Large Patterns Make Great Symbols: An Example of Learning from Example

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

Abstract

We look at distributed representation of structure with variable binding, that is natural for neural nets and allows traditional symbolic representation and processing. The representation supports learning from example. This is demonstrated by taking several instances of the mother-of relation implying the parent-of relation, by encoding them into a mapping vector, and by showing that the mapping vector maps new instances of mother-of into parent-of.

1 Introduction

Distributed representation is used commonly with neural nets, as it is in ordinary computers, to encode a large number of attributes or things with a much smaller number of variables or units. In this paper we assume that the units are binary so that the encodings of things are bit patterns or bit vectors ('pattern' and 'vector' will be used interchangeably in this paper). In normal symbolic processing the bit patterns are arbitrary identifiers, and it matters only whether two patterns are identical or different, whereas bit patterns in a neural net are somewhat like an error-correcting code: similar patterns (highly correlated, small Hamming distance) mean the same or similar things—hence the use of neural nets as classifiers and as low-level sensory processors.

Computer modeling of high-level mental functions, such as language, led to the development of symbolic processing and has remained mostly in that domain. However, the systems that have been built are rigid and brittle rather than lifelike. It is natural to look to neural nets for a remedy. Their error-correcting properties should make the systems more forgiving. Consequently, there is major research effort into combining symbolic and neural approaches. One research direction is the hybrid approach: use both, and use each for what it does the best. Wermter (1997) reviews such approaches for language processing. Another direction is distributed representation: encode structure in a way that suits neural nets. The present paper explores this second direction.

Much has been written and debated about the encoding of structure for neural nets (see, e.g., Sharkey, 1991) without reaching a clear consensus. Hinton (1990) has discussed it in depth and has introduced the idea of *reduced representation*. This idea is realized in Plate's (1994, 1997) Holographic Reduced Representation (HRR) and in my binary Spatter Code (Kanerva, 1996). I will use the latter here because it allows the simplest examples. The research on representation by Shastri and Ajjanagadde (1993) and by Hummel and Holyoak (1997) demonstrate different solutions to problems that we share.

---

[1] Real World Computing Partnership. The 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.

2 Binary Spatter-coding of Structure

The binary Spatter Code is a form of Holographic Reduced Representation. It is summarized below in terms applicable to all HRRs, and in traditional symbolic terms using a two-place relation $r(x, y)$ as an example.

Space of Representations

HRRs work with large random patterns, or very-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 $\mathbf{x}$ is the N-dimensional vector (N-vector for short) that represents the variable or role x, and $\mathbf{a}$ is the N-vector that represents 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). It 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, $\mathbf{x}\otimes(\mathbf{x}\otimes\mathbf{a}) = \mathbf{a}$ finds the vector to which $\mathbf{x}$ is bound in $\mathbf{x}\otimes\mathbf{a}$ (i.e., what's the value of $\mathbf{x}$?).

Merging

Merging is the second level of composition in which identifiers and bound pairs are combined into a single item. It is also called 'super(im)posing', 'bundling', and 'chunking'. It is done by a *normalized sum* vector, and the merging of $\mathbf{G}$ and $\mathbf{H}$ is written as $\langle\mathbf{G} + \mathbf{H}\rangle$, where $\langle\ldots\rangle$ stands for normalization. The relation $r(A, B)$ can be represented by merging the representations for r, '$r1 = A$', and '$r2 = B$', where $r1$ and $r2$ are the first and second roles of the relation r. It is encoded by

$$\mathbf{R} = \langle\mathbf{r} + \mathbf{r1}\otimes\mathbf{A} + \mathbf{r2}\otimes\mathbf{B}\rangle$$

The normalized sum 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 (i.e., **R** is similar to **r**, **r1**⊗**A**, and **r2**⊗**B**).

