Robert Anderson

PSYC78D7: Research Dissertation

CC1718-04

A Relational Priming Approach to Solving Raven's Progressive Matrices

Birkbeck College, MSc Cognition and Computation

Robert Anderson

A Relational Priming Approach to Solving Raven's Progressive Matrices

Birkbeck College, MSc Cognition and Computation

This work is submitted in partial completion of the MSc in Cognition and Computation.

I state on this date of 23 August 2019 that this work is my own.

Word count (excluding abstract and references): 10,736

A Relational Priming Approach to Solving Raven's Progressive Matrices

Abstract

Analogy has been viewed as a central and core component of human intelligence. Many theories of analogical reasoning assume it involves higher order cognitive processes such as abstraction, reasoning and creativity. Relational priming has been shown to provide an account whereby the capacity to complete semantic-based proportional analogies could arise as a by-product of simpler cognitive processes: perception, memory and pattern-completion. In a recurrent connectionist network, the perception of a novel stimulus (the analogue target) is influenced by the context of the previously perceived exemplars (the analogue source) producing a novel output which completes the analogy. Here, we present a model which adapts the relational priming approach to a visual task often regarded as the hallmark of analogical reasoning tests, Raven's Progressive Matrices. The network is trained to learn how basic shapes are transformed through simple operations such as scaling, rotation and shading. The model is then able to complete many 2x2 matrices using only the relational priming mechanism and without recourse to semantic representations nor procedures for structure alignment. Furthermore, we demonstrate a novel approach that combines the result of multiple simultaneous priming operations in order to solve much more complicated 3x3 matrices. We propose that this concurrent relational priming enables more complex analogical reasoning and suggest that it could be a driver for general intelligence.

Acknowledgements

I would like to thank my supervisor, Denis Mareschal, for his many contributions to this project. It would not have been possible without his helpfulness, friendliness, enthusiasm, knowledge and encouragement. I have learned so much through this work.

I would also like to thank Rick Cooper for suggesting Raven's Progressive Matrices over a year ago when I had little idea what to do for my research. They have proved to be a very rich topic.

Finally, thanks to my lovely wife Zaklina, and my amazing children Tasha and Novak, for their love, support and encouragement throughout this project.

Table of Contents

A Relational Priming Approach to Solving Raven's Progressive Matrices

Introduction

Analogy-making, the capacity to perceive and exploit patterns of relational similarity between perceptions, memories and concepts, has been argued to form one of the key components of human intelligence (French, 1995; Hofstadter, 1996).

Traditionally, analogical reasoning has been described as the transfer of relational information from one source domain (or analogue) to a target domain (Vosniadou & Ortony, 1989). Sometimes the things being compared are very similar, but they can be different such as comparing an atom to the solar system. Solving an analogical problem requires that one finds some similarity between the source and target domains and how the features within each domain map to one another.

Aristotle defined analogy as 'an equality of proportions … involving at least four terms … when the second is related to the first as the fourth is to the third.' (Aristotle, *Poetics*). An example would be *west* is to *east* as *left* is to *right*, often written in the form *west* : *east* :: *left* : *right* or in matrix form. This type of formulation, known as a *proportional analogy*, is often used in intelligence tests where the task is to complete puzzles such as those in Figure 1.

tree	leaf
flower	?

tomato	red
broccoli	?

sphere	circle
cube	?

cup	Dionysus
shield	?

Figure 1: Typical examples of proportional analogies shown in matrix form. The task is to complete the empty square with the word or concept which best relates to the word in the bottom left, in the same way as the word in the top-right relates to the top-left. The last example is due to Aristotle and illustrates the problem of cultural and educative bias in these tests. The answers are *petal*, *green*, *square* and *Ares*.

From the cognitive perspective, how we make analogies is not fully understood. It appears to call on many faculties: perception, memory, reasoning, logic (inference, deduction), pattern-matching, creativity, learning (Gust, Krumnack, Kuhnberger, & Schwering, 2008). Two main theories dominate the debate.

When a person comes across a novel problem, they are reminded of something similar, retrieve it and use it, with some adaptation, to solve the problem. Many models of analogy work in a similar fashion. The source analogue is perceived. A similar representation is fetched from memory. The corresponding features in the source and target are mapped to each other. This mapping is then used to adapt the example's solution and thereby solve the problem (VanLehn, 1998). By this account, the perception step occurs before the mapping step. This is the view endorsed, among others, by advocates of structure-mapping theory (Gentner, 1983).

The alternative view is that analogy-making is an automatic and simultaneous cognitive function inseparable from perception. In the same way as context influences our perception of colours or sounds, it also influences the analogies we are likely to form. On this account, the mapping between two analogues cannot be separated from the process of initially perceiving them. Moreover, the supporters of this view argue that the role of analogy is central to what they term *high-level perception* encompassing a far wider range of abilities: categorisation, episodic memory and learning, among others (Chalmers, French, & Hofstadter, 1992; French, 1995; Hofstadter, 1996; Hofstadter & Sander, 2013).

Both of these approaches have engendered many successful computational models of analogy in various domains, but neither offers much in the way of explanation as to how such abilities might arise developmentally. Leech et al (2008) propose that analogy-completion can arise through a simpler mechanism called *relational priming*. Their model is tasked with solving a test of analogical reasoning based on semantic-based proportional analogies of the

form *A* is to *B* as *C* is to *D*. The model is connectionist and uses a biologically-plausible learning algorithm. Contrary to many models, it avoids explicitly targeting analogy itself, rather analogy completion emerges automatically and simultaneously from the more basic mechanisms of perception, memory and pattern-matching. Specifically, analogies are completed using a mechanism in which the perception of the novel input (the C component of the analogy) is influenced or *primed* by the context of the exemplars (the A and B components).

In this paper we present a model inspired by the Leech et al. (2008) model. Our target domain is a visual one: Raven's Progressive Matrices (RPM), often considered the hallmark of tests of analogical reasoning ability. The following section describes RPMs in more detail and what they measure.

Raven's Progressive Matrices

The proportional analogy task is verbal in nature, such that some level of language proficiency and education is a prerequisite. Even after attempts to circumvent this by using pictures instead of text, such as in Goswami & Brown's study of analogical reasoning in children (1990), there are still suggestions of cultural and educational bias since domain knowledge is still required.

Raven's Progressive Matrices (1938), provide an alternative non-verbal test frequently used by researchers. The test requires inductive reasoning about abstract geometric patterns and the relationships between them. It is designed to be usable by subjects of any age, gender, nationality or education and covers a wide range of mental ability. It is widely used across educational, vocational and clinical populations.

Figure 2 shows a typical example similar to the ones used by Raven[1]. Each problem is a 2x2 or 3x3 matrix with the bottom-right cell left blank. To solve the problem the subject must choose the missing picture from the eight candidates labelled A to H.

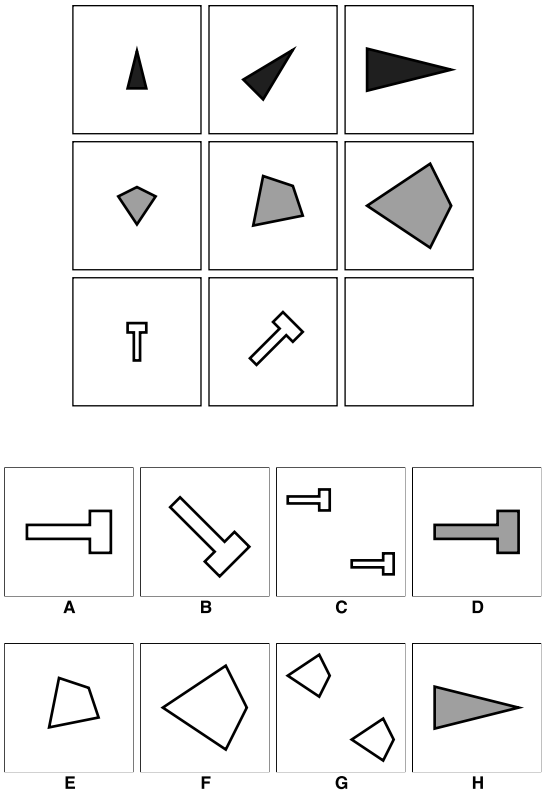

Figure 2: A typical 3x3 problem similar to problems from Raven's Progressive Matrices. The answer in this case is A.

RPMs were developed by John C. Raven as part of his Masters dissertation. His supervisor, Charles Spearman, coined the term *general intelligence* to explain the high correlations of schoolchildren's scores on different academic tests. He believed that the intercorrelations could largely be explained by a single underlying factor, *general cognitive*

[1] Note that official Raven's Progressive Matrices test materials are protected by copyright. Here, and elsewhere in the paper, depicted 2x2 and 3x3 matrices are not actual problems but Raven-like constructed equivalents.

ability, which became known as Spearman's *g*. Raven's matrices were designed to measure *g* (Carpenter, Just, & Shell, 1990).

Raven's tests gained popularity because they were simple to administer and proved to be a very strong predictor of ability on many other intelligence tests, not just of abstract reasoning, but on a wide variety of verbal, mathematical and geometric tests. In a classic scaling analysis study (Figure 3), Snow et al. (1984) placed RPMs at the very centre indicating that it is the single most predictive test of general intelligence.

In spite of their success and wideness of use, there is still much discussion about what exactly the tests measure (Mackintosh & Bennett, 2005). Raven and Spearman believed that they measure the *eductive* component of *g*, from the Latin for 'to draw out': 'the ability to make meaning out of confusion' (Raven, 2000, p. 2). For Lovett and Forbus, RPMs measure 'individuals' abilities to make effective analogies' (Lovett & Forbus, 2017, p. 60).

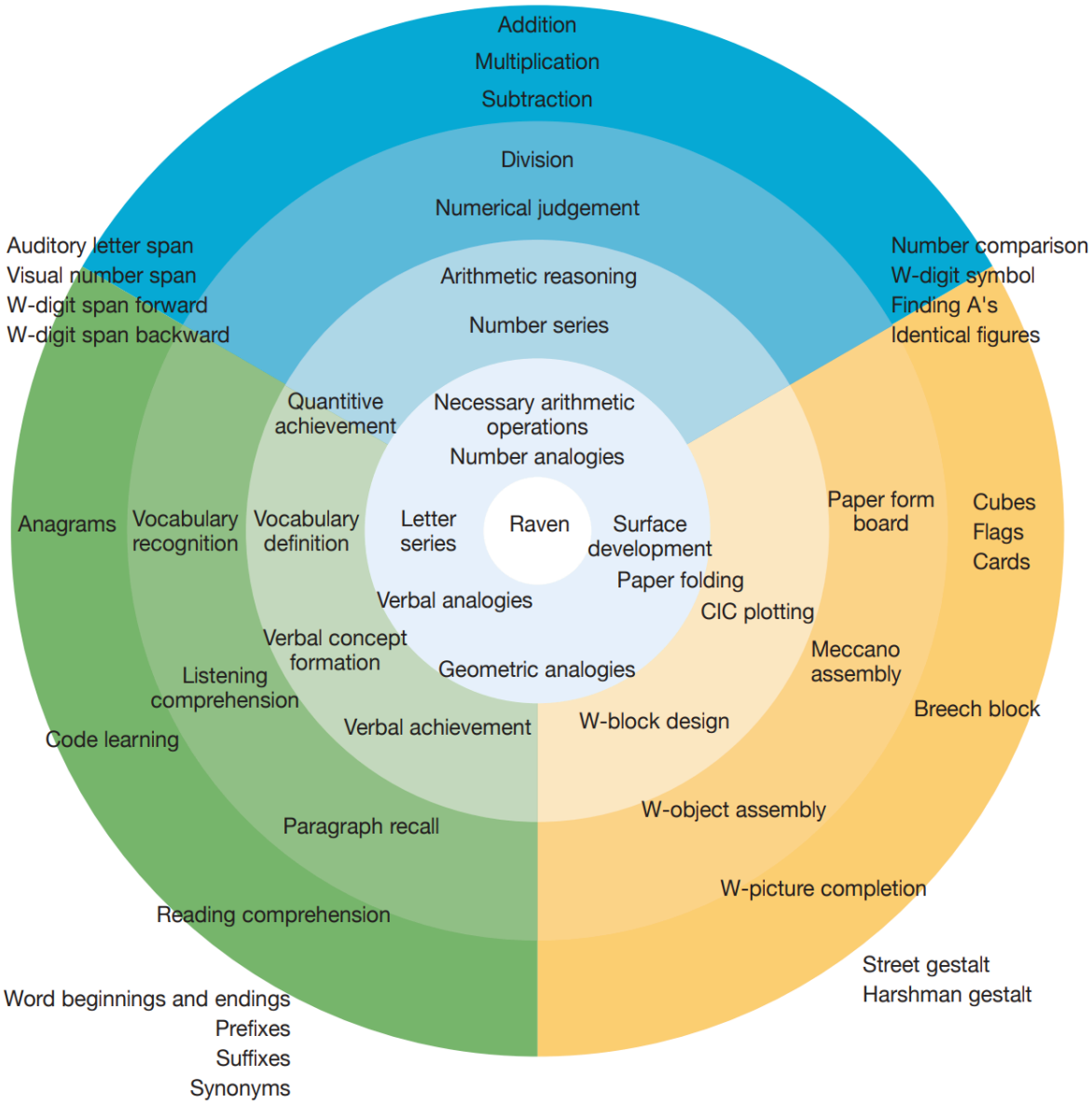

Figure 3: Raven's Progressive Matrices are central to a multidimensional scaling analysis of ability tests across several datasets. Highly correlated tests appear near each other. The outwardly radiating concentric circles indicate decreasing test complexity. The coloured sections group tests by figural, verbal and numerical flavour. The original figure is from Snow, Kyllonen & Marshalek (1984, p. 84) (Reprinted from the adapted version in Gray & Thompson, 2004, p. 472)

There are several versions of Raven's Progressive Matrices. The original and most common are the Standard Progressive Matrices (Raven, 1938). This set contains five sections

of twelve problems each labelled A to E in increasing order of difficulty. Sections A and B are 2x2 matrices; sections C, D and E are 3x3. All problems are presented in black and white. Subsequently several other RPM sets were published targeting different populations: the Advanced Progressive Matrices, designed for above-average intelligence adolescents and adults; and the Coloured Progressive Matrices, designed for children aged 5-11 (Raven, Raven, & Court, 1998).

As a domain for cognitive modelling, Raven's Progressive Matrices offer several advantages over proportional analogies. RPMs are an established psychometric test, so there exists a wealth of human performance statistics with which we can contrast a model's performance. Problems can be categorised by difficulty level more readily than word-based or concept-based proportional analogies. They are devoid of cultural and educational bias. Above all, they seem to capture the essence of analogical reasoning: the transfer of relational information from a source to a target.

Research background

This section examines the existing literature, firstly that pertaining to cognitive models that aim to solve RPMs, followed by some relevant models of analogy from other domains.

Existing cognitive models targeting Raven's Progressive Matrices

The earliest work on algorithmic solving of RPMs is due to Hunt (1974) who presented a theoretical account including two unimplemented algorithms for solving RPMs: one visual, which he named *Gestalt* using graphical representations and perceptually-based operations such as extending lines; the second he named *analytic*, used feature-based representations and logical operations. He demonstrated that the first seven of twelve problems in Set I of the Advanced Progressive Matrices could be solved by the Gestalt method, but that only the analytic approach could solve all of them. Mackintosh & Bennett

(2005) suggest this is evidence of separate perceptual and analytical cognitive abilities that contribute differently to subjects' scores.

Carpenter et al. (1990) is the most influential early work. It includes a psychological study of human performance including eye-fixations and verbal reports of subjects attempting the Advanced Progressive Matrices, the hardest version of the test. It also presents a symbolic rule-based production system using tailored text-based representations and includes frequently cited taxonomy of the types of transformations that appear in the Raven's tests.

