ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

Neurocomputing 69 (2006) 2127–2139

NEUROCOMPUTING

www.elsevier.com/locate/neucom

Deductive rules in holographic reduced representation

Vladimír Kvasnička, Jiří Pospíchal*

Institute of Applied Informatics, Faculty of Informatics and Information Technologies, Slovak Technical University, 842 16 Bratislava, Slovakia

Received 4 March 2004; received in revised form 15 September 2005; accepted 16 September 2005
Available online 8 February 2006
Communicated by T. Heskes

Abstract

Holographic reduced representation is based on a suitable distributive coding of structured information in conceptual vectors, which elements satisfy normal distribution $N(0,1/n)$. Existing applications of this approach concern various models of associative memory that exploit a simple algebraic operation of scalar product of distributed representations to measure an overlap between two structured concepts. This paper describes an inference process based on the rules *modus ponens* and *modus tollens*.

Keywords: Holographic; Reduced representation; Distributed representation; Convolution

1. Introduction

A modern view of the relation between brain and mind is based on the neuroscience paradigm [9], according to which the architecture of the brain is determined by connections between neurons, their inhibitory or excitatory character and also by the strength of the connections. Human brain displays a great plasticity, synapses are perpetually formed (but also deleted) during a learning process. It can be stated, that *an ability of brain to perform not only cognitive activities, but also to serve as memory and control center for our motoric activities, is fully encoded by its architecture.* The metaphor of the brain as a computer should be therefore reformulated: the computer in the metaphor should be specified as a *parallel distributed computer* (containing many billions of neurons, elementary processors interconnected into a complex neural network). A program in such a parallel computer is directly encoded in the architecture of the neural network, i.e. human brain is a single-purpose parallel computer represented by a neural network, which cannot be reprogrammed without a change of its architecture.

It follows that the mind and the brain create one integral unit, which is characterized by a complementary *dualism*. In this approach, the mind is a program carried out by the brain. This program is specified by architecture of the distributed neural network representing the brain. The brain and the mind are two different aspects of the same object:

(1) When talking about the brain, we refer to a "hardware" structure, biologically determined by neurons and their synaptic connections (formally represented by a neural network).
(2) When talking about the mind, we refer to cognitive and other similar activities of the brain, which are carried out on a symbolic level, where the transformation of symbolic information is processed by (simple) rules.

Complementary dualism between brain and mind causes certain difficulties in the interpretation of cognitive activities of the mind. A purely neural approach to the interpretation of cognitive activities of the mind focuses on the search of neural correlates of cognitive activities (connectionism). The application of the neural paradigm to the interpretation of symbolic cognitive activities has a

*Corresponding author. Tel.: ++ 421 2 60291679; fax: ++ 421 2 65420587.

E-mail addresses: kvasnicka@fiit.stuba.sk (V. Kvasnička), pospichal@fiit.stuba.sk (J. Pospíchal).

0925-2312/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2005.09.011

"side effect" in "dissolving" of these activities in their microscopic description; symbols quasi "disappear" in the detailed description of activities of neurons, strengths of synaptic connections, etc. On the other side, the absolute acceptance of symbolic paradigm in interpretation of cognitive activities of mind (cognitivism) ignores the fact, that mind is thoroughly embedded in brain. Purely symbolic paradigm usually leads to a conceptual sterility. An effort to explain cognitive activities of human mind only in the phenomenological terms derived from the concept of symbol is mostly fruitless. It leads to symbolic constructs (methods, algorithms, etc.) without experimental support in neuroscience.

The goal of this paper is to highlight an alternative approach, which may overcome the gap between the connectionist and cognitivist approaches to the description and interpretation of cognitive activities of the human brain [25,27,28,29]. Distributed representation, where mental representations (symbols) are specified by distributed patterns of neural activities, allows to integrate connectionism and cognitivism. Formal algebraic operations can be introduced over these distributed patterns. These operations allow mathematical modeling of cognitive operations, as well as simulation of information storage and retrieval processes in memory.

We shall turn our attention to a nontraditional style of performing calculation by using distributed patterns. This approach is substantially different from classical numeric and symbolic computations and it is a suitable model tool for understanding global properties of neural networks. Such a "neurocomputing" is based on extensive randomly created patterns (represented by multidimensional vectors with random entries). This approach, which basic principles were formulated already at the end of the sixties [3,10,11,24,26], was followed by Willshaw [30], Borselino [1], and Murdock [12,13], and crowned by a series of works by Tony Plate [17–19] on "*holographic reduced representation*" (HRR). Kanerva [6–8] in the middle of nineties proposed a certain alternative to HRR, which is based on randomly generated binary vectors. Rachkovskij and Kussul suggested a new representation and processing of structures with binary sparse distributed codes [22,23]. Jane Neumann [14–16] showed that a simple gradient descent approach can be used to learn the holistic transformation of HRR from examples. The acquired knowledge can be generalized to structures containing unseen elements and to structures more complex than the training examples.

Our results as well as results of Plate [19] indicate, that transformation vector need not be constructed by a gradient optimization method, but it may be constructed by one-step constructive approach.

The goal of the present paper is to apply HRR to model simple higher level cognitive processes of reasoning by application of rules modus ponens and modus tollens in propositional and predicate logic. To keep the paper readable, in its first part we shortly reformulated basic results obtained by Plate [19].