Distributivity

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

$$\mathbf{x}\otimes\langle\mathbf{G}+\mathbf{H}+\mathbf{I}\rangle=\langle\mathbf{x}\otimes\mathbf{G}+\mathbf{x}\otimes\mathbf{H}+\mathbf{x}\otimes\mathbf{I}\rangle$$

Distributivity is a key to analyzing HRRs.

Probing

To find out, for example, what **r1** is bound to in **R** (i.e., what's the value of **r1** in **R**?), we probe **R** with **r1** using the unbinding operator. For example, if **R** represents the above relation, probing it with **r1** yields a vector **A′** that is recognizable as **A** (**A′** will retrieve **A** from the item memory). The analysis is as follows:

$$\mathbf{A}'=\mathbf{r1}\otimes\mathbf{R}=\mathbf{r1}\otimes\langle\mathbf{r}+\mathbf{r1}\otimes\mathbf{A}+\mathbf{r2}\otimes\mathbf{B}\rangle$$

which, by distributivity, becomes

$$\mathbf{A}'=\langle\mathbf{r1}\otimes\mathbf{r}+\mathbf{r1}\otimes(\mathbf{r1}\otimes\mathbf{A})+\mathbf{r1}\otimes(\mathbf{r2}\otimes\mathbf{B})\rangle$$

and simplifies to

$$\mathbf{A}'=\langle\mathbf{r1}\otimes\mathbf{r}+\mathbf{A}+\mathbf{r1}\otimes\mathbf{r2}\otimes\mathbf{B}\rangle$$

Thus **A′** is similar to **A**; it is also similar to **r1**⊗**r** and to **r1**⊗**r2**⊗**B** (see *Merging* above), but they are not stored in the item memory and thus act as random noise.

Holistic Mapping and Simple Analogical Retrieval

The functions described so far are sufficient for traditional symbol processing; for example, for realizing a Lisp-like list-processing system. Holistic mapping is a parallel alternative to sequential search and substitution of traditional symbolic processing.

Probing is the simplest form of holistic mapping: It approximately maps a composed pattern into one of its bound constituents, as seen above. However, much more than that can be done in a single mapping operation. For example, it is possible to do several *substitutions* at once, by constructing a mapping vector from individual substitutions (each substitution appears as a bound pair, which than are merged; the kernel map **M*** discussed at the end of the next section is an example). I have demonstrated this kind of mapping between things that share structure (same roles, different objects; Kanerva, 1998). In the next section we do the reverse: we map between structures that share objects (same objects in two different relations). The mappings are constructed from examples, so that this is a demonstration of analogical retrieval or inference.

3 Learning from Example

We will look at two relations, one of which implies the other: 'If *x* is the mother of *y*, then *x* is the parent of *y*', represented symbolically by $m(x, y) \rightarrow p(x, y)$. We take a specific example $m(A, B)$ of the mother-of relation and compare it to the corresponding parent-of relation $p(A, B)$, to get a mapping M_1 between the two. We then use this mapping on another pair (U, V) for which the mother-of relation holds, to see whether and how well M_1 maps $m(U, V)$ into the corresponding parent-of relation $p(U, V)$.

We will encode *'A* is the mother of *B'*, or *m*(*A*, *B*), with the random *N*-vector **mAB** = ⟨**m** + **m1**⊗**A** + **m2**⊗**B**⟩, where **m** encodes (it names) the relation and **m1** and **m2** encode its two roles. Similarly, we will encode *'A* is the parent of *B'*, or *p*(*A*, *B*), with **pAB** = ⟨**p** + **p1**⊗**A** + **p2**⊗**B**⟩. Then

$$\mathbf{M}_1 = \mathbf{M}_{AB} = \mathbf{mAB} \otimes \mathbf{pAB}$$

maps a specific instance of the mother-of relation into the corresponding instance of the parent-of relation, because **mAB**⊗$\mathbf{M}_{AB}$ = **mAB**⊗(**mAB**⊗**pAB**) = **pAB**. The mapping $\mathbf{M}_{AB}$ is based on one example; is it possible to generalize based on only one example? When the mapping is applied to another instance *m*(*U, V*) of the mother-of relation, which is encoded by **mUV** = ⟨**m** + **m1**⊗**U** + **m2**⊗**V**⟩, we get the vector **W**:

$$\mathbf{W} = \mathbf{mUV} \otimes \mathbf{M}_{AB}$$

Does **W** resemble **pUV**?

