Solving Proportional Analogy Problems using Tensor Product Networks with Random Representations

William H. Wilson
Computer Science and Engineering
University of New South Wales
Sydney 2052 Australia
billw@cse.unsw.edu.au

Deborah J. Street
School of Mathematical Sciences
University of Technology Sydney
Box 123, Broadway 2007 Australia
deborah@maths.uts.edu.au

Graeme S. Halford
Department of Psychology
University of Queensland
Queensland 4072 Australia
gsh@psy.uq.oz.au

ABSTRACT

This paper describes the use of vectors with randomly generated components for representing concepts in a system for solving proportional analogy problems using a memory based on a tensor product network. Both dense random vectors (all or most components non-zero) and sparse random vectors (most components zero) are used. In both cases systems which are able to solve proportional analogy problems were successfully produced: the degree of success varied with the number of components in the vectors and/or the proportion of non-zero components.

1 Introduction

The STAR (Structured Tensor Analogical Reasoning) model [4,5,6] solves analogical problems using a distributed connectionist approach. In this respect it contrasts with ACME [7] and SME [3]. The original versions of STAR relied on distributed representations for concepts which comprised an orthonormal set of vectors. The STAR model does not set out to be faithful model of cognitive processing at the neural level, but this restriction to using orthonormal sets of vectors seemed worth exploring. This paper describes two attempts to remove the restriction, replacing the orthonormal set of representation vectors by vectors with random components.

Section 2 outlines the basics of tensor product networks and briefly describes the way in which the STAR model uses them to solve proportional analogy problems. Section 3 describes the process of choosing random components for representation vectors, and modifications necessary to make the model work with them. Section 4 describes the results of experiments with networks using random representation vectors.

2 Tensor Product Networks in STAR

Tensor product networks have been used for purposes like the modelling of variable binding and for working memory structures in connectionist implementations of production system architectures [2,9]. They have the property, sometimes an advantage, of being one-shot learning systems. A tensor product network has a *rank*: all networks used in the experiments described in this paper were of rank 3, and we will describe tensor product networks in terms of the rank 3 case.

2.1 Representing and Retrieving Facts

In general, a rank 3 tensor product network has 3 dimensions - of size *p*,*q*, and *r*, say. Its processing units include *pqr binding units*, and input/output units grouped into 3 vectors of length *p*, *q* and *r*. In the underlying application for the experiments described in this paper, one of the I/O vectors represented a *predicate*, and the other two represented a pair of *arguments*. The information represented in the tensor (i.e. in the binding unit structure) was thus relational information such as larger–than(mare, foal) (the analogical reasoning tasks that the STAR system undertook included proportional analogies such as *mother* is to *baby* as *mare* is to *what?*)

To store the fact larger–than(mare, foal) in the tensor, one computes the tensor product larger–than⊗ mare⊗ foal of vectors representing the concepts larger–than, mare, and foal and adds it to the values stored in the binding units: thus if **l** is a vector that represents larger–than, **m** represents mare, and **f** represents foal, one would add $l_i * m_j * f_k$ to the value stored in binding unit b_{ijk}, for each index-value of i, j, and k.

A simple mechanism allows retrieval ("unbinding") of the stored information. It is the case that if the vectors used for predicates (i.e. for one "axis" of the tensor) form an orthonormal set, and similarly for the vectors used for first arguments and for second arguments, then given any two "pieces" of a relational instance, one can retrieve the third.

For example, given **m** (mare) and **f** (foal), one can retrieve the relationship(s) between them (such as **l** (larger-than)) that are stored in the tensor. In general, more than one relationship may hold between the arguments (e.g. mother-of(mare, foal)). Thus what is retrieved will be a sum of vectors representing predicates - we term this sum a *predicate bundle*. As well, given all three "pieces" one can check whether the tensor holds this relational instance, and in fact, given any one "piece", one can retrieve from the tensor a tensor of lower rank representing the relational instances involving that "piece". In the case of retrieving predicates given **m** (mare) and **f** (foal), the computation is as follows, where p_i signifies the i-th component of the predicate bundle retrieved:

$$p_i = \Sigma_{jk} b_{ijk} * m_j * f_k$$

Figure 1 shows the binding units and input/output vectors in an 8x8x8 tensor product network.