2. A mathematical formulation of holographic representation

In this section we shall outline basic properties of holographic representation [1,6–8,11–19,30]. Its essential term is a *conceptual vector* [4], which is represented by an n-dimensional vector

$$\boldsymbol{a} \in R^n \Rightarrow \boldsymbol{a} = (a_0, a_1, ..., a_{n-1}). \tag{1}$$

Its components are random numbers from a normal distribution

$$a_i = N(0, 1/n) \quad \forall i \in \{0, 1, ..., n-1\}, \tag{2}$$

where $N(0,1/n)$ is a random number from a normal distribution with mean value of 0 and standard deviation $1/n$.

A binary operation "*convolution*", defined over conceptual vectors, assigns to a couple of vectors a third vector, $\otimes : R^n \times R^n \to R^n$, or

$$\boldsymbol{c} = \boldsymbol{a} \otimes \boldsymbol{b}. \tag{3}$$

The components of the resulting vector $c = (c_0, c_1, \ldots, c_{n-1})$ are determined by a formula

$$c_i = \sum_{j=0}^{n-1} a_j b_{[i-j]} \quad (i = 0, 1, \ldots, n-1), \tag{4}$$

where the index in the square brackets, $[k]$, is defined using a *modulo n* operation as follows[1]

$$[k] = k \ \ mod \ \ n. \tag{5}$$

For example, standard expression of convolution of two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ for $n = 3$ has the following form

$$\begin{aligned} c_0 &= a_0 b_0 + a_1 b_2 + a_2 b_1, \\ c_1 &= a_0 b_1 + a_1 b_0 + a_2 b_2, \\ c_2 &= a_0 b_2 + a_1 b_1 + a_2 b_0. \end{aligned} \tag{6}$$

The convolution satisfies the following properties:

(1) commutativity, $\boldsymbol{a} \otimes \boldsymbol{b} = \boldsymbol{b} \otimes \boldsymbol{a}$,
(2) associativity, $(\boldsymbol{a} \otimes \boldsymbol{b}) \otimes \boldsymbol{c} = \boldsymbol{a} \otimes (\boldsymbol{b} \otimes \boldsymbol{c})$,
(3) distributiveness, $\boldsymbol{a} \otimes (\alpha \boldsymbol{b} + \beta \boldsymbol{c}) = \alpha(\boldsymbol{a} \otimes \boldsymbol{b}) + \beta(\boldsymbol{a} \otimes \boldsymbol{c})$,
(4) an existence of a unit vector, $\mathbf{1} \otimes \boldsymbol{a} = \boldsymbol{a}$ $(\mathbf{1} = (1, 0, \ldots, 0))$.

Let us define a unary operation *involution*

$$(\,)^* : R^n \to R^n \tag{7}$$

by a formula

$$\boldsymbol{b} = \boldsymbol{a}^* = \left(a_{[0]}, a_{[-1]}, \ldots, a_{[-n+2]}, a_{[-n+1]}\right). \tag{8}$$

[1] A standard definition of an arithmetic operation *k modulo n* is determined as a remainder after integer division by a number *n*. It is necessary to comment, that the used definition of the operation *k modulo n* is different from this standard definition for negative numbers *k*. While the standard definition provides a result with a negative value, if the result is negative in our definition, then it is transformed by adding *n* to it.

The operation of involution satisfies the equations

$$(\boldsymbol{a}+\boldsymbol{b})^* = \boldsymbol{a}^* + \boldsymbol{b}^*, \tag{9a}$$

$$(\boldsymbol{a}\otimes\boldsymbol{b})^* = \boldsymbol{a}^* \otimes \boldsymbol{b}^*, \tag{9b}$$

$$\left(\boldsymbol{a}\otimes\boldsymbol{b}^*\right)\cdot\boldsymbol{c} = \boldsymbol{a}\cdot(\boldsymbol{b}\otimes\boldsymbol{c}), \tag{9c}$$

$$\boldsymbol{a}^{**} = \boldsymbol{a}. \tag{9d}$$

A numerical implementation of the convolution can be carried out by two different approaches. The first one is based on formula (4); it offers a simple implementation, but its running time is $O(n^3)$. The second one is much more sophisticated and complex and it corresponds to an application of fast Fourier transform (FFT) [21]; its running time is $O(n \log n)$, see Fig. 1.

We prove that an involution $\boldsymbol{c}^*$ is roughly proportional to an inverse vector $\boldsymbol{c}^{-1}$, $\boldsymbol{c}^*\otimes\boldsymbol{c}\approx\mathbf{1}$. Let us study the ith component of convolution $\boldsymbol{c}^*\otimes\boldsymbol{c}$

$$\left(\boldsymbol{c}^*\otimes\boldsymbol{c}\right) = \sum_{k=0}^{n-1} c_k^* c_{[i-k]} = \sum_{k=0}^{n-1} c_{[-k]}c_{[i-k]} = \begin{cases} \boldsymbol{c}\cdot\boldsymbol{c} & (\text{for } i=0), \\ \sum\limits_{k=0}^{n-1} c_{[-k]}c_{[i-k]} & (\text{for } i>0). \end{cases} \tag{10}$$

The zero-component of convolution $(\boldsymbol{c}^*\otimes\boldsymbol{c})_0$ corresponds to a scalar product $\boldsymbol{c}\cdot\boldsymbol{c}$ expressed as a sum of positive "diagonal" terms c_i^2, whereas other components $(\boldsymbol{c}^*\otimes\boldsymbol{c})_i$, for $i\geqslant 1$, are determined by sums of "nondiagonal" terms $c_i c_j$, which have random signs. This observation has an important consequence that $(\boldsymbol{c}^*\otimes\boldsymbol{c})_0$ is much greater than absolute values of remaining components $(\boldsymbol{c}^*\otimes\boldsymbol{c})_i$, for $i\geqslant 1$, then we proved that $\boldsymbol{c}^*\otimes\boldsymbol{c}\approx\mathbf{1}$, which was to be proved (see Fig. 2).

