indications that ANNs’ success may stem more from “contamination” – exposure to similar questions in their training data – rather than from genuine abstract reasoning^{13,14}. This dependency on specific training data is further emphasized by findings that minor changes in problem phrasing, which do not affect human performance, can render problems unsolvable for ANNs^{6,15}. Thus, while ANNs may exhibit some analogical reasoning capabilities, it is disputed that these are based on pattern matching or memorization rather than on general intelligence comparable to that of humans.

In this work, we investigated whether ANN tools commonly used in machine learning are capable of demonstrating general abstract reasoning. Specifically, we asked if these networks could solve intelligence test problems with novel inputs, relying only on the information provided by the specific problem at hand, without drawing on prior memorization.

Certain intelligence tests, by their nature, require some level of prior knowledge. For instance, a human unfamiliar with English or the relationship between “cow” and “milk” would struggle to relate “chicken” to “egg” in an analogy test. Consequently, general intelligence in humans is often assessed using *visual* reasoning tests, utilizing abstract shapes like squares and triangles to minimize the influence of language or cultural knowledge. Thus, to evaluate the abstract reasoning of ANN models, we employed visual abstract reasoning tests. These visual reasoning tests require identifying relations in a sequence of stimuli, a skill common to many intelligence tests^{1–3,16}.

For our network models, we used Relation Networks (RNs)¹⁷, as members of this class of models were shown to be capable of identifying abstract relations and solving intelligence tests after extensive training^{9,18}. Notably,

¹The Edmond and Lily Safra Center for Brain Sciences, The Hebrew University, Jerusalem, Israel. ²Department of Cognitive Sciences, The Federmann Center for the Study of Rationality, The Alexander Silberman Institute of Life Sciences, The Hebrew University, Jerusalem, Israel. ✉email: tomer.barak@mail.huji.ac.il

RNs were shown to successfully solve word-analogy problems without specific training on those problems, but with specific training on relevant relationships^{19,20}. In contrast to these previous studies, our focus was on the ability of “naive” RNs, who were not exposed to *any* pre-training, to identify relations in visual reasoning tests and use them for solving the tests.

The structure of this paper is as follows: We begin by introducing the visual reasoning tests and the network models employed in our study. Next, we present the model's performance, showcasing its ability to solve non-trivial problems. We then analyze the mechanisms underlying this performance. Finally, as the model's problem-solving involves an optimization process that modifies network parameters in a manner similar to learning, we examine the relationship between the model's problem-solving capabilities and the networks' accumulation of knowledge.

Results

Sequential visual reasoning tests

We constructed a set of artificial problems in which the task is to evaluate the consistency of an image with a sequence of its preceding images (Fig. 1). Each problem comprises 5 gray-scale images and 4 optional-choice images. The images, 224×224 pixels each, are composed of identical abstract objects and differ along several dimensions: the shape of the objects, their size, their color, their number, and their arrangement. By construction, one of these features changes predictably over the 5 images. Formally, an image is characterized by a low-dimensional vector of features, f_j , where f_j^i denotes the value of feature i in image j . An image in pixel space, x_j , is constructed according to its characterizing features by a generative function $x_j = G(f_j)$. One of the features f_j^p changes predictably along the sequence according to a simple deterministic rule $f_{j+1}^p = U(f_j^p)$ while the other features are either constant over the images or change randomly (values are i.i.d). Considering the optional-choice images, the predictable rule is followed in only one of them, and the task is to select this image. The other features are either constant in all 9 images (5 of sequence and 4 of optional choices) or change randomly (see Methods). We refer to a problem's predictably changing feature as the problem's Predictive Feature (PF) and to the randomly changing features as distracting features or *distractors*. Intuitively, the number of distractors is a measure of a problem's difficulty.

The computational task

Each image is characterized by a small number of features, of which one changes predictably. The challenge is to simultaneously identify the features and the rule that relates the features of the different images. Relation Networks¹⁷ do exactly that. Taking a set of stimuli, they learn two functions: an encoder function $Z_\phi(x)$ that extracts relevant feature(s) from the stimulus, that is, a low-dimensional representation of the stimuli x and a relation module $R_\theta(Z_\phi(x_i), Z_\phi(x_j))$ that characterizes the relationship between the features of pairs of stimuli x_i and x_j . In practice, the encoder and the relation modules are functions (typically networks) whose parameters (ϕ and θ , respectively) are learned from examples. It should be noted that, to some extent, the complexities of the encoder and the relation module are interchangeable. The reason is that a sufficiently-complex relation module

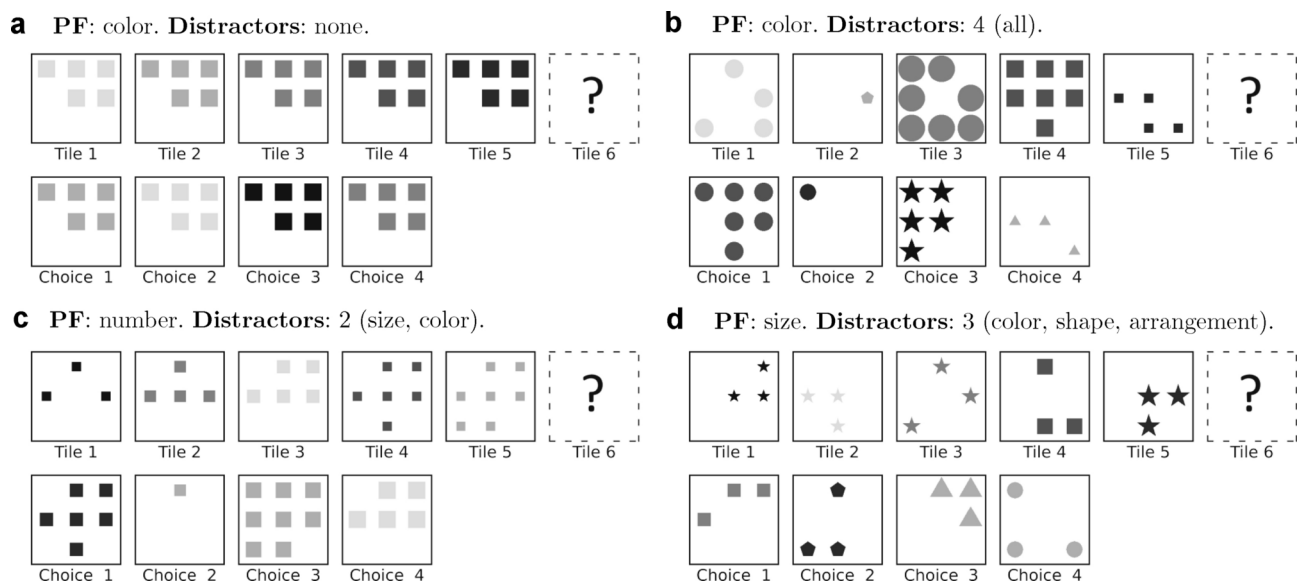


Fig. 1. Visual reasoning problems. The problems are characterized by the *Predictive Features* (PF) that can be the color (a, b), number (c), or size (d) of the abstract shapes. The values of the predictive features linearly increase along the sequence. The rest of the features (non-predictive) are either constant or random. We refer to the random features as *Distractors*, and their number determines the problem difficulty. *Note:* the shapes' type and arrangement are always non-predictive, and can either be constant or distracting. The correct choices in this figure are all 3.

can incorporate the feature extraction. Similarly, a sufficiently complex encoder can operate on the extracted features as to simplify the relation between them. For example, any monotonous relation between the features is also a linear relation between a (nonlinear) transformation of the features.