A structure-mapping model was proposed and refined in several iterations (Lovett & Forbus, 2017; Lovett, Forbus, & Usher, 2007, 2010), the most successful of which succeeded on 93% of 60 problems. The approach is based on Gentner's structure-mapping theory (1983). It operates over structured representations, *i.e.*, symbolic representations consisting of entities, attributes, and relations, which the model automatically extracts from vector graphics.

The account by (McGreggor, Kunda, & Goel, 2010) uses a fractal coding and transformation approach which solved 58% of 60 problems attempted. Fractal coding uses an algorithm to encode transformations from a source to a target image as a set of repeated affine transformations. The model is unusual in that it extracts similarity metrics directly from scans of the Raven's test booklet along with the resultant imprecise alignments and pixel artefacts. At no point are the inputs converted to coded representations of shapes, lines or edges – only the raw pixel intensities are used.

A system using the ACT-R cognitive architecture (Ragni & Neubert, 2012) successfully solved 92% of 66 problems attempted. The graphical elements of the matrix were represented symbolically. A hybrid approach which combined a symbolic rule layer with a deep-learning neural network (Mekik, Sun, & Yun Dai, 2017, 2018) used abstract visual features extracted from raw images with no intervention from human annotators. It

solved 78% of 108 3x3 software-generated problems similar to Raven's. A Bayesian account (Little, 2012) with hand-coded inputs solved 78% of 66 items attempted.

From a more AI perspective, further models have been proposed (Barrett, Hill, Santoro, Morcos, & Lillicrap, 2018; Steenbrugge, Leroux, Verbelen, & Dhoedt, 2018; Hill, Santoro, Barrett, Morcos, & Lillicrap, 2019) which explore performance on Raven-like tasks using various state of the art neural network architectures. Although the focus of these papers is algorithmic rather than cognitive, some of their approach can be leveraged for deeper understanding of specific aspects of analogy. Specifically, Barrett et al. (2018) cover methods for separating test and training data in such a way as to differentiate problems which require interpolation from those that require extrapolation.

Other studies relevant to this research include the work by Matzen et al. (2010) who developed matrix generation software based on an analysis of the structure of Raven's matrices. It includes a norming study which demonstrates that the matrices generated by their software have psychometric properties comparable to the original RPM problems. Other approaches to the generation of Raven-like matrices exist (Wang & Su, 2015; Barrett et al., 2018).

Comparisons of the performance of these models is difficult. Carpenter (1990) and Matzen (2010) among others have attempted to divide matrix problems into categories and provide taxonomies of rules required to solve them. These taxonomies are complex and illustrate the richness and diversity within the matrices. Accordingly, both cognitive accounts and computational models tend to focus on subsets of tests which exhibit certain characteristics. Researchers frequently report a percentage of success based on performance against a hand-picked subset, or, when the full tests are used, any out-of-scope problems are scored at chance. Matrix-generation software allows researchers to target problems of a specific category (Barrett et al., 2018; Matzen et al., 2010; Wang & Su, 2015) and some

researchers, e.g., (Mekik et al., 2017, 2018) report performance against normed equivalent generated matrices instead of the official Raven's tests.

Existing cognitive models of analogy in other domains

Analogy is a vast area of research in cognitive science. We focus here on only two influential models of analogy that have been successfully applied to multiple domains.

The Structure-Mapping Engine (Falkenhainer, Forbus, & Gentner, 1989) has been applied to a wide variety of analogical reasoning tasks, including Raven's Progressive Matrices (Lovett et al., 2010). It draws on the influential structure-mapping theory developed by Dedre Gentner (1983) which provides a generalised algorithm for applying analogy, *i.e.*, it does not require a specific algorithm for each analogy, or even domain of comparison so long as it is provided with an appropriately constructed representation. The analogical processes are decomposed into three stages: *access*, in which the representation of a similar case to the base analogue is retrieved from long term memory; *mapping and inference*, in which correspondences are formed between base and target; and *evaluation and use*, during which the correspondences are checked for validity and relevance before being applied to produce a solution. The algorithms for each of these steps are detailed and elaborate and the theory has many strengths. Notably, it makes a distinction between analogy and other types of comparison such as abstraction, anomaly-detection or simple appearance (Morrison & Dietrich, 1995).

Another significant model of analogy is Copycat, designed to illustrate the theory of analogy as high-level perception (Hofstadter & Mitchell, 1994; Mitchell, 1993), in which representations at a conceptual level arise from the interaction between high-level concepts and low-level perceptual processes. The system takes as input three short sequences of alphabetical characters. The first two are exemplars demonstrating a mapping of some sort. The goal is to apply the same mapping to the third input sequence.

abc	zyx
abd	?

abc	aabbcc
ijk	?

abc	axbxcx
def	?

aabc	aabd
ykk	?

Figure 4. Examples of stimuli using in the Copycat model of analogy (Hofstadter & Mitchell, 1994). Sometimes there is no 'correct' answer, such as in the rightmost example and human subjects provide a variety of explanations to defend their answers.

Contrary to most models of analogy, Copycat is non-deterministic. High-level behaviour emerges from the collective actions of many small processing agents, called *codelets*, competing or cooperating in parallel, each with its own activation level. The codelets rely on a network of simple pre-programmed concepts (such as *successor*, *leftmost* or *opposite*) called the *slipnet* which is likened to long-term memory. One of the strengths of the model is that it avoids fixed representations by allowing these concepts to be flexible. They are interrelated, but their links can shrink and grow and provide a mechanism for what Hofstadter calls *slippage*, when a concept is replaced with another (Hofstadter & Mitchell, 1994). The codelets vie for attention (purported to represent bottom-up processes) and the current state of the slipnet determines probabilistically which codelets are run (top-down influences). Copycat provides a plain text running commentary of its 'thought process' as it tackles a problem which gives us insight into how it reasons, especially pertinent when analogies have no absolute 'correct' answer such as the last example in Figure 4. Mitchell (1993) compares transcripts of human subjects' reasoning with that of Copycat.

The relational priming model

The study that has most inspired this research is that presented by Leech et al. (2008). The Leech model is a connectionist model of analogy which relies on the notion of relational priming. A neural network is exposed to an input image and an output image which primes

the nodes in the hidden layer that encode the relationship between them. When a novel input pattern is presented without resetting the hidden layer's activations, it can apply the primed relationship to the new input by propagating the activation throughout the network and thus obtain a new output. This output forms the basis of analogical completion. The model successfully solved semantic-based proportional analogies without relying on tailored representations and manifested some of the patterns observed in children's performance on similar tasks.

Most cognitive models of analogy, including the Structure-Mapping Engine, express relations using structured predicate and argument terms. The Leech model, instead, views relations as simple transformations between items. Representations do not involve tailored semantics since they are simply input stimuli expressed as vectors. Transformations are similarly encoded as vectors and do not require structured rules or symbolic algorithms for mapping. The network is trained to complete the activation patterns present in input/transformation/output triples and analogy-completion arises as a by-product of pattern-completion.

Methodology

Overview

The goal of this study is to determine whether we can apply the relational priming approach proposed by Leech et al. (Leech et al., 2008), replacing the inputs with vector representations of shapes similar to those found in Raven's Progressive Matrices.

The Leech model targeted analogies involving simple causal relationships between objects. The task, borrowed from Goswami and Brown (1990) who investigated the development of analogical reasoning in children, involves analogies of the form '*A is to B as C is to ?*'. Using simple pictures, they tested 3-6 year-old subjects' abilities to accurately

complete a set of pictured representations of everyday objects undergoing simple transformations such as bread or lemons being cut into slices.

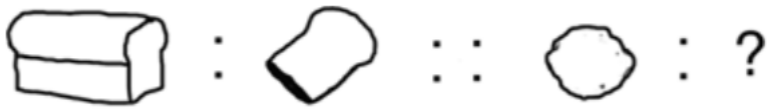

Figure 5. An example of the type of problem used in the Leech et al. (2008) study of relational priming.

A corresponding analogy in our model might look like this:

Figure 6. An example of a problem we hope to solve with the relational priming approach. In matrix form, it would resemble a 2x2 RPM.

The structure of the remainder of this paper is as follows. Firstly, we will describe how we have limited the scope of target problems to consider – there are many different types of RPMs and would be impossible to target them all in a study of this length. Secondly, we will explain how shapes and transformations are represented in our model. Thirdly, we will describe the learning mechanism (Contrastive Hebbian Learning).

We will then present two computational models. The first, Experiment 1, will mirror the architecture of the Leech et al. (2008) paper as far as possible in order to establish whether the relational priming approach is appropriate for solving Raven-like matrices. We will then discuss some weaknesses of the architecture of the network with regard to assumptions made in the Leech model that do not map very convincingly to the new domain.

In Experiment 2 we will seek to improve and simplify the network architecture in the light of those weaknesses to present a more convincing model. Finally, we will consider the data from Experiment 2 to shed light on findings from cognitive research on similarity and analogy.

Scope

It is difficult to make use of the official matrices prepared by Raven in computational experiments. They are subject to strict copyright. In order to prevent their content entering the public domain, they are sold only to professionals with the necessary qualifications to administer the tests. Also, while they number very few, 108 in total across the Standard and Advanced sets (Raven et al., 1998), they encompass a very broad variety of problems. Many of these are beyond the scope of this study, either because they are too simple (the 12 problems in Section A of the Standard Progressive Matrices are not in matrix form – they consist of patterned fields with a section missing), or because they require strategies like counting, combining multiple shapes into a composite, or logic such as XOR, none of which our model is designed to encapsulate. Another difficulty is the variety of shapes and patterns present in the official RPMs. Although some are based on simple shapes like squares and triangles, others make use of dots, dashes, wavy lines or cross-hatched areas.

Given these challenges and following many of the ideas from Matzen et al. (2010), our software incorporates matrix-generation routines which operate on a restricted lexicon of permissible shapes and transformations. We have restricted the basic shapes to the same ones in the Matzen paper, along with their variants, shown in Figure 7.

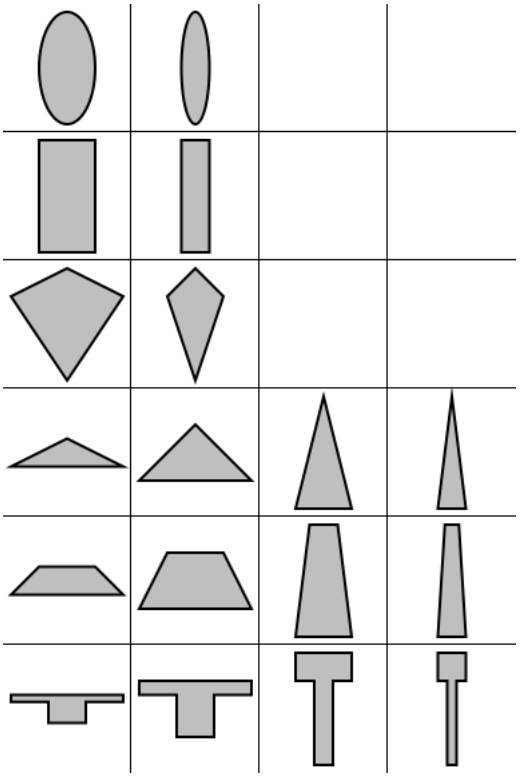

Figure 7. The shapes and their variants used in the matrix generation routines.

Our study considers two types of Raven-like matrix, 2x2 matrices and 3x3 matrices. Each cell of a matrix contains an element which can vary along five dimensions: shape, size, orientation, shading and number. The shape dimension remains consistent on each horizontal row of the matrix.

For the top-left cell of the matrix, a random starting element is generated for the first column, along with a random transformation. The transformation may sometimes be a null transformation signifying no change, or it may contain up to three feature modifications. The transformation is then applied to the starting element to obtain the element for the next column in the matrix. In the 3x3 case, another random transformation is generated and applied to the second column to obtain the third column.

On subsequent rows of the matrix, a new basic shape is generated with its features inherited from the starting element. Then the same transformations as in the first row are applied to produce the remaining columns. The last position in the matrix is left blank.

In the resulting matrix feature modifications take place horizontally and shape modifications vertically. However, by rearranging the cells within the rows we can obtain matrices that have diagonally-distributed feature symmetry, known as a *distribution of 3 values* in the Carpenter (1990) nomenclature. Thus, for each 3x3 matrix in our dataset, the direction of the feature modifications can be horizontal, left-diagonal or right-diagonal.

For each matrix, eight candidate answers are generated. The first candidate is the correct answer for the matrix. The others are incorrect variations of it as follows: one candidate where one of transformed features has been applied with the wrong magnitude; two candidates where incorrect features were features; two candidates where the transformation is correct, but the base shape is wrong; and two where both the base shape and the transformed features are wrong.

Our method of generation leaves out many of the types of problems present in the official tests. For instance, there is no possibility of what Matzen (2010) calls *outward* matrices and Carpenter calls *pair-wise progression*, where a feature (such as figure numerosity) is monotonically increased across the columns, but has different starting values on each row. There is equally no support for *addition* and *subtraction* type problems, where multiple base shapes are combined in different ways. See the second and third matrices in Figure 4 for examples of these.

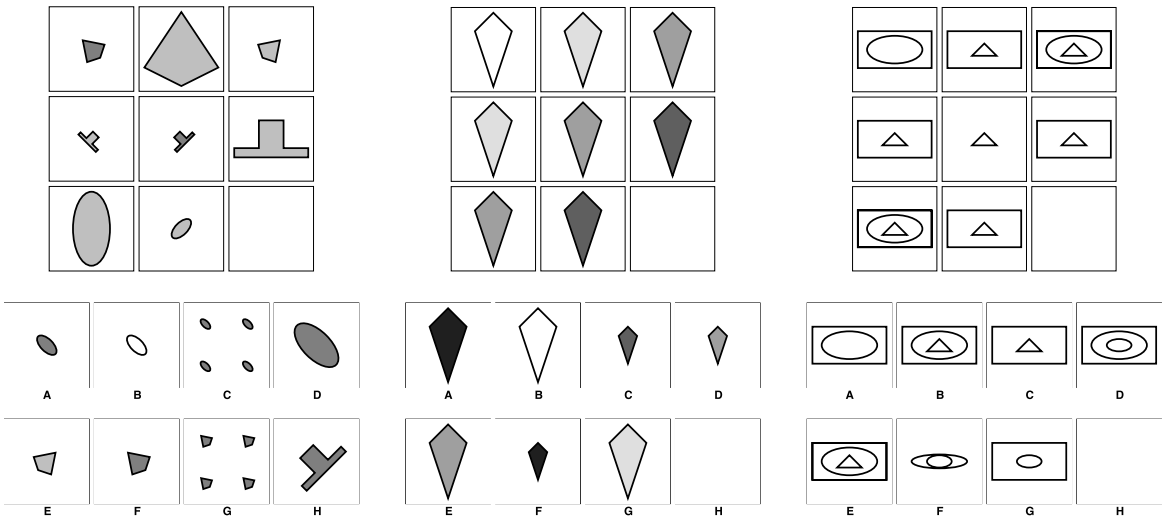

Figure 8. Three examples of 3x3 matrices. Although all of these can be generated by our software, only the first is within the scope of the model presented in this paper. The second and third examples are targets for future research.

These restrictions on scope are considerable. Since most of the official Raven problems consist of a combination of rules, only 14 of the 60 problems in the Standard Progressive Matrices conform exactly to this subset. However, we believe that although the reduced scope is a valid criticism of this research, there is little to suggest that it could not be broadened in future research. The problems shown in Figures 5 and 6 demonstrate the wide variety that even this limited subset encompasses.