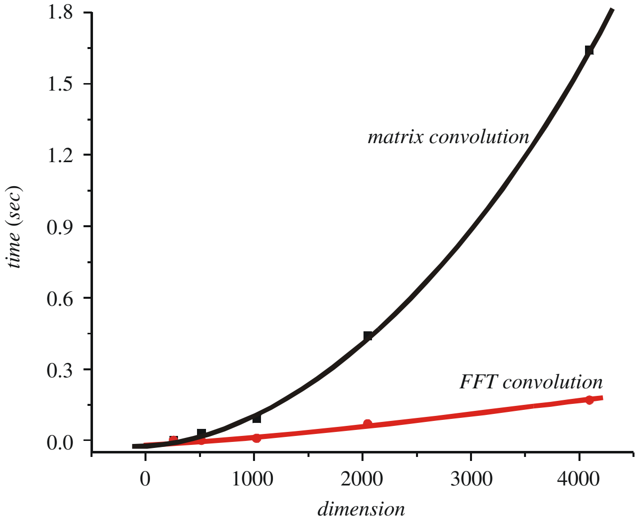

Fig. 1. Plots of computational time vs. problem dimensions for a realization of convolution product by two different methods: (1) matrix convolution is implemented by the formula (4), the time is proportional to n^3; (2) FFT convolution, the time is proportional to $n\times\log(n)$. We observed that up to $n=30$ the matrix convolution is faster, but for substantially greater values of n the convolution based on FFT is much more effective.

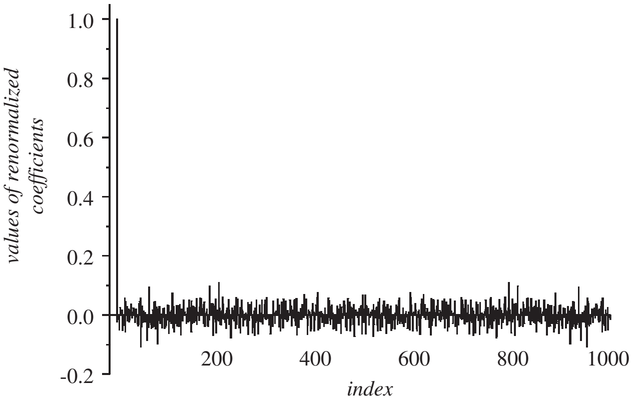

Fig. 2. A histogram of single components of $\boldsymbol{c}^*\otimes\boldsymbol{c}$, where $\boldsymbol{c}$ is randomly generated conceptual vector for $n=1000$. We see from this plot that an absolute value of the "first" component $(\boldsymbol{c}^*\otimes\boldsymbol{c})_0$ is two orders of magnitude greater than absolute values of remaining components $(\boldsymbol{c}^*\otimes\boldsymbol{c})_i$, for $i\geqslant 1$. It means that the product $\boldsymbol{c}^*\otimes\boldsymbol{c}$ after a proper normalization plays approximately a role of unit vector $(\boldsymbol{c}\cdot\boldsymbol{c})^{-1}(\boldsymbol{c}^*\otimes\boldsymbol{c})\doteq\mathbf{1}=(1,0,\ldots,0,0)$.

One of the basic aspects of the holographic representation is the possibility of reconstruction of the original components, which were used for construction of convolution of two vectors. This possibility is very important, since it allows to decode the original information from the complex conceptual vectors. *Reconstruction of* $\boldsymbol{x}$ from $\boldsymbol{c}\otimes\boldsymbol{x}$ is based on the formula proved above, $\boldsymbol{c}^*\otimes\boldsymbol{c}\approx\mathbf{1}$

$$\tilde{\boldsymbol{x}} = \boldsymbol{c}^*\otimes(\boldsymbol{c}\otimes\boldsymbol{x}) = \left(\boldsymbol{c}^*\otimes\boldsymbol{c}\right)\otimes\boldsymbol{x} \doteq (\boldsymbol{c}\cdot\boldsymbol{c})1\otimes\boldsymbol{x} = (\boldsymbol{c}\cdot\boldsymbol{c})\boldsymbol{x} \tag{11}$$

according to which the convolution $\boldsymbol{c}^*$ with the vector $\boldsymbol{c}\otimes\boldsymbol{x}$ produces the vector $\tilde{\boldsymbol{x}}$, which is similar to the original vector $\boldsymbol{x}$, $\tilde{\boldsymbol{x}}\approx\boldsymbol{x}$. This result can be reformulated in the form

$$\frac{1}{(\boldsymbol{c}\cdot\boldsymbol{c})}\tilde{\boldsymbol{x}} = \begin{pmatrix} x_0 \\ x_1+\eta_1 \\ \vdots \\ x_{n-1}+\eta_{n-1} \end{pmatrix} = \boldsymbol{x}+\eta, \tag{12}$$

where the elements of $\boldsymbol{\eta}$ can be regarded as normally distributed random noise with a zero mean and a standard deviation much smaller than the standard deviation of the elements of $\boldsymbol{x}$.