Previous studies have shown that with sufficient examples, relational networks can learn to extract the relevant features and their relations at a level sufficient for solving intelligence tests^{9,18,20}. The challenge here is to perform a similar task without any pre-training. To do so, we defined the following loss function on a sequences of 5 images:

$$\mathcal{L}(\theta, \phi) = \frac{1}{4} \sum_{i=1}^4 [R_{\theta}(Z_{\phi}(\mathbf{x}_i), Z_{\phi}(\mathbf{x}_{i+1}))] \quad (1)$$

\mathbf{x}_i is the i^{th} image in the sequence, the encoder $Z_{\phi} : \mathbb{R}^{224 \times 224} \rightarrow \mathbb{R}^n$ is a function that takes 224×224 pixel images to an n dimensional latent space and the relation module $R_{\theta} : \mathbb{R}^{2n} \rightarrow \mathbb{R}^+$ takes two consecutive latent variables, each of dimension n , and outputs a positive 1D relation score.

This loss is minimized for a relation function $R_{\theta}(Z_{\phi}(\mathbf{x}_i), Z_{\phi}(\mathbf{x}_j))$ that outputs a minimal relation score for consecutive sequence images ($j = i + 1$), requiring the identification of the regularity that characterizes these consecutive images. We updated the networks' weights θ and ϕ with 10 optimization steps over the loss $\mathcal{L}(\theta, \phi)$ using the RMSprop optimizer²¹ (learning rate of 10^{-5} , the rest of the parameters are set to PyTorch²² default). Eventually, after optimization, we evaluated the consistency of each choice image with the sequence based on their relation value R when they were placed as the sixth sequence image $R_{\theta}(Z_{\phi}(\mathbf{x}_5), Z_{\phi}(\cdot))$ and selected the choice image with the lowest relation value as the answer.

To clarify, in these settings, the model does not need to learn the features and their relations in the generative sense to pass a test successfully. Instead, it is enough to find image representations and rules that are *sufficiently* correlated with a problem's predictive feature for selecting the most consistent image out of four options.

Vanilla model performance

The success of the model would depend on the specific choice of R and Z (their network structure), as they can be inductively biased towards certain types of features and rules. In our vanilla model, the encoder Z was a small CNN from input space to a 1D latent neuron, composed of 3 convolutional layers followed by 5 fully-connected (FC) layers with a single output neuron (see Methods and Supplementary Information Fig. S1). For the relation module, we used a simple function that asserts a linearly changing relation between the latent variables,

$$R_{\theta}(Z_{\phi}(\mathbf{x}_i), Z_{\phi}(\mathbf{x}_j)) = (Z_{\phi}(\mathbf{x}_i) - Z_{\phi}(\mathbf{x}_j) + \theta)^2 \quad (2)$$

where θ is a trainable constant that does not depend on Z .

We evaluated the performance of the vanilla model on the different tests, in which the predictive feature's values increased linearly, and found that it performed substantially better than chance (0.25) in almost all tasks and all levels of difficulty (Fig. 2). Without distractors, its performance on some tasks was close to perfect. We also found that performance decreased with the number of distractors, verifying that the number of distractors is a good measure of the task's difficulty. All these results were obtained using networks that were randomly initialized before each problem, thus demonstrating that ANNs can perform abstract reasoning that does not depend on memorization. Averaged over all conditions, the model's performance was 0.58 ± 0.01 . From this point in the paper, we use this global performance measure for comparisons (see Methods; complete performance results are in the Supplementary Information).

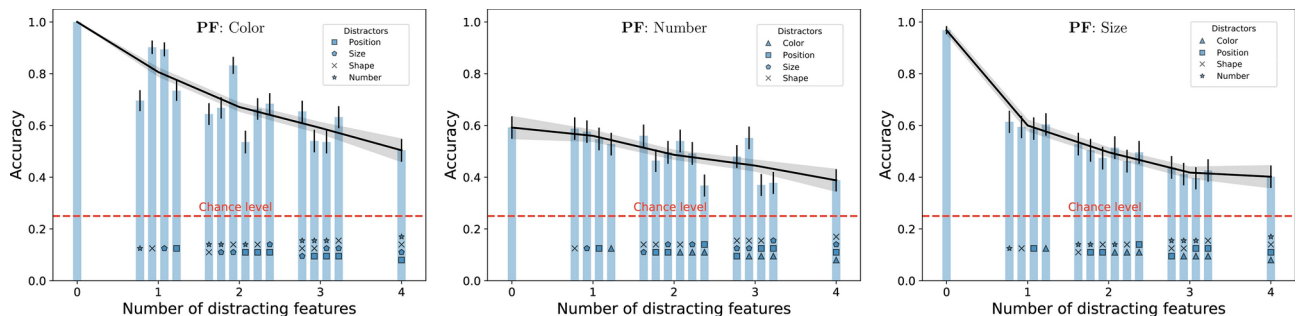


Fig. 2. Vanilla model performance. The performance of naive ANNs on the three Predictive Features (PFs): Color (left), Number (center), and Size (right). For each predictive feature, we tested the networks over 16 test conditions where the predictive feature was linearly changing along the sequence, and the non-predictive features were either distractors (marked according to the legend) or constant (not marked). Each test condition included 500 randomly generated problems. Error bars are 95% Confidence Intervals (CI). The black line and its shade are the average accuracy per difficulty and the corresponding 95% CI. The dashed line denotes the chance level of problems with four choice images (0.25).

Determinants for success

Our model consists of two main components: the encoder Z and the relation module R . The parameters of both were changed in the direction of minimizing the loss function on the images of each problem, a process that we will refer to as optimization. To study the relative contribution of these components to problem-solving, we studied the model's performance when the parameters of only one of these components, either Z or R , were optimized. We found that optimizing the encoder was essential: when the parameters of the encoder Z remained unchanged, the model's performance, averaged over all conditions, was close to the chance level, 0.30 ± 0.01 (Fig. S2). By contrast, using random parameters for the relation module R and not optimizing it had no significant effect on performance, resulting in an average performance of 0.58 ± 0.01 (Fig. S3), which is not significantly different from that of the vanilla model. These results motivated us further to study the role of the encoder in the task.

The encoder

The encoder is an 8-layer network with 3 convolutional layers followed by 5 Fully-Connected (FC) layers. Removing the convolutional layers and connecting the FC layers directly to the inputs impaired the average performance of the model, reducing its performance to 0.48 ± 0.01 (Fig. S4), indicating that the convolutional layers are important for performance. In the vanilla model, the parameters of both the convolutional layers and the FC layers are optimized in the direction of minimizing the loss function. However, it turns out that the optimization of the parameters of the convolutional layers does not contribute to the performance. The average performance when the weights of the convolutional layers remained random, 0.57 ± 0.01 , was not significantly different than that of the vanilla model (Fig. S5). By contrast, keeping the FC network weights fixed at their randomly-initialized values during problem-solving was detrimental to the performance (0.34 ± 0.01 , Fig. S6).

So far, we saw that freezing either the weights of the convolutional layers or the relation module at their initial random values does not impair performance. This insensitivity does not change when *both* are frozen (0.58 ± 0.01 , Fig. S7).

We conclude that the convolutional layers effectively operate as frozen feature extractors (features in the more general sense – not necessarily the features used for constructing the images) while the parameters of the FC layers are optimized to solve the task.

To test how the FC layers contribute to this task, we note that the task could be perfectly solved if the encoder could learn to identify the inverse generative function of the problem images $G^{-1}(x)$ and use it to extract the underlying predictive feature f^p and its rule $U(f^p)$. If this is done, we expect the optimization steps to increase the correlation of the encoder's output neuron with the predictive feature (but not with the distracting features). We tested this hypothesis in the vanilla model for all the predictive features. Indeed, the absolute Pearson correlation of the output neuron with the predictive feature (see Methods) increases with optimization steps, as depicted in Fig. 3a (black).

