Dual Role of Analogy in the Design of a Cognitive Computer

Pentti Kanerva
RWCP[1] Theoretical Foundation SICS[2] Laboratory
SICS, Box 1263, SE-164 29 Kista, Sweden
E-mail: kanerva@sics.se

Abstract

This paper is about the computer analogy of the brain and how it can both help and hinder our understanding of the human mind. It is based on the assumptions that the mind can be understood in terms of the working of the brain, and that the brain's function is to process information: that it is some kind of a computer, as contrasted for example with the heart which is a pump. It is a computer whose design we do not understand but try to, by analogy; that is, by making a model—a "cognitive computer"—based on our understanding of computers, brains, and the working of the mind.

Human intelligence and language are fundamentally analogical and figurative whereas lower forms of intelligence and conventional computers treat meaning literally. Therefore, the challenge in designing a cognitive computer is to find the kinds of information representation and operations that make figurative meaning come out naturally. The paper discusses holistic representation, which is unconventional and looks promising and worthy of investigation—it easily encodes recursive (list) structure, for example—and points out a danger in taking too literally cognitive models that have been developed on conventional computers, such as the following of rules.

Introduction

The human mind is unlike any computer or program we know. It is not literal, and when meaning is taken literally, the result can be funny or total nonsense. That's the humor of puns. This must mean that the human mind, although capable of being literal, is fundamentally figurative or symbolical or analogical. How else could we judge a literal interpretation as being at once both accurate and wrong?

The growth of the human mind—our grasp of things—is largely due to analogical perceiving and thinking. Some things are meaningful to us at birth or without learning; they are mostly things necessary for survival. The rest we learn through experience. Some learning is associative, as when we learn cause and effect. This kind of learning is basic to all animals.

To follow an example, or imitate, is a more advanced form of learning and is common at least in mammals and birds. It involves a basic form of analogy. The learner identifies with a role model—perceives one as the other, makes an analogical connection or mapping between oneself and the other.

Full-fledged analogy is central to human intelligence. We relate the unfamiliar to the familiar, and we see the new in terms of the old. This is most evident in language, which is thoroughly metaphorical. New and unfamiliar things are expressed and explained in familiar terms that are understood not literally but figuratively. It is possible that full-fledged analogy and human language need each other and that our faculties for them have coevolved.

Analogy is such an integral part of us that we hardly notice it nor pay it its proper dues. That is, until we try to program a computer to act like a human. AI has puzzled over the programming of humanlike behavior for three decades. At first it was thought that programming computers to understand language, to translate, and the like, were just around the corner, waiting only for computers to get large and fast enough. Now they are large and fast,

. 1. Real World Computing Partnership. Support of this research by Japan's Ministry of International Trade and Industry through the RWCP is gratefully acknowledged.
2. Swedish Institute of Computer Science.

many things have been tried and much has been learned, but the puzzle remains and we have no clear idea of how to solve it.

This paper is a personal view of the lessons this holds for us. The theme is that we must rethink computing, put figurative meaning and analogy at its center, and find computing mechanisms that make it come out naturally. This can be construed as designing a new kind of computer, a "cognitive computer," that is a better model of the brain than present-day computers are. I will also try to verbalize things that students of connectionist architectures take for granted but that might puzzle others, the main idea being that implementation matters when we try to understand how the mind works.

The Computer as a Brain and the Brain as a Computer

Equating computers with brains is an example of analogical thinking. Early computers were dubbed electronic brains, computers have memory, and we even say that a program knows, wants, or believes so and so. Such anthropomorphizing seems natural to us and it serves a purpose. It brings a technological mystery within the realm of the familiar, since we already have an idea of what the brain does even if we don't know just how it does it.

We also talk of the brain as a computer. Its appeal is in that whereas the mechanisms of the brain are hidden, those of the computer are available to us, and through them we could possibly understand the brain's mechanisms. The principle is sound and is the thesis behind Turing's imitation game: If we can build a machine that behaves in the same way as a natural system does, we have understood the natural system.

Analogies not only help our thinking but they also channel and limit it. The computer analogy of the brain or of the mind has certainly done so, as modeling in cognitive science and AI has been dominated by programs written for the computer, while philosophical and qualitative treatment of issues is looked upon with suspicion.