The overlap of the resulting vector $\tilde{\boldsymbol{x}}$ with the original vector $\boldsymbol{x}$ is determined by cosine of these two vectors with real values

$$-1\leqslant\cos(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \frac{\boldsymbol{x}\cdot\tilde{\boldsymbol{x}}}{|\boldsymbol{x}||\tilde{\boldsymbol{x}}|}\leqslant 1, \tag{13}$$

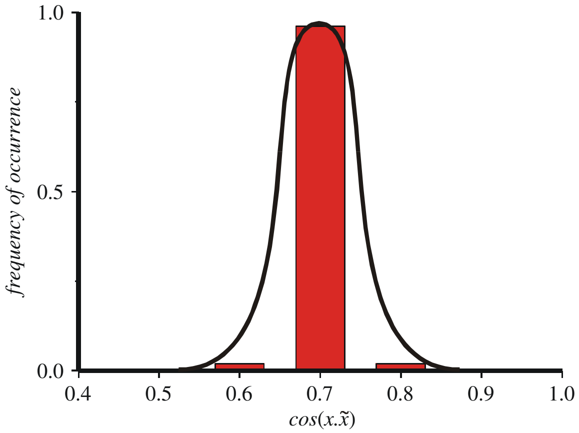

Fig. 3. A histogram of overlaps determined by cosine of the angle between vectors $\tilde{\boldsymbol{x}}$ and $\boldsymbol{x}$ (of dimensions $n = 1000$) has the highest frequency around 0.7, from which follows, that the vectors $\tilde{\boldsymbol{x}}$ and $\boldsymbol{x}$ are similar.

where the inequalities result directly from the Schwartz's inequality from linear algebra. The closer this value is to its maximum value, the more similar[2] the vectors $\tilde{\boldsymbol{x}}$ and $\boldsymbol{x}$ are.

The Fig. 3 shows a histogram of overlaps of vectors $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$, where the most common overlap between $\tilde{\boldsymbol{x}} = \boldsymbol{c}^{*} \otimes \boldsymbol{c} \otimes \boldsymbol{x}$ and $\boldsymbol{x}$ is around 0.7 (vectors $\boldsymbol{c}$ and $\boldsymbol{x}$ are randomly generated different conceptual vectors of dimension $n = 1000$). It follows that the vectors $\tilde{\boldsymbol{x}}$ and $\boldsymbol{x}$ are similar, $\tilde{\boldsymbol{x}} \approx \boldsymbol{x}$.

Formula (12) may be also verified by the approach called the "*superposition memory*". Suppose we have a set containing $p+q$ randomly generated conceptual vectors, $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_p, \boldsymbol{x}_{p+1}, ..., \boldsymbol{x}_{p+q}\}$, while $p<q$. Using the first p vectors from X allows us to define a *memory vector* $\boldsymbol{t}$ as their sum

$$\boldsymbol{t} = \sum_{i=1}^{p} \boldsymbol{x}_i. \tag{14}$$

The vector $\boldsymbol{t}$ represents a superposition memory, which by a simple additive way contains vectors from the set X. The decision, whether some vector $\boldsymbol{x} \in X$ is contained in $\boldsymbol{t}$ must be based on the value of the cosine (15)

$$\cos(\boldsymbol{x}, \boldsymbol{t}) = \frac{\boldsymbol{x} \cdot \boldsymbol{t}}{|\boldsymbol{x}||\boldsymbol{t}|}. \tag{15}$$

If this value is greater than a predefined threshold value, $\cos(\boldsymbol{x}, \boldsymbol{t}) \geqslant \vartheta$, then the vector $\boldsymbol{x}$ is included in the superposition memory $\boldsymbol{t}$, in the opposite case, if $\cos(\boldsymbol{x}, \boldsymbol{t}) < \vartheta$, then the vector $\boldsymbol{x}$ is not included in $\boldsymbol{t}$ (see Fig. 4).

The illustrative example (see Fig. 4) shows, that it is possible to determine the presence of conceptual vectors in superposition memory. Such vectors can "appear" in other complex conceptual vectors, where these complex conceptual vectors can be the result of complicated previous calculations — transformations. The method used for decoding of superposition memory vector is called "*clean-up*" (after Plate [19]) and it is specified as follows: let us have a set of vectors $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ and some vector $\boldsymbol{t}$. We face the decision, whether the memory vector (trace) $\boldsymbol{t}$ contains a superposition component, which is similar (or which is not similar) to some vector from the set X. This problem can be solved by calculating a cosine (15), formally