To better understand how such correlations emerge in the FC network, we also computed the absolute Pearson correlations of these features with the activities of all other neurons in the FC network (see Methods). Comprehensive results in Supplementary Fig. S8). These correlations, averaged over all neurons in a layer, are depicted in Fig. 3a. We found that the correlations with the predictive feature increase with the layer depth.

The higher the correlation of the output neuron with the predictive feature, the easier it is for the relation module to identify the regularity in the sequence of images. Along the same lines, we also expected the optimization process to decrease the correlation of the encoder output neuron with the other irrelevant features. This, however, is not the case. Considering the same features in problems in which they are not predictive features (either constant or distracting), we found that the correlation of the output neuron with these features also increases on average in the optimization process, albeit to a lesser extent (Fig. 3b). Considering the correlations of these features with neurons in the hidden layers of the encoder, we found that the correlations with these irrelevant features also increased with the layer depth.

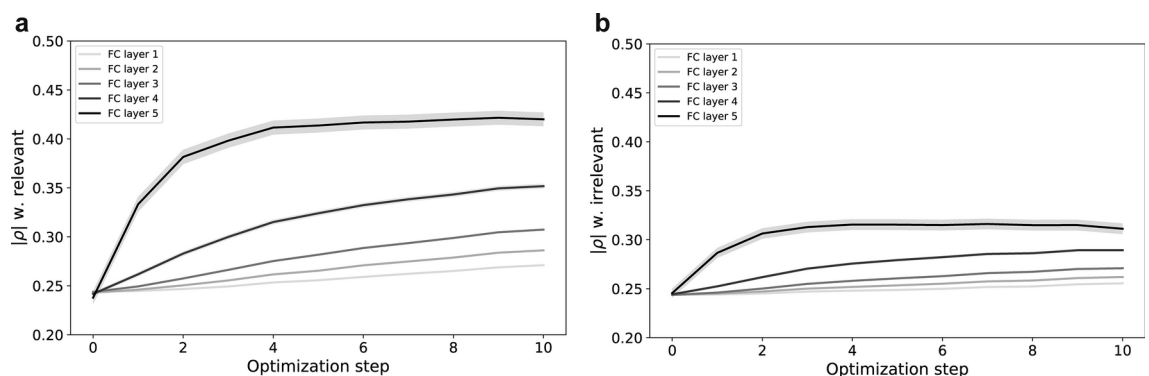


Fig. 3. The encoder's FC layers feature correlations. The average absolute correlations of encoders' FC layers with (a) the specific predictive feature of the problems they solved (either Color, Number, or Size), and (b) the other two non-predictive features (from either Color, Number, or Size). Error shades represent the 95% CI, based on the standard error of the means. The calculation of the correlations is detailed in the Methods section.

At the end of the optimization procedure, the output neuron of the encoder network is correlated with both the predictive feature and the irrelevant features (both distractors and constant). The stronger the correlation of the output neuron with the predictive feature, relative to its correlation with the distracting features, the better we expected the performance to be. To test this, we focused on the six problems in which the relevant feature was either color, number, or size, and there was one distracting feature, again: color, number, or size. For each of these problems, we computed the absolute Pearson correlations of the output neuron with the predictive and distracting features (taken from Fig. S8). We expected that performance in each of these problems would increase with the correlation with the relevant feature and decrease with the correlation with the irrelevant feature. Indeed, as depicted in Fig. 4, the logit of the performance ($\log \frac{p}{1-p}$, where the accuracies p are taken from Fig. 2) is correlated with the ratio of the absolute Pearson correlation of the output neuron with the predictive feature and the distracting feature (Wald t-test, p -value = 0.028).

Next, we studied how the correlation with the features increases during optimization. The loss function “seeks” some 1D predictable representation of the sequence of inputs. Considering the individual neurons at the output layer of the convolutional network part of the encoder, some co-vary with the sequence while others do not. Examples of two such neurons are depicted in Fig. 5a (left). As shown in Fig. 5a-b for a single problem, the optimization process makes larger changes to the synaptic weights from those neurons that co-vary strongly with the sequence order (e.g., blue in Fig. 5a-b) compared with low co-variance neurons (e.g., orange in Fig. 5a-b). This is the case across all problems (Fig. 5c). Consequently, the neurons in the encoder’s FC layer become strongly correlated with the sequence order (Fig. 5d). As a result, the encoder amplifies the representation of those features that co-vary with the sequence order (independently of whether they are the predictive or irrelevant features).

Together, our results indicate that the ANN’s ability to execute abstract reasoning without prior learning stems from two important properties: (1) The random convolutional layers extract features that are correlated with the relevant features. (2) The optimization process amplifies the response to those features that monotonically vary along the sequence.

The relation module

By construction, the vanilla model’s relation module is simple, implicitly assuming that the features change linearly. Therefore, one may naively expect that identifying a non-linear change in the predictive feature will be more challenging. However, any monotonically changing rule can be mapped into a linearly changing rule with a sufficiently complex encoder. Therefore, we tested our vanilla model in problems in which the change in the feature was *non-linear* (Fig. 6). We found that when the relevant feature was size, the performance for an exponential increase or a square root increase of this feature was comparable to that of a linear increase (Linear: 0.53 ± 0.01 ; Exp: 0.54 ± 0.01 ; Sqrt: 0.52 ± 0.01 , Fig. S9–10 right). Similarly, when the relevant feature was color, the model achieved comparable performance to the linear case, although with higher variability: performance was better for an exponential increase and worse for a square root increase (Linear: 0.70 ± 0.01 ; Exp: 0.75 ± 0.01 ; Sqrt: 0.64 ± 0.01 , Fig. S9–10 left). These results suggest that a relation module that assumes linear relationships can capture general monotonic relationships, substantially downsizing the hypothesis space of possible relationships. It would, however, be more difficult for the model to deal with non-monotonic rules. Indeed, when tested in problems where the predictive feature alternated between two of its values, the vanilla model performance was at a chance level (0.24 ± 0.01 , Fig. S11).

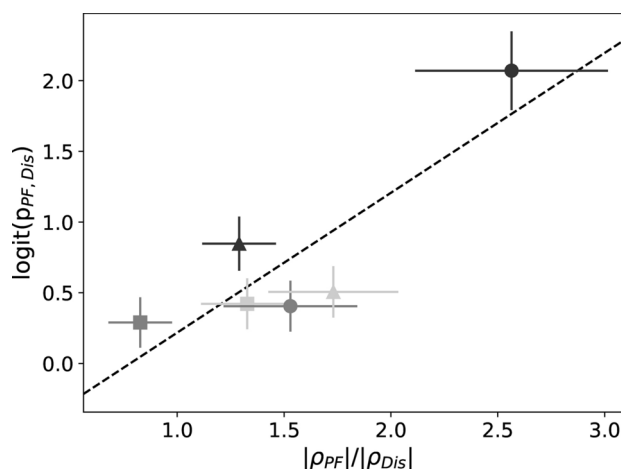


Fig. 4. The effect of distractors on accuracy. The figure depicts the relationship between the absolute correlation ratio with the relevant Predictive Feature ($|\rho_{PF}|$) and the Distracting feature $|\rho_{Dis}|$, and its consequential effect on networks’ accuracy in problems of that predictive feature with the corresponding distracting feature ($p_{PF,Dis}$). The predictive features were either Color (Dark Gray), Number (Medium Gray), or Size (Light Gray). The distracting features were either Color (Square), Number (Triangle), or Size (Circle). Error bars represent the 95% CI. The black dashed line depicts a linear regression analysis.