Many things are modeled successfully on computers, such as weather, traffic flow, strength of materials and structures, industrial processes, and so forth. However, there are special pitfalls when the thing being modeled—the brain—is itself some kind of a computer: the danger is that our models begin to look like the computers they run on or the programming languages they are written in. For example, we talk of human short-term or working memory and think of the computer's active registers, or we talk of human long-term memory and think of the computer's permanent storage (RAM or disk), or we talk of the grammar of a language and think of a tree-structure or a set of rewriting rules programmed in Lisp. Of course these are analogical counterparts, but there is a danger of taking them too literally. Human memory works very differently from computer memory, and the brain is not a Lisp machine nor the mind a logic program. Some analogical comparisons have not been at all useful in understanding the working of the mind; for example, equating the brain with the computer's hardware and the mind with its software. Finally, there is a worse danger of failing to notice what is missing in our models of the mind because it is missing or invisible in computers. To safeguard against it, we must treat the subject qualitatively: Our models may behave as advertised, but is that how people behave; for example, how they use language?

Artificial Neural Nets as Biologically Motivated Models of Computing

The computer's and brain's architectures are very different. Perhaps the differences account for the difficulty in programming computers to be more lifelike and less literal-minded. This has motivated the study of alternative computing architectures called (artificial) neural nets (NN), or parallel distributed processing (PDP), or connectionist architectures. The hope is that an architecture more similar to the brain's should produce behavior more similar to the brain's, which is a valid analogical argument. Unfortunately it does not tell us what in the architecture matters and what is incidental, and unfortunately our neural nets are not significantly more figurative than traditional computers.

Neural-net research has made a valuable contribution by focusing our attention on representation. Computer theoreticians and engineers know, for example, that the representation of numbers has a major effect on circuit design. A representation that works well for addition works reasonably well also for multiplication, whereas a representation that allows very fast multiplication is useless for addition. Thus a representation is a compromise that favors some operations and hinders others.

Information in computers is stored locally, that is, in records with fields. Local representation—one unit per concept—is common also in neural nets. The alternative is to distribute information from many sources over shared units. It is more brainlike, at least superficially, and it has been studied and used with neural nets for a long time. I take distributed representation to be fundamental to the brain's operation and believe that a cognitive computer should be based on it, and that therefore we should find out all we can about the encoding of information into, and operating with, distributed representations.

Neural-net research has shown that these representations are robust and support some forms of generalization: representations (patterns) that are similar on the surface—close according to some metric—are treated similarly, for example as belonging in the same or similar classes. The representations are also suitable for learning from examples. The learning takes place by statistical averaging or clustering of representations (self-organizing). It is not very creative but it can be subtle and lifelike, which makes it cognitively interesting. It can produce behavior that looks like rule-following although the system has no explicit rules, as was demonstrated with the learning of the past tense of English verbs by Rumelhart and McClelland (1986). This is a significant discovery, in that it demonstrates a principle that probably governs the working of the brain in general and should govern the working of a cognitive computer. What we see and describe as rule-following is an emergent phenomenon that reflects an underlying mechanism. However, the rules do not produce the behavior even if they may accurately describe it.

Description vs. Explanation

The distinction between description and explanation of behavior is so central that I will highlight it with an example. Consider heredity. Long before the genetic bases of heredity were known, people knew about dominant and recessive traits and had figured out the basic laws of inheritance. For example, a plant species may come in three varieties, with white, pink, or red flowers, and cross-pollinating the white with the red always produces plants with pink flowers. The specific rule is that all of the first generation is pink, and when pink-flowered plans are crossed with each other, one-fourth of the offspring is white, one-fourth red, and half pink. So we can say that the inheritance mechanism works by this rule. However, no mechanism in the reproductive system keeps counting the numbers of offspring to make sure that the proportions come out right: "I have made so and so many white flowers, it's time to make the same number of red flowers." It is not the rule that makes the proportions come out in a certain way. The proportions are an outward reflection of the mechanism that passes traits from one generation to the next. It is significant, however, that long before chromosomes or genes, or RNA and DNA were discovered, people speculated correctly about a hereditary mechanism that would produce offspring in those proportions. Clearly, the laws provided a useful description of the behavior, and accurate description often leads to discovery and explanation.