2.2 Analogical Reasoning in STAR

The STAR model of human analogical reasoning [4,6] uses tensor product networks based on orthonormal sets of representation vectors, with *exact unbinding* [9]. As an aim of the STAR model is to provide a distributed model of analogical reasoning, it is desirable for the representation vectors to have a high proportion of non-zero components. This can be achieved in a systematic way by using the rows of a Hadamard matrix, suitably normalised [11].

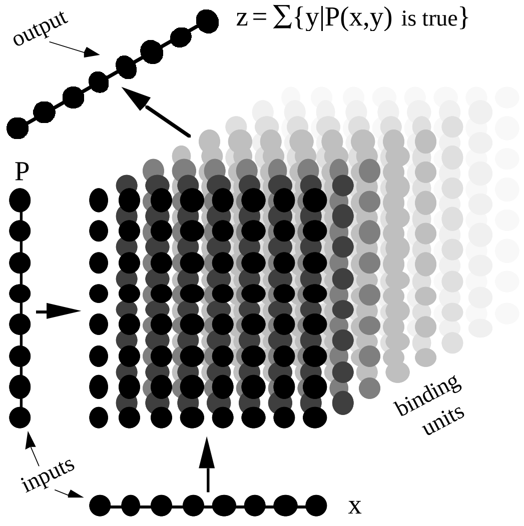

Fig. 1: Using a tensor product network to find { y |P(x,y)} . From [11].

The STAR model associates a representation vector with each concept that is to be represented. In [4,5,6,11], the vectors used for each axis of the tensor form an orthonormal set (and they have no zero components). A collection of facts involving the concepts of interest, some relevant to the reasoning problem, some not, are stored in the binding units.

In tensor networks, the I/O vectors can serve both as input and as output vectors. We shall illustrate the STAR method of solving proportional analogies, using the problem *mother* is to *baby* as *mare* is to *what?*

(1) representations of mother and baby are presented as inputs to the argument vectors in the tensor, and the output is a predicate bundle: the sum of the vectors representing predicates relating mother to baby: e.g. loves, feeds, larger-than.

(2) this unanalysed predicate bundle is then presented as an input to the tensor product network on the predicate axis, along with the representation of mare, on the first argument axis. The output on the second argument axis is a linear combination of argument concepts. For example, if larger-than(mare, rabbit) is in the tensor, then rabbit will show up in the output of this step, with a coefficient which, in effect, indicates how likely it is as a solution to this analogy problem. This is illustrated in Figure 2.

3 Using Random Representations in STAR

In [11], it was found that the STAR model is robust, in the sense that damaging significant numbers of binding units, so that they either produced no output or noisy output, only mildly degraded the performance of the system. Typical output from an intact system, for our example problem *mother:baby::mare:what?* would be scores of, say, 3 for *foal* (corresponding to the 3 predicates *mother-of, larger-than,* and *feeds* in common between *mother/baby* and *mare/foal,* and stored in the tensor) and 1 for *rabbit* (larger-than). Typical output from a degraded system might be 1.294 for *foal,* 0.331 for *rabbit,* and scattered smaller scores, some negative, for other concepts (like *mother, baby, mare*). When the tensor binding units are "damaged" in this way, the information in the tensor no longer corresponds exactly to orthonormal sets of representation vectors.

