Knowledge-based Learning with KBCC

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

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

László Egri
School of Computer Science
McGill University
3480 University Street, Montreal, QC H3A 2B4, Canada
laszlo.egri@mail.mcgill.ca

Jean-Philippe Thivierge
Department of Psychology
McGill University
1205 Penfield Avenue, Montreal, QC H3A 1B1, Canada
jpthiv@ego.psych.mcgill.ca

***Abstract* – A constructive learning algorithm, knowledge-based cascade-correlation (KBCC), recruits previously-learned networks in addition to the single hidden units recruited by ordinary cascade-correlation. This enables learning by analogy when adequate prior knowledge is available, learning by induction from examples when there is no relevant prior knowledge, and various combinations of analogy and induction. A review of experiments with KBCC indicates that recruitment of relevant existing knowledge typically speeds learning and sometimes enables learning of otherwise impossible problems. Current limitations of this approach are discussed.**

Index Terms – Knowledge-based learning, neural networks, transfer.

I. Introduction

Neural networks are among the most successful approaches to modeling learning and development [1, 2]. However, a significant limitation of such neural learning is that it is typically conducted *from scratch*, without allowing for the influence of existing knowledge. For simulation purposes, this is unfortunate because people make extensive use of their existing knowledge when learning [3-9]. Use of knowledge is largely responsible for the ease and speed with which people are able to learn, as well as for interference of learning by existing knowledge. Typical neural networks fail to use knowledge while learning because they start learning from random connection weights.

Here we examine a somewhat unusual neural-learning algorithm that does use its knowledge to learn new problems. This algorithm is an extension of cascade-correlation (CC), a constructive learning algorithm that has proved useful in simulating more than a dozen phenomena in learning and cognitive development [2]. Ordinary CC creates a network topology by recruiting new hidden units into a feed-forward network as needed in order to reduce error [10]. The algorithm extension, called knowledge-based cascade-correlation (KBCC), recruits whole previously-learned networks in addition to the single hidden units recruited by CC [11].

KBCC is similar in spirit to recent neural-network research on inductive transfer [12, 13], multitask learning [14], sequential learning [15], lifelong learning [16], input re-coding [17], knowledge insertion [18], and modularity [19]. KBCC incorporates and integrates many of these ideas by learning, storing, searching for, mapping, and recruiting knowledge within a constructive-learning approach. It often outperforms other knowledge-based learners, and it can potentially model a range of psychological phenomena.

We describe the KBCC algorithm, review its performance on toy and realistic problems, and then discuss its advantages and limitations in the context of current literature on knowledge and learning in neural networks.

II. KBCC

As noted, 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 [10]. The principal innovation in KBCC is the potential to recruit previously-learned networks or indeed any differentiable function, in competition with single hidden units. The computational device that gets recruited is the one whose output correlates best with existing network error, just as in ordinary CC.

At the start of learning, a KBCC network (see Fig. 1 for an example) has only a bias unit and input units fully connected to output units with small randomized weights. During the initial learning phase, known as the *output phase*, connection weights feeding the output units are trained to minimize the sum of squared error, which is defined as

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

where $V_{o,p}$ is the activation of output o in response to training pattern p, and $T_{o,p}$ is the corresponding target activation value that the network is learning to produce.

If error reduction stagnates or a certain number of epochs is reached without learning success, then the algorithm shifts to the so-called *input phase*. In input-phase, a pool of candidate units and 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 recruitment candidate. Those weights are then trained to maximize a covariance between each candidate's outputs c_o and residual error in the target network. This covariance for each candidate is defined as:

$$G_c = \frac{\sum_{c_o}\sum_{o} Cov(V_c, E)^2_{c_o,o}}{\sum_{o}\sum_{p} E^2_{o,p}} \quad (2)$$

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 $Cov(V_c,E)$ the covariance matrix relating output activations and errors.

Input-phase training continues until a maximum number of epochs is reached or until the increases in correlations stagnate. When the input phase is finished, the candidate with the highest G value is 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 some maximum number of epochs expires. A hypothetical KBCC target network with three recruits is illustrated in Fig. 1.