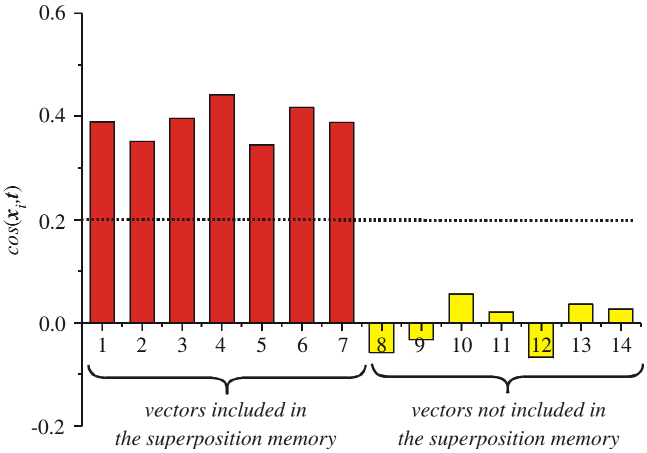

Fig. 4. Illustration of the superposition memory for the first seven vectors of the set X, which contains 14 randomly generated conceptual vectors of the dimension $n = 1000$. The threshold value ϑ can be in this case set to 0.2.

$$\boldsymbol{x} \approx \boldsymbol{t} = \begin{cases} \text{yes} & (\cos(\boldsymbol{x}, \boldsymbol{t}) \geqslant \vartheta), \\ \text{no} & (\cos(\boldsymbol{x}, \boldsymbol{t}) < \vartheta), \end{cases} \tag{16}$$

where ϑ is a chosen *threshold value* of acceptance of the size of the cosine as the positive answer. The result of this cleaning-up process is a subset of vectors

$$X(\boldsymbol{t}) = \{\boldsymbol{x} \in X; \boldsymbol{x} \approx \boldsymbol{t}\} \subseteq X. \tag{17}$$

We can also ask, whether the memory vector $\boldsymbol{t}$ is similar to any of the vectors from the set X? This more general question shall be decided from the maximum value of the cosine

$$\cos(\boldsymbol{t}, X) = \max_{x \in X} \cos(\boldsymbol{t}, \boldsymbol{x}). \tag{18}$$

Then we can rewrite (16) in the form

$$\boldsymbol{t} \approx X = \begin{cases} \text{yes} & (\cos(\boldsymbol{t}, X) \geqslant \vartheta), \\ \text{no} & (\cos(\boldsymbol{t}, X) < \vartheta). \end{cases} \tag{19}$$

3. Associative memory

The construction of the associative memory belongs to the main results of the holographic reduced representation, which can be further generalized by the so called chunking. Let us have a set of conceptual vectors $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ and a training set $A_{\text{train}} = \{\boldsymbol{c}_i/\boldsymbol{x}_i\ ;\ i = 1, 2, \ldots, m\}$, which contains $m<n$ associated couples of conceptual vectors $\boldsymbol{c}_i/\boldsymbol{x}_i$, where $\boldsymbol{c}_i$ is the *input* to the associative memory (*cue*)

[2] If the overlap value approaches -1, then the vectors $\tilde{\boldsymbol{x}}$ and $\boldsymbol{x}$ are also similar, even though they have opposite orientation (they are anticolinear).

and $\boldsymbol{x}_i$ is the output from the memory. Let's create a memory vector $\boldsymbol{t}$ representing *the associative memory* created from the training set A_{train}

$$\boldsymbol{t} = \boldsymbol{c}_1 \otimes \boldsymbol{x}_1 + \cdots + \boldsymbol{c}_m \otimes \boldsymbol{x}_m = \sum_{i=1}^{m} \boldsymbol{c}_i \otimes \boldsymbol{x}_i \qquad (20)$$

Let us suppose, that we know in advance only the inputs $\boldsymbol{c}_i$ to the associative memory, we do not know the possible outputs from the set $X_{\text{train}} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$. The response of the associative memory to the input—cue $\boldsymbol{c}_i$ is determined by the process of "*cleaning-up*" represented by formulas (16)–(19). In the first step we shall calculate the vector $\tilde{\boldsymbol{x}}_i = \boldsymbol{c}_i^* \otimes \boldsymbol{t}$, then by a process based on the maximum value of the cosine we shall find whether $\tilde{\boldsymbol{x}}_i \approx \boldsymbol{x}_i \in X$

$$\cos(\tilde{\boldsymbol{x}}_i, X) = \max_{x \in X} \cos(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}). \qquad (21)$$

The concept of associative memory will be illustrated by the following two examples.

4. Sequence of symbols

The aim of this section is to show, that a holographic distributed representation is able to process a linear sequence of symbols, which are represented by a sequence of conceptual vectors (see Plate [19]).

Let us study a sequence of 6 conceptual vectors of the dimension $n = 1000$

$$sequence = \left\{\boldsymbol{a} \to \boldsymbol{b} \to \boldsymbol{c} \to \boldsymbol{d} \to \boldsymbol{e} \to \boldsymbol{f}\right\}. \qquad (22)$$

For these vectors we shall construct a memory vector

$$\begin{aligned} \boldsymbol{t}_0 = {} & \boldsymbol{a} + \boldsymbol{a} \otimes \boldsymbol{b} + \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} + \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d} \\ & + \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d} \otimes \boldsymbol{e} + \mathbf{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d} \otimes \boldsymbol{e} \otimes \boldsymbol{f}. \end{aligned} \qquad (23)$$

Let us suppose, that we get a memory vector, which is constructed in such a way from a known bigger set of vectors, but we do not know which vectors were selected and in what order. We shall show, that by the clean up procedure we can reconstruct the original sequence (22) from the vector $\boldsymbol{t}_0$ step by step using the following procedure:

1. *step*:$\boldsymbol{a} = clean\_up(\boldsymbol{t}_0)$, $\quad \boldsymbol{t}_1 := \boldsymbol{t}_0 - \boldsymbol{a}$,
   $\tilde{\boldsymbol{t}}_1 := \boldsymbol{a}^* \otimes \boldsymbol{t}_1$,
2. *step*:$\boldsymbol{b} = clean\_up(\tilde{\boldsymbol{t}}_1)$, $\quad \boldsymbol{t}_2 := \boldsymbol{t}_1 - \boldsymbol{a} \otimes \boldsymbol{b}$,
   $\tilde{\boldsymbol{t}}_2 := (\boldsymbol{a} \otimes \boldsymbol{b})^* \otimes \boldsymbol{t}_2$,
3. *step*:$\boldsymbol{c} = clean\_up(\tilde{\boldsymbol{t}}_2)$, $\quad \boldsymbol{t}_3 := \boldsymbol{t}_2 - \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c}$,
   $\tilde{\boldsymbol{t}}_3 := (\boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c})^* \otimes \boldsymbol{t}_3$,
4. *step*:$\boldsymbol{d} = clean\_up(\tilde{\boldsymbol{t}}_3)$, $\quad \boldsymbol{t}_4 := \boldsymbol{t}_3 - \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d}$,
   $\tilde{\boldsymbol{t}}_4 := (\boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d})^* \otimes \boldsymbol{t}_4$,
5. *step*:$\boldsymbol{e} = clean\_up(\tilde{\boldsymbol{t}}_4)$, $\quad \boldsymbol{t}_5 := \boldsymbol{t}_4 - \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d} \otimes \boldsymbol{e}$,
   $\tilde{\boldsymbol{t}}_5 := (\boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d} \otimes \boldsymbol{e})^* \otimes \boldsymbol{t}_5$,
6. *step*:$\boldsymbol{f} = clean\_up(\tilde{\boldsymbol{t}}_5)$.

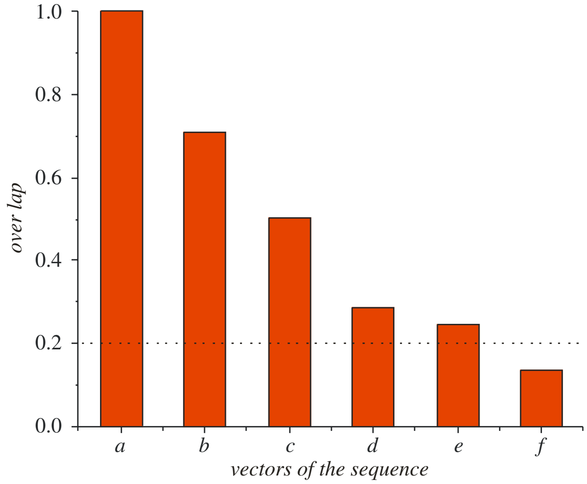

Fig. 5. The cosines for single vectors from the sequence (22), which were obtained by the reconstruction from the vector $\boldsymbol{t}_0$ (see (23)). The figure shows a fast degradation of the quality of reconstruction, already the sixths vector *f* is reconstructed with a probability smaller than 0.20.

The function *clean_up*(·) carries out the clean up process for the given vector $\boldsymbol{t}$ with respect to the set of vectors $X = \{\boldsymbol{a}, \boldsymbol{b}, \ldots, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{h}, \ldots\}$. The single steps of the reconstruction of the sequence of conceptual vectors – symbols are shown in Fig. 5, from which follows, that the process of the reconstruction of a sequence of symbols rather quickly degrades, already for the sixth vector the cosine is smaller than 0.2. When all further possible vectors in the sequence during the reconstruction have a subthreshold cosine value, the clean-up procedure is finished, we assume, that the sequence is complete. Modifications of the HRR encoding and decoding processes can lessen the degradation for long sequences [19].

5. Coding of relations

Holographic reduced representation can serve also as a suitable means for encoding relations (predicates). Let us study a binary relation $P(x,y)$, when the Pascal code is used, this relation is formally specified by the head

$$function\ \ P(x : type_1;\ \ y : type_2) : type_3 \qquad (24)$$

The single arguments of the relation are specified by the types $type_1$ and $type_2$, which specify the domain of conceptual space (i.e., a set of symbols representing conceptual items), over which are these variables defined; similarly also the relation P itself is understood as a function, which set of values is specified by the $type_3$. In many cases the domain of all variables is postulated beforehand; therefore their specifications can be omitted, which substantially reduces the holographic representation of relations. The reduced form of relation (24) looks as follows

$$function\ \ P(x\,; y) \qquad (25)$$

where we know in advance the type of variables x, y, and also the type of the relation P itself. The holographic representation of relation (24) can have the following form:

$$\boldsymbol{t} = \boldsymbol{P} + \boldsymbol{variable}_1 + \boldsymbol{variable}_2 + \boldsymbol{P} \otimes \left(\boldsymbol{type}_3 + \boldsymbol{variable}_1 \otimes \left(\boldsymbol{x} + \boldsymbol{type}_1\right) + \boldsymbol{variable}_2 \otimes \left(\boldsymbol{y} + \boldsymbol{type}_2\right)\right), \tag{26}$$

where the vectors ***variable***$_1$ and ***variable***$_2$ are used to distinguish the arguments in (25). Decoding of the representations is carried out step-by-step. In the first step we use the clean up procedure to recognize the relation ***P*** and also its variables ***x*** and ***y***. In the second step we identify the type ***type***$_3$ of the relation ***P***, in the last, third step we use previous results to identify variables ***x***, ***y*** and also their types ***type***$_1$ and ***type***$_2$. In many cases the representation of the relation $P(x,y)$ is satisfactory in the following simplified form (see (25)):

$$\boldsymbol{t} = \boldsymbol{P} + \boldsymbol{variable}_1 \otimes \boldsymbol{x} + \boldsymbol{variable}_2 \otimes \boldsymbol{y}. \tag{27}$$