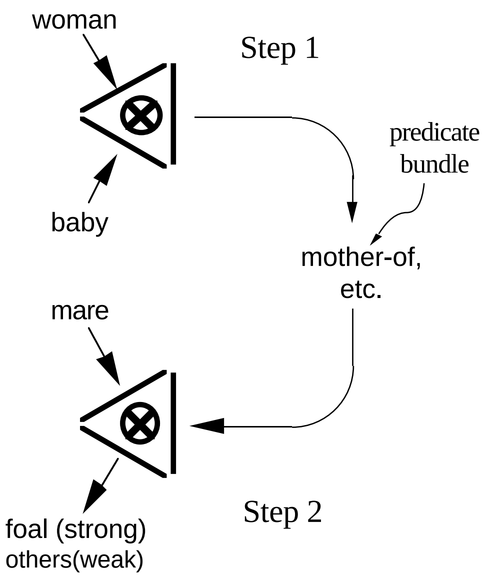

Figure 2: Simple analogical reasoning with a tensor product network - the triangular symbols represent a rank 3 tensor product.

3.1 Dense Random Representations

This suggested that the requirement for orthonormal sets of vectors can be weakened to "approximately orthonormal". It is the case that pairs of vectors composed of uniform random numbers have an expected inner product of 0 (but with a significant variance). Such vectors could be considered to be "approximately orthogonal," and thus good candidates for the use of the *self-addressing unbinding procedure* described by Smolensky [9].

The STAR model was modified so as to use normalised random vectors generated by choosing each component to be a uniform random number from the range [–1,+1], and normalising the resulting vector **v**. Note that normalisation has the effect that vectors with more components, generated as described, will (on average) have smaller components than vectors with fewer components, since |**v**|, the length of the vector, will increase with the number of components. This suggests that experiments with these vectors should investigate the effect of increasing the number of components of the vectors on the performance of the networks.

3.2 Sparse Random Representations

Having a distributed representation does not of course require that all or even most nodes have non-zero activations, just that any concept should be represented by a significant number of active nodes. This consideration, together with data of McNaughton et al. quoted in [8], to the effect that percent activity of neurons in the different regions of the hippocampal system ranges from about 0.5% to about 9%, suggests that a possible design principle for vectors representing concepts in a memory system is "have most of the components of the representation vector zero (inactive)." Again, we stress that we are not attempting a faithful neural-level model of analogical reasoning in STAR, but rather borrowing ideas about the human memory system.

The version of our model which uses sparse random vectors constructs the vectors as follows: all *k* vectors have the same number *n* of components, and a fixed number *s* of these are non-zero (*s*<<*n*). The *s* non-zero components of a vector are chosen by generating *s* different random positions between 1 and *n* at which the non-zero components will be placed, generating *s* uniform random numbers from [–1,+1], placing them at the chosen positions, and then normalising the resulting vector.

For very small values of *sk* (relative to *n*), this will sometimes result in sets of *k* vectors which are actually orthonormal, simply because the random positions chosen happen not to overlap each other. Such networks produce perfect results, but, of course, they are also minimally distributed (for *s*=1 they are in fact localist nets), and, as described in the next section, they occasionally throw up sets of vectors which perform abysmally.

4 Experiments with Random Representations

4.1 Dense Random Vector Experiments

Systems with vectors containing varying numbers of components were constructed and tested on a standard analogy problem *woman:baby::mare:what?*. With a perfect network, there would be activation scores of (foal=3, rabbit=1, others=0). For each system generated, the scores for *foal* and *rabbit* were recorded, along with the highest score for the rest of the represented concepts. In some cases the scores for all concepts were recorded.

For each number of components used, 100 runs (with different random number seeds) were done. We deemed a result *good* if the score for *foal* was at least 20% better than that for *rabbit,* and the score for *rabbit* was at least 20% better than the highest of the remaining concept scores. A result was rated *fair* if, though not *good*, the scores for *foal, rabbit*, and the highest other score were in descending order. A result was rated *poor* if it was neither *fair* nor *good*, but the score for *foal* was at least 20% better than the highest other score (in particular, *poor* results all rated *rabbit* third highest, or worse). Any other results would be rated *bad* (in fact, no *bad* results occurred). The outcomes are shown in Table 1.