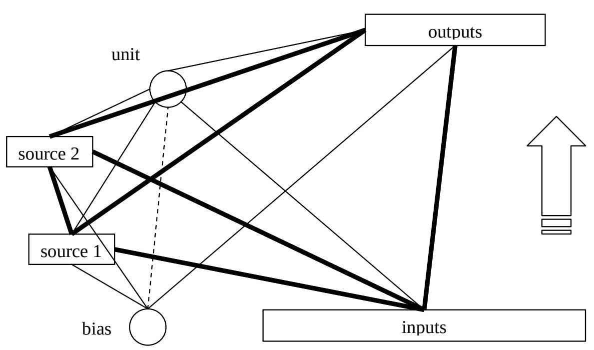

Fig. 1 Hypothetical KBCC network that has recruited two source networks and a sigmoid unit. The dashed line represents a single connection weight, thin solid lines represent weight vectors, and thick solid lines represent weight matrices. The arrow indicates direction of activation flow.

II. APPLICATIONS OF KBCC

KBCC has been applied to both toy and realistic problems.

A. *Toy Problems*

Although toy problems may not seem as challenging as realistic problems and simulations, they can play an important role in understanding the behavior and ability of complex algorithms such as KBCC.

1) Geometric shapes: The first batch of toy problems we explored involved learning about two-dimensional geometric shapes under various transformations such as translation, rotation, and size changes, as well as compositions of complex shapes from simpler shapes [11]. Networks had to learn to distinguish points within a target shape from points outside the shape. Plots of activation outputs enabled evaluation of a network's knowledge representations.

An illustration of KBCC compositionality is presented in Fig. 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 [11]. Notice that the recruited components of the cross in Figure 2 are very similar to their original sources and that the original sources remain unaltered in the composition. These are characteristics of classical, concatenative compositionality, in which the components are incorporated unaltered. This is supposed to be impossible for neural networks [20, 21].

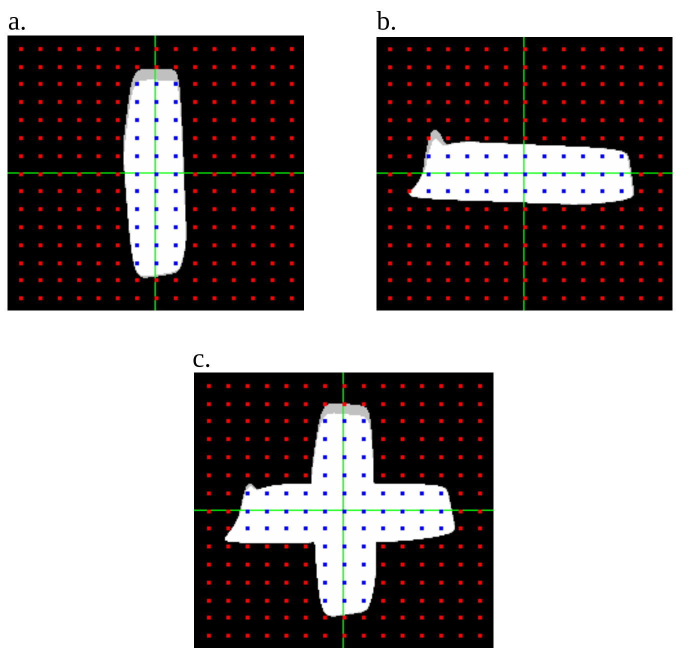

Fig. 2. Output activation diagrams for a KBCC network that learned a cross target (*c*) by recruiting its two components (*a* and *b*). Dark dots represent training points inside a shape; light dots training points outside a shape. White background indicates generalization to test points inside a shape,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

Learning time without relevant knowledge was 4 to 16 times longer than with relevant knowledge on these kinds of problems, depending on particular conditions. There was a strong tendency to recruit relevant knowledge whenever it was available.