The chosen method of the holographic representation of relation can be easily generalized also for more complex (higher order) relations, where the variables are predicates as well, e.g. $P(x, Q(y, z))$, where the "inner" predicate Q is characterized by

$$\mathit{function}\ Q(y : \mathit{type}_3;\ z : \mathit{type}_4) : \mathit{type}_5. \tag{28}$$

In order to create a higher order relation $P(x, Q(y, z))$, we must presume a type compatibility of the second variable of the relation P and of the type of relation Q, i.e. $type_2 = type_5$. In the simplified approach, where all the types are the same, it is not necessary to distinguish the types of single variables and the relations themselves. A simplified holographic representation of relation (28) has the following form:

$$\boldsymbol{t}' = \boldsymbol{Q} + \boldsymbol{variable}_1 \otimes \boldsymbol{y} + \boldsymbol{variable}_2 \otimes \boldsymbol{z}. \tag{29}$$

By exchanging the representation (29) for the variable ***y*** in the representation (27) we get the following resulting representation of the higher order relation $P(x, Q(y, z))$

$$\begin{aligned} \boldsymbol{t} &= \boldsymbol{P} + \boldsymbol{variable}_1 \otimes \boldsymbol{x} \\ &\quad + \boldsymbol{variable}_2 \otimes (\boldsymbol{Q} + \boldsymbol{variable}_1 \otimes \boldsymbol{y} + \boldsymbol{variable}_2 \otimes \boldsymbol{z}) \\ &= \boldsymbol{P} + \boldsymbol{variable}_1 \otimes \boldsymbol{x} + \boldsymbol{variable}_2 \otimes \boldsymbol{Q} + \\ &\quad + \boldsymbol{variable}_1 \otimes \boldsymbol{variable}_2 \otimes \boldsymbol{y} \\ &\quad + \boldsymbol{variable}_2 \otimes \boldsymbol{variable}_2 \otimes \boldsymbol{z}. \end{aligned} \tag{30}$$

5.1. Illustrative example – a similarity between geometric figures

In Fig. 6 there are presented $48 = 6 \times 8$ geometric patterns, which contain either in horizontal or in vertical settings two objects, which moreover can be of two sizes, small and big. Let us mark holographic representations of corresponding atomic concepts as follows:

Objects: ***tr*** (triangle), ***sq*** (square), ***ci***(circle), ***st*** (star)
Unary relations: ***sm*** (small), ***lg*** (large)
Binary relations: ***hor*** (horizontal), ***ver*** (vertical)
Variables: ***ver_var₁*** (1st variable for binary relation ***ver***), ***ver_var₂*** (2nd variable for binary relation ***ver***), ***hor_var₁*** (1st variable for binary relation ***hor***), ***hor_var₂*** (2nd variable for binary relation ***hor***)

Single figures from Fig. 6 are characterized by relations given in Table 1.

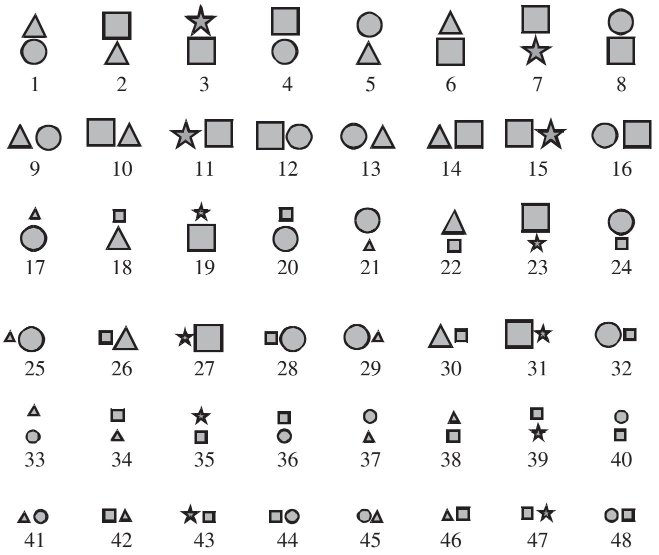

Fig. 6. A set of 48 similar figures, each of which contains two objects. The objects are placed either next to each other, or on top of each other and the objects are either small or big.

