Compositionality in a Knowledge-based Constructive Learner

François Rivest

Département d'Informatique et de Recherche Opérationnelle
Université de Montréal
CP 6128 succursale Centre Ville, Montréal, QC H3C 3J7, Canada
francois.rivest@mail.mcgill.ca

Thomas R. Shultz

Department of Psychology and School of Computer Science
McGill University
1205 Penfield Avenue, Montreal, QC H3A 1B1, Canada
thomas.shultz@mcgill.ca

Abstract

A knowledge-based constructive learning algorithm, KBCC, simplifies and accelerates the learning of parity and chessboard problems. Previously learned knowledge of simpler versions of these problems is recruited in the service of learning more complex versions. A learned solution can be viewed as a composition in which the components are not altered, showing that concatenative compositionality can be achieved in neural terms.

The Nature of Compositionality

The idea of compositionality is that mental representations are built out of parts and possess a meaning that is derived from the meanings of the parts and how the parts are combined. A symbolic expression can exhibit what is called *concatenative* compositionality, which means that the expression incorporates its constituents without changing them. This kind of compositionality makes it possible for symbolic propositions to express the hierarchical, tree-like structure of sentences in a natural language. It has been argued that compositionality exists in a wide variety of psychological processing, not just in language (Bregman, 1977). Because neural networks supposedly cannot express compositionality, it is claimed that they cannot simulate comprehension and production of language, nor other forms of thought, at least some of which are considered to be language like (Fodor & Pylyshyn, 1988; Pinker, 1997).

In response, some connectionists have argued that current neural networks exhibit a unique functional form of compositionality that may be able to model the compositional character of cognition even if the constituents are altered when composed into a complex expression (van Gelder, 1990). Here the original constituents are retrievable from the complex expression by natural connectionist means. For example, an encoder network can learn to encode simple syntactic trees on distributed hidden unit representations and then decode them back into the same syntactic trees at the outputs (Pollack, 1990).

In this paper, we apply a knowledge-based neural algorithm to the problem of learning compositional structures.

Knowledge-based Cascade-correlation

A relatively new algorithm called knowledge-based cascade-correlation (KBCC) recruits previously learned source networks as well as single hidden units (Shultz & Rivest, 2001). This algorithm arguably implements a new kind of neural compositionality in which recruited components are preserved intact as they are combined into an overall solution of the target task, as in classical concatenative compositionality.

KBCC is a variant of cascade-correlation (CC), a feedforward constructive algorithm that grows a network while learning, essentially by recruiting new hidden units as needed (Fahlman & Lebiere, 1990). The innovation in KBCC is the potential to recruit previously learned networks, in competition with single sigmoid hidden units. The computational device that gets recruited is the one whose output correlates best with existing network error, just as in classical CC.

At the start of training, a KBCC network has only a bias unit and input units fully connected to output units with small randomized weights. During the first phase, called the *output phase*, weights entering the output units are trained to minimize the sum of squared error, which is defined as follows:

$$E = \sum_o \sum_p \left(V_{o,p} - T_{o,p}\right)^2 \quad (1)$$

where V is the activation of output o in response to training pattern p, and T is the corresponding target activation value that the network is learning to produce.

If reduction of error E stagnates or a maximum number of epochs is reached without the problem being learned, then the algorithm shifts to what is called *input phase*. Here a pool of candidate units and previously trained source networks is collected, and small random weights are used to connect every non-output unit of the target network to each input of each candidate recruit. Those weights are then trained to maximize a scaled covariance between each candidate's outputs and the residual error in the target network. This scaled covariance for each candidate is defined as follows:

$$G_c = \frac{\left\| Cov\left(V_c, E\right) \right\|^F}{\sum_o \sum_p E^2{}_{o,p}} \quad (2)$$

where $E_{o,p}$ is the error at output unit o for pattern p, V_c is the matrix of candidate output activations for all patterns, E is the matrix of target network error for all patterns, and $\|C\|^F$ is the Frobenius norm of the covariance matrix, defined as:

$$\|C\|^F = \sum_{i,j} C^2{}_{i,j} \quad (3)$$

This input phase training continues until a maximum number of epochs is reached or until the increases in correlations stagnate. When the input phase is over, the candidate with the highest G value is retained and installed in the network with new connection weights from the outputs of the recruit to the target network's output units. These new weights are initialized with small random values having the negative of the sign of the correlation.