We will measure the similarity of vectors by their correlation ρ (i.e., by normalized covariance; $-1 \le \rho \le 1$). The correlations reported in this paper are exact (mathematical mean values or expectations). They have been calculated by starting with a set of "primary" vectors made of all possible bit combinations (i.e., *k* primary vectors of 2^k bits each).

If we start with randomly selected (primary) vectors **m**, **m1**, **m2**, **p**, **p1**, **p2**, **A**, **B**, …, **U, V** that are pairwise uncorrelated (ρ = 0), we observe first that **mAB** and **pAB** are uncorrelated but **mAB** and **mUV** are correlated because they both include **m** in their composition; in fact, ρ(**mAB**, **mUV**) = 0.25 and, similarly, ρ(**pAB**, **pUV**) = 0.25. When **W** is compared to **pUV** and to other vectors, there is a tie for the best match: ρ(**W**, **pUV**) = ρ(**W**, **pAB**) = 0.25. All other correlations with **W** are lower: with the related (reversed) parent-of relations **pBA** and **pVU** it is 0.125, with an unrelated parent-of relation **pXY** it is 0.0625, and with **A**, **B**, …, **U, V**, **mAB**, and **mUV** it is 0. So based on only one example, *m*(*A*, *B*) → *p*(*A*, *B*), it cannot be decided whether *m*(*U, V*) should be mapped to the original "answer" *p*(*A*, *B*) or should generalize to *p*(*U, V*).

Let us now look at generalization based on three examples of mother implying parent: What is *m*(*U, V*) mapped to by M_3 that is based on *m*(*A*, *B*) → *p*(*A*, *B*), *m*(*B*, *C*) → *p*(*B*, *C*), and *m*(*C*, *D*) → *p*(*C*, *D*)? This time we will use a mapping vector $\mathbf{M}_3$ that is the sum of three binary vectors,

$$\mathbf{M}_3 = \mathbf{M}_{AB} + \mathbf{M}_{BC} + \mathbf{M}_{CD}$$

where $\mathbf{M}_{AB}$ is as above, and $\mathbf{M}_{BC}$ and $\mathbf{M}_{CD}$ are defined similarly. Since $\mathbf{M}_3$ itself is not binary, mapping **mAB** or **mUV** with $\mathbf{M}_3$ cannot be done with an XOR. However, we can use an equivalent system in which binary vectors are bipolar, by replacing 0s by 1s, 1s by −1s, and bitwise XOR (⊗) by coordinatewise multiplication (×), and then the mapping can be done with multiplication, vectors can be compared with correlation, and the results obtained with $\mathbf{M}_1$ still hold. Notice that now $\mathbf{M}_{AB}$ = **mAB**×**pAB**, for example.

Mapping with $\mathbf{M}_3$ gives the following results: To check that it works at all, consider $\mathbf{W}_{AB}$ = **mAB**×$\mathbf{M}_3$; it is most similar to **pAB** (ρ = 0.71) as expected because $\mathbf{M}_3$ contains $\mathbf{M}_{AB}$. Its other significant correlations are with **mAB** (0.41) and with **pUV** and **pVU** (0.18). Thus the mapping M_3 strongly supports *m*(*A*, *B*) → *p*(*A*, *B*). It also supports the generalization *m*(*U, V*) → *p*(*U, V*) unambiguously, as seen by comparing $\mathbf{W}_{UV}$ = **mUV**×$\mathbf{M}_3$ with **pUV**. The correlation is ρ($\mathbf{W}_{UV}$, **pUV**) = 0.35; the other significant correlations of $\mathbf{W}_{UV}$ are with **pAB** and **pVU** (0.18) and with **pBA** (0.15) because they all include the vector **p** (parent-of).