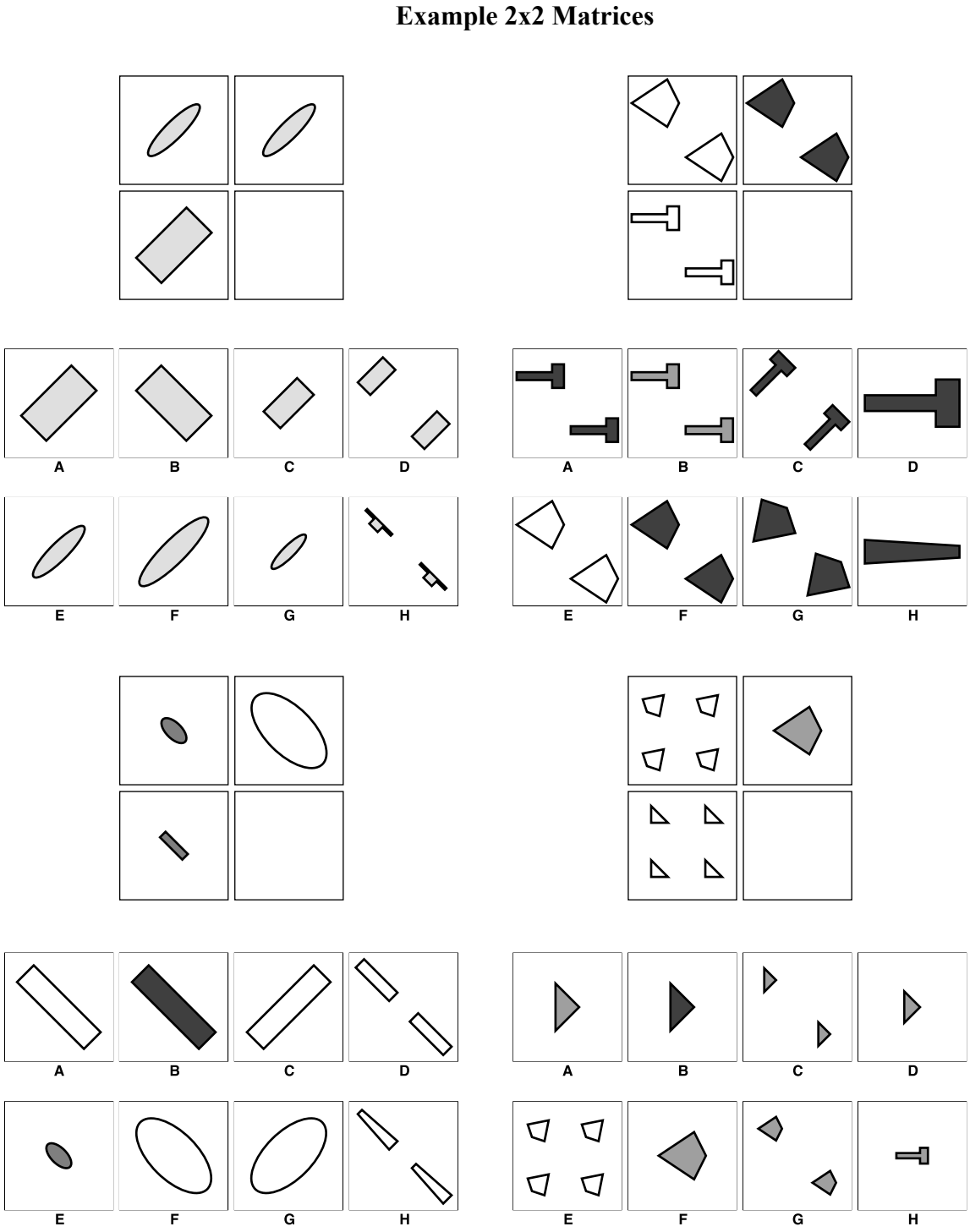

Figure 9. Examples of 2x2 Raven-like matrices used in testing the network. The shapes remain constant across the rows; the features (scale, orientation, shading and numerosity) are constant across the columns. The number of features modified ranges from 0 to 3. Starting from top-left are a 0-, 1-, 2- and 3-relational problem. The correct answer in all cases is A.

Example 3x3 matrices

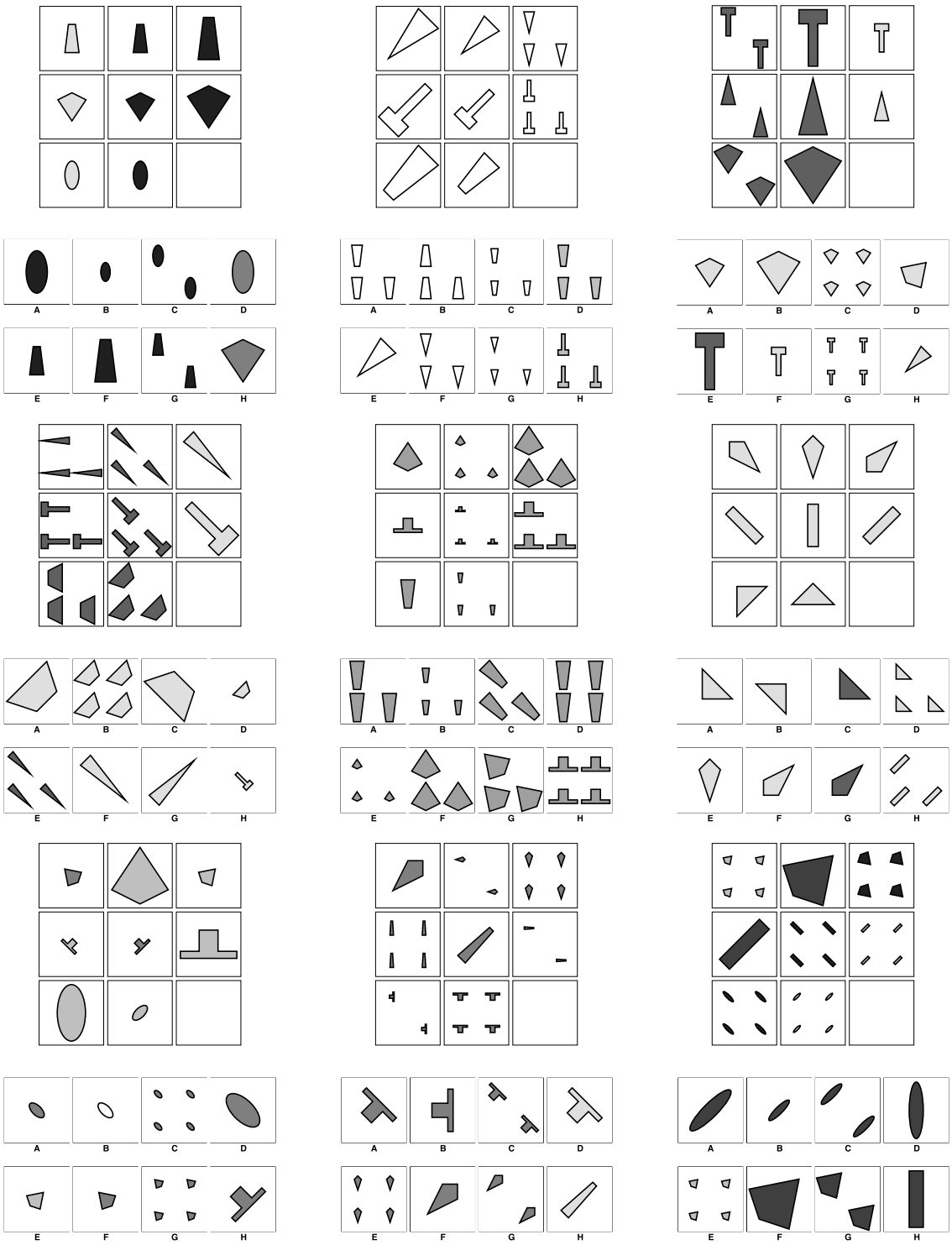

Figure 10. Examples of 3x3 Raven-like matrices illustrating the additional complexity compared to the 2x2 matrices. The last three examples show *distributions of 3*, where each row contains three exemplars but not in the same columns. As with the 2x2 matrices the shapes remain constant across the rows, but the features (scale, orientation, shading and numerosity) can vary vertically or along either of the diagonals. The answer in all cases is A.

Vector representation of shapes and transformations

How are these shapes and transformations represented? Each element is encoded as a vector of size 11 comprising of the shape itself and its parameters features (scale, rotation, shading and number). Scale and numerosity can take any of four possible values; rotation and shading any of eight. All elements of the vector are normalised to the range [0, 1].

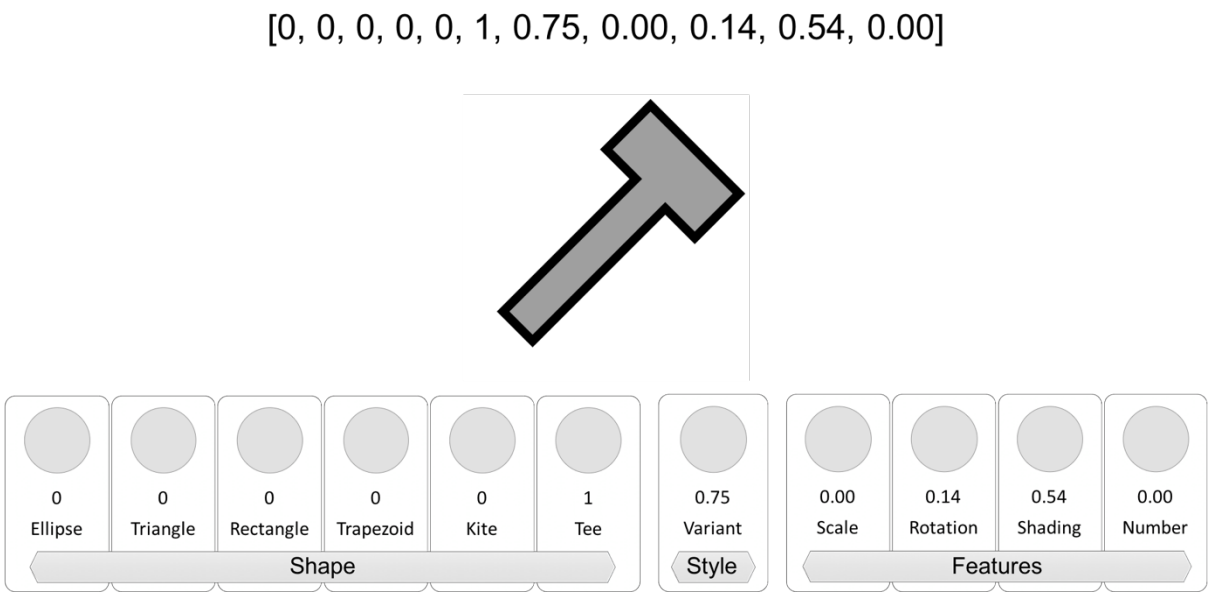

Figure 11. The vector representation of a shape and its features. The same representation is used for input and output shapes.

Each transformation is made up of four feature modifications corresponding to each of scale, rotation, shading and number. These are also normalised to the range [0, 1] such that a value of 0.5 corresponds to no change. A value above 0.5 will correspond to an increase in the value of the corresponding feature; a value below 0.5 to a decrease. The transformations are generated in a manner to ensure the transformed feature remains within the bounds of the feature – for instance, a shape with shading of 80% will never be the subject of a transformation increasing shading by 30%.

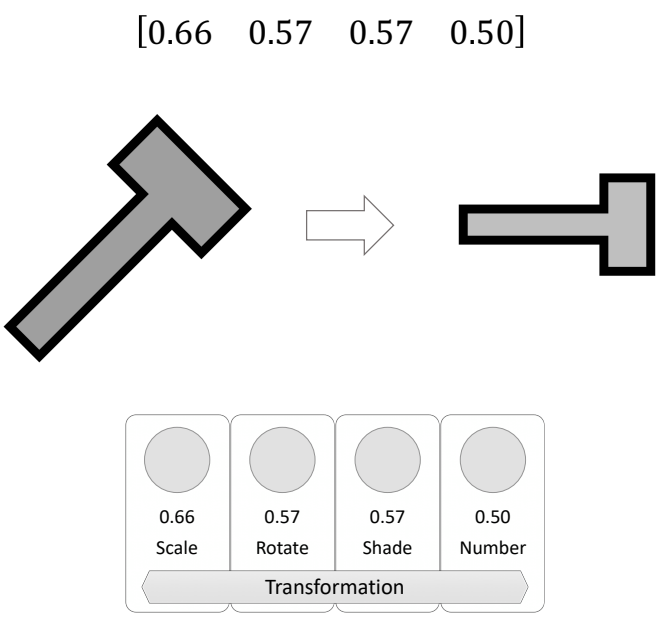

Figure 12. The vector representation of a transformation.

In choosing this encoding, we have sought to avoid semantic content as much as possible. Each of the six possible shapes has been replaced with a progressive numeral index encoded as a 1-hot vector. It is merely an unlexicalised index into the list of shapes and has no semantic meaning. The feature units are simple scalars in the range [0,1] corresponding to the magnitude of each feature. They carry no semantic meaning beyond their numeric value. The positions of the feature units are interchangeable without affecting the outcome of the experiment, as are the indices for the shapes. Likewise, each of the four units in the transformation vector are simple scalars corresponding to the magnitude of the change to a given feature. 0.5 corresponds to 'unchanged'. 0 corresponds to 'decrease as much as possible'; 1 to 'increase as much as possible'. Again, the position of these units is irrelevant.

The chosen representation is not designed for analogical completion, only for visual description. The individual units within both the shape and the transformation vector could conceivably be the outputs of lower-level feature detectors.

Generation of training data

In each of the experiments, the network is trained to complete missing data when presented with an example of a simple transformation, that is, an input shape, a transformation and an output shape. At no point is the network trained to complete analogies, nor is it ever exposed to vector representations of complete RPM matrices.

Each item of training data is comprised three vectors: a starting shape vector; a transformation vector made up of between zero and three feature modifications; and the resulting output shape vector. Figure 4 shows some examples of input/output pairs similar to those used in training.

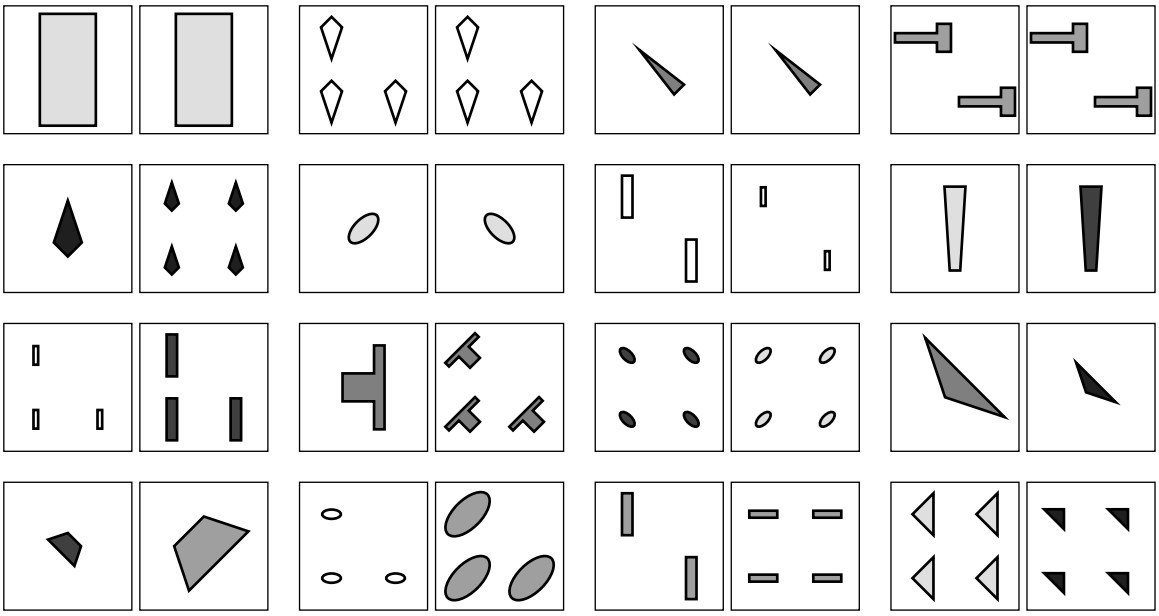

Figure 13. Examples of training patterns grouped by row in input/output pairs. The first row shows 0-relational training patterns. The subsequent rows show 1-, 2- and 3-relational training patterns.