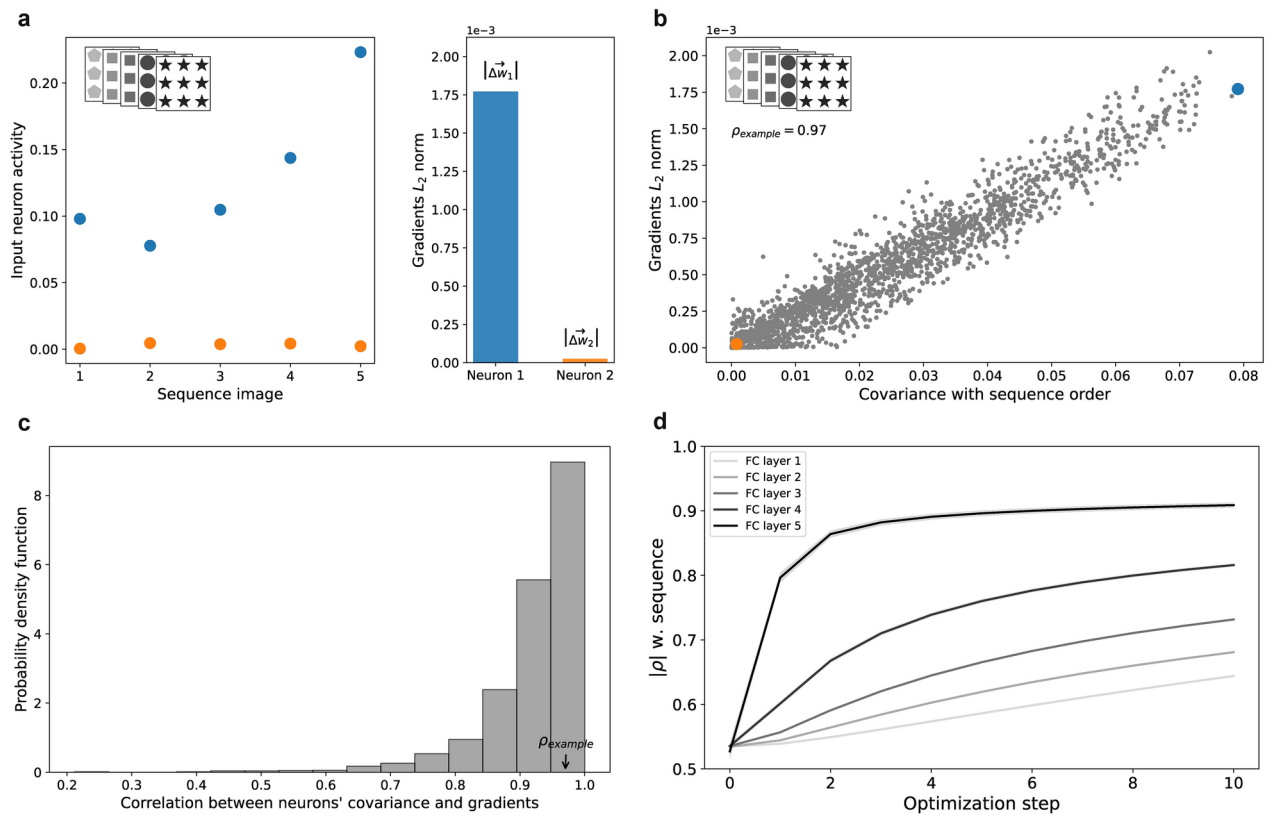


Fig. 5. Problem-solving mechanism. **(a)** Two example neurons' activity from the convolutional layers' output of a network (before optimization) when presented with the example problem of the inset. The blue neuron has a large covariance with the problem's image order, and the orange neuron has a small covariance with the order. The neurons' L2 gradient norms correlate with their respective image order covariances. **(b)** In this example network, the L2 gradient norms of the convolutional layers' output neurons are strongly correlated with their image-order covariances ($\rho_{\text{example}} = 0.97$). The two example neurons presented in (a) are highlighted. **(c)** Distribution of the correlations between L2 gradient norms and images' sequence-order across all problems. **(d)** The absolute correlation of the encoder's FC layers with the sequence order during the optimization process. Error shades represent 95% CI. Neurons' covariance and correlation calculations are explained in Methods.

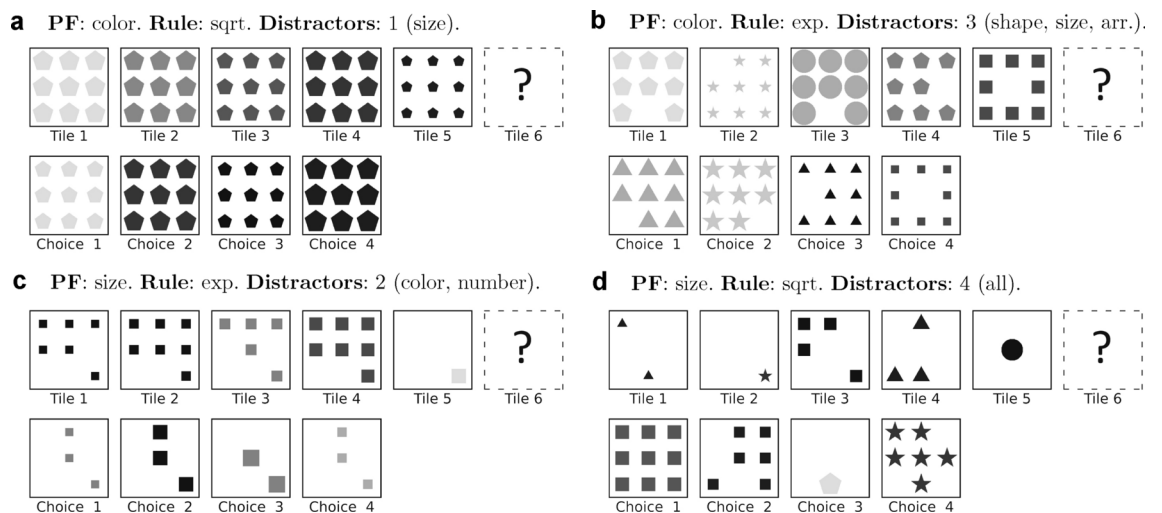


Fig. 6. Non linear rules. The predictive features' values in these tests increased as a square root (**a** and **d**) or exponentially (**b** and **c**). The rest of the features (non-predictive) are either constant or random. *The correct choices are all 3.*

In the vanilla model, the relation module is simple and general, and the encoder that finds appropriate image representations carries most of the “computational load”. However, we expect the complexity of the encoder and the complexity of the relation module to be interchangeable, to some extent. Thus, we can move some of the computational load from the encoder to the relation module without changing the performance. To test this, we simplified the encoder by removing the fully-connected layers, leaving only the convolutional layers, and complicated the relation module, by making it a more complex and expressive,

$$R_{\theta}(Z_{\phi}(\mathbf{x}_i), Z_{\phi}(\mathbf{x}_j)) = H_{\theta}(Z_{\text{conv}}(\mathbf{x}_i) \oplus Z_{\text{conv}}(\mathbf{x}_j)) \quad (3)$$

where the relation module H_{θ} takes a concatenation of the convolutional layers’ outputs to a single output neuron and has a network architecture similar to the vanilla model’s encoder’s FC layers (with twice the input dimension). Rather than optimizing both the encoder and the relation module, as in the vanilla model, we optimized only the relation module. This version of the model achieved an average accuracy of 0.59 ± 0.01 (Fig. S12) comparable to that of the vanilla model, demonstrating that it is possible to move the computational load from the encoder to the relation module without paying in performance.