Direct comparison revealed that KBCC networks were faster to learn translation problems than were Multitask

Learning (MTL) networks [22]. An MTL network is trained in parallel on several tasks from the same domain, with a single output for each task [14]. MTL networks typically learn a common hidden-unit representation, which can be useful for learning subsequent tasks from the same domain.

2) Parity: Other toy, but difficult problems involved learning high-level parity problems with knowledge of smaller parity problems. Parity problems require a network to turn on an output unit only when an odd number of binary inputs are on. Generalization in such problems has been notoriously difficult to demonstrate. KBCC networks learned parity-8 problems (with 8 binary inputs) faster and with fewer recruits than did CC networks when parity-4 networks were included in the KBCC candidate source pool [23]. Such parity-4 networks tended to be recruited by KBCC target networks whenever available.

3) Chessboard Shapes: Similarly, KBCC networks learned complex chessboard shapes from knowledge of simpler chessboards [23]. As with parity, networks here used simpler previous knowledge to compose a solution to a similar, but more complex problem and learning was speeded up accordingly. Fig. 3 shows output activations of a KBCC network having learned an 8x8 chessboard shape after recruiting a 4x4 chessboard source network. The striped pattern indicates that this network learned alternating hyperplanes demarcating positive vs. negative output regions. This is an efficient solution that generalizes well even beyond the range of the training patterns.

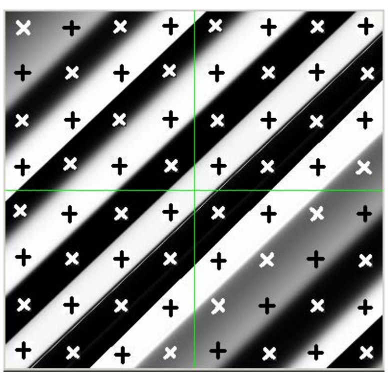

Fig. 3 Output activation plot for a KBCC network that learned an 8x8 chessboard shape. The training patterns are marked with + for positive output and *x* for negative output. White background represents positive generalizations, black background represents negative generalizations, and gray background represents intermediate values.

B. *Realistic Problems*

KBCC has also been applied to several realistic problems.

1) Vowel Recognition: KBCC networks with knowledge of vowels from one sort of speaker (e.g., adult males) learned to recognize vowels spoken by other sets of speakers (e.g., children and adult females) faster than networks without such knowledge [24].

2) Splice Junctions: KBCC was also applied to gene splice-junction determination using biological rules (previously learned as networks) as source knowledge. Because not all sources were recruited, KBCC was useful in identifying the knowledge that was required. On this task, KBCC required fewer recruits than did knowledge-free CC networks, but did not outperform CC on learning speed and accuracy, probably due to the limited utility of some of the rules [25].

Rather than having to be learned, symbolic rules can be injected into a source network as in the KBANN (Knowledge-Based Artificial Neural Networks) algorithm [18]. However, unlike KBANN, which combines inputs with only ANDs and ORs, a variant of KBCC, called rule-based cascade-correlation (RBCC), uses an *n-of-m* scheme in which *n* of *m* inputs can turn on the output unit [26]. This is more general and more flexible than using only AND (essentially *m-of-m*) and OR (*1-of-m*).

Moreover, unlike KBANN networks that involve injecting rules by hand into a target network, RBCC itself decides which rule-based sources to recruit, based on the standard CC criterion of selecting the source whose output activations correlate best with target-network error. RBCC (and ordinary KBCC) networks learned the splice-junction problem more accurately and faster than did KBANN networks [26]. RBCC networks also generalized better than did KBANN networks on this problem. An advantage of RBCC is that rules can be represented more crisply than if inductively learned by a CC source network.

3) Primality: The most recent realistic problem to which we applied KBCC is prime-number detection. An integer greater than 1 is *prime* if it has exactly two divisors, 1 and itself. An integer greater than 1 having more than two divisors is termed *composite*. The integer 1 is by definition neither prime nor composite.