The learning of these patterns of shapes and their transformations takes place via Contrastive Hebbian Learning which is the topic of the next section.

Contrastive Hebbian Learning

Contrastive Hebbian Learning (CHL) is an algorithm that can be used to perform supervised learning in a neural network (Movellan, 1990). It is more neurologically plausible than algorithms which rely on backpropagation (described in Rumelhart, Hinton, & Williams, 1986), yet under many conditions it has been shown to be equivalent (Xie & Seung, 2003).

The network is made up of a layer of input neurons, a layer of output neurons and a layer of hidden neurons. The synaptic connections between adjacent layers are bilateral and symmetric, that is, activation can flow both forward and backward between layers. This recurrent nature of the network makes it intrinsically unstable and it takes multiple cycles before the activation propagating through the network eventually settles. The number of passes is sometimes used as a proxy for response times or similar behavioural measures (Seidenberg & McClelland, 1989).

As with the restricted Boltzman machine (Smolensky, 1986) from which CHL is inspired, we can train the network by performing Hebbian updates in two distinct phases. In the first phase, only the input units are clamped to a given example of training data and activation is allowed to propagate throughout the remaining units. This is known as the *unclamped*, *negative* or *unlearning* phase. The resulting activation state corresponds to the response of the network to the given input.

The second phase, also known as the *clamped*, *positive* or *learning* phase, involves clamping not only the input unit but also clamping the output to the corresponding desired result. Activation again propagates through the network and settles, but only the hidden units' activation stabilises since all other units are constrained by the clamping. Here the resulting state is the desired activation of the network given the input. Each negative phase thereby corresponds to an *expectation*; each positive phase to a *confirmation* (O'Reilly, 1998).

This provides a method for calculating local error by comparing the state of activation during the clamped phase with that during the unclamped phase and using the difference to modify the synaptic weights so as to reduce the difference between them. After both phases, the synaptic weights are updated according to the difference of the cross-products of the activations of the two phases,

$$w_{i,j} \leftarrow w_{i,j} + \eta(y_i^+ y_j^+ - y_i^- y_j^-)$$

where w represents the weight between two neurons, i and j; y represents the output of a neuron; and η is the learning rate (Movellan, 1990). The equation above is known as *synchronous*, because although the two products $y_i^+ y_j^+$ and $y_i^- y_j^-$ are computed at different times (at the end of the positive and negative phases respectively), the weights are updated at the same moment. This is more efficient, but biologically implausible because it requires storing the products until the update of weights takes place. A more plausible alternative is to perform *asynchronous* weight updates where the product $y_i y_j$ is calculated and used immediately to update the weights, with the sign of the update depending on the phase.

$$w_{i,j} \leftarrow w_{i,j} + \begin{cases} +\eta(y_i y_j) & \text{if the phase is positive} \\ -\eta(y_i y_j) & \text{if the phase is negative} \end{cases}$$

Over time, the activation in the minus phase comes to replicate that during the positive phase. When the network is presented with partial inputs, activation will spread towards the nearest attractor state and thus complete the missing portions. This corresponds to learning.

Implementation details

All software was written in Python 3.6. Simulations were run on cloud-based Linux Ubuntu 16.04 instances.

The routines to generate shape and transformation vectors and display them as Raven-like matrices were motivated by the work of Matzen et al. (2010). For the graphical display of

the elements within the matrices, the software makes use of the *pyRavenMatrices* library (Mekik, 2018), with some modifications. For the replication of the neural network architecture and the Contrastive Hebbian Learning algorithm used in Leech et al. (2008), the Matlab source code from Robert Leech's doctoral dissertation (Leech, 2004) was invaluable.

Experiment 1

Architecture

The goal of this experiment is to demonstrate that the relational priming approach used in the Leech et al. model (Leech, 2004; Leech et al., 2008) can be adapted to complete the Raven-like analogy problems provided by the matrix generation routines discussed in previous sections. Some aspects of this architecture are motivated by the word analogy domain targeted by the Leech model and may be unnecessary or suboptimal. Nevertheless, this experiment aims to replicate that architecture as far as possible with only minor modifications. Experiment 2 will aim to improve and simplify the architecture presented here.

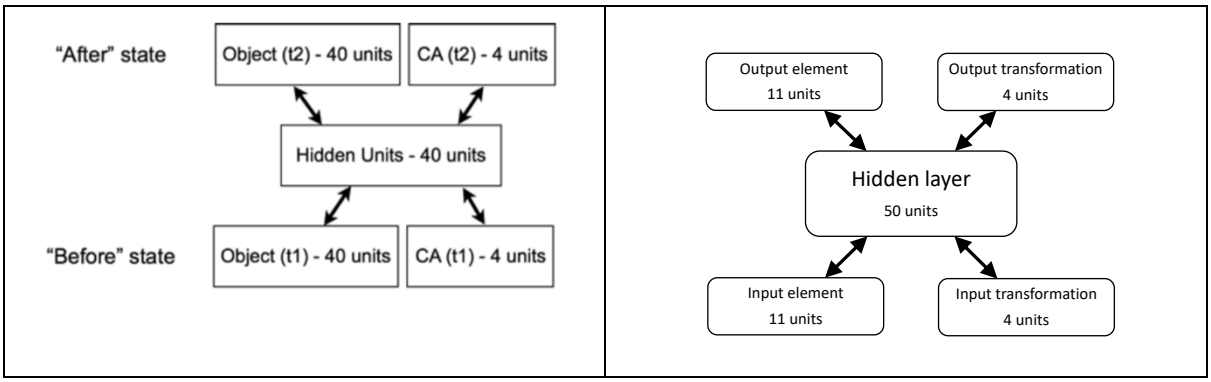

Figure 14: A comparison of the model architectures. The left schema is reprinted from the Leech et al. (2008, p. 364, Figure 3) model used to complete word analogies. The right schema is used in Experiment 2 to complete visual analogy problems similar to Raven's Progressive Matrices.

The network consists of five banks of units arranged in three layers: the input layer, an 11-unit bank representing the input shape and a 4-unit bank representing the

transformation; the hidden layer, a single 50-unit bank; and the output layer, 11- and 4- unit banks for the output shape and a copy of the transformation. The connections between the layers are symmetric and bidirectional, allowing activation to flow in all directions. There are no connections laterally between the units within the same layer.

In the left image in Figure 14, *CA* refers to the notion of *causal agent*, that is, an object which provides a context for the transformation, but which remains unchanged. For example, if the analogue pairs were *apple* and *slices of apple,* the causal agent might be a *knife*. The knife remains unchanged after the cutting transformation, so it is replicated in the bank labelled *CA*(*t2*).

In our domain of visual shapes and simple transformations, the input/output objects have been replaced with input/output shapes and the causal agents with the 4-unit transformation vector corresponding to the changes in scale, orientation, shading and numerosity. We will discuss the validity of the correspondence between causal agents and transformations further in the criticisms of Experiment 1.

Training

The network was trained with 1000 unique vector triples consisting of an input shape, a transformation and the resulting output shape. Initial weights were randomised uniformly in the interval [-0.5, 0.5]. Training was performed for 25,000 epochs, testing the model at 50 epoch intervals. The logistic function,

$$\mathrm{f(x)} = \frac{1}{1 + e^{-\mathrm{kx}}}$$

was used as the activation function, with logistic growth rate k (often referred to as temperature) set to 0.1. The learning rate η was set to 0.05. An adaptive bias was used for

each layer, updated asynchronously at the same time as the weight updates. As often as possible, these settings are kept in line with the Leech model[2].

As in the source code from the appendix of Leech's doctoral thesis (2004), the network was trained using a slightly modified version of asynchronous Contrastive Hebbian Learning in which there is an extra fully-unclamped step in the negative phase. During the first step in the negative phase, both the input element and the input transformation units are clamped to a training exemplar; the output element, the output transformation and the hidden layer are all unclamped. Activation is propagated through the network and allowed to settle.

At this point, in a deviation from canonical CHL, all units are unclamped, and the network is allowed to settle a second time. Without this step, the network learns to complete the output layers, but never learns to complete the input transformation layer, necessary for the priming phase (Leech, 2004, p. 110). After the fully unclamped network has settled anew, the resulting activations are used to update the weights and biases according to the negative update rule as normal.

---

[2] For a more detailed comparison of Experiment 1 to the Leech model, refer to Appendix A.

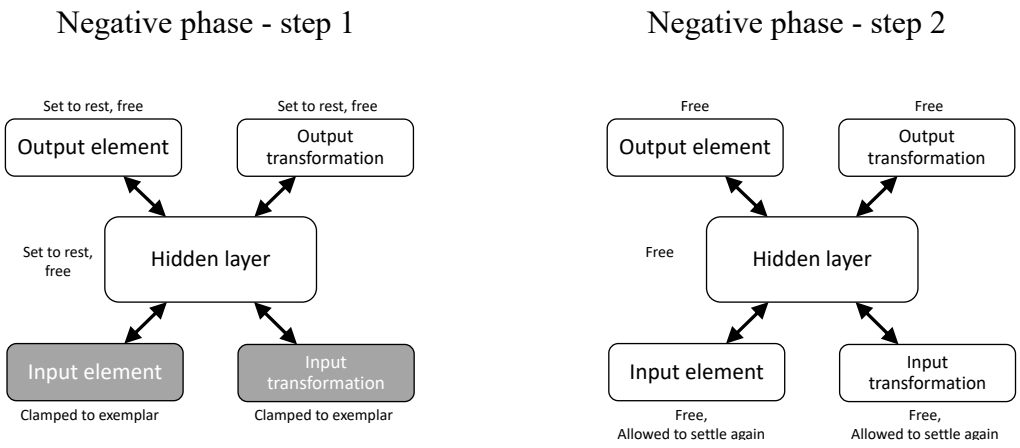

Figure 15: The deviation from the canonical form of the negative phase of contrastive Hebbian learning used in both Experiment 1 and the Leech (2004) model. Shaded units indicate clamps. In step 1, the inputs are clamped and the network settles. In step 2, the inputs are unclamped without resetting any activations and the network settles again. By this mechanism, the network learns to complete not only the outputs, but also to complete the inputs, especially the input transformation which is required for the priming phase.

In the positive phase, all input and output layers are clamped to the training exemplars and the network is allowed to settle. The weights and biases are updated for the positive phase. Over multiple epochs the difference between the activations during the positive and negative phases is reduced, such that the network has learned to reconstruct the missing layers. When presented with inputs, the outputs correspond to the expectation of the network, and the accuracy of this expectation increases over time.

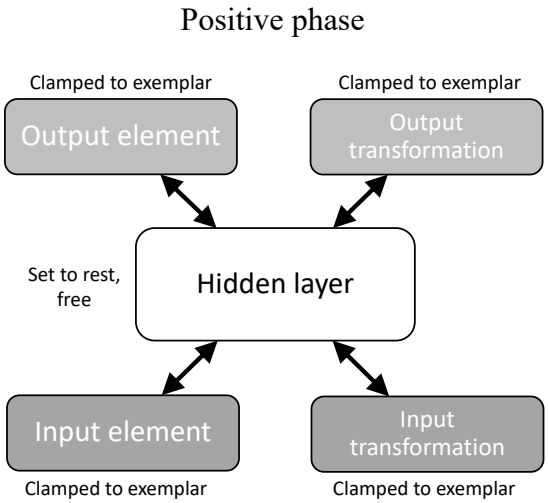

Figure 16: The positive, clamped learning phase. Both inputs and outputs are clamped, and the hidden layer is allowed to settle.

One way of interpreting these layers is by seeing them as representations of the *before* and *after* states of objects in the world. In our everyday experience, our visual system is constantly exposed to shapes changing their features: size, orientation, colour, *etc.*, such as when a baby observes the oval of its mother's face enlarge as it nears. The assumption is that these pairs of before and after percepts, and the corresponding transformation between them, are learned over time.

Accuracy and loss during training

Training loss was calculated using the least mean squares method. Two separate measures for loss were computed. Firstly, we measure the network's ability to learn to produce the correct output shape and output transformation vectors on being presented with an input vector and an input transformation vector. Training accuracy was measured by selecting the nearest pattern (in Euclidean distance terms) out of all the output patterns in the training set and testing for equality with the expected target vector. Secondly, we measure the network's ability to internalise the transformations, that is, whether it produces the correct transformation vector (encoded in the input transformation layer) when presented with a pair of input/output shapes.

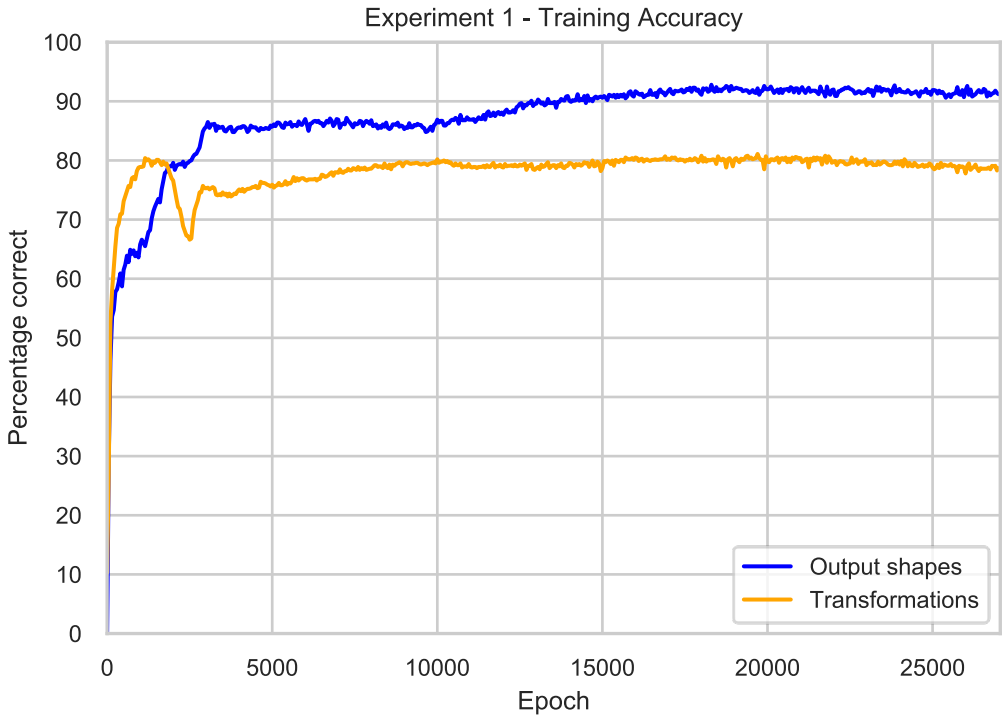

Figure 17: Plot of the network's training accuracy by epoch.

Figure 18: Plot of the network's training loss by epoch.

The plots of accuracy and loss by epoch in Figures 17 and 18 show that both shapes and transformations are being learned gradually over time. In this particular run there is a moment at about epoch 2000 where the transformation accuracy and loss suffer temporarily but ultimately recover. Over many runs with different parameters (*e.g.*, learning rate, number of hidden units), it was observed that these non-linearities are typical.