Table 1: Classification of solutions produced by dense random vector experiments (1000 runs for each number of components; 20 facts; solving woman:baby::mare:what?)

components	% good	% fair	% poor
25	40.9	10.3	48.8
40	55.5	7.5	37.0
70	72.5	9.3	18.2
100	84.8	4.3	10.9
130	92.6	3.5	3.9
160	94.5	2.5	3.0
190	96.7	1.8	1.5
220	97.4	1.4	1.2
250	98.7	1.2	0.1

In the case of 40 components, we investigated the distribution of the scores for the individual concepts. They looked approximately normally distributed, with means around 3 (foal), 1 (rabbit), and 0 (all other concepts). Testing for normality was done by the Kolmogorov-Smirnov method [10]: 6 of 8 concepts were confirmed as having scores distributed normally; the remaining 2 concepts were not too far off normal.

The classification of the outcomes involves comparing the scores for *foal* and *rabbit* with the largest of the scores for the other six concepts: this will be distributed as the 6th order statistic of the essentially normal distributions of those 6 concepts. We found that this data could be modelled by the gamma distribution, whose probability density function is:

$$f(x) = b^{-a}x^{a-1}e^{-x/b}/\Gamma(a),\ a>0,\ b>0.$$

The maximum likelihood estimates of a and b using the method of Choi and Wette [1]. The parameter estimates were $\hat{a} = 2.86$ and $\hat{b} = 0.297542$.

4.2 Sparse Random Vector Experiments

Our experiments with sparse random representations used vectors of length 500 (and so 125 million binding units in the tensor)[1]. Each experimental run used a fixed number of non-zero components, ranging from 1 to 25, and chose the components initially as uniform random numbers in [–1,+1], but then normalized the resulting vectors. Twenty runs were done with each number of non-zero components.

Let *nzc* denote the number of non-zero components. One would expect the results for low *nzc* to range from excellent (when a completely orthonormal basis happens to be generated) to maverick (when similar

[1] A sparse data structure was used for the tensor, viz. a 2-dimensional array of lists of (index, value) pairs.

vectors are generated for different concepts). Similar vectors are more probable for low *nzc*. Furthermore, the average length of random vectors increases as √*nzc*, so normalisation reduces the size of components of vectors with small *nzc* less than for vectors with large *nzc*. Thus “bad” components should do more damage, for small *nzc*.

Results, as shown in Fig. 3, are good compared with those for the dense random vectors (compare Table 1).

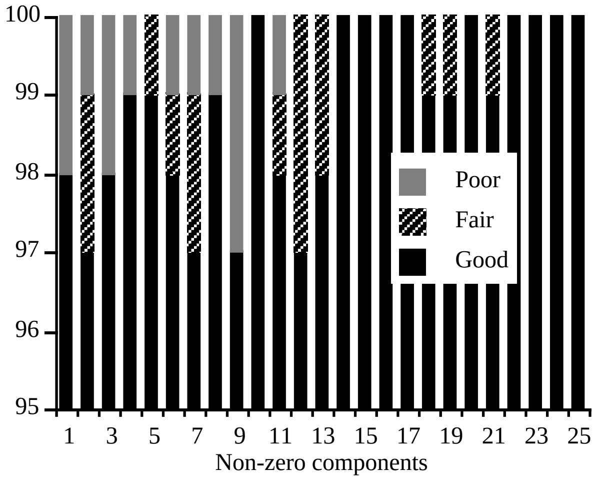

Fig. 3: Classification of solutions produced by sparse random vector experiments. There were 100 runs for each number of non-zero components between 1 and 50. For more than 25 non-zero components, all solutions were classified as “good”.

As with the dense random representations, we inspected the distributions of the scores for *foal*, *rabbit*, and the other argument concepts known to the system (i.e. the concepts which could possibly be produced as solutions to the analogy problem). The cases with smaller *nzc* (1-13)[2], as expected, produced rather wild and spiky distributions (the worst case being an instance with *nzc* = 1 which produced a score of 2 for *foal*, –2 (sic) for *rabbit*, and 0 for the rest.)