The algorithm then shifts back to output phase to adjust all target-network output weights in order to use the new recruit effectively. KBCC continues to cycle back and forth between output and input phases until learning is complete, or the maximum number of epochs expires. A hypothetical KBCC target network with three recruits is illustrated in Figure 1.

KBCC has so far been applied to learning about geometric shapes under various transformations such as translation, rotation, and size changes (Shultz & Rivest, 2001), learning to recognize vowels (Rivest & Shultz, 2002), and learning to identify splice junctions in a genetic database (Thivierge & Shultz, 2002).

An illustration of KBCC compositionality from past work is presented in Figure 2. To learn a cross shape, this network recruited previously-learned vertical and horizontal rectangles, greatly shortening learning time and lessening the number of recruits and connection weights (Shultz & Rivest, 2001). Here the target and source networks had to distinguish points inside a shape from those outside the shape.

It is worth noting that the recruited components of the cross in Figure 2 are very similar to their original sources and the original sources remain unaltered by the composition. All of this is characteristic of concatenative compositionality.

Additional details of both CC and KBCC can be found elsewhere (Shultz & Rivest, 2001; Shultz, 2003). In this paper, we apply KBCC to two well-known problems in the machine-learning literature for which post-learning generalization has been difficult – parity and chessboards.

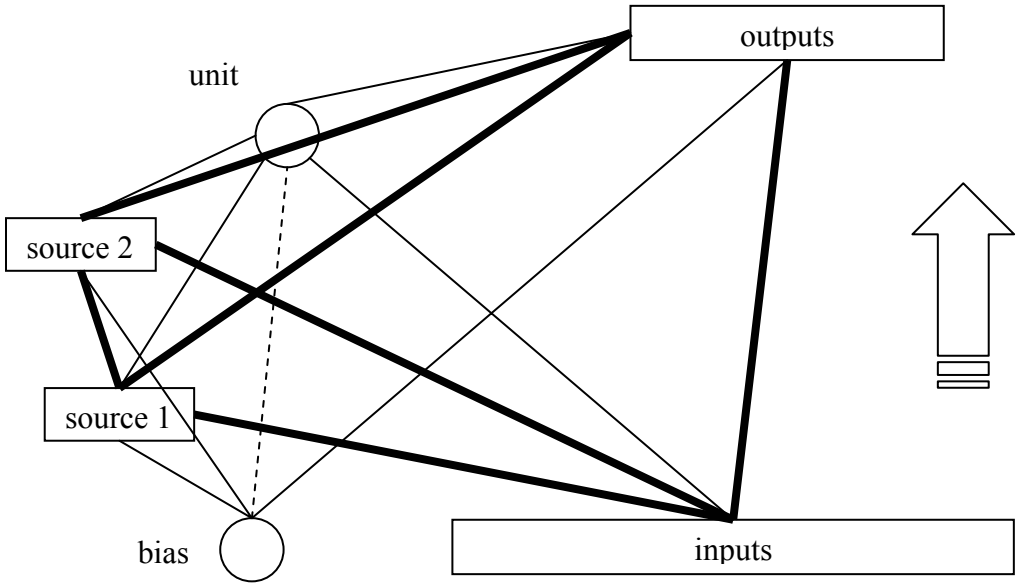

Figure 1. Hypothetical KBCC target network that has recruited two source networks and a sigmoid unit. The dashed line represents a single connection weight, solid thin lines represent weight vectors, and solid thick lines represent weight matrices. The arrow indicates direction of activation flow.

a. b. c.