Testing

The network was tested on its ability to solve 2x2 and 3x3 matrices. One hundred unique examples of each type were generated using the matrix generation routines. Each starting shape was verified to be distinct form the 1000 training inputs so that all of the analogies are novel.

The trained network completes analogies using the relation priming mechanism. The following sections explain how this is accomplished for both the 2x2 matrices (very similar to the Leech model), but also the 3x3 matrices where a novel approach is necessary.

Analogy completion for 2x2 matrices

First, the input shape and output shape layers are set to the vectors corresponding to the two shapes on the top row. These layers are then clamped; the transformation layers and the hidden layers are free and set to rest. Activation is propagated through the network and settles. The network is now primed (Figure 19).

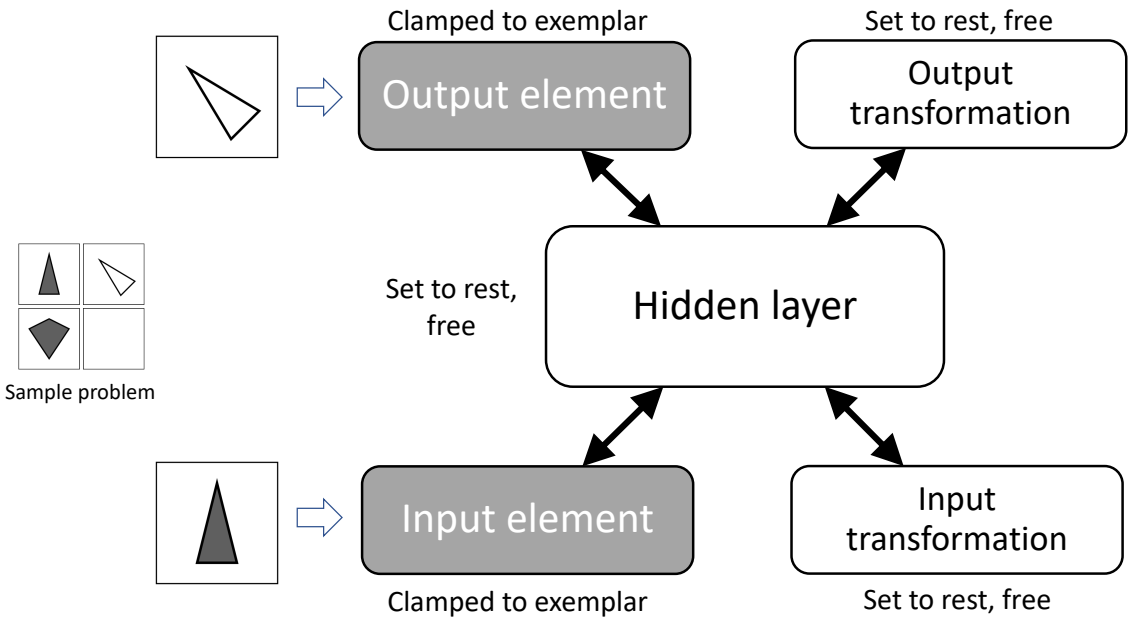

Figure 19: The priming phase of analogy completion for a 2x2 matrix. The vectors corresponding to the first row of the matrix are presented as exemplars. Shaded units indicate clamps.

Next, the target analogue (i.e., the bottom-left element of the matrix) replaces the source input. The output shape layer is set to rest. The other layers are left with the activations from the previous priming phase. In other words, the network is perceiving the target analogue in the context of the source input/output pair.

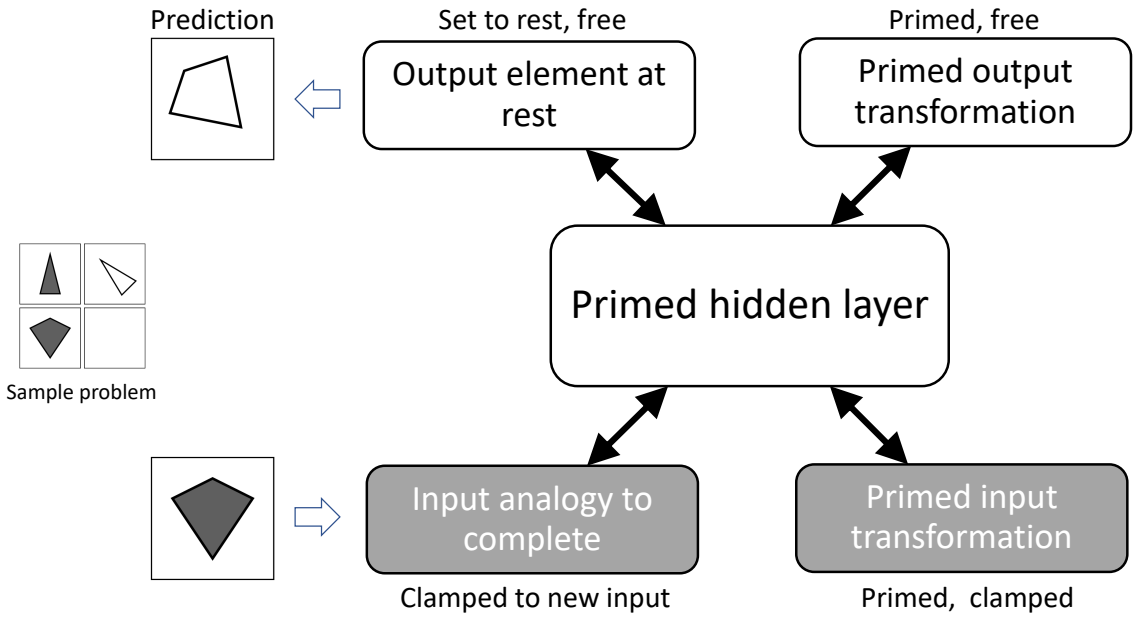

Figure 20: The analogy completion phase. The input shape is set to the target analogue, which is clamped, along with the input transformation which has been primed by the context of the first row.

Since the input and output elements have changed, the network is no longer in equilibrium and settles into a new attractor state which is biased by the primed hidden and transformation layer units. Consequently, the output element vector corresponds to the expectation of the perceived input influenced by the context of the source examplars, in other words, the completion of the analogy.

Analogy completion for 3x3 matrices

Whereas the 2x2 matrix maps conveniently onto the proportional analogy template with its $A : B :: C : D$ components, 3x3 matrices are considerably more complex. In whatever procedure we propose, three constraints must be satisfied.

Firstly, the procedure must be capable of solving a substantial portion of the problems in the scope of our matrix generation routines. Ideally, the method should be extensible to cover the broader scope of the official RPM tests. Secondly, in order to support the

hypothesis that analogy completion occurs automatically and concurrently with perception, the procedure cannot be overly algorithmic. An algorithmic mechanism would imply the involvement of higher-order cognitive functions such as reasoning, logic and planning. Thirdly, the procedure should consistent with what we know of human behaviour during similar tasks, such as the eye-tracking studies by Carpenter et al. (1990). The procedure we propose for completing 3x3 RPMs relies on two insights.

The first insight is that many matrices can be solved by only considering the four bottom-right cells as a 2x2 matrix and ignoring the top- and left-most cells. This is true of any matrix where the features are either constant in a vertical column or constant in a horizontal row. The same holds for three other 2x2 subsets of the cells as shown in Figure 21.

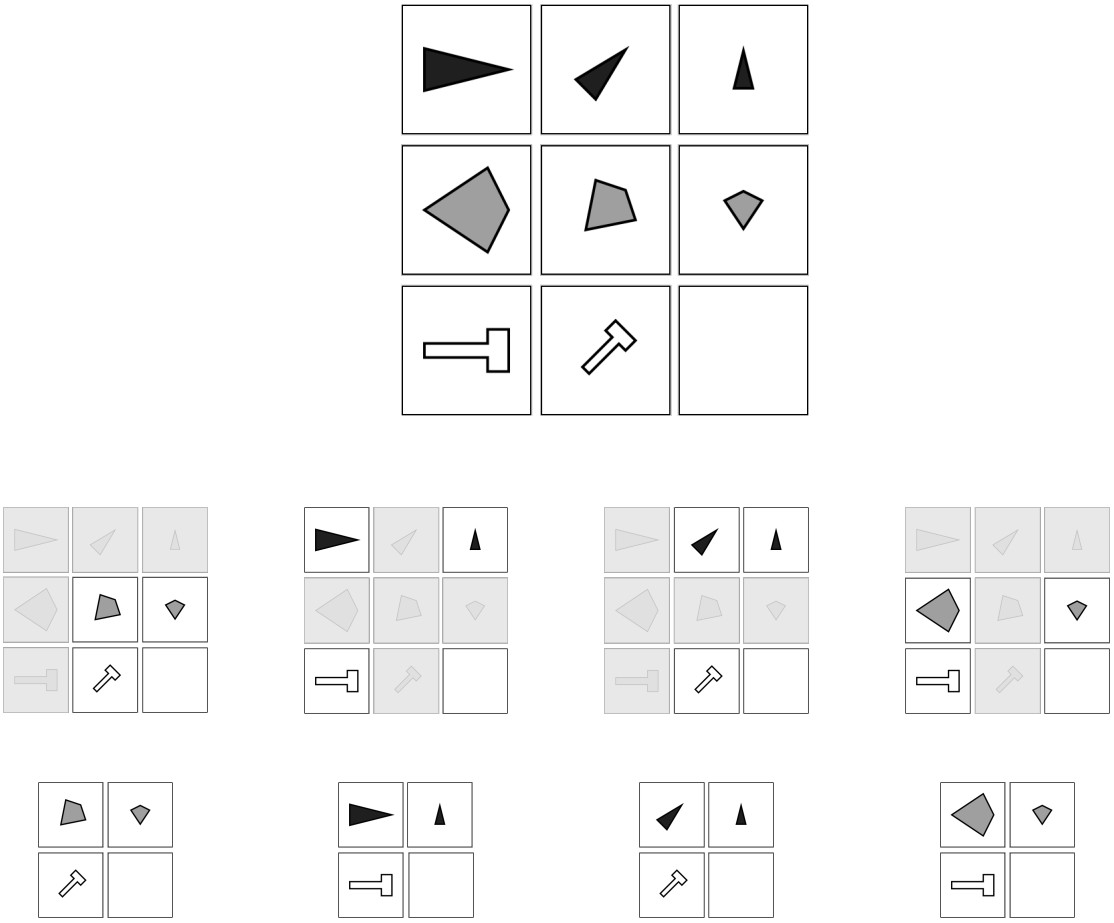

Figure 21: Many 3x3 matrices can be solved by considering them as four separate 2x2 matrices. The network selects the candidate answer from the four predictions by consensus.

The second insight is that the same method can be adapted to solve many diagonal matrices too. The example in Figure 16 illustrates how the first and second rows in a matrix with a diagonal distribution are right-shifted to make the alignment of the feature transformations vertical instead. Thus, matrices with modifications along either diagonal can also be treated as four separate 2x2 matrices (with different constituent cell coordinates than in the vertical situation).

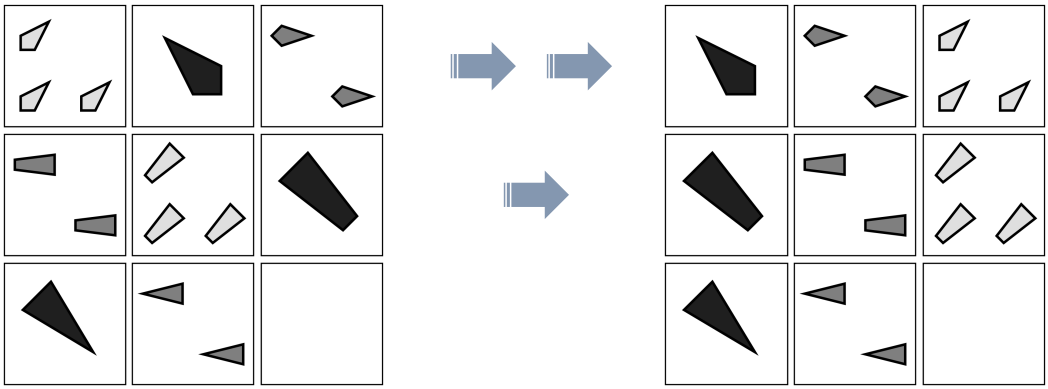

Figure 22: Matrices with diagonally aligned axes of transformation are often equivalent to a vertically aligned matrix. When this is the case, they can be treated as four separate 2x2 matrices.

Therefore, the approach for 3x3 matrices is as follows: consider the cells as four individual 2x2 matrices three times over, once assuming features are constant vertically, once assuming they are constant along the left-diagonal and finally that they are constant along the right-diagonal. Then select the direction and candidate with the most consensus.

This method fulfils our criteria. The procedure is effective for solving many types of 3x3 problems. It does not involve any procedural logic – it is purely a tally of the votes from multiple 2x2 matrices. Finally, it is consistent with the behavioural findings of Carpenter et al. (1990) who found that subjects tend to fixate back and forth between pairs of figures when solving 3x3 matrices. In particular this observation resonates:

> *Perhaps the most striking facet of the eye fixations and verbal protocols was the demonstrably incremental nature of the processing. The way that the subjects solved a problem was to decompose it into successively smaller subproblems, and then proceed to solve each subproblem. (Carpenter et al., 1990, p. 8)*

Unfortunately, the method in its current form is not universal enough to solve all types of problem – it is possible to construct matrices for which it will fail. Figure 17 has *both* a vertical direction (for constancy of shading) and a diagonal direction (for constancy of rotation), typical of many matrices in the advanced sections of the official Raven's test. This type of matrix is beyond the scope of the generation routines used in this study. In spite of this shortcoming, it may be possible to extend or adapt the method to cater for additional cases.

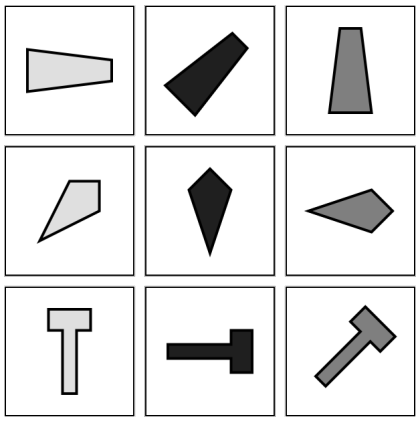

Figure 23: An example adapted Matzen et al. (2010) showing a 'one-diagonal outward relation' (p. 533). This type of problem cannot be solved by considering four separate 2x2 matrices.

Results of Experiment 1

Maximum performance on the 2x2 test matrices occurred at epoch 14,150 with 90% of the 100 test matrices solved correctly. The maximum for 3x3 matrices occurred at epoch 21,950 with 89% solved.