One might think that the primality of an integer *n* could be determined by checking whether *n* is divisible by any integers between 2 and $n - 1$. Indeed it can be, but such testing can be much more efficient. The only divisors needed are primes from 2 to the integer part of $\sqrt{n}$ [27]. Further increases in efficiency can be gained by starting with the smallest prime and increasing divisor size until finding a divisor that works. This is because the smaller the prime divisor, the more composites it can detect in a fixed range of integers.

KBCC target networks learning to classify integers as prime or composite came to perform in just this fashion when their pool of source knowledge contained networks that knew whether an input integer could be divided by each of a range of divisors [28]. There was, for example, in the candidate pool a divide-by-2 network, a divide-by-3 network, etc., up to a divisor of 20. KBCC target networks recruited only source networks involving prime divisors below the square root of the largest number they were trained on (360). That is to say, they avoided recruiting single hidden units or source networks with composite divisors, any divisors greater than square root of 360 even if prime, and divisor networks with randomized connection weights. Moreover, they recruited their divide-by sources in order from small to large, installing all recruits on a single layer.

Installing more than one recruit on the same layer is enabled by a variation of CC called sibling-descendant cascade-correlation (SDCC) [29]. One-half of the candidate recruits (known as siblings) have no inputs from the previous layer of hidden units; the other half of the candidates (known as descendants) do have such inputs just as in classical CC. Sibling and descendant candidates compete with each other to be recruited based on their relative values of *G* as computed in (2). In this fashion, SDCC can build a variety of network topologies, depending on which recruit's activations correlate best with network error at the time of recruitment. Extension of the sibling-descendant idea to KBCC allows previously-learned networks to likewise be installed as either siblings or descendants.

To our initial surprise, KBCC target networks never recruited a divide-by-2 source network, but it turned out this was because they instead learned to use the last binary digit of *n* to easily determine if *n* was odd or even. As with humans, this is an effective shortcut to actually dividing by 2.

Developing in this fashion, KBCC target networks learned to classify their training integers in about one third of the epochs required by knowledge-free control networks, with fewer recruits on fewer network layers, and they generalized almost perfectly to novel test integers. In contrast, even after learning the training patterns to perfection, knowledge-free networks generalized less well than automatic guessing that the integer was composite, which was true 81% of the time in the integers employed.

A knowledge-representation analysis of the KBCC networks further revealed that they had composed an understanding of primality characterized by the Boolean expression: $\neg(n \text{ is divisible by } 2) \wedge \neg(n \text{ is divisible by } 3) \wedge \ldots \wedge \neg(n \text{ is divisible by the largest prime} <= \sqrt{n})$. In other words, these KBCC networks represented primality as a composition of divisor components, thus contradicting the popular view that compositionality is beyond the ability of artificial neural networks [20, 21]. Because the internals of the recruited source networks were preserved, it was argued that this type of compositionality is fully *concatenative*, unlike the mere *functional* compositionality of networks that are unable to recruit existing knowledge [30, 31].

Although not really a simulation of the psychology of prime-number testing, these results do bear some interesting similarities to the continued educational use of the ancient *sieve of Eratosthenes* (circa 200 BC) in teaching about prime numbers. Essentially, both methods order divisors from small to large and use only prime divisors below $\sqrt{n}$.

III. NEUROLOGICAL CORRESPONDENCE

Although not a detailed model of brain circuits, KBCC is inspired by evidence about how the brain supposedly works and thus incorporates many neurological features. Like other artificial neural networks, KBCC contains generic neurons with sigmoid activation functions having an activation floor and ceiling and a sharp, but continuous threshold between them. Like ordinary CC, KBCC implements both direct and indirect connectivity between sites and grows by recruiting new computational devices, suggestive of synapto- and neuro-genesis [32]. The topology of a KBCC network changes, not only by growing, but also by pruning relatively unused connections [33]. KBCC also mimics two fundamental features of brain organization, functional specialization and integration [34]. Source networks in KBCC are typically specialized and can be integrated into solutions of new tasks through learning. Unlike a variety of hybrid systems for combining symbolic and neural methods [35-38], RBCC is implemented in a homogeneous neural fashion, as are brain networks. RBCC has been used to model the interactions between frontal and temporal cortices during resolution of lexical ambiguities [32].