The cases with 14-25 were examined in more detail. For these, the results for *foal* and *rabbit* conformed to a normal distribution with means 3.015 and 1.015 respectively, and standard deviations 0.226 and 0.200 respectively. The other concepts were non-normal: typically a symmetric distribution with mean close to 0, and standard deviations mostly from 0.13–0.17 (one at 0.23), and with a tall spike at the mean. This is illustrated in Figure 4. Again, testing for normality was done by the Kolmogorov-Smirnov method.

As with the dense random vectors, the classification in Fig. 3 entails comparing scores for *foal* and *rabbit* with the largest score among the other six concepts: this will be distributed as the 6th order statistic of the spiked, normal-like distribution of those 6 remaining concepts. This data could also be modelled by the gamma distribution, with parameter estimates $\hat{a}$ = 1.39 and $\hat{b}$ = 0.159416.

```
                    .
                    :
                    :
                    :
                    :
                    :
                   :::
                ..:::::....
---+---------+---------+---
-1.0        0.0       1.0
```

Fig. 4: Spiky, symmetric distribution of scores for the concept *burrow*. Each dot represents 12 points.

4.3 Another Analogy and More Facts

One might reasonably ask whether the results found with one particular analogy and set of facts would hold over a wider range of analogies and fact sets. At present our results here are incomplete, but confirm what might be inferred from § § 4.2 and 4.3. We have run the dense random vector experiments with the analogy problem *woman:house::rabbit:what?* and a set of 40 facts including the 20 facts used in the experiments reported above. With a perfect network, the activation scores would be burrow=1, others=0.

During each run, we also tested fact recognition: that is, recognition scores were calculated for each of the 40 facts, and also for 20 non-facts (i.e. propositions like *feeds(burrow, house)* which are expressible in terms of the concepts known to the network, but which have not been taught to the network.)

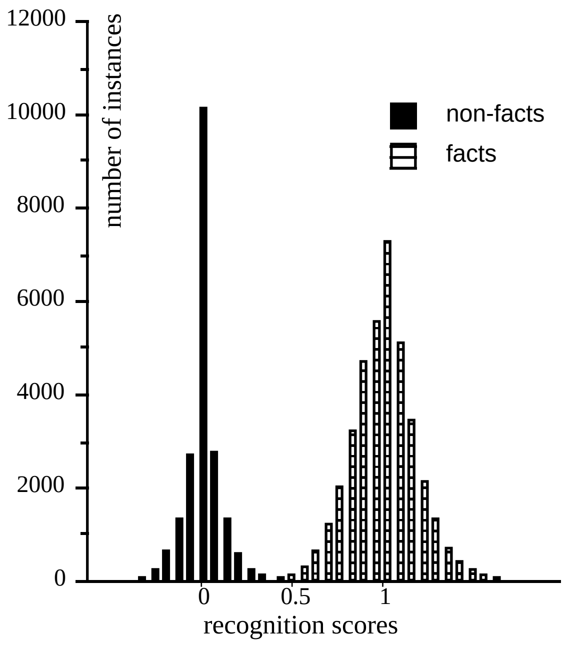

Fig. 5: distribution of recognition scores for 40 facts and 20 non-facts for a tensor with dense random representations and axis length 250, over one thousand runs.

For each fact, and for every tensor size simulated (from 40x40x40 to 250x250x250) we found that the mean recognition score for every *non-fact* was close to 0, and the mean recognition score for every *fact* was close to 1. The variance decreased, for both facts and non-

---

2 Nine was the largest value of *nzc* for which an orth onormal basis was generated during the experi ment, but larger values also often produced scores that were close to zero for all concepts but *rabbit* and *foal*. This perturbs the distribution of the order statistic.

facts, as the tensor became larger. Fig. 5 shows the distribution of recognition scores for the 250x250x250 tensor. If a recognition score ≥ 0.48 is used as the criterion of acceptance, then in the 250x250x250 case, 0.0625% of facts would have been classified as non-facts, and 0.035% of non-facts would have been classified as facts.