The situation is similar with regard to language and to mental functions at large. For example, we attribute the patterns of a language to its grammar and we devise sets of rules by which the grammar works. However, it is not the grammar that generates sentences in us when we speak or write. The regularities captured in the grammar are an outward expression of our underlying mechanisms for language—the grammar is an emergent phenomenon. This distinction is easily lost when we produce language output with computers, for there we actually use the grammar to generate sentences, and we work hard to develop a comprehensive grammar for a language. And when we think of the computer as a model of the brain and use computers to model mental functions, we tacitly assume that the brain uses grammar rules to generate language. Formal logic as a model of thinking can be criticized on similar grounds. It may describe rational thought but it does not explain thinking. Our understanding of the mechanisms of mind is not yet sufficient to allow us to explain thinking and language. The best we can do is to describe them, but as our descriptions improve, our chances for discovering the mechanisms improve.

The Brain as a Computer for Modeling the World, and Our Model of the Brain's Computing

It is useful to think of the brain as a computer if we make the analogy between the two sufficiently abstract. So what in computers should we look at? The organization of computation as a sequence of programmed instructions for manipulating pieces of data stored in memory seems like an overly restricted a model of how the brain or the mind works. A more useful analogy is made at the level of computers as state machines, the states being realized as configurations of matter, or patterns in some physical medium. Mental states and subjective experience then correspond to—or are caused by—physical states so that when a physical state repeats, the corresponding subjective experience repeats. Thus the patterns that define the states are the objective counterpart of the subjective experience. Our senses are the primary source of the patterns, and our built-in faculties for pleasure and pain give primary meaning to some of the patterns. Brains are wired for rich feedback, and when the feedback works in such a way that an experience created by the senses—i.e., a succession of states—can later be created internally, we have the basis for learning. With learning, rich networks of meaningful states can be built.

The evolutionary function of this computer is to make the world predictable: the brain models the world as the world is presented to us by our senses. It appears to compute with patterns of activity over large sets of neurons. To study such computing mathematically, we can model the patterns by large patterns of bits, emphasizing the large size of the patterns, as that gives the models their power. The key question is, how do patterns that have already been established and have become meaningful, give rise to new patterns; how do existing concepts give rise to new concepts?

I have used the binary Spatter Code (Kanerva, 1996) to model computing with large patterns. The code is related to Plate's Holographic Reduced Representation (HRR; Plate, 1994) and allows simple demonstrations of it. The representation is distributed so that every item of information that is included in a composed pattern—every constituent pattern—contributes to every bit of the composed pattern: the patterns are holographic or holistic.

Computing with Large Patterns

The following description is in traditional symbolic terms and uses a two-place relation $r(x, y)$ and a triplet $t = (x, y, z)$ as examples.

Space of Representations

All HRRs, including the Spatter Code, work with large random patterns, or high-dimensional random vectors. All things—variables, values, composed structures, mappings between structures—are elements of a common space: they are very-high-dimensional random vectors with independent, identically distributed components. The dimensionality of the space, denoted by N, is usually between 1,000 and 10,000. The Spatter Code uses dense binary vectors (i. e., 0s and 1s are equally probable). The vectors are written in boldface, so that $\mathbf{x}$ stands for an N-vector representing the variable or role x, and $\mathbf{a}$ stands for an N-vector representing the value or filler a, for example.

Item Memory or Clean-up Memory

Some operations produce approximate vectors that need to be cleaned up (i.e., identified with their exact counterparts). That is done with an item memory that stores all valid vectors known to the system, and retrieves the best-matching vector when cued with a noisy vector, or retrieves nothing if the best match is no better than what results from random chance. The item memory performs a function that, at least in principle, is performed by an autoassociative neural memory.

Binding

Binding is the first level of composition in which things that are very closely associated with each other are brought together. A variable is bound to a value with a binding operator that combines the N-vectors for the variable and the value into a single N-vector for the bound pair. The Spatter Code binds with coordinatewise (bitwise) Boolean Exclusive-OR (XOR, $\otimes$), so that the variable x having the value a (i.e., $x = a$) is encoded by the N-vector $\mathbf{x}\otimes\mathbf{a}$ whose nth bit is the bitwise XOR $x_n \otimes a_n$ (x_n and a_n are the nth bits of $\mathbf{x}$ and $\mathbf{a}$, respectively). An important property of all HRRs is that binding of two random vectors produces a random vector that resembles *neither* of the two.

Unbinding

The inverse of the binding operator breaks a bound pair into its constituents: finds the filler if the role is given, or the role if the filler is given. The XOR is its own inverse function, so that, for example, $(\mathbf{x}\otimes\mathbf{a})\otimes\mathbf{a} = \mathbf{x}$ finds the vector to which $\mathbf{a}$ is bound in $\mathbf{x}\otimes\mathbf{a}$.