Until recently, it has not been clear how brains might implement complex learning algorithms like KBCC. However, the learning rules used in the CC family of algorithms can be rewritten in a mathematical form that is a small extension of the Hebb rule, which is widely regarded as being biologically realistic [39].

IV. DISCUSSION

In closing, we address some of the advantages and limitations of knowledge-based learning as implemented in KBCC.

A. *Advantages of KBCC*

KBCC is a general learner with considerable power and flexibility. Like CC, KBCC can automatically construct a network to suit the particular problem being learned, and escape from Fodor's paradox about not being able to learn anything genuinely novel [40, 41]. KBCC seamlessly integrates inductive and analogical learning such that each can compensate for the other. It learns by analogy to what it already knows whenever it can, resorts to learning by induction from examples when it knows nothing relevant, and achieves required combinations of learning by analogy and induction [11].

Existing knowledge is automatically selected from various sources without the intervention of a human programmer and without regard for the number and order of network inputs and outputs [11]. KBCC selects, maps, and tweaks existing knowledge automatically to aid new learning, and is able to compose solutions without changing the recruited components [28], as in classical, concatenative compositionality. Building on existing knowledge in these ways allows learning to be fast and accurate [11] and is sometimes necessary for neural learning to generalize [28].

B. *Limitations of KBCC*

One of the biggest limitations of KBCC is that its search for source knowledge is computationally expensive, especially in a mature and experienced network that has learned a lot. However, the difficulty that humans have in finding analogous knowledge that they are known to possess suggests that this limitation may actually enhance data coverage in psychology simulations. Humans are known to sometimes require hints in their search for analogous knowledge [42, 43]. Perhaps KBCC could similarly benefit from hints, conceivably by biased weighting of G in (2) [32]. Such biasing of G would have to be implemented in some realistic and tractable fashion.

At this point, there are still too few psychology simulations using KBCC. This is partly due to the fact that most psychology experiments on knowledge-based learning use quite simple linearly-separable problems. KBCC, by starting in output phase, can learn such problems without having to recruit any knowledge. A possible solution might involve starting KBCC in input phase instead, where it would try to recruit relevant knowledge even for simple linear problems. Another solution would be to continue working with more complex, non-linear problems as here, and bring these into the laboratory for psychological study. Non-linear problems such as prime-number testing might be possible to use with ordinary human participants.

Another limitation is that, in its current form, KBCC never modifies the source networks that it recruits. Only the input weights to the source and the output weights from the source get modified with learning. Freezing of internal connection weights and installation of copies of source networks are computationally effective, but may be psychologically unrealistic. They seem to conflict with new psychological research on memory consolidation that indicates that memories are labile just after retrieval [44]. A natural way to modify weights inside of source networks would be to back-propagate error signals through all layers of source networks. It is an open question whether this would damage KBCC learning and performance and cover emerging results on memory reconsolidation.

Still another problem is that the technique currently used in pruning KBCC (and CC) networks is not psychologically realistic. In this technique, weights are pruned from a network until error on an additional, generalization test set begins to increase [33], a method known as *early stopping*. This is unrealistic because such test sets rarely occur in natural human learning. Perhaps simpler pruning methods such as removing very small weights would work nearly as well and be more psychologically realistic.

A final limitation of KBCC is that segmentation of learning tasks is done by the experimenter. It would be more natural for a learning algorithm to autonomously determine whether a learning task is new or old. This is a problem that is common in machine-learning algorithms and it has received relatively little attention. Perhaps the passage of time, changes in content or context, or changes in input or output coding could signal that a learning task is new and thus requires a fresh target network. An automatic novelty detector might also help with this problem [45].

Acknowledgment

This research was supported by a grant from the Natural Sciences and Engineering Research Council of Canada to Shultz and an FQRNT Graduate Fellowship to Thivierge. We are grateful to Yoshio Takane for some excellent comments.

References

[1] J. L. Elman, Bates, E. A., Johnson, M. H., Karmiloff-Smith, A., Parisi, D., & Plunkett, K., *Rethinking innateness: A connectionist perspective on development*. Cambridge, MA: MIT Press, 1996.