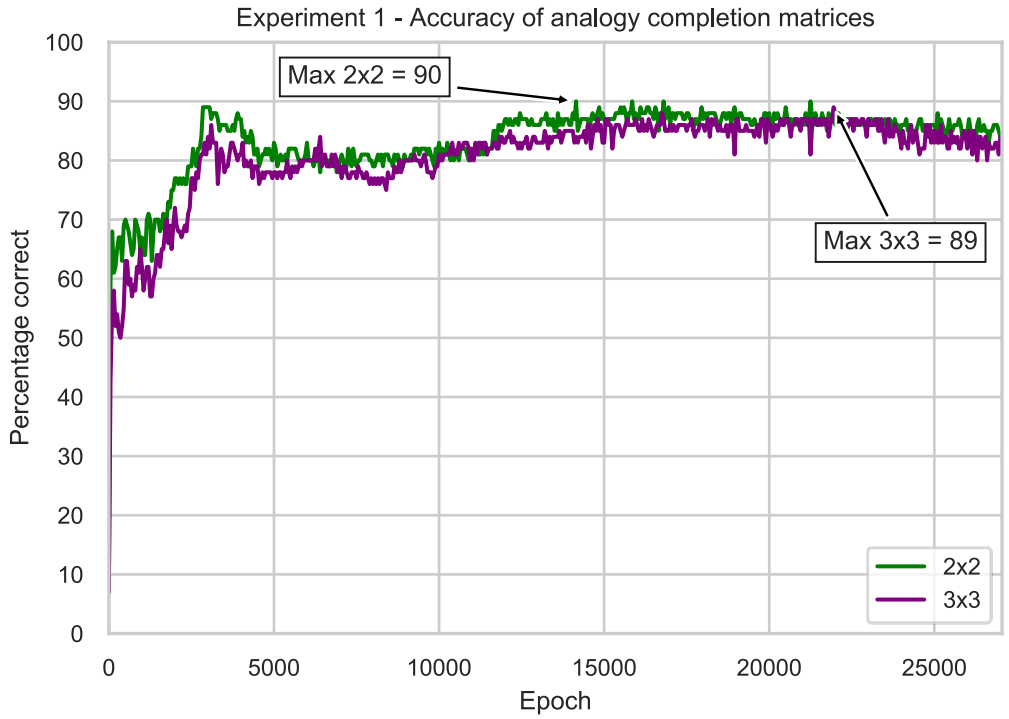

Figure 24: Plot of accuracy of analogy completion on the 2x2 and 3x3 matrix tests.

Both 2x2 and 3x3 success rates achieve above 84% accuracy early on at about epoch 3000, and then dip before surpassing 85% accuracy at about epoch 12,000. This reflects the dip seen in the learning of transformation patterns in the training plots discussed earlier. These non-linearities are sometimes seen as evidence that the network has internalised information that achieves a certain performance level, but that in order to exceed this, it must 'unlearn' in order to improve further. In terms of the solution space, the network has found a point of local maximum.

Discussion of Experiment 1

The goal of this experiment was to determine whether the relational priming approach could work for a visual analogy task similar to Raven's Progressive Matrices. Overall, performance on the test data was found to be high with 90% and 89% appropriate analogical completions for the 2x2 and 3x3 matrices respectively. In this regard, the goal has been achieved. The network can solve 2x2 matrices which are the type of problem most similar to

the word analogies targeted by the Leech model. Furthermore, the same mechanism can be used multiple times in parallel to solve many considerably more complex 3x3 matrices including those where the direction of feature modification is diagonal.

One notable difference compared to the Leech model is the number of transformations which must be learned. The Leech et al. model (2008) used only 4 possible one-hot encoded transformations. In our RPM experiments, a transformation represents a modification of zero or more of the four features leading to $4 \times 8 \times 8 \times 4 = 1024$ possible transformations. As a result, the network is sensitive to reductions in the width of the hidden layer – with fewer hidden units the learning curve plateaus early at an much lower accuracy rate.

In spite of the promising performance of the model in Experiment 1, we have refrained from analysing the data further to address some criticisms of its architecture. We attempt to correct for some of the issues in Experiment 2, after which the data will be further analysed from a more cognitive standpoint.

Criticisms of Experiment 1

The principal goal of Experiment 1 was to adapt the model used by Leech et al. to target the RPM task. In spite of the success of the model, some aspects of its architecture seem overly complex and artificial.

Firstly, the notion of causal agents seems unnecessary. In the word analogies targeted by the Leech model, the causal agent provides a context for the transformation, for instance when the network learns the relation *cutting*, the transition from *apple* to *slices of apple* occurs in the context of the concept of *knife* as the causal agent. However, it is not clear what the parallel would be with regard to the transformation of a shape's features. When we look at the transition from *ellipse* to *rotated ellipse* or from *triangle* to *shaded triangle*, we do not usually think of those transitions as occurring in the context of a *rotator* concept or a *shader* concept. When we explain a transformation such as rotation to a child, we do not usually

explain in terms of a causal agent, a machine for rotating things. In a sense, the transformation seems less accessible to us than the input and output shapes.

Secondly, the Leech model placed the causal agent as an explicit output of the network, the motivation being that the input and output layers represent *before* and *after* states of the world. Before the transformation, there is an apple and a knife; after the transformation there are some slices of apple and the unchanged knife. But there is no obvious parallel in the visual shapes domain – the transformation is not associated with any causal agent, unchanged or otherwise.[3]

Thirdly, in the Leech model, many of the relations considered were not reversible, such as cutting, melting, burning. As such it made sense to place the input causal agent layer as a separate input, alongside the layer representing object it acts upon. But when we look at a pair of shapes, say a triangle and a shaded triangle, the transformation can be seen as an increase or, conversely, a decrease in shading, depending on the direction. Consequently, it may be inaccurate to place the transformation layer as a separate explicit input and we should consider an architecture in which the transformation layer can be influenced by the input and output shapes, that is, with synaptic connections between the shape layers and the transformation layer.

Finally, the clamping mechanism is complex. As we have seen the negative phase is complicated by the necessity of training two separate pattern-completion behaviours: completion of the transformation (in the priming phase) and completion of the output shape (in the completion phase). Perhaps a different architecture would obviate the need for this complexity.

---

[3] In a separate run of Experiment 1, the network was modified to remove the output transformation layer. The performance and learning metrics were virtually unchanged.

These considerations inform the second experiment. In Experiment 2 we propose a different network architecture with the principal aim of refining and simplifying the model proposed in Experiment 1, both from the perspective of the connectivity of the layers and the clamping regime.

Experiment 2

Architecture

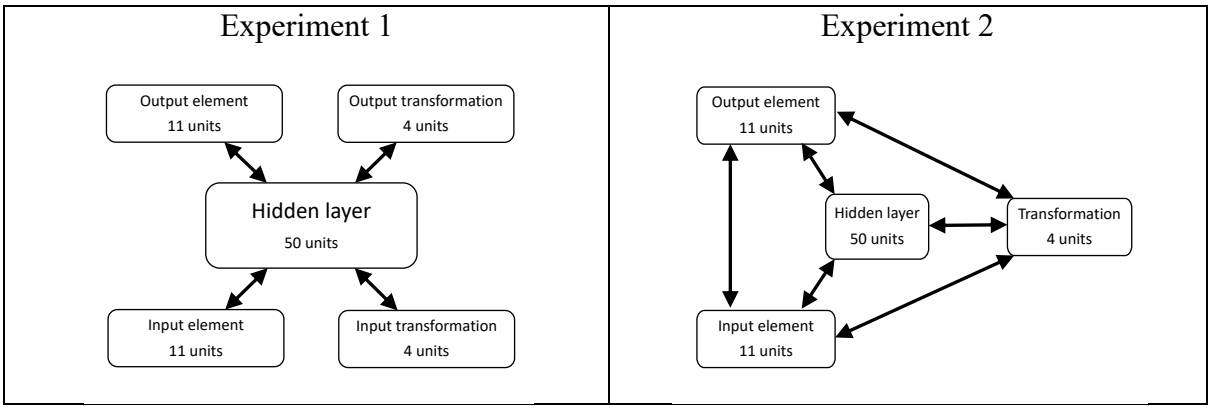

Figure 25: A comparison of the network architectures of the two experiments.

Figure 26 shows the new configuration of the layers and connections. The output transformation layer, the *CA(t2)* in the Leech model, has been removed altogether. The input transformation layer has been moved so that it lies between the input and output layers alongside the hidden layer. All layers are connected bidirectionally and symmetrically to each other. In this way, the input, output and transformation layers all have influence over each other, both directly, and via the hidden layer.

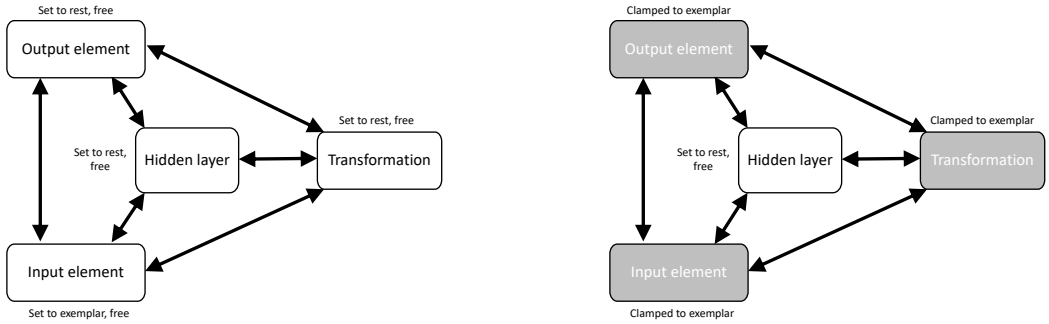

Figure 26: The training phases of the architecture used in experiment 2. On the left, the unclamped, negative phase. On the right, the clamped, positive phase.

Training takes place as before, via asynchronous CHL. However, the clamping regime has been simplified and no longer includes a two-step negative phase. Instead, the negative phase is completely unclamped. Only the units of the input layer are set to the vector representing the input exemplar. The positive phase is unchanged. All three visible layers are set to their exemplars and clamped.

While there is still a slight deviation from canonical CHL (the input layer would usually be clamped during the negative phase), the chosen regime was found during experimentation to provide the best performance, especially when using the asynchronous flavour of CHL. Further investigation to understand the precise implications of different clamping regimes is beyond the scope of this research.

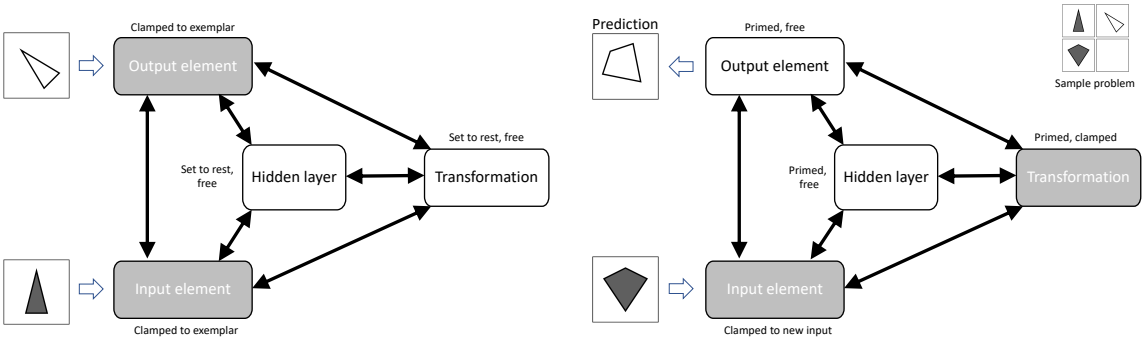

Figure 27: The analogy completion phases for experiment 2. On the left the priming phase. On the right the analogy completion phase.

The priming phase is straightforward. The elements in the first row of the matrix are presented to the input and output layers. The resulting activation in the network is allowed to settle, updating the transformation layer. The elements in the second row are then presented, that is, the input units are set to the vector representing the target analogue and the output element (the empty cell) is set to rest. The transformation layer, primed with the activations from the first row, is clamped along with the input layer. The network is no longer in equilibrium. The latent activations from the primed and hidden units influence the new activations in the input layer to settle into a new attractor state. The output units now correspond to the predicted completion of the analogy.

Training

The same training stimuli were used as for Experiment 1, consisting of 1000 input/transformation/output triples with an even mix of 0-, 1-, 2- and 3- feature modifications.

Training was performed for 16,000 epochs, testing the model at 50 epoch intervals. All hyperparameters were kept the same as for Experiment 1 (logistic temperature $k = 0.01$, hidden units = 50), except for the learning rate η which was reduced to 0.001 (compared to 0.05 in Experiment 1). Training accuracy and loss plots are shown in Figures 28 and 29.

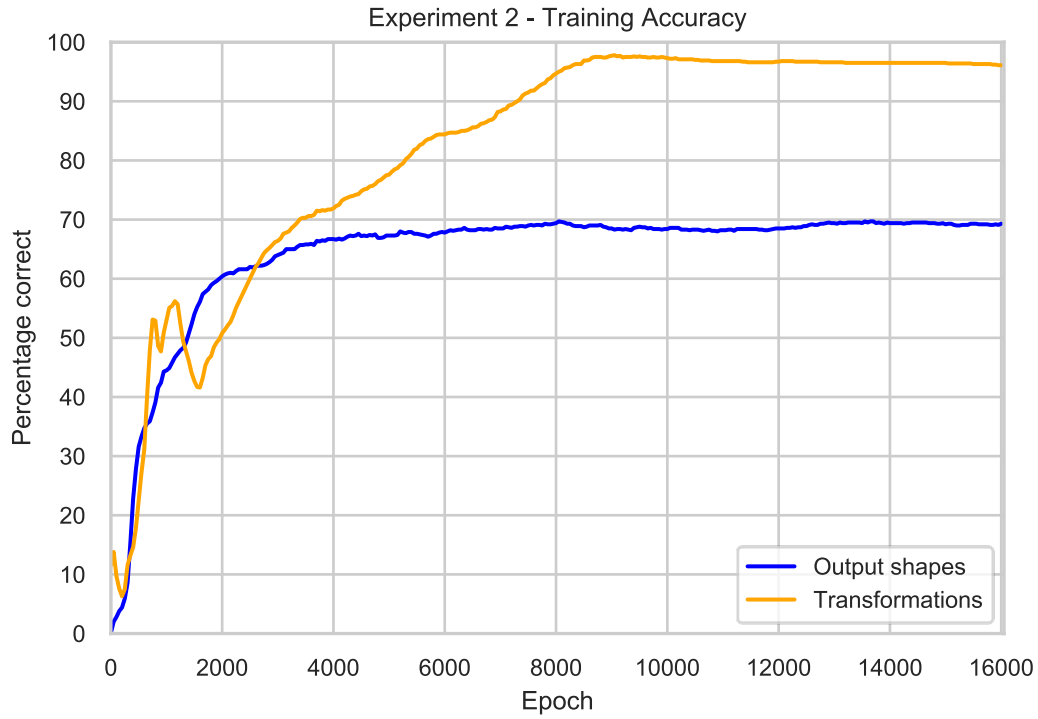

Figure 28: A plot showing the network's accuracy at predicting output shapes and transformations from the training data.

Figure 29: A plot of training loss by epoch.

Similar to the findings from the previous experiment, the learning of transformations seems to happen in stages, where a reversal occurs before the accuracy attains new maxima. Contrary to the plots from Experiment 1, transformation accuracy exceeds output shape accuracy from about epoch 2500.

Results of Experiment 2

The same test stimuli of one hundred 2x2 and one hundred 3x3 matrices were used as in the previous experiment. The network successfully solved 87% of the 2x2 and 86% of the 3x3 matrices. These maximum levels of accuracy for the two types of test occurred very close together, at epoch 2500 and 2750 respectively.