Merging

Merging is the second level of composition in which identifiers and bound pairs are combined into a single item. It has also been called 'superimposing' (superposition), 'bundling', and 'chunking'. It is done by a (*normalized*) *mean* vector, and the merging of $\mathbf{G}$ and $\mathbf{H}$ is written as $[\mathbf{G} + \mathbf{H}]$, where $[\ldots]$ stands for normalization. The relation $r(a, b)$ can be represented by merging the representations for r, '$x = a$', and '$y = b$'. It is encoded by

$$\mathbf{R} = [\mathbf{r} + \mathbf{x}\otimes\mathbf{a} + \mathbf{y}\otimes\mathbf{b}]$$

The normalized mean of binary vectors is given by bitwise majority rule, with ties broken at random. An important property of all HRRs is that merging of two or more random vectors produces a random vector that resembles *each* of the merged vectors.

Distributivity

In all HRRs, the binding and unbinding operators distributes over the merging operator, so that, for example,

$$[\mathbf{G} + \mathbf{H} + \mathbf{I}]\otimes\mathbf{a} = [\mathbf{G}\otimes\mathbf{a} + \mathbf{H}\otimes\mathbf{a} + \mathbf{I}\otimes\mathbf{a}]$$

Distributivity is a key to analyzing HRRs.

Probing

To find out whether the vector $\mathbf{a}$ appears bound in another vector $\mathbf{R}$, we probe $\mathbf{R}$ with $\mathbf{a}$ using the unbinding operator. For example, if $\mathbf{R}$ represents the above relation, probing it with $\mathbf{a}$ yields a vector $\mathbf{X}$ that is recognizable as $\mathbf{x}$ ($\mathbf{X}$ will retrieve $\mathbf{x}$ from the item memory). The analysis is as follows:

$$\mathbf{X} = \mathbf{R}\otimes\mathbf{a} = [\mathbf{r} + \mathbf{x}\otimes\mathbf{a} + \mathbf{y}\otimes\mathbf{b}]\otimes\mathbf{a}$$

which becomes

$$\mathbf{X} = [\mathbf{r}\otimes\mathbf{a} + (\mathbf{x}\otimes\mathbf{a})\otimes\mathbf{a} + (\mathbf{y}\otimes\mathbf{b})\otimes\mathbf{a}]$$

by distributivity and simplifies to

$$\mathbf{X} = [\mathbf{r}\otimes\mathbf{a} + \mathbf{x} + \mathbf{y}\otimes\mathbf{b}\otimes\mathbf{a}]$$

Thus $\mathbf{X}$ is similar to $\mathbf{x}$; it is also similar to $\mathbf{r}\otimes\mathbf{a}$ and $\mathbf{y}\otimes\mathbf{b}\otimes\mathbf{a}$, but they are not stored in the item memory and thus act as random noise.

The functions described so far are sufficient for traditional symbol processing, for example, for realizing a Lisp-like list-processing system. Holistic mapping, which is discussed next, is a parallel alternative to what is traditionally accomplished with sequential search and substitution.

Holistic Mapping and Simple Analogical Retrieval

Probing is the simplest form of holistic mapping. It approximately maps a composed pattern into one of its bound constituents, as discussed above and seen in the following example. Let $\mathbf{F}$ be a holistic pattern representing France: that its capital is Paris, geographic location is Western Europe, and monetary unit is franc. Denote the patterns for capital, Paris, geographic location, Western Europe, money, and franc by **ca**, **Pa**, **ge**, **WE**, **mo**, and **fr**. France is then represented by the pattern

$$\mathbf{F} = [\mathbf{ca}\otimes\mathbf{Pa} + \mathbf{ge}\otimes\mathbf{WE} + \mathbf{mo}\otimes\mathbf{fr}]$$

Probing $\mathbf{F}$ for "the Paris of France" is done by mapping (XORing) it with **Pa** and it yields

$$\mathbf{F}\otimes\mathbf{Pa} = [\mathbf{ca} + \mathbf{ge}\otimes\mathbf{WE}\otimes\mathbf{Pa} + \mathbf{mo}\otimes\mathbf{fr}\otimes\mathbf{Pa}]$$

(see 'Probing' above) and is approximately equal to **ca**:

$$\mathbf{F}\otimes\mathbf{Pa} \approx \mathbf{ca}$$

XORing with **Pa** has mapped $\mathbf{F}$ approximately into **ca**, meaning that Paris is France's capital.