To conclude this section, we demonstrated two ways for carrying the computational load. Either the encoder carries most of the load by extracting the relevant feature in a manner that a simple linear relation module is sufficient for capturing the rule. Alternatively, the relation module can carry the computational load. In that case, the relation module finds a *specific relation* between high-dimensional input representations, keeping the input representations fixed during problem-solving.

Knowledge crystallization

Our focus so far was the ability of the networks to solve problems without any training, that is, without any accumulation of information between problems. Embedded in our model, however, is the ability to accumulate knowledge. This is because problem-solving in our model is achieved through changes in synaptic weights. This motivated us to study how solving multiple problems affects performance. In humans, the improvement of performance due to the accumulation of knowledge by training is referred to as knowledge crystallization²³.

We first studied the extent to which the model can improve its performance on one predictive feature by practicing on that feature. Notably, in these practice sessions there was no feedback about the correct answer (in fact, the networks were exposed only to the sequences of 5 images and not to the possible answers). We found that networks that solved 1,000 easy problems with a specific predictive feature (without resetting the weights between problems) improved their accuracy on problems with that same predictive feature to 0.74 ± 0.01 (averaged over the three predictive features, Fig. 7), a substantial improvement from the average accuracy without prior training (0.58 ± 0.01). Notably, the improvement was not uniform across features. While performance on Number and Size substantially improved (Size: from 0.53 ± 0.01 to 0.69 ± 0.01 , Number: from 0.50 ± 0.01 to 0.84 ± 0.01), training on Color did not affect performance in Color problems (0.70 ± 0.01 in both conditions).

Interestingly, freezing the weights of the relation module resulted in an even better performance (0.80 ± 0.01 , Fig. S13). On the other hand, the improvement was only modest when the convolutional layers’ weights were frozen (0.65 ± 0.01 , Fig. S14).

This improvement in performance is analogous to knowledge crystallization. However, will training on one predictive feature improve performance when other predictive features are used? In humans, training on one abstract reasoning task does not necessarily generalize to other tasks²⁴. Similarly, we found that while training on one predictive feature improved performance in problems with that same predictive feature, it was detrimental when the networks were tested on problems with a different predictive feature (Fig. S15).

Will training on several predictive features improve network performance on those trained features? To address this question, we focused on the two predictive features that exhibited improvement with training, Number and Size. We used block training and tested performance on the most difficult problems of both types.

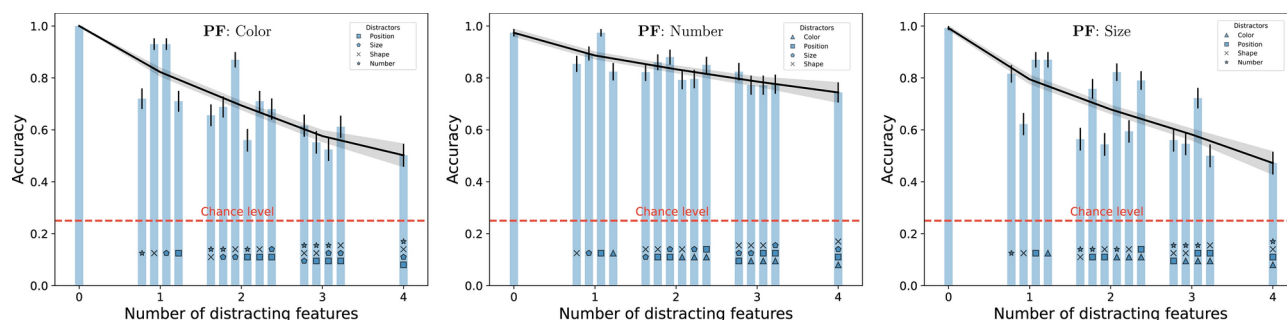


Fig. 7. Knowledge crystallization. The performance of networks that trained on 1,000 easy problems (without distracting features) of a certain predictive feature and tested on the different test conditions of that same predictive feature. Error bars correspond to 95% CI. The black line and its shade are the average accuracy per difficulty and 95% CI corresponding to the mean. The dashed line denotes the chance level given four choice images (0.25).

Considering the first block of training, extensively training the network with one predictive feature improves performance on that feature but not on the other feature (Fig. S16). Considering the second block, when this network is trained on the other predictive feature, the network quickly improves on that feature, but improvement on the first predictive feature quickly diminishes when the network trains on the other feature (Fig. S16a, b). Trying to resolve this by interleaving these two predictive-feature problems in short blocks of 5 problems does not change the result and the network seems unable to simultaneously improve on two predictive features (Fig. S16c, d).

To minimize conflict between the two features, we trained and tested the network in problems in which the competing non-predictive feature (Size or Number) was set at *the same constant value* (see Methods). We found that when training was done in two long blocks, the network only improved on the trained feature (Fig. 8a, b). By contrast, when training was done by interleaving many short blocks of 5 problems, the network improved in both features (Fig. 8c, d).

The fact that the network forgets one feature when training on the other is known in the machine learning literature as *catastrophic forgetting*²⁵, and indeed, interleaving has been shown to address this problem effectively²⁶. Similarly, the fact that interleaving is more effective than block training for learning is also well known in the cognitive literature as the interleaving effect^{27,28}.

Discussion

We found that naive randomly-initialized ANNs can perform abstract reasoning that does not rely on memorization when they are optimized at test time. This result has implications both the cognitive sciences and for machine learning.

Traditionally, abstract reasoning in humans has been considered a symbolic computation – a type of digital processing distinctly different from the analog nature of computation in ANNs^{29,30}. Recently, however, studies have shown that complex computations once attributed solely to symbolic processing can be accomplished by extensively trained ANNs^{31,32}. This is especially evident with large language models, which appear capable of performing certain forms of abstract reasoning¹². Nevertheless, critics of abstract reasoning in ANNs argue that this success may be due more to sophisticated memory retrieval than to genuine abstract reasoning^{13,14}. Our contribution is that we show that the tools used for training ANNs can also be used for exhibiting what resembles symbolic abstract reasoning without any training, hence without relying on memory recall.

A key element in our network's ability to perform abstract reasoning tasks is the convolutional part of the encoder. We found that these random convolutional layers are instrumental in extracting features correlated with relevant latent features. Interestingly, optimization of the convolutional layers was not necessary for achieving the performance, but did support knowledge accumulation. These results resonate with human-brain studies. In humans, early visual cortex regions act as general-purpose feature extractors, sensitive to basic features like