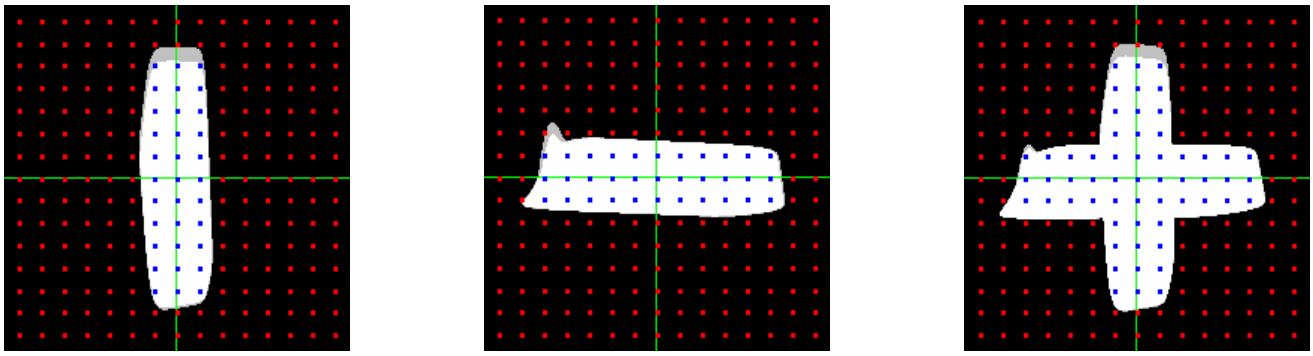

Figure 2. Output activation diagrams for a network that learned a cross target (*c*) by recruiting its two components (*a* and *b*). Dark dots represent training points inside a shape; light dots training points outside a shape. A white background indicates generalization to test points inside a shape, black background indicates generalization to test points outside a shape, and gray background indicates intermediate values.

Parity

The parity problem has a fixed number of binary inputs. A parity-learning network learns to turn on its binary output unit if an odd number of inputs are on, and otherwise turn off this output unit. Our experiment involved training 30 KBCC networks to learn a standard parity-8 problem under four different knowledge-source conditions, each of which had 12 different computational devices for possible recruitment.

Condition *a*, with 12 sigmoid units, was essentially a control condition without prior knowledge of parity or anything else. One-half of the recruitment candidates in condition *b* were networks trained on parity-2 problems, while the other half were sigmoid units. In condition *c*, one-half of the candidates were networks trained on parity-4 problems; the other half were sigmoid units. In condition *d*, there were 4 sigmoid candidates, 4 networks trained on parity-2, and 4 networks trained on parity-4.

Results are shown in Table 1 in terms of mean recruits and mean number of epochs needed to master the parity-8 target problem. In each condition, sigmoid units were heavily recruited. Parity-2 networks were rarely recruited, but parity-4 networks were often recruited when available.

A factorial ANOVA of the total number of recruits was significant, $F(3, 116) = 6.6$, $p < .001$. Multiple comparison of means revealed that conditions *c* and *d*, with parity-4 candidates, needed fewer recruits than did conditions *a* and *b*, $p < .05$.

A factorial ANOVA of the number of epochs to learn yielded similar results, $F(3, 116) = 11$, $p < .001$. Multiple comparisons indicated that learning was faster in conditions *c* and *d*, where parity-4 sources were available, than in conditions *a* and *b*, $p < .05$.

Chessboards

The chessboard problem is a generalized version of the exclusive-or problem which, in turn, can be viewed as a two-unit parity problem. There are two inputs which represent the coordinates of the squares in a chessboard. For one color of squares, the output is to be turned off; for the other color, the output is to be turned on.

We trained 30 KBCC networks to learn an 8x8 chessboard problem under four different knowledge source conditions, each of which once again had 12 different possible recruits. Condition *a*, with 12 sigmoid units, was again a control condition without prior knowledge of the problem. Six of the recruitment candidates in condition *b* were networks trained on 2x2 chessboard problems, while the other six candidates were sigmoid units. In condition *c*, six of the candidates were networks trained on 4x4 chessboard problems, while the other six candidates were once again sigmoid units. In condition *d*, there were 4 sigmoid candidates, 4 networks trained on 2x2 chessboard problems, and 4 networks trained on 4x4 chessboard problems. Figure 3 shows training patterns and some generalization results for 4x4 (2a) and 8x8 (2b and 2c) chessboards.