To track the trend further, we check generalization based on five examples, $m(A, B) \rightarrow p(A, B)$, $m(B, C) \rightarrow p(B, C)$, …, $m(E, F) \rightarrow p(E, F)$, giving the mapping vector

$$\mathbf{M}_5 = \mathbf{M}_{AB} + \mathbf{M}_{BC} + \mathbf{M}_{CD} + \mathbf{M}_{DE} + \mathbf{M}_{EF}$$

When applied to **mAB**, we get $\rho(\mathbf{mAB}\times\mathbf{M}_5, \mathbf{pAB}) = 0.63$ (the other correlations are lower, as they were for $\mathbf{M}_3$), and when applied to **mUV**, we get $\rho(\mathbf{mUV}\times\mathbf{M}_5, \mathbf{pUV}) = 0.40$ (again the other correlations are lower).

When the individual mappings $\mathbf{M}_{xy}$ are analyzed, each is seen to contain the vectors $\mathbf{m}\times\mathbf{p}$, $\mathbf{m1}\times\mathbf{p1}$, and $\mathbf{m2}\times\mathbf{p2}$, plus others that act as noise and average out as more and more $\mathbf{M}_{xy}$s are added together. These three kernel vectors are responsible for the generalization, and from them we can construct a *kernel mapping* from mother-of to parent-of:

$$\mathbf{M}^* = \mathbf{m}\times\mathbf{p} + \mathbf{m1}\times\mathbf{p1} + \mathbf{m2}\times\mathbf{p2}$$

When **mUV** is mapped with it, we get a maximum correlation with **pUV**, as expected, namely, $\rho(\mathbf{mUV}\times\mathbf{M}^*, \mathbf{pUV}) = 0.43$; correlations with other parent-of relations are $\rho(\mathbf{mUV}\times\mathbf{M}^*, \mathbf{pXY}) = 0.14$, ($\mathbf{X} \neq \mathbf{U}$, $\mathbf{Y} \neq \mathbf{V}$) and 0 with everything else.

The results are summarized in Figure 1 that relates the amount of data to the strength of inference and generalization. The data are examples or instances of mother-of implying parent-of, $m(x, y) \rightarrow p(x, y)$, and the task is to map either an old (Fig. 1a) or a new (Fig. 1b) instance of mother into parent. The data are taken into account by encoding it into the mapping vector $\mathbf{M}_n$.

Figure 1a shows the effect of new data on old examples. Adding examples into the mapping makes it less specific, and consequently the correlation for old inferences (**pAB**) decreases, but it decreases also for all incorrect alternatives. Figure 1b shows the effect of new data on generalization. When the mapping is based on only one example, generalization is inconclusive ($\mathbf{mUV}\times\mathbf{M}_1$ is equally close to **pAB** and **pUV**), but when it is based on three examples, generalization is clear, as $\mathbf{M}_3$ maps **mUV** much closer to **pUV** than to any of the others. Finally, the kernel mapping $\mathbf{M}^*$ represents a very large number of examples, a limit as n approaches infinity, and then the correct inference is the clear winner.

4 Discussion

We have used a simple form of Holographic Reduced Representation to demonstrate mapping between different information structures, which is a task that has traditionally been in the domain of symbolic processing. Similar demonstrations have been made by Chalmers (1990) and by others (e.g., Bodén & Niklasson, 1995) using Pollack's (1990) Recursive Auto-Associative Memory (RAAM). The lesson from such demonstrations is that certain kinds of representations and operations on them make it possible to do traditionally symbolic tasks with distributed representations suitable for neural nets. Furthermore, when patterns are used as if they were symbols (Gayler & Wales, 1998), we do not need to configure different neural nets for different data structures. A general-purpose neural net that operates with such patterns is then a little like a general-purpose computer that runs programs for a variety of tasks.