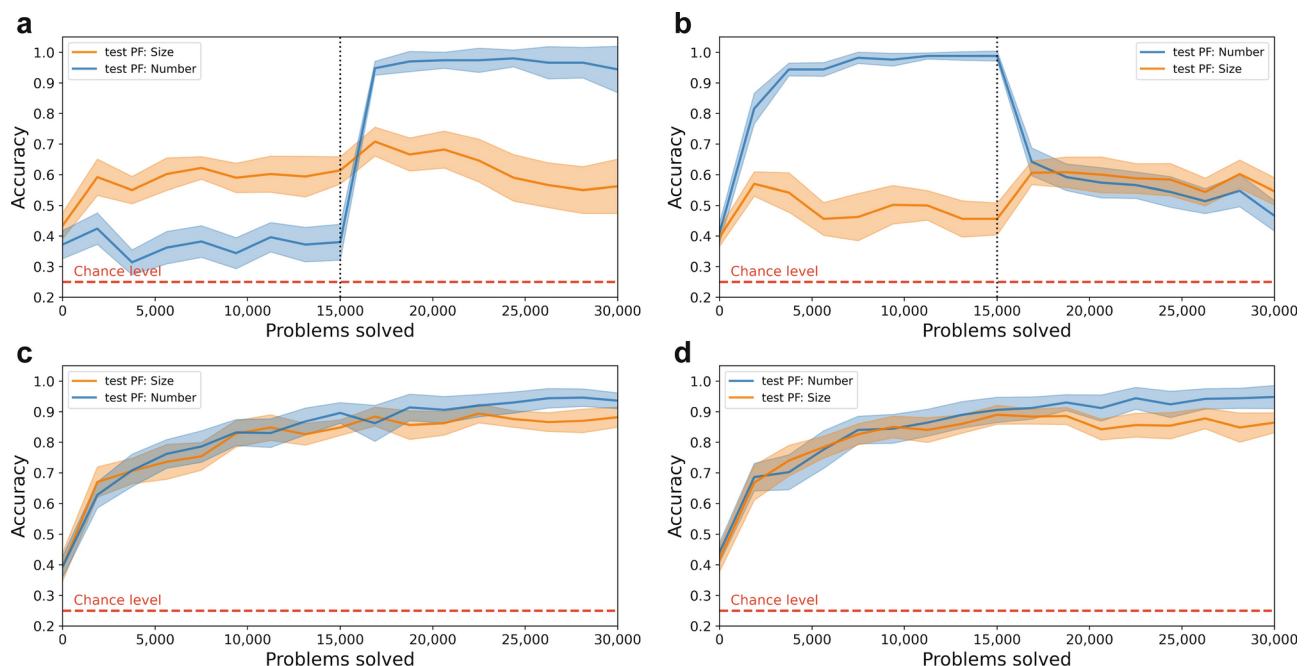


Fig. 8. Interleaving effect. Networks were trained on 15,000 problems in which the predictive feature was Size and 15,000 problems with predictive feature Number. The training problems were either presented in two large consecutive blocks (Size and then Number (a); Number and then Size (b)) or interleaved at a rate of five problems per predictive feature (Size and then Number (c); Number and then Size (d)). Errors correspond to 95% CI (see Methods).

orientation and direction. It seems unlikely that this low-level feature extraction changes with every problem presented to a human participant. They may, however, change with extensive training³³.

Highlighting the features that are relevant for the particular sequence of images of a particular problem was done in the deep layers of the encoder (fully-connected layers), together with the relation module. In humans, imaging studies suggest that higher cortical regions, such as the lateral prefrontal cortex, play an important role in abstract reasoning³⁴ and rule learning³⁵. We hypothesize that these higher cortical regions perform the analog of the optimization-based computation of highlighting the relevant feature (fully-connected layers of the encoder) and identifying its regularity (the relation module).

Notably, the computations performed by the fully-connected layers of the encoder and the relation module are somewhat interchangeable. This is because either of those networks can carry the main computational load. This suggests an interesting approach for finding relations by implementing a few very simple and general (applicable to different problems) relation modules, transferring a significant computational load of finding appropriate input representations to the encoder. Furthermore, given the interchangeability of complexity in the fully-connected layers of the encoder and the relational module, the separation between the encoder and relational module in the brain analog of this computation may not exist.

Our framework naturally generalizes to explaining knowledge acquisition through problem-solving (the practice in Fig. 7 was unsupervised, with no feedback). We found that training with many short interleaved blocks was substantially more effective than training with two long blocks. This resembles a similar observation in the cognitive sciences known as the interleaving effect^{27,28}. In the cognitive sciences, two competing theories have been used to explain this effect. In one, the interleaving effect is due to the enhanced problem-identification and feature-distinction required when solving two types of problems in close proximity³⁶. The second theory explains the interleaving effect by proposing that with interleaved training, the brain is continually engaged at retrieving the responses from memory – a process that enhances the consolidation of those memories^{37,38}. In contrast to these theories, our model has no explicit problem-identification or memory-consolidation mechanisms implemented. Rather, the interleaving effect is a manifestation of the well-known catastrophic forgetting phenomenon in machine learning^{25,26}.

We focused in this paper on the abstract reasoning of ANNs optimized by gradient descent. These models have shown to achieve performance levels that sometimes rival or even exceed human capabilities, especially in areas like language and visual processing, key aspects of human cognition^{10,39}. However, these successes have been achieved by scaling up model size and training data, with models trained on datasets vastly larger than what human children require to learn comparable skills^{40,41}. Thus, the sustainability of simply increasing network size and data volume as a path to further improvements has been doubted^{42,43}. Our findings suggest a potential alternative approach, in which ANNs, by optimizing their weights at test time, exhibit computational capacity in the absence of massive datasets. This approach offers a promising direction for AI development that prioritizes efficiency over scale.

Abstract reasoning consists of several computational facets. In this work, we focused on only one of them: the identification of relationships between images in order to infer the sequence completion, also termed inductive reasoning. Inductive reasoning is needed for solving many types of intelligence tests¹⁶. While modeling this facet, our model does not encapsulate other facets of abstract reasoning observed in humans. Specifically, our model does not incorporate working memory, limiting the regularities it can identify. It also does not explicitly perform the mapping computation required for analogical reasoning²⁰. Additionally, the model cannot solve a problem by breaking it into its sub-components⁴⁴. For example, to solve a Raven Progression Matrix, humans use the strategy of identifying common regularities in the rows and the columns. Our model was constructed only to find a regularity in a sequence. As with most ANN models, the model cannot interpret its choices. Finally, it lacks the ability to generate new images that follow the regularity it identifies. These limitations present opportunities for future research and suggest areas for improvement.

In humans, evidence suggests that abstract reasoning operates as a general computational process, analogous to a general-purpose computer that can handle any input. For example, an individual's performance on various cognitively demanding tests tends to correlate⁴⁵. As the tests require different prior knowledge, these correlations are taken as support for the hypothesis that a general ability, often termed general intelligence⁴⁶, underlies these diverse cognitive skills. Additionally, training on a specific cognitive task usually does not improve performance on unrelated tasks²⁴. This lack of transfer suggests that human abstract reasoning is indeed general, relying on general cognitive processes rather than specific learned patterns or memorized solutions. This somewhat resembles our model.

In conclusion, our work demonstrates that ANNs can exhibit abstract reasoning abilities without reliance on memory recall, opening pathways for further exploration of abstract reasoning mechanisms in both artificial systems and humans.

Methods

Code availability

The code for this paper was written using PyTorch²². The code that generates test problems and applies the model to solve them is available at https://github.com/Tomer-Barak/learning-independent_abstract_reasoning.

Network architectures

The encoder ($Z_\phi(\mathbf{x})$) consisted of two main components (see Supplementary Fig. S1): three convolutional layers (kernel sizes: 2, 2, and 3; strides: all 1; padding: all 1) and five Fully-Connected (FC) layers (number of neurons: 200, 100, 50, 10, 1). Three ReLU activation functions were applied after each convolutional layer, and two Max-Pool layers (kernels: 4 and 6, strides: all 1) were applied after the second and third convolutional (+ReLU) layers.

Four tanh activation functions were applied after each FC layer, except the last one, which had no activation function and remained a linear transformation.

The vanilla model's relation module consisted of a single parameter as written in equation (2). The more complex relation module written in equation (3) was implemented by a five-layer fully-connected network (number of neurons: 200, 100, 50, 10, 1). Four tanh activation functions were applied after each of this relation module's layers, except the last one, which had no activation function and remained a linear transformation.

Sequential visual reasoning tests

Each image of the tests was constructed using the following five features: the number of objects in an image (possible values: 1 to 9), their shade (6 linearly distributed grayscale values), their shape (circle, triangle, square, star, hexagon), their size (6 linearly distributed values for the shapes' enclosing circle circumference), and arrangement (a vector of grid positions that was used to place the shapes in order).