For the second analogy problem, *woman:house:: rabbit:what?*, a new classification of good/bad/etc. results was necessary, as there is in this case no second answer, like *rabbit* in *woman:baby::mare:what?* with any plausibility. Thus we classified a solution as *good* if the score for *burrow* was at least 20% larger than the next highest score, *poor* if the solution was not *good*, but the score for *burrow* was greater than any other, and *bad* otherwise.

Table 2 shows that, according to these criteria, this analogy was well-solved by all but the smallest networks.

Table 2: Classification of solutions produced by dense random vector experiments (1000 runs for each number of components; 40 facts; solving woman:house::rabbit:what?) With more than 100 components, all solutions found by the simulations were classified as good.

components	% good	% poor	% bad
25	99.2	0.4	0.4
40	99.9	0.1	0.0
70	100.0	0.0	0.0
100	100.0	0.0	0.0

5 Conclusions

Tensor product networks using dense random representations are reasonably successful at solving our target analogy problems. The success rate increases with the number of components (along with the time and space required) in the cases studied.

Tensor product networks using sparse random representations are very successful at solving our target analogy problem, though there are two problems for small numbers of non-zero components: one conceptual - such representations are not truly distributed and one practical: performance ranges from near-perfect to absurdly wrong without intermediate cases. The high success rate does not vary much with the number of components in the cases studied.

Our experiments were limited by the amount of memory available on the computer used for the simulations, and it would be interesting to check that the results are confirmed for larger dense random vectors and larger sparse random vectors (both more components and more non-zero components). It would also be interesting to investigate the robustness of the random representations to destruction or randomization of binding units (cf. [11]).

It would also be interesting to replicate the experiments with larger fact bases and concept spaces, and different analogy problems.

References

[1] S.C. Choi and R. Wette, Maximum likelihood estimation of the parameters of the gamma distribution and their bias, *Technometrics* **11** (1969) 683-690.

[2] C.P. Dolan and P. Smolensky, Tensor product production system: A modular architecture and representation, *Connection Science* **1** (1989) 53-68.

[3] Falkenhainer, B., Forbus, K.D. and Gentner, D., The structure-mapping engine: algorithm and examples, Artificial Intelligence **41** (1990) 1-63.

[4] G.S. Halford, J. Wiles, M.S. Humphreys, & W.H. Wilson, Parallel distributed processing approaches to creative reasoning: Tensor models of memory and analogy, in *AI and Creativity* ed. T. Dartnall, S. Kim, and F. Sudweeks, AAAI Technical Report (1993).

[5] Graeme S. Halford and William H. Wilson, Creativity and capacity for representation: Why are humans so creative, *AISB Quarterly* **85** Autumn 1993, 32-41.

[6] G.S. Halford, W.H. Wilson, J. Guo, J. Wiles, and J.E.M. Stewart (in press), Connectionist implications for processing capacity limitations in analogies, pages 363-415 in K.J. Holyoak and J. Barnden (editors) *Advances in Connectionist and Neural Computation Theory: Volume 2: Analogical Connections*, Norwood, NJ: Ablex, 1994.

[7] Holyoak, K.J., and Thagard, P., Analogical mapping by constraint satisfaction, *Cognitive Science* **13**(3) (1989) 295-355.

[8] Randall C. O'Reilly and James L. McClelland, Hippocampal Conjunctive Encoding, Storage, and Recall : Avoiding a Tradeoff, Technical Report PDP.CNS.94.4, Carnegie Mellon University, June 1994.

[9] P. Smolensky, Tensor product variable binding and the representation of symbolic structures in connectionist systems, *Artificial Intelligence* **46** (1990) 159-216.

[10] M.A. Stephens, Kolmogorov-Smirnov-type tests of fit, 398-402 in *Encyclopedia of Statistical Science* **4**, edited N.L. Johnson and S. Kotz, New York: Wiley, 1983.

[11] William H. Wilson and Graeme S. Halford, Robustness of tensor product networks using distributed representations, in *Proceedings of the Fifth Australian Conference on Neural Networks, ACNN'94*, edited by A.C. Tsoi & T. Downs, Brisbane, 31 January-2 February 1994, 258-261.