Much more than that can be done in a single mapping operation, as shown in the following two examples. Let **S** be a holistic pattern for Sweden with capital Stockholm (**St**), located in Scandinavia (**Sc**), and with monetary unit krona (**kr**). This information about Sweden is then represented by the pattern

$$\mathbf{S} = [\mathbf{ca}\otimes\mathbf{St} + \mathbf{ge}\otimes\mathbf{Sc} + \mathbf{mo}\otimes\mathbf{kr}]$$

We can now ask 'What is the Paris of Sweden?' If we take the question literally and do the mapping $\mathbf{S}\otimes\mathbf{Pa}$, as above, we get nothing recognizable, so we must take Paris in a more general sense. 'Paris of France' gave us a recognizable result above (i.e., approximately **ca**), so we can use it: we can map **S** (XOR it) with $\mathbf{F}\otimes\mathbf{Pa}$ and we get

$$\mathbf{S}\otimes\mathbf{F}\otimes\mathbf{Pa} \approx \mathbf{St}$$

which is recognizable as the pattern for Stockholm. The derivation is based on distributivity and is similar to the one given under 'Probing'. The significant thing in $\mathbf{S}\otimes\mathbf{F}\otimes\mathbf{Pa}$ is that $\mathbf{S}\otimes\mathbf{F}$ can be thought of as a binding of two composed patterns of equal status, rather than a binding of a variable to a value, and also as a holistic mapping between France and Sweden, capable of answering analogy questions of the kind 'What is the Paris of Sweden?' and 'What is the krona of France?'

Holistic mapping allows *multiple substitutions* at once. What will happen to the pattern for France if we substitute Stockholm for Paris, Scandinavia for Western Europe, and krona for franc, all at once, and how is the substitution done? We create a mapping pattern as above, by binding the corresponding items to each other with XOR and by merging the results:

$$\mathbf{M} = [\mathbf{Pa}\otimes\mathbf{St} + \mathbf{WE}\otimes\mathbf{Sc} + \mathbf{fr}\otimes\mathbf{kr}]$$

Mapping the pattern for France with **M** then gives

$$\begin{aligned}
&\mathbf{F}\otimes\mathbf{M}\\
&\quad = [\mathbf{ca}\otimes\mathbf{Pa} + \mathbf{ge}\otimes\mathbf{WE} + \mathbf{mo}\otimes\mathbf{fr}]\\
&\qquad \otimes [\mathbf{Pa}\otimes\mathbf{St} + \mathbf{WE}\otimes\mathbf{Sc} + \mathbf{fr}\otimes\mathbf{kr}]\\
&\quad = [\ \mathbf{ca}\otimes\mathbf{Pa}\\
&\qquad\quad \otimes [\mathbf{Pa}\otimes\mathbf{St} + \mathbf{WE}\otimes\mathbf{Sc} + \mathbf{fr}\otimes\mathbf{kr}]\\
&\qquad + \mathbf{ge}\otimes\mathbf{WE}\\
&\qquad\quad \otimes [\mathbf{Pa}\otimes\mathbf{St} + \mathbf{WE}\otimes\mathbf{Sc} + \mathbf{fr}\otimes\mathbf{kr}]\\
&\qquad + \mathbf{mo}\otimes\mathbf{fr}\\
&\qquad\quad \otimes [\mathbf{Pa}\otimes\mathbf{St} + \mathbf{WE}\otimes\mathbf{Sc} + \mathbf{fr}\otimes\mathbf{kr}]\ ]
\end{aligned}$$

by distributivity, which becomes

$$\begin{aligned}
&[\ [\mathbf{ca}\otimes\mathbf{Pa}\otimes\mathbf{Pa}\otimes\mathbf{St} + \mathbf{ca}\otimes\mathbf{Pa}\otimes\mathbf{WE}\otimes\mathbf{Sc}\\
&\quad + \mathbf{ca}\otimes\mathbf{Pa}\otimes\mathbf{fr}\otimes\mathbf{kr}]\\
&+ [\mathbf{ge}\otimes\mathbf{WE}\otimes\mathbf{Pa}\otimes\mathbf{St} + \mathbf{ge}\otimes\mathbf{WE}\otimes\mathbf{WE}\otimes\mathbf{Sc}\\
&\quad + \mathbf{ge}\otimes\mathbf{WE}\otimes\mathbf{fr}\otimes\mathbf{kr}]\\
&+ [\mathbf{mo}\otimes\mathbf{fr}\otimes\mathbf{Pa}\otimes\mathbf{St} + \mathbf{mo}\otimes\mathbf{fr}\otimes\mathbf{WE}\otimes\mathbf{Sc}\\
&\quad + \mathbf{mo}\otimes\mathbf{fr}\otimes\mathbf{fr}\otimes\mathbf{kr}]\ ]
\end{aligned}$$