The examples above were encoded with the binary Spatter Code because of its simplicity. It uses very simple mathematics and yet demonstrates the properties of Holographic Reduced Representation. However, the thresholding of the vector sum, which is a part of the Spatter Code's merging operation, discards a lot of information. Therefore the sums themselves, rather than thresholded sums, were used for the mappings $\mathbf{M}_n$ and $\mathbf{M}^*$. This put the mappings "outside the

system" (they are not binary vectors; in HRRs, all things are elements of the same space), and the results had to be measured by correlation rather than by Hamming distance. In fact, the system we ended up with (binding and mapping with coordinatewise multiplication, merging with vector addition) has been used by Ross Gayler (personal communication) to study distributed representation of structure. Plate's (1994) real-vector HRRs also use the vector sum for merging, but they use circular convolution for binding, which involves more computation.

The correlations obtained with HRRs tend to be low. Even the highest correlation for generalization is only 0.43 and it corresponds to the kernel map (Fig. 1b), and we usually have to distinguish between two quite low correlations, such as 0.3 and 0.25. That would not be possible if the vectors had only few elements, but when they have several thousands, even small correlations and small differences in them are significant. This explains the need for very-high-dimensional vectors ($N > 1{,}000$). The statistics show that $N = 10{,}000$ allows very large systems to work reliably and that $N = 100{,}000$ will always be unnecessarily large.

The binary system has its peculiarities due to binding and mapping with XOR that is its own inverse function and is also commutative. When we form the mapping from mother-of to parent-of, which is a valid inference, it is the same mapping as from parent-of to mother-of, which is valid evidence but not valid inference. The binary system with XOR obviously is not the final solution to the distributed representation of structure, although it is a good introduction.

Our example of learning from example uses a traditional symbolic setting with roles and fillers, an operation for binding the two, and another operation for combining bound pairs (and singular identifiers) into new (higher level) compound entities such as relations. The same operations can also be used to encode general mappings, such as the kernel map between two relations. Individual instances, or examples, of the mapping were encoded with the binding operator—by binding corresponding instances of the two relations—and such examples were combined by simple averaging into an estimate of the general (kernel) map. That the averaging of vectors for structured entities should be meaningful, is a consequence of the representation used and has no counterpart in traditional symbolic representation. It suggests the possibility of structured representation based on examples without explicitly encoding roles and fillers but that can be interpreted as consisting of roles and fillers—representation from which the abstract notions of role and filler, and may others, could be extracted. We would still use the same binding and merging operators, and possibly one or two additional operators, but instead of binding the objects of a new instance to abstract roles, as was done in the examples above, we would bind them to the objects of an old instance: an old instance, rather than an abstract frame, would serve as the template. This would correspond rather naturally to how people learn. Such possibilities point to the need for further research on the subject.

References

Bodén, M.B., and Niklasson, L.F. (1995) Features of distributed representation for tree structures: A study of RAAM. In L.F. Niklasson & M.B. Bodén (eds.), *Current Trends in Connectionism*, pp. 121–139. Hillsdale, NJ: Erlbaum.

Chalmers, D.J. (1990) Syntactic transformations on distributed representations. *Connection Science* 2(1–2):53–62.

Gayler, R.W., and Wales, R. (1998) Connections, binding, unification and analogical promiscuity. 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* (proceedings of workshop in Sofia), pp. 181–190. Sofia: New Bulgarian University.

Hinton, G.E. (1990) Mapping part–whole hierarchies into connectionist networks. *Artificial Intelligence* 46(1–2):47–75.

Hummel, J.E., and Holyoak, K.J. (1997) Distributed representation of structure: A theory of analogical access and mapping. *Psychological Review* 104:427–466.

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.

Kanerva, P. (1998) Dual role of analogy in the design of a cognitive computer. 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* (proceedings of workshop in Sofia), pp. 164–170. Sofia: New Bulgarian University.