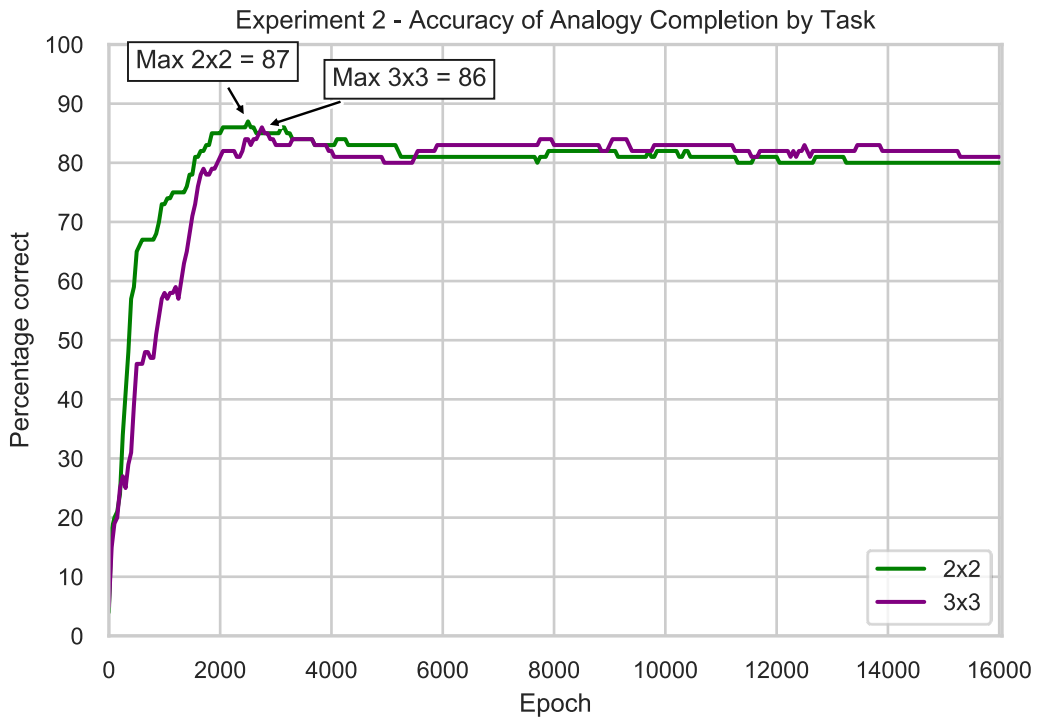

Figure 30: A plot illustrating the network's ability to perform accurate analogy completion for the 2x2 and 3x3 matrices in the testing dataset.

Comparison of multiple runs with different numbers of hidden layers shows a somewhat surprising result. Whereas in Experiment 1, the number of hidden units was a key parameter in obtaining adequate performance, it seems that this revised network is much

more robust to reductions in the number of hidden units. Figure 31 shows a comparison of performance in the 3x3 matrix task with 50, 15 and zero hidden units. Even with no hidden layer at all, the network was able to correctly complete almost 70% of the test matrices.

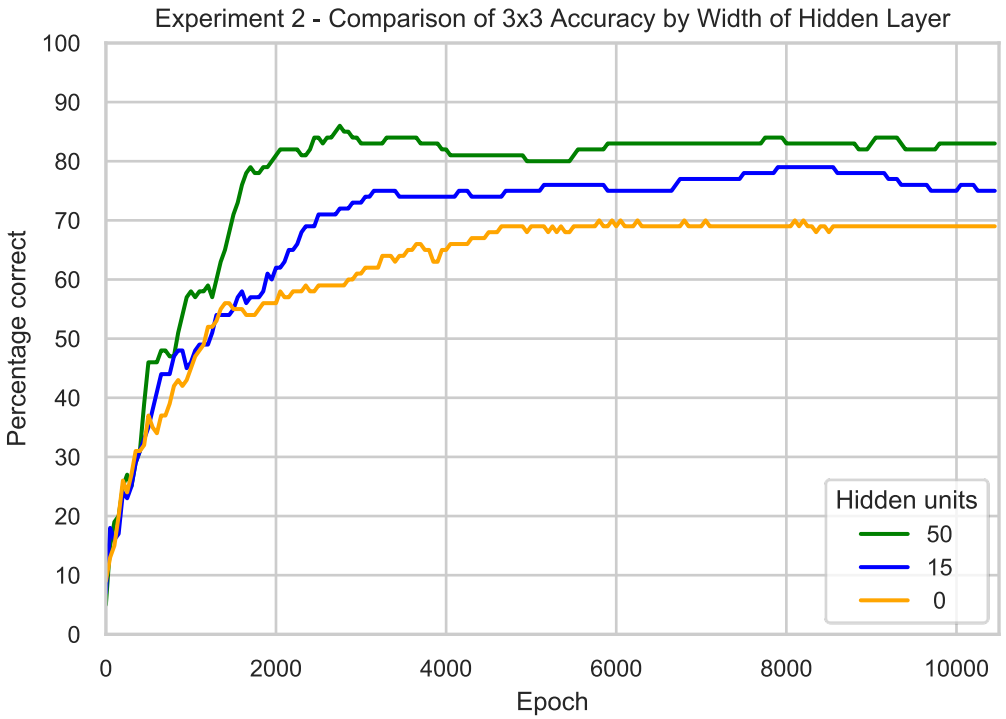

Figure 31: A comparison of the performance of the network with different numbers of hidden units.

Discussion of Experiment 2

The maximum accuracy rates for 2x2 and 3x3 analogy completion were 87% and 86% respectively for Experiment 2 (compared with 90% and 89% for Experiment 1). Although the performance of Experiment 2 is not quite as good, the revised architecture has several advantages.

Firstly, the architecture is simpler – the unnecessary transformation output layer has been removed. It no longer relies on the concept of causal agents and the transformation layer is in an arguably more logical place. Secondly, the clamping mechanism is uncomplicated. The extra step present in the negative phase in Experiment 1 (and the Leech model) is no

longer necessary. Thirdly the system requires fewer hidden units and can even function adequately with none.

Also, in spite of slightly underperforming the network in Experiment 1, we feel there is more opportunity for further improvements with this model. The limitations of time and resources prevented a full exploration of different clamping regimes, hyperparameters and connectivity between the different layers.

The next section delves deeper into the data from experiment 2 to categorize different types of analogy-completion and to compare with data from studies of human cognition.

Further analysis – implications for cognition

Research has shown that analogy completion is modulated by the size of the transformation involved (Leech, Mareschal, & Cooper, 2007). Subjects were asked to complete proportional word analogies which were manipulated so that the semantic distance between concepts was either small or large. It was shown that performance was significantly better on the analogies with large transformations.

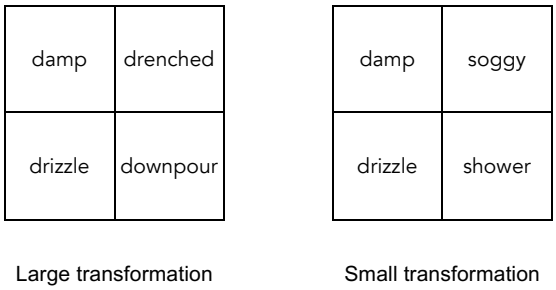

Figure 18: Example word analogies which differ in the size of the transformation involved. Adapted from Leech, Mareschal, & Cooper (2007)

To investigate whether the same pattern is apparent in our network, 100 additional 2x2 matrices involving only 1-relational transformations were generated. These were divided into two categories: *small* and *large* depending on the absolute value of the change in feature value. When these new analogies are completed by the trained network (after 7500 epochs), 94.6% of the matrices with large transformations compared with 68.2% for those with small

transformations. By comparison, the Leech et al. study (2007) of 55 subjects each tested on 14 analogies found means of 76.1% (*SD* = 4.22) and 58.9% (*SD* = 4.18) for the large and small conditions respectively; $t(54) = -5.30$, $p < 0.0001$.

Furthermore, we can examine how this difference evolves as the network learns. Figure 32 shows the progression of the large transformation accuracy compared to the small. It seems that accuracy plateaus early (at over 80%, epoch 500) for analogies involving large transformations, but that the performance on those with finer-grained transformations takes grows more slowly and takes longer to achieve a plateau (at around the 80% mark after 1,900 epochs). It would be interesting to investigate whether the same pattern exists in human development, by comparing, say, the performance of young children vs adolescents on the same 2x2 matrices.

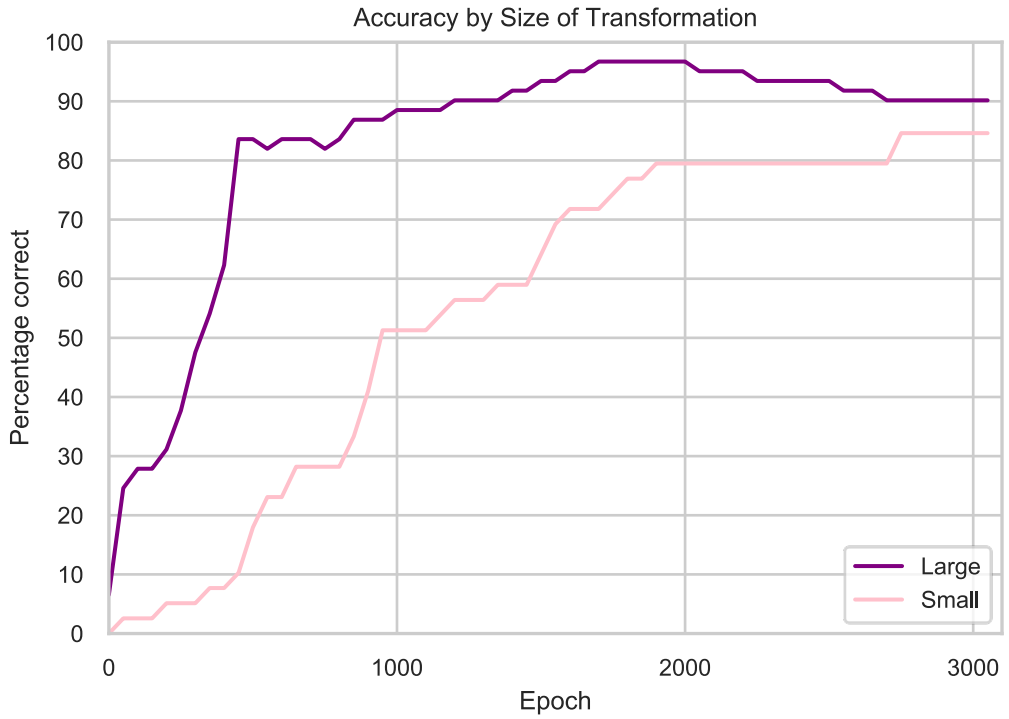

Figure 32: Progression of accuracy by size of transformation over time. Accuracy is established first for analogies involving large transformations.

Complexity of transformation is another way of breaking down the data. Work by Hahn, Chater & Richardson (2003) aims to describe *similarity* in terms of transformational relationships. In one experiment, subjects were asked to compare pairs of shapes and rate how similar they were. The second shape is the same as the first shape with some number of features modified according to a set of transformations similar to the ones considered in this paper (rotation, shading, *etc.*)

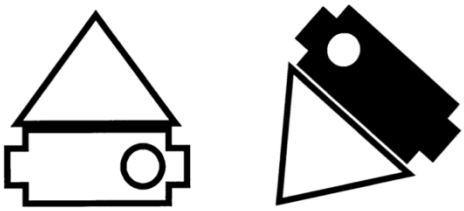

Figure 33: Sample stimuli from an experiment into similarity and transformation. Reprinted from Hahn et al. (2003, p. 16, Figure 3).

The study found that people consider two shapes to be more similar the fewer the transformations involved and that the relationship between transformation distance and similarity was approximately linear. Furthermore, the paper suggests that similarity and analogy are related and that 'establishing the simplicity of a transformation (or sequence of transformations) as a measure of the strength of an analogy' (p. 7) would be an interesting direction for research.

Simplicity as a measure of the strength of an analogy is supported by our model. Figure 34 shows the accuracy by epoch of the network hidden units, broken down by relational complexity. In line with Hahn et al. (2003), successful analogy completion correlates inversely with the number of features involved in the modification, such that the analogies involving fewer feature changes are more likely to be correctly completed.

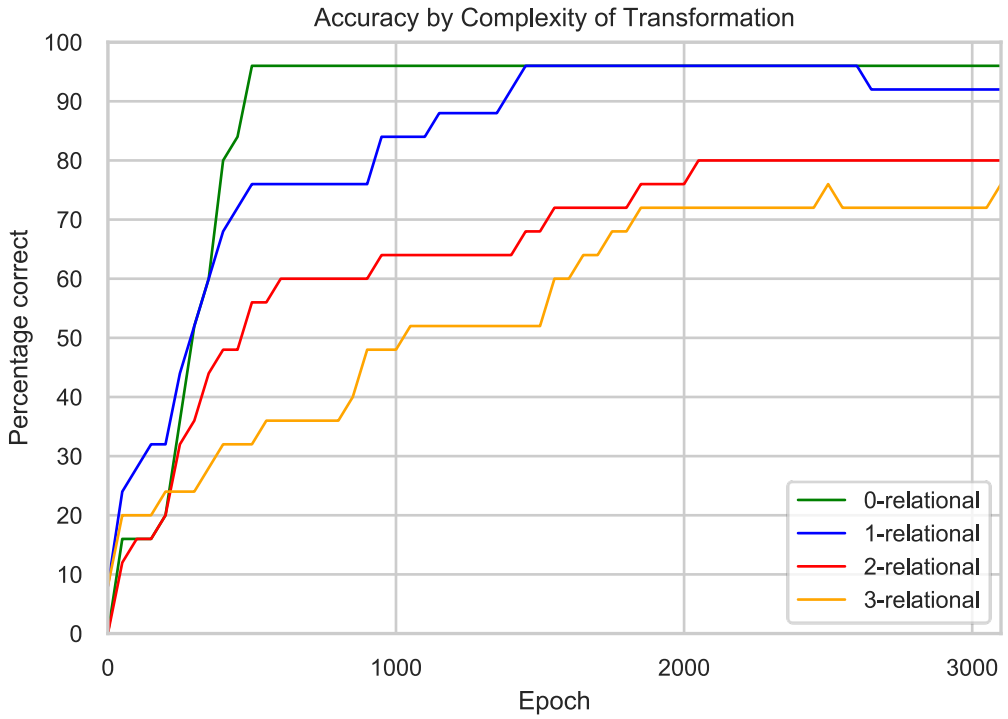

Figure 34: A comparison of performance of the network from Experiment 2 by epoch separated by relational complexity. An *n*-relational transformation is one in which *n* of the features (scale, orientation, shading and number) is modified.

Overall discussion

First and foremost, the experimental results demonstrate that the relational priming approach can successfully be applied to a cognitive task involving abstract visual analogies. Without recourse to any hierarchical or symbolic description of shapes and features, nor explicit specification of rules for comparison, the trained network is capable of completing 2x2 and 3x3 analogies similar to Raven's Progressive Matrices, often considered the hallmark of analogical reasoning tests and central to studies of general intelligence.

The problems used are derived from a restricted subset of the official Raven's Progressive Matrices and while this subset may not reflect the full analogical complexity present in the official tests, they nevertheless embody a significant portion of the challenges involved in the task.

Two different network architectures were presented, both of which learn via a biologically plausible update mechanism. Furthermore, both experiments show patterns mostly in line with cognitive data: analogies involving small transformations produce fewer errors than those involving large transformations; and errors increase when problems involve multiple feature modifications.

The same architecture employed by Experiment 1 has already been shown to successfully target word analogies in the Leech model. Here it has been adapted with success to a visual task, uncontaminated by cultural or educational bias, demonstrating that the approach can transfer to different domains.

Experiment 2 introduced simplifications to the architecture and mechanisms, even to the extent that hidden units may be unnecessary for relational priming to work. This simplicity supports the idea that relational priming may be a more universal cognitive function, more likely to emerge developmentally or evolutionarily.

Perhaps the greatest success of the models presented here is that they provide a demonstration that the simple mechanism of relational priming can solve a complex visual analogical reasoning task that on the surface, looks like it must enlist many cognitive faculties, not only perception and memory, but also higher-order functions such as reasoning, inference, deduction, creativity and imagination. The experiments show that complex problems can be solved by aggregating the results of multiple parallel relational priming operations, each made up of simple low-level cognitive functions. We therefore propose that *concurrent relational priming* is a broader cognitive mechanism for solving complex analogical reasoning tasks.