again by distributivity. That simplifies to

$$\begin{aligned}
&[\ [\mathbf{ca}\otimes\mathbf{St} + \mathbf{ca}\otimes\mathbf{Pa}\otimes\mathbf{WE}\otimes\mathbf{Sc}\\
&\quad + \mathbf{ca}\otimes\mathbf{Pa}\otimes\mathbf{fr}\otimes\mathbf{kr}]\\
&+ [\mathbf{ge}\otimes\mathbf{WE}\otimes\mathbf{Pa}\otimes\mathbf{St} + \mathbf{ge}\otimes\mathbf{Sc}\\
&\quad + \mathbf{ge}\otimes\mathbf{WE}\otimes\mathbf{fr}\otimes\mathbf{kr}]\\
&+ [\mathbf{mo}\otimes\mathbf{fr}\otimes\mathbf{Pa}\otimes\mathbf{St} + \mathbf{mo}\otimes\mathbf{fr}\otimes\mathbf{WE}\otimes\mathbf{Sc}\\
&\quad + \mathbf{mo}\otimes\mathbf{kr}]\ ]
\end{aligned}$$

and is recognizable as

$$[\mathbf{ca}\otimes\mathbf{St} + \mathbf{ge}\otimes\mathbf{Sc} + \mathbf{mo}\otimes\mathbf{kr}]$$

In other words,

$$\mathbf{F}\otimes\mathbf{M} \approx \mathbf{S}$$

so that a single mapping operation composed of multiple substitutions changes the pattern for France to an approximate pattern for Sweden, recognizable by the clean-up memory.

Toward a New Model of Computing

Holistic representation and holistic mapping hint at the possibility of organizing computing around analogy. However, the examples that I have shown are not very strong. This could mean that large random patterns and the suggested operations on them are not a good way to compute, but it is also possible that they are, but that we are not using them correctly. What stands out about the examples is that they are built around established notions of variable, value, property, relation, and the like. These are high-level abstractions that help us describe abstract things to each other, but they may be poor indicators of what goes on in the brain. For example, should a pattern for a variable, such as capital city in the above examples, be related to patterns that stand for individual cities, and how should those be related to the patterns for the countries they are capitals of? There are many questions to answer before we can decide about the utility or futility of computing with large patterns.

What is appealing about large random patterns is that they have rich and subtle mathematical properties, and they lend themselves to parallel computing. Furthermore, the brain's connections and patterns of activity suggest that kind of computing.

For a computer to work like the human mind, it must be extremely flexible in its use of symbols. It cannot stumble on the multiplicity of meanings that a word can have but rather it must be able to benefit from the multiplicity. The human mind conquers the unknown by making analogies to that which is known, it understands the new in terms of the old. In so doing it creates ambiguity or, rather, it creates rich networks of mental connections and becomes robust.

My hunch is that after we understand how the brain handles analogy—how it treats one thing as another—and have programmed it into computers, programming computers to handle language will be an easy task, but it will not be easy before.

References

Kanerva, P. (1996) Binary spatter-coding of ordered *K*-tuples. In C. von der Malsburg, W. von Seelen, J.C. Vorbrüggen, and B. Sendhoff (eds.), *Artificial Neural Networks* (Proc. ICANN '96, Bochum, Germany), pp. 869–873. Berlin: Springer.

Plate, T. A. (1994) Distributed Representation and Nested Compositional Structure. Ph.D. Thesis. Graduate Department of Computer Science, University of Toronto.

Rumelhart, D.E, and McClelland, J.L. (1986) On learning the past tenses of English verbs. In J.L. McClelland and D.E. Rumelhart (eds.), *Parallel Distributed Processing 2: Applications,* pp. 216–271. Cambridge, Mass.: MIT Press.

In K. Holyoak, D. Gentner, and B. Kokinov (eds.), *Advances in Analogy Research: Integration of Theory and Data from the Cognitive, Computational, and Neural Sciences.* Workshop proceedings, pp. 164-170. Sofia: New Bulgarian University, 1998.