[2] T. R. Shultz, *Computational developmental psychology*. Cambridge, MA: MIT Press, 2003.

[3] A. Robins, "Transfer in cognition," *Connection Science*, vol. 8, pp. 185-203, 1996.

[4] G. Nakamura, "Knowledge-based classification of ill-defined categories," *Memory and Cognition*, vol. 13, pp. 377-384, 1985.

[5] E. Heit, "Models of the effects of prior knowledge on category learning," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 20, pp. 1264-1282, 1994.

[6] F. C. Keil, "Conceptual development and category structure," in *Concepts and Conceptual Development: Ecological and Intellectual Factors in Categorization*, U. Neisser, Ed. Cambridge: Cambridge University Press, 1987.

[7] G. L. Murphy, "A rational theory of concepts," *The Psychology of Learning and Motivation*, vol. 29, pp. 327-359, 1993.

[8] M. J. Pazzani, "Influence of prior knowledge on concept acquisition: Experimental and computational results," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 17, pp. 416-432, 1991.

[9] E. J. Wisniewski, "Prior knowledge and functionally relevant features in concept learning," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 21, pp. 449-468, 1995.

[10] S. E. Fahlman, & Lebiere, C., "The cascade-correlation learning architecture," in *Advances in neural information processing systems 2*, D. S. Touretzky, Ed. Los Altos, CA: Morgan Kaufmann, 1990, pp. 524-532.

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

[12] L. Y. Pratt, "Discriminality-based transfer between neural networks," in *Advances in neural information processing systems 5*. San Mateo, CA: Morgan Kaufmann, 1993, pp. 204-211.

[13] B. Rehder, & Murphy, G. L., "A knowledge-resonance (KRES) model of category learning," in *Proceedings of the Twenty-third Annual Conference of the Cognitive Science Society*. Mahwah, NJ: Erlbaum, 2001, pp. 821-826.

[14] R. Caruana, "Multitask learning," *Machine Learning*, vol. 28, pp. 41-75, 1997.

[15] D. Silver, & Mercer, R., "The parallel transfer of task knowledge using dynamic learning rates based on a measure of relatedness," *Connection Science*, vol. 8, pp. 277-294, 1996.

[16] S. Thrun, & Mitchell, T., "Integrating inductive neural network learning and explanation-based learning," in *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, R. Bajcsy, Ed. San Mateo, CA: Morgan Kaufmann, 1993.

[17] A. Clark, & Thornton, C., "Trading spaces: Computation, representation, and the limits of uninformed learning," *Behavioral and Brain Sciences*, vol. 20, pp. 57-97, 1997.

[18] J. W. Shavlik, "A framework for combining symbolic and neural learning," *Machine Learning*, vol. 14, pp. 321-331, 1994.

[19] M. I. Jordan, & Jacobs, R. A., "Hierarchical mixtures of experts and the EM algorithm," *Neural Computation*, vol. 6, pp. 181-214, 1994.

[20] J. A. Fodor, & Pylyshyn, Z. W., "Connectionism and cognitive architecture: A critical analysis," *Cognition*, vol. 28, pp. 3-71, 1988.

[21] S. Pinker, *How the mind works*. New York: Norton, 1997.

[22] T. R. Shultz, & Rivest, F., "Using knowledge to speed learning: A comparison of knowledge-based cascade-correlation and multi-task learning," in *Proceedings of the Seventeenth International*

Conference on Machine Learning. San Francisco: Morgan Kaufmann, 2000, pp. 871-878.

[23] F. Rivest, & Shultz, T. R., "Compositionality in a knowledge-based constructive learner," in *Papers from the 2004 AAAI Fall Symposium*, *Technical Report FS-04-03*. Menlo Park, CA: AAAI Press, 2004, pp. 54-58.

[24] F. Rivest, & Shultz, T. R., "Application of knowledge-based cascade-correlation to vowel recognition," *IEEE International Joint Conference on Neural Networks*, pp. 53-58, 2002.