As written in the paper, the choice images' non-predictive features followed the same rules they abide by in the sequence (constant or randomly changing). The predictive feature followed the sequence rule only in the correct choice and was randomly chosen from the remaining feature values in the incorrect choices. We restricted the possibility of having a repeated choice image in the same problem. If a repeated image was generated by chance, we generated another one to replace it.

Average accuracies

In the paper, we report networks' average accuracies/performance in different experiments. For example, the vanilla model's average accuracy was 0.58 ± 0.01 . These numbers were obtained (except in the knowledge crystallization section, discussed below) in the following way. For each predictive feature relevant to the experiment, we considered all its test conditions of different difficulties. There were five features, one predictive and the other four either constant or distracting, amounting to $2^4 = 16$ test conditions per predictive feature. We tested randomly initialized networks in 500 problems in each test condition (each problem with a different initialized network) and obtained their success rate in that test condition. To estimate the errors, we calculated the standard error of the mean of a sample of Binomial random variables based on the success rate and the number of samples (500). To obtain the average accuracy of that predictive feature, we averaged the success rates over all test conditions and propagated the errors accordingly. For the total average accuracy, we averaged the accuracies of the experiment's relevant predictive features and propagated the errors.

In the knowledge crystallization section, the average accuracies (e.g., Fig. 7) were obtained by training 50 networks in each predictive feature on 1000 easy problems (without distractors) of that predictive feature. After training, the networks solved 10 test problems in each of the 16 test conditions of a given predictive feature (results for networks that trained on one predictive feature and tested on another are shown in Fig. S15). We then calculated the average success rate of the networks in each test condition, using the standard error of the mean of Binomial random variables as errors. For the total average accuracy, we averaged across the different test conditions and all the relevant predictive features of the experiment, propagating the errors.

In the blocks versus interleaving experiments (Fig. 8, S16-S18), we trained 20 networks (in each of the figure panels) on 30,000 easy problems of two training predictive features. The training was either in two big blocks or interleaved into small five-problem blocks. After every 1875 training problems, we tested the networks on 25 difficult problems of the two predictive features. In Fig. S16, the easy and difficult problems were such that there were no distractors or all of the distractors. In Fig. 8, both the easy and difficult problems of Size had a fixed value of Number (5 shapes). Accordingly, the easy and difficult problems of Number had a fixed value of Size (the 5th size value).

Correlations

To calculate an encoder neurons' correlations with a particular feature (color, number, or size; Fig. 3 and S9), we generated artificial testing examples corresponding to that feature: 20 images for each of the feature's six possible values (120 examples overall) where the rest of the features' values were drawn randomly. We applied these examples to the network and recorded its neurons' activity. Based on the neurons' activity, we calculated each of the neurons' correlation with the feature values. Finally, we averaged the correlations across the layers.

To generate Fig. 3a, we average the correlations of networks that solved the three possible predictive features with the predictive features they solved. For Fig. 3b, we calculated the correlations with the other (non-predictive) features. In both figures, we averaged over the 16 test conditions of a given predictive feature, 50 problems per test condition, each problem solved by a different naive network. Thus, overall, the results are average over $3 \times 50 \times 6 = 900$ networks. The complete results of these simulations, before averaging over networks and test conditions, are shown in Fig. S8. To calculate the errors of the correlations, we estimated the standard error of the mean of the average correlations of the different networks. We propagated these errors when we averaged the correlations across test conditions and different predictive features.

To calculate the correlations (or covariance) of neurons with the sequence (Fig. 5b-d), we applied the sequence images to the network and recorded its neural activity. Then, we calculated for each neuron the correlation (or covariance) between its activity and the sequence order indices of the images. In Fig. 5b-c, we calculated the covariance with the sequence of neurons taken from the output of the encoder's 3rd convolutional layer, while in Fig. 5d, we calculated correlations with the sequence of the FC layers' neurons. The errors in the latter case were calculated like those of the feature correlations.

Figure 4

To obtain the values of this plot, we considered test conditions with one distracting feature that was either Color, Number, or Size (as those are the features for which we were able to calculate networks' correlations). In total,

there were 6 such test conditions (2 for each predictive feature). For each of the test conditions, the y-axis value is the logit function ($\text{logit}(p) = \ln \frac{p}{1-p}$) of the success rate of 500 problems of that test condition, each solved by a different naive network, obtained from Fig. 2. For the errors of these values, we propagated the success rate errors through the logit function. For the x-axis values, we obtained networks' average absolute correlations with the predictive feature of the problems they solved (after solving them) and compared that with the absolute correlation of the distracting features. The values and errors were obtained from Fig. S8, and the errors were propagated through the ratio. The linear regression analysis was conducted using SciPy's⁴⁷ linear regression function.

Figure 5c

The histogram in Fig. 5c was obtained by considering the three predictive features (color, number, and size) and each of their 16 test conditions, 50 problems in each test condition. For each problem, we calculated the correlations between the convolutional layers' output neurons' co-variance with the sequence order and the L2 gradient norm of those neurons and averaged over the neurons. Finally, we plotted the histogram of those averages.

Data availability

No datasets were generated or analyzed during the current study. The visual reasoning tests were generated in real-time by an algorithm (included in the Supplementary materials).