Results are shown in Table 2 in terms of mean recruits and training epochs needed to master the 8x8 target problem. In each condition, sigmoid units were heavily recruited. Both 2x2 and 4x4 chessboard networks were recruited, and 4x4 networks were preferred when available. A factorial ANOVA of the total number of recruits was significant, $F(3, 116) = 41$, $p < .001$. Multiple comparison of means revealed that conditions *c* and *d*, with 4x4 source candidates, needed fewer recruits than did conditions *a* and *b*, $p < .05$. Condition *b*, with 2x2 source candidates, needed fewer recruits than control condition *a* with only sigmoid hidden units, $p < .05$.

An analogous ANOVA of epochs to learn was also significant, $F(3, 116) = 51$, $p < .001$. Multiple comparison of means revealed that conditions *c* and *d*, with 4x4 candidates, learned faster than did conditions *a* and *b*, $p < .05$. Condition *b*, with 2x2 source candidates, learned faster than control condition *a*, $p < .05$.

Output activation plots for three representative networks are shown in Figure 3. The network in Figure 3a learning the 4x4 chessboard could recruit only sigmoid hidden units, the network in Figure 3b learning the 8x8 chessboard could recruit either sigmoid units or 4x4 chessboard source networks, and the network in Figure 3c could recruit sigmoid units or chessboard sources of either 2x2 or 4x4 size.

The striped patterns in Figures 3a and 3b indicate that these two networks learned alternating hyperplanes demarcating positive or negative output regions. These hyperplanes can have positive (*b*) or negative (*a*) slopes. Mesh-like representations can arise when recruited hyperplanes are oriented differently than in the target network (*c*). Such orientation differences can even occur when the source and target hyperplanes are initially oriented in the same direction, if the source hyperplanes happen to get rotated during input-phase training while inputs to the source are adjusted. In general, mesh-like solutions were more common with more complex recruited sources.

The 4x4 hyperplanes are particularly useful to recruit in learning about an 8x8 chessboard, but they need to be made more numerous and refined to deal with the 8x8 case.

Table 1. Mean recruits and epochs to learn a parity-8 problem in four different source conditions.

Sources to recruit	Mean recruits				Epochs
	Sigmoid	Parity-2	Parity-4	Total	
a. 12 sigmoids	6.5	n/a	n/a	6.5	971
b. 6 sigmoids, 6 parity-2 networks	5.7	0.9	n/a	6.6	992
c. 6 sigmoids, 6 parity-4 networks	2.6	n/a	2.1	4.7	702
d. 4 sigmoids, 4 parity-2, 4 parity-4	2.6	0.3	2.3	5.2	737

Table 2. Mean recruits and epochs to learn an 8x8 chessboard in four different source conditions.

Sources to recruit	Mean recruits				Epochs
	Sigmoid	2x2	4x4	Total	
a. 12 sigmoids	10.9	n/a	n/a	10.9	1523
b. 6 sigmoids, 6 2x2 networks	6.0	2.9	n/a	8.9	1222
c. 6 sigmoids, 6 4x4 networks	1.4	n/a	5.5	6.8	927
d. 4 sigmoids, 4 2x2, 4 4x4	2.1	0.6	4.3	7.0	942