[25] J. P. Thivierge, & Shultz, T. R., "Finding relevant knowledge: KBCC applied to splice-junction determination," *IEEE International Joint Conference on Neural Networks*, pp. 1401-1405, 2002.

[26] J. P. Thivierge, F. Dandurand, and T. R. Shultz, "Transferring domain rules in a constructive network: Introducing RBCC," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, 2004, pp. 1403-1409.

[27] T. Nagell, *Introduction to number theory*. New York: Wiley, 1951.

[28] L. Egri, & Shultz, T. R., "A compositional neural-network solution to prime-number testing," in *Proceedings of the Twenty-eighth Annual Conference of the Cognitive Science Society*: Mahwah, NJ: Erlbaum, 2006, in press.

[29] S. Baluja and S. E. Fahlman, "Reducing network depth in the cascade-correlation learning architecture.," School of Computer Science, Carnegie Mellon University Technical Report CMU-CS-94-209, 1994.

[30] J. Pollack, "Recursive distributed representations," *Artificial Intelligence*, vol. 46, pp. 77-105, 1990.

[31] T. van Gelder, "Compositionality: A connectionist variation on a classical theme," *Cognitive Science*, vol. 14, pp. 355-364, 1990.

[32] J. P. Thivierge, Titone, D., & Shultz, T. R., "Simulating frontotemporal pathways involved in lexical ambiguity resolution," in *Proceedings of the Twenty-seventh Annual Conference of the Cognitive Science Society*. Mahwah, NJ: Erlbaum, 2005, pp. 2178-2183.

[33] J. P. Thivierge, Rivest, F., & Shultz, T. R., "A dual-phase technique for pruning constructive networks," in *IEEE International Joint Conference on Neural Networks*, 2003, pp. 559-564.

[34] K. Friston, "Learning and inference in the brain," *Neural Networks*, vol. 16, pp. 1325-1352, 2003.

[35] F. G. Ashby, Alfonso-Reese, L. A., Turken, A. U., & Waldron, E. M., "A neuropsychological theory of multiple systems in category learning," *Psychological Review*, vol. 105, pp. 442-481, 1998.

[36] M. A. Erickson, & Kruschke, J. K., "Rules and exemplars in category learning," *Journal of Experimental Psychology: General*, vol. 127, pp. 107-140, 1998.

[37] T. J. Palmeri, & Nosofsky, R. M., "Generalizations by rule models and exemplar models of category learning," in *Proceedings of the Fifteenth Annual Meeting of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum, 1993, pp. 794-799.

[38] A. Vandierendonck, "A parallel rule activation and rule synthesis model for generalization in category learning," *Psychonomic Bulletin and Review*, vol. 2, pp. 442-459, 1995.

[39] F. Rivest, & Shultz, T. R., "Learning with both adequate computational power and biological realism.," in *Proceedings of the 2005 Canadian Artificial Intelligence Conference: Workshop on Correlation Learning*. University of Victoria, Victoria, BC, 2005, pp. 15-23.

[40] S. R. Quartz, "Neural networks, nativism and the plausibility of constructivism," *Cognition*, vol. 48, pp. 223-42, 1993.

[41] D. Mareschal, & Shultz, T. R., "Generative connectionist networks and constructivist cognitive development," *Cognitive Development*, vol. *11*, pp. 571-603, 1996.

[42] M. L. Gick, & Holyoak, K. J., "Analogical problem solving," *Cognitive Psychology*, vol. 12, pp. 306-355, 1980.

[43] M. L. Gick, & Holyoak, K. J., "Schema induction and analogical transfer," *Cognitive Psychology*, vol. 15, pp. 1-38, 1983.

[44] K. Nader, Schafe, G. E., & LeDoux, J. E., "Fear memories require protein synthesis in the amygdala for reconsolidation after retrieval," *Nature Medicine*, vol. 406, pp. 722-726, 2000.

[45] T. Kohonen, *Self-organization and associative memory*, 2nd ed. New York: Springer-Verlag, 1988.