Plate, T.A. (1994) Distributed Representations and Nested Compositional Structure. PhD thesis. Graduate Department of Computer Science, University of Toronto. (Available at Ftp-host: ftp.cs.utoronto.ca as Ftp-file: /pub/tap/plate.thesis.ps.Z)

Plate, T.A. (1997) A common framework for distributed representation schemes for compositional structure. In F. Maire, R. Hayward, & J. Diederich (eds.), *Connectionist Systems for Knowledge Representation and Deduction* (Proc. CADE '97, Townsville, Australia), pp. 15–34. Queensland U. of Technology.

Pollack, J.P. (1990) Recursive distributed representations. *Artificial Intelligence* 46(1–2):77–105.

Sharkey, N.E. (1991) Connectionist representation techniques. *AI Review* 5(3):143–167.

Shastri, L., and Ajjanagadde, V. (1993) From simple associations to systematic reasoning. *Behavioral and Brain Sciences* 16(3):417–494.

Wermter, S. (1997) Hybrid Approaches to Neural-Network-Based Language Processing. Report ICSI TR-97-030, International Computer Science Institute, Berkeley, California.

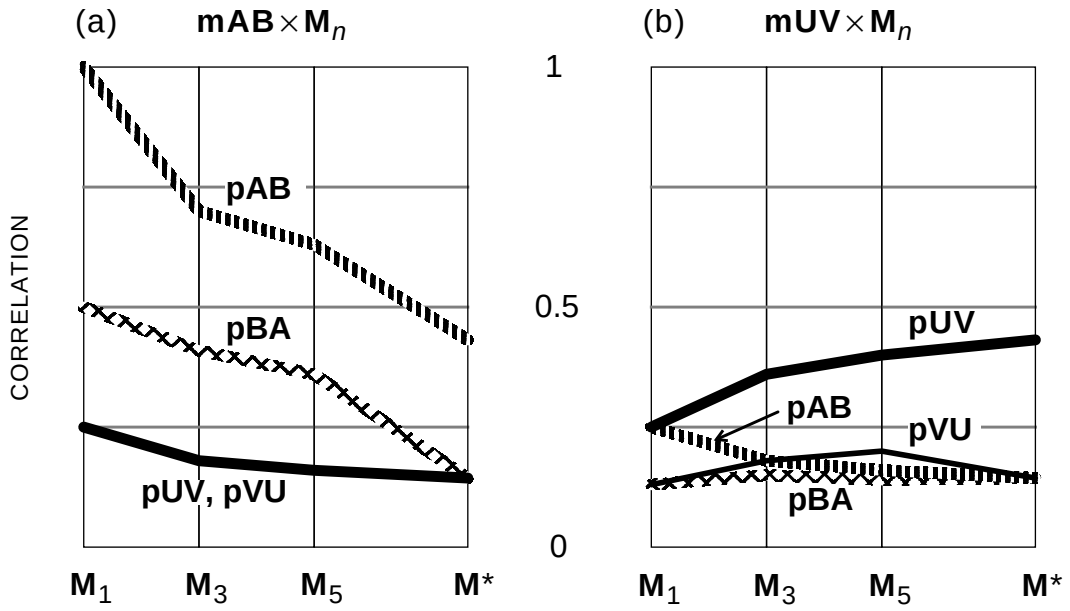

Figure 1. Leaning from example: The mappings $\mathbf{M}_n$ map the mother-of relation to the parent-of relation. They are computed from *n* specific instances or examples of **mXY** being mapped to **pXY**. Each map $\mathbf{M}_n$ includes the instance '**mAB** maps to **pAB**' and excludes the instance '**mUV** maps to **pUV**'. The mappings are tested by mapping an "old" instance **mAB** (a) and a "new" instance **mUV** (b) of mother-of to the corresponding parent-of. The graphs show the closeness of the mapped result (correlation) to different alternatives for parent-of. The mapping **M*** is a kernel map that corresponds to an infinite number of examples. Graph b shows good generalization (**mUV** is mapped closest to **pUV**) and discrimination (it is mapped much further from, e.g., **pAB**) based on only three examples ($\mathbf{M}_3$).