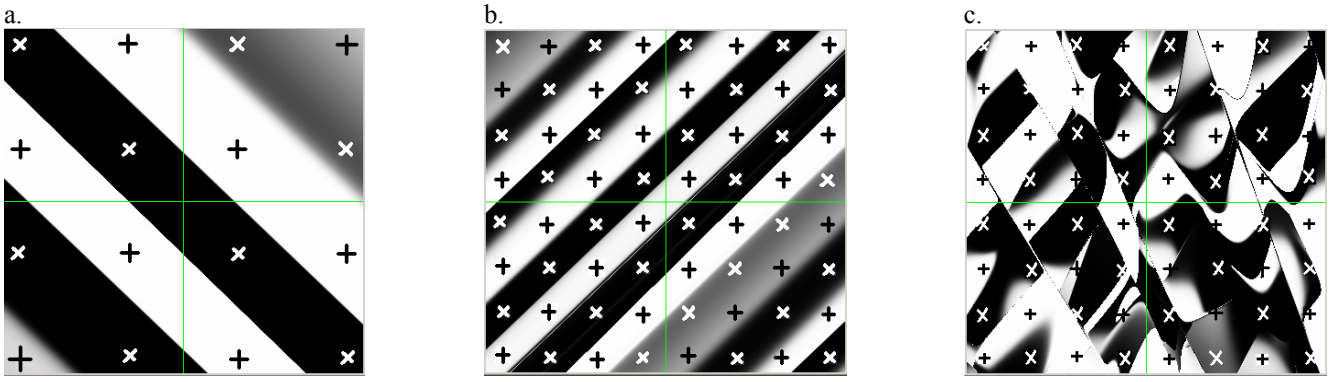

Figure 3. Output activation plots for three different networks learning a 4x4 chessboard (*a*) or an 8x8 chessboard (*b* and *c*). The training patterns are marked with + for positive output and *x* for negative output. A white background represents positive generalizations, black background represents negative generalizations, and gray background represents intermediate values.

Discussion

Both experiments show that KBCC recruits simpler previous knowledge to compose a solution to a similar, but more complex problem. Knowing about parity-4 problems helped networks to learn more complex parity-8 problems.

It makes sense that parity-2 sources were less effective and less likely to be recruited than parity-4 sources because only a single hidden unit is recruited for a parity-2 problem. A parity-2 source network is thus no more powerful than a single sigmoid unit. The solutions to parity-8 learning were a composition of parity-4 source networks and sigmoid units.

Similarly, knowing about smaller chessboard problems helped networks to learn more complex chessboards. As with parity problems, the solutions to 8x8 chessboards were a composition of simpler source chessboard networks and sigmoid units.

In all of these source-network recruitments, the simpler source networks retain their original representational and computational abilities. This is because KBCC does not alter the internal structure of either recruits or their original source networks. The connection weights entering and leaving a recruit inside of a target network are altered to tweak the source knowledge and integrate it into the newly learned solution, respectively. But importantly, connection weights internal to the recruited source network are not changed.

Our evidence shows that compositional learning can be implemented in neural terms and that it can simplify and accelerate learning. The essential characteristics of concatenative compositionality are preserved in these cases because the components are not altered as the composition is formed.

This is not the on-the-spot compositionality involved in, say, creating a novel sentence, but rather a knowledge-based-learning sort of compositionality in which both the learning and the representations are computationally specified. Our results suggest that connectionist accounts of compositionality may be worth pursuing despite the criticisms that such attempts have garnered.

It will be interesting to apply these techniques to a wider range of compositional problems and to simulations of human participants in compositional learning experiments.

Acknowledgment

This research was supported by a grant to T. R. S. from the Natural Sciences and Engineering Research Council of Canada.

References

Bregman, A. S. 1977. Perception and behavior as compositions of ideals. *Cognitive Psychology, 9*, 250-292.

Fahlman, S. E., and Lebiere, C. 1990. The cascade-correlation learning architecture. In D. S. Touretzky ed., *Advances in Neural Information Processing Systems 2*, pp. 524-532. Los Altos, CA: Morgan Kaufmann.

Fodor, J. A., and Pylyshyn, Z. W. 1988. Connectionism and cognitive architecture: A critical analysis. *Cognition, 28*, 3-71.

Pinker, S. 1997. *How the Mind Works*. New York: Norton.

Pollack, J. 1990. Recursive distributed representations. *Artificial Intelligence, 46*, 77-105.

Rivest, F., and Shultz, T. R. 2002. Application of knowledge-based cascade-correlation to vowel recognition. *IEEE International Joint Conference on Neural Networks 2002* (pp. 53-58).

Shultz, T. R. 2003. *Computational Developmental Psychology*. Cambridge, MA: MIT Press.

Shultz, T. R., and Rivest, F. 2001. Knowledge-based cascade-correlation: Using knowledge to speed learning. *Connection Science, 13*, 1-30.

Thivierge, J.-P., and Shultz, T. R. 2002. Finding relevant knowledge: KBCC applied to splice-junction determination. *IEEE International Joint Conference on Neural Networks 2002* (pp. 1401-1405).

van Gelder, T. 1990. Compositionality: A connectionist variation on a classical theme. *Cognitive Science, 14*, 355-364.