Conclusions

In their introduction to the Structure Mapping Engine (1989), Falkenhainer et al. state:

> *Cognitive simulation studies can offer important insights for understanding the human mind. […] Unfortunately, cognitive simulations tend to be complex and computationally expensive. Complexity can obscure the relationship between the theory and the program. (Falkenhainer et al., 1989, p. 2)*

Yet, the many steps of the structure mapping algorithms are detailed and elaborate. Structure mapping theory is an example of an approach where the perception step and the analogy-making steps are modelled as separate cognitive functions. By contrast, relational priming's mechanism is simple, its implementation is neurologically, developmentally and evolutionarily plausible. Perception and analogy-completion occur simultaneously and

Chalmers, French and Hofstadter in their critique of the Structure Mapping Engine (1992) insist that in modelling human cognition, content-dependent, flexible, easily-adaptable representations are to be preferred over hand-coded rigid representations. They argue that perceptual processes cannot be separated from other cognitive processes. Further, that 'it is a mistake to try to skim off conceptual processes from the perceptual substrate on which they rest, and with which they are tightly intermeshed.' (p. 210)

If Chalmers et al. (1992) are correct, then how are these conceptual and perceptual processes intermeshed? In the networks presented here, analogy completion in a 2x2 matrix is the process of perceiving a novel object (the input from the second row) through the lens of its context (the exemplars from the first row). The result is a novel concept (the output from the primed network) which is used to complete the analogy. In this domain of 2x2 and 3x3 matrices, the source analogue is fixed by the first row of the matrix, but perhaps in the broader domain of human cognition, the context can come from beyond the immediate visual environment such that our perception is influenced by our goals, our beliefs, our knowledge and our memories. Our model was extended to solve more complex 3x3 matrices by solving

multiple simpler matrices in parallel. Perhaps more complex novel concepts are the product of multiple parallel analogical completions sourced from the simultaneous contexts of an individual's present perceptions, goals, beliefs, knowledge and memories.

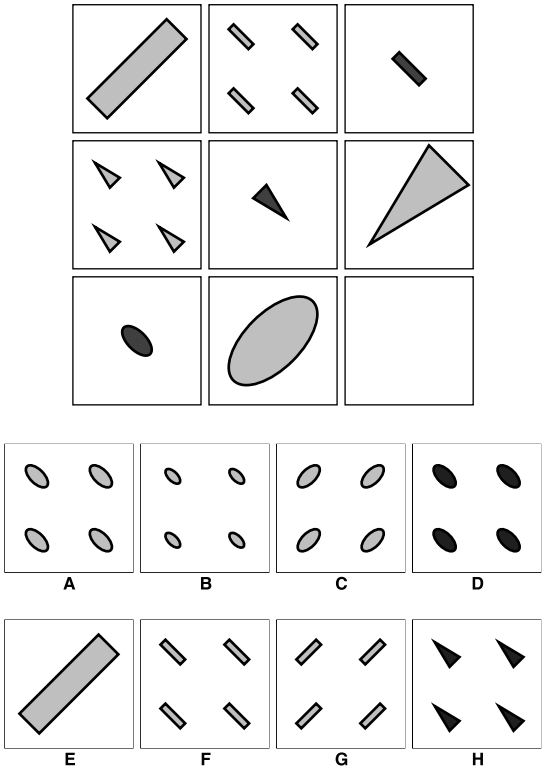

Figure 35: An example of a 3x3 matrix that was correctly solved by the networks from both Experiment 1 and Experiment 2.

The evidence here shows that fairly complex matrices can be solved by aggregating parallel applications of the relational priming mechanism. Moreover, we know that Raven's Progressive Matrices are one of the best single predictors of ability on a range of further intelligence tests. Perhaps the capacity to simultaneously combine percepts with multiple elements from prior knowledge, past experience and the current environment simultaneously, explains performance on these other correlated tests. Perhaps concurrent relational priming is the driver for general intelligence.

References

Barrett, D., Hill, F., Santoro, A., Morcos, A., & Lillicrap, T. (2018). Measuring abstract reasoning in neural networks. *International Conference on Machine Learning*, 511–520. Retrieved from http://proceedings.mlr.press/v80/barrett18a.html

Carpenter, P. A., Just, M. A., & Shell, P. (1990). What one intelligence test measures: A theoretical account of the processing in the Raven Progressive Matrices Test. *Psychological Review*, *97*(3), 404–431.

Chalmers, D. J., French, R. M., & Hofstadter, D. R. (1992). High-level perception, representation, and analogy: A critique of artificial intelligence methodology. *Journal of Experimental & Theoretical Artificial Intelligence*, *4*(3), 185–211. https://doi.org/10.1080/09528139208953747

Falkenhainer, B., Forbus, K. D., & Gentner, D. (1989). The structure-mapping engine: Algorithm and examples. *Artificial Intelligence*, *41*(1), 1–63. https://doi.org/10.1016/0004-3702(89)90077-5

French, R. M. (1995). *The Subtlety of Sameness: A Theory and Computer Model of Analogy-making*. Cambridge, MA, USA: MIT Press.

Gentner, D. (1983). Structure-Mapping: A Theoretical Framework for Analogy*. *Cognitive Science*, *7*(2), 155–170. https://doi.org/10.1207/s15516709cog0702_3

Goswami, U., & Brown, A. L. (1990). Melting chocolate and melting snowmen: Analogical reasoning and causal relations. *Cognition*, *35*(1), 69–95. https://doi.org/10.1016/0010-0277(90)90037-K

Gray, J. R., & Thompson, P. M. (2004). Neurobiology of intelligence: Science and ethics. *Nature Reviews. Neuroscience*, *5*(6), 471–482. https://doi.org/10.1038/nrn1405

Gust, H., Krumnack, U., Kuhnberger, K.-U., & Schwering, A. (2008). *Analogical Reasoning: A Core of Cognition*. 5.

Hahn, U., Chater, N., & Richardson, L. B. (2003). Similarity as transformation. *Cognition*, *87*(1), 1–32. https://doi.org/10.1016/S0010-0277(02)00184-1

Hill, F., Santoro, A., Barrett, D. G. T., Morcos, A. S., & Lillicrap, T. (2019). Learning to Make Analogies by Contrasting Abstract Relational Structure. *ArXiv:1902.00120 [Cs]*. Retrieved from http://arxiv.org/abs/1902.00120

Hofstadter, D. R. (1996). *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought*. New York, NY, USA: Basic Books, Inc.

Hofstadter, D. R., & Mitchell, M. (1994). The Copycat project: A model of mental fluidity and analogy-making. In *Advances in Connectionist and Neural Computation Theory, Vol. 2. Analogical connections* (pp. 31–112). Westport, CT, US: Ablex Publishing.

Hofstadter, D. R., & Sander, E. (2013). *Surfaces and essences: Analogy as the fuel and fire of thinking*. New York, NY, US: Basic Books.

Hunt, E. (1974). Quote the raven? Nevermore! In L. W. Gregg (Ed.), *Knowledge and Cognition* (pp. 129–158). https://doi.org/10.4324/9780203781579

Leech, R. (2004). *A connectionist account of the development of analogical reasoning* (Doctoral Dissertation). Birkbeck, University of London, London.

Leech, R., Mareschal, D., & Cooper, R. P. (2007). Relations as transformations: Implications for analogical reasoning. *Quarterly Journal of Experimental Psychology*, *60*(7), 897–908. https://doi.org/10.1080/17470210701288599

Leech, R., Mareschal, D., & Cooper, R. P. (2008). Analogy as relational priming: A developmental and computational perspective on the origins of a complex cognitive skill. *The Behavioral and Brain Sciences*, *31*(4), 357–378; discussion 378-414. https://doi.org/10.1017/S0140525X08004469

Little, D. R. (2012). *A Bayesian Model of Rule Induction in Raven's Progressive Matrices*. 7.

Lovett, A. M., & Forbus, K. D. (2017). Modeling visual problem solving as analogical reasoning. *Psychological Review*, *124*(1), 60–90. https://doi.org/10.1037/rev0000039

Lovett, A. M., Forbus, K. D., & Usher, J. (2007). *Analogy with Qualitative Spatial Representations Can Simulate Solving Raven ' s Progressive Matrices*.

Lovett, A. M., Forbus, K. D., & Usher, J. (2010). A structure-mapping model of Raven's Progressive Matrices. *Proceedings of CogSci10*. Presented at the Proceedings of CogSci10. Retrieved from https://www.scholars.northwestern.edu/en/publications/a-structure-mapping-model-of-ravens-progressive-matrices

Mackintosh, N. J., & Bennett, E. S. (2005). What do Raven's Matrices measure? An analysis in terms of sex differences. *Intelligence*, *33*(6), 663–674. https://doi.org/10.1016/j.intell.2005.03.004

Matzen, L. E., Benz, Z. O., Dixon, K. R., Posey, J., Kroger, J. K., & Speed, A. E. (2010). Recreating Raven's: Software for systematically generating large numbers of Raven-like matrix problems with normed properties. *Behavior Research Methods*, *42*(2), 525–541. https://doi.org/10.3758/BRM.42.2.525

McGreggor, K., Kunda, M., & Goel, A. (2010). A Fractal Analogy Approach to the Raven's Test of Intelligence. *Proceedings of the 7th AAAI Conference on Visual Representations and Reasoning*, 69–75. Retrieved from http://dl.acm.org/citation.cfm?id=2908587.2908598

Mekik, C. (2018). pyRavenMatrices (Version 0.1.0) [Python]. Retrieved from https://github.com/cmekik/pyRavenMatrices

Mekik, C., Sun, R., & Yun Dai, D. (2017, May 14). *Deep Learning of Raven's Matrices*. Presented at the Proceedings of the Fifth Annual Conference on Advances in Cognitive Systems, Rensselaer Polytechnic Institute, Troy, New York.

Mekik, C., Sun, R., & Yun Dai, D. (2018, July 1). *Similarity-Based Reasoning, Raven's Matrices, and General Intelligence*. 1576–1582. https://doi.org/10.24963/ijcai.2018/218

Mitchell, M. (1993). *Analogy-making As Perception: A Computer Model*. Cambridge, MA, USA: MIT Press.

Morrison, C. T., & Dietrich, E. (1995). Structure-Mapping vs. High-level Perception: The Mistaken Fight over the Explanation of Analogy. *In Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society*, 678–682.

Movellan, J. R. (1990). Contrastive hebbian learning in the continuous hopfield model. *In*, 10–17.

O'Reilly, R. (1998). Biologically Plausible Error-Driven Learning Using Local Activation Differences: The Generalized Recirculation Algorithm. *Neural Comput*, *8*. https://doi.org/10.1162/neco.1996.8.5.895

Ragni, M., & Neubert, S. (2012). Solving Raven's IQ-tests: An AI and Cognitive Modeling Approach. *Proceedings of the 20th European Conference on Artificial Intelligence*, 666–671. https://doi.org/10.3233/978-1-61499-098-7-666

Raven, J. (1938). Standardization of Progressive Matrices. *British Journal of Medical Psychology*, *19*(1), 137–150. https://doi.org/10.1111/j.2044-8341.1941.tb00316.x

Raven, J. (2000). The Raven's Progressive Matrices: Change and Stability over Culture and Time. *Cognitive Psychology*, *41*(1), 1–48. https://doi.org/10.1006/cogp.1999.0735

Raven, J., Raven, J. C., & Court, J. H. (1998). *Manual for Raven's progressive matrices and vocabulary scales*. San Antonio, TX: Pearson.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. https://doi.org/10.1038/323533a0

Seidenberg, M. S., & McClelland, J. L. (1989). A distributed, developmental model of word recognition and naming. *Psychological Review*, *96*(4), 523–568.

Smolensky, P. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1* (D. E. Rumelhart, J. L. McClelland, & C. PDP Research Group, Eds.). Retrieved from http://dl.acm.org/citation.cfm?id=104279.104290

Snow, R. E., Kyllonen, P. C., & Marshalek, B. (1984). The topography of ability and learning correlations. In R. J. Sternberg (Ed.), (Vol. 2, pp. 47-103). In *Advances in the psychology of human intelligence* (Vol. 2, pp. 47–103). Hillsdale, NJ: Lawrence Erlbaum Associates.

Steenbrugge, X., Leroux, S., Verbelen, T., & Dhoedt, B. (2018). *Improving Generalization for Abstract Reasoning Tasks Using Disentangled Feature Representations*. Retrieved from https://arxiv.org/abs/1811.04784

VanLehn, K. (1998). Analogy Events: How Examples are Used During Problem Solving. *Cognitive Science*, *22*(3), 347–388. https://doi.org/10.1207/s15516709cog2203_4

Vosniadou, S., & Ortony, A. (Eds.). (1989). Introduction. In *Similarity and analogical reasoning*. https://doi.org/10.1017/CBO9780511529863

Wang, K., & Su, Z. (2015). Automatic Generation of Raven's Progressive Matrices. *Proceedings of the 24th International Conference on Artificial Intelligence*, 903–909. Retrieved from http://dl.acm.org/citation.cfm?id=2832249.2832374

Xie, X., & Seung, H. S. (2003). Equivalence of Backpropagation and Contrastive Hebbian Learning in a Layered Network. *Neural Computation*, *15*(2), 441–454. https://doi.org/10.1162/089976603762552988

Appendix A – Comparison of Experiment 1 with the Leech model

In experiment 1, while we attempted to follow the Leech model (2004; Leech et al., 2008) as far as possible, this appendix details some of the similarities and differences.

Table 1.

Parameters used in Experiment 1 compared to the Leech model

Parameter	Experiment 1	Leech model
Input element units	11	40
Input transformation units	4	4
Hidden units	50	40
Output element units	11	40
Output transformation units	4	4
Learning mechanism	asynchronous CHL	synchronous CHL
Propagation mechanism	threshold-based	20 cycles
Learning rate η	0.05	0.1
Logistic growth rate k	0.1	0.1

In the Leech model, during the primed phase, the transformation is set to rest, and only the new input pattern is clamped. The outputs are left primed. In our model, this does not lead to adequate performance – the transformation nodes do not get completed accurately enough during priming. Instead, we set the outputs to rest and leave the transformation primed. We also clamp both the new input and the primed transformation.

The Leech model uses the synchronous version of contrastive Hebbian learning. Both experiment 1 and experiment 2 use the more biologically plausible asynchronous variety.

In the Leech network, activation is propagated through the network for 20 cycles. In our experiments, propagation continues until the delta of activation in the hidden layer falls below a threshold of 1×10^{-5} with a maximum of 100 cycles.

In the unclamped negative phase of the Leech model, the clamped units were unclamped after 2 cycles and activation was allowed to propagate freely for the remaining 18 cycles. (This is the aforementioned deviation from canonical CHL). In our implementation, as explained in the text, propagation was allowed to settle twice, firstly with the inputs clamped, and then again with all units unclamped. This performed better than the Leech regime, possibly on account of the aforementioned asynchronous update strategy.

In experiment 2, all the values in Table 1 remain the same as for Experiment 1, except for the learning rate which was 0.001.

Appendix B – Possible further investigations

The types of matrix used in this study were a relatively limited subset of the full possibilities of RPM format. It would be feasible to extend the model to cope with 3x3 properties of progression, distribution of 2, addition and subtraction, composite shapes.

In terms of further testing of the existing model, it would be interesting to explore interpolation vs extrapolation similar to the methods in Barrett et al. (2018). Can a network trained on even numbered feature values solve test data consisting only of odd numbered ones (interpolation)? Can a network trained on bottom half values solve top half ones (extrapolation)?

It would straightforward to adapt the networks to target the dataset of Raven-like problems provided by Matzen et al. (2010) which has the advantage of being normed against human subjects. This would allow for a better comparison with human performance.

The same relational priming approach could be extended to the domain of letter sequences, such as those from the Copycat model (Hofstadter & Mitchell, 1994). Then the approach would have been applied to the three most prevalent domains in studies of analogy: fluid intelligence tests such as RPMs, letter sequences and proportional analogies.