Received: 28 November 2023; Accepted: 31 October 2024

Published online: 08 November 2024

References

1. Lohman, D. F. Complex Information Processing and Intelligence. In Sternberg, R. J. (ed.) *Handbook of Intelligence*, 285–340, <https://doi.org/10.1017/CBO9780511807947.015> (Cambridge University Press, 2000).
2. Sternberg, R. J. Component processes in analogical reasoning. *Psychol. Rev.* **84**, 353–378. <https://doi.org/10.1037/0033-295X.84.4.353> (1977).
3. Sternberg, R. J. Components of human intelligence. *Cognition* **15**, 1–48. [https://doi.org/10.1016/0010-0277\(83\)90032-X](https://doi.org/10.1016/0010-0277(83)90032-X) (1983).
4. Kaplan, R. M. & Saccuzzo, D. P. *Psychological Testing: Principles, Applications, and Issues*. (Wadsworth/Thomson Learning, 2009).
5. Raven, J., Raven, J. C. & Court, J. H. *Manual for Raven's Progressive Matrices and Vocabulary Scales* (Pearson, 1998) (OCLC: 697438611).
6. Mitchell, M. How do we know how smart AI systems are?. *Science* **381**, eadj5957. <https://doi.org/10.1126/science.adj5957> (2023) <https://www.science.org/doi/pdf/10.1126/science.adj5957>.
7. Hersche, M., Zeqiri, M., Benini, L., Sebastian, A. & Rahimi, A. A neuro-vector-symbolic architecture for solving Raven's progressive matrices. *Nat. Mach. Intell.* **5**, 363–375. <https://doi.org/10.1038/s42256-023-00630-8> (2023).
8. Santoro, A. et al. A simple neural network module for relational reasoning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17*, 4974–4983 (Curran Associates Inc., Red Hook, NY, USA, 2017).
9. Barrett, D., Hill, F., Santoro, A., Morcos, A. & Lillicrap, T. Measuring abstract reasoning in neural networks. In *International conference on machine learning*, 511–520 (PMLR, 2018).
10. OpenAI. GPT-4 Technical Report, <https://doi.org/10.48550/arXiv.2303.08774> (2023). [ArXiv:2303.08774](https://arxiv.org/abs/2303.08774) [cs].
11. Kim, Y., Shin, J., Yang, E. & Hwang, S. J. Few-shot visual reasoning with meta-analogical contrastive learning. *Adv. Neural. Inf. Process. Syst.* **33**, 16846–16856 (2020).
12. Webb, T., Holyoak, K. J. & Lu, H. Emergent analogical reasoning in large language models. *Nature Human Behaviour* 1–16 (2023). Publisher: Nature Publishing Group UK London.
13. McCoy, R. T., Yao, S., Friedman, D., Hardy, M. & Griffiths, T. L. Embers of Autoregression: Understanding Large Language Models Through the Problem They are Trained to Solve, <https://doi.org/10.48550/arXiv.2309.13638> (2023). [ArXiv:2309.13638](https://arxiv.org/abs/2309.13638) [cs].
14. Biever, C. ChatGPT broke the Turing test - the race is on for new ways to assess AI. *Nature* **619**, 686–689. <https://doi.org/10.1038/d41586-023-02361-7> (2023).
15. Azulay, A. & Weiss, Y. Why do deep convolutional networks generalize so poorly to small image transformations?. *J. Mach. Learn. Res.* **20**, 1–25 (2019).
16. Siebers, M., Dowe, D. L., Schmid, U., Hernández-Orallo, J. & Martínez-Plumed, F. Computer models solving intelligence test problems: Progress and implications. *Artif. Intell.* **230**, 74–107. <https://doi.org/10.1016/j.artint.2015.09.011> (2015) (ISBN: 9780999241103 Publisher: Elsevier B.V.).
17. Sung, F. et al. Learning to compare: Relation network for few-shot learning. *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.
18. Hill, F., Santoro, A., Barrett, D., Morcos, A. & Lillicrap, T. Learning to Make Analogies by Contrasting Abstract Relational Structure (2018).
19. Lu, H., Wu, Y. N. & Holyoak, K. J. Emergence of analogy from relation learning. *Proceedings of the National Academy of Sciences* **116**, 4176–4181, <https://doi.org/10.1073/pnas.1814779116> (2019).
20. Lu, H., Ichien, N. & Holyoak, K. J. Probabilistic analogical mapping with semantic relation networks. *Psychol. Rev.* **129**, 1078–1103. <https://doi.org/10.1037/rev0000358> (2022).
21. Dauphin, Y. N., de Vries, H., Chung, J. & Bengio, Y. RMSProp and equilibrated adaptive learning rates for non-convex optimization. *CoRR arxiv:1502.04390* (2015)
22. Paszke, A. et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems* **32**, 8024–8035 (Curran Associates, Inc., 2019).
23. Horn, J. L. Intelligence-why it grows, Why it declines. In *Human Intelligence*, 22 (Routledge, 1972).
24. Jaeggi, S. M., Buschkuhl, M., Jonides, J. & Perrig, W. J. Improving fluid intelligence with training on working memory. *Proc. Natl. Acad. Sci.* **105**, 6829–6833 (2008).
25. French, R. M. Catastrophic forgetting in connectionist networks. *Trends Cogn. Sci.* **3**, 128–135. [https://doi.org/10.1016/S1364-6613\(99\)01294-2](https://doi.org/10.1016/S1364-6613(99)01294-2) (1999).
26. Robins, A. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connect. Sci.* **7**, 123–146. <https://doi.org/10.1080/09540099550039318> (1995).
27. Kornell, N. & Bjork, R. A. Learning concepts and categories: Is spacing the enemy of induction?. *Psychol. Sci.* **19**, 585–592. <https://doi.org/10.1111/j.1467-9280.2008.02127.x> (2008).
28. Brunmair, M. & Richter, T. Similarity matters: A meta-analysis of interleaved learning and its moderators. *Psychol. Bull.* **145**, 1029–1052. <https://doi.org/10.1037/bul0000209> (2019).

29. Turing, A. M. & others. On computable numbers, with an application to the Entscheidungsproblem. *J. Math.* **58**, 5 (1936).
30. Turing, A. M. Computing machinery and intelligence. *Mind* **LIX**, 433–460. <https://doi.org/10.1093/mind/LIX.236.433> (1950) <https://academic.oup.com/mind/article-pdf/LIX/236/433/30123314/lix-236-433.pdf>.
31. Krizhevsky, A., Sutskever, I. & Hinton, G. E. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, vol. 25 (Curran Associates, Inc., 2012).
32. Campbell, D., Kumar, S., Giallanza, T., Cohen, J. D. & Griffiths, T. L. Relational constraints on neural networks reproduce human biases towards abstract geometric regularity. <https://doi.org/10.48550/arXiv.2309.17363> (2023). [ArXiv:2309.17363](https://arxiv.org/abs/2309.17363) [q-bio].
33. Ahissar, M. & Hochstein, S. The reverse hierarchy theory of visual perceptual learning. *Trends Cogn. Sci.* **8**, 457–464. <https://doi.org/10.1016/j.tics.2004.08.011> (2004).
34. Gray, J. R., Chabris, C. F. & Braver, T. S. Neural mechanisms of general fluid intelligence. *Nat. Neurosci.* **6**, 316–322. <https://doi.org/10.1038/nn1014> (2003).
35. Mansouri, F. A., Freedman, D. J. & Buckley, M. J. Emergence of abstract rules in the primate brain. *Nat. Rev. Neurosci.* **21**, 595–610. <https://doi.org/10.1038/s41583-020-0364-5> (2020).
36. Rohrer, D. Interleaving helps students distinguish among similar concepts. *Educ. Psychol. Rev.* **24**, 355–367. <https://doi.org/10.1007/s10648-012-9201-3> (2012).
37. Dunlosky, J., Rawson, K. A., Marsh, E. J., Nathan, M. J. & Willingham, D. T. Improving students' learning with effective learning techniques: Promising directions from cognitive and educational psychology. *Psychol. Sci. Public Interest* **14**, 4–58 (2013).
38. Krug, D., Davis, T. B. & Glover, J. A. Massed versus distributed repeated reading: A case of forgetting helping recall?. *J. Educ. Psychol.* **82**, 366 (1990).
39. Ho, J., Jain, A. & Abbeel, P. Denoising diffusion probabilistic models. <https://doi.org/10.48550/arXiv.2006.11239> (2020). [ArXiv:2006.11239](https://arxiv.org/abs/2006.11239).
40. Laurence, S. & Margolis, E. The poverty of the stimulus argument. *Br. J. Philos. Sci.* **52**, 217–276. <https://doi.org/10.1093/bjps/52.2.217> (2001).
41. Frank, M. C. Bridging the data gap between children and large language models. *Trends Cogn. Sci.* [SPACE] <https://doi.org/10.1016/j.tics.2023.08.007> (2023).
42. Sun, C., Shrivastava, A., Singh, S. & Gupta, A. *Revisiting Unreasonable Effectiveness of Data in Deep Learning Era*, 843–852 (2017).
43. Hestness, J. *et al.* Deep learning scaling is predictable, empirically. <https://doi.org/10.48550/arXiv.1712.00409> (2017). [ArXiv:1712.00409](https://arxiv.org/abs/1712.00409) [cs, stat].
44. 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. *Psychol. Rev.* **97**, 404–431. <https://doi.org/10.1037/0033-295X.97.3.404> (American Psychological Association, 1990).
45. Gottfredson, L. S. *The General Intelligence Factor* (1998).
46. Spearman, C. General intelligence, objectively determined and measured. *Am. J. Psychol.* **15**, 201–293. <https://doi.org/10.2307/1412107> (1904).
47. Virtanen, P. *et al.* SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nat. Methods* **17**, 261–272. <https://doi.org/10.1038/s41592-019-0686-2> (2020).

Acknowledgements

This work was supported by the Gatsby Charitable Foundation. Y.L. holds the David and Inez Myres Chair in Neural Computation.

Author contributions

T.B. conducted the experiments, T.B. and Y.L. analyzed the results, wrote, and reviewed the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1038/s41598-024-78530-z>.

Correspondence and requests for materials should be addressed to T.B.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2024