Table 1
Relations characterizing figures from Fig. 6

Row	Specification
1	***ver***(***lg***(***x***),***lg***(***y***))
2	***hor***(***lg***(***x***),***lg***(***y***))
3	***ver***(***sm***(***x***),***lg***(***y***)) and ***ver***(***lg***(***x***),***sm***(***y***))
4	***hor***(***sm***(***x***),***lg***(***y***)) and ***hor***(***lg***(***x***),***sm***(***y***))
5	***ver***(***sm***(***x***),***sm***(***y***))
6	***hor***(***sm***(***x***),***sm***(***y***))

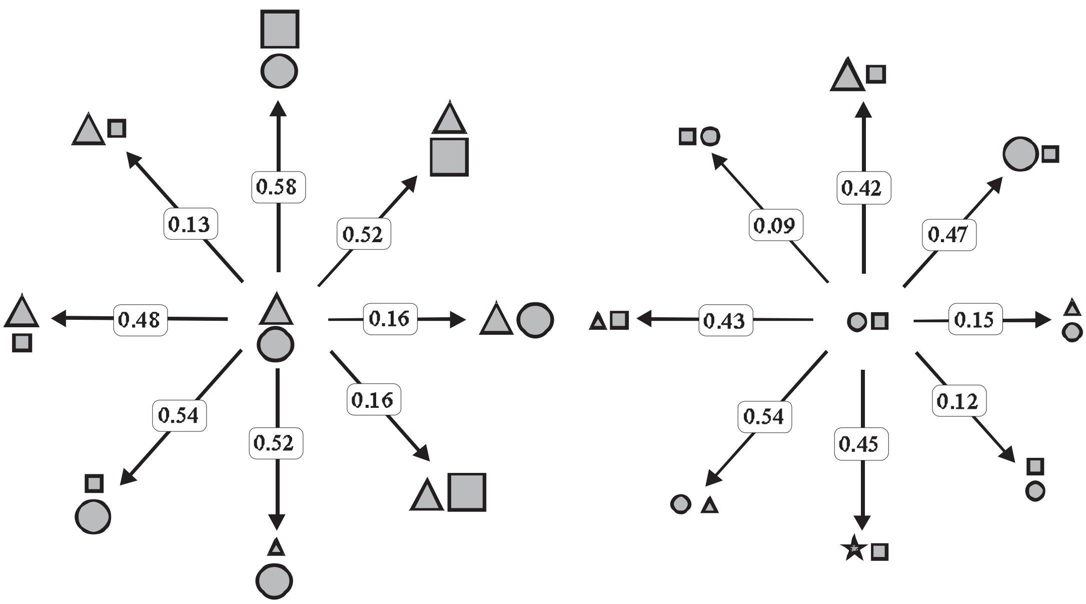

Fig. 7. Illustrative presentation of similar figures for two chosen figures 1 and 48 (see Fig. 6). Single arrows are marked by the cosine between vectors representing the figures calculated by formula (32).

Holographic representations of single cases from this table have the following form (compare with Eq. (27)).

$$\begin{aligned}
\boldsymbol{t}_{1,x,y} &= \boldsymbol{ver} + \left\langle \boldsymbol{ver\_var}_1 \otimes \boldsymbol{lg} \otimes \boldsymbol{x} + \boldsymbol{ver\_var}_2 \otimes \boldsymbol{lg} \otimes \boldsymbol{y} \right\rangle,\\
\boldsymbol{t}_{2,x,y} &= \boldsymbol{hor} + \left\langle \boldsymbol{hor\_var}_1 \otimes \boldsymbol{lg} \otimes \boldsymbol{x} + \boldsymbol{hor\_var}_2 \otimes \boldsymbol{lg} \otimes \boldsymbol{y} \right\rangle,\\
\boldsymbol{t}_{3,x,y} &= \begin{cases} \boldsymbol{ver} + \left\langle \boldsymbol{ver\_var}_1 \otimes \boldsymbol{lg} \otimes \boldsymbol{x} + \boldsymbol{ver\_var}_2 \otimes \boldsymbol{sm} \otimes \boldsymbol{y} \right\rangle,\\ \boldsymbol{ver} + \left\langle \boldsymbol{ver\_var}_1 \otimes \boldsymbol{sm} \otimes \boldsymbol{x} + \boldsymbol{ver\_var}_2 \otimes \boldsymbol{lg} \otimes \boldsymbol{y} \right\rangle, \end{cases}\\
\boldsymbol{t}_{4,x,y} &= \begin{cases} \boldsymbol{hor} + \left\langle \boldsymbol{hor\_var}_1 \otimes \boldsymbol{lg} \otimes \boldsymbol{x} + \boldsymbol{hor\_var}_2 \otimes \boldsymbol{sm} \otimes \boldsymbol{y} \right\rangle,\\ \boldsymbol{hor} + \left\langle \boldsymbol{hor\_var}_1 \otimes \boldsymbol{sm} \otimes \boldsymbol{x} + \boldsymbol{hor\_var}_2 \otimes \boldsymbol{lg} \otimes \boldsymbol{y} \right\rangle, \end{cases}\\
\boldsymbol{t}_{5,x,y} &= \boldsymbol{ver} + \left\langle \boldsymbol{ver\_var}_1 \otimes \boldsymbol{sm} \otimes \boldsymbol{x} + \boldsymbol{ver\_var}_2 \otimes \boldsymbol{sm} \otimes \boldsymbol{y} \right\rangle,\\
\boldsymbol{t}_{6,x,y} &= \boldsymbol{hor} + \left\langle \boldsymbol{hor\_var}_1 \otimes \boldsymbol{sm} \otimes \boldsymbol{x} + \boldsymbol{hor\_var}_2 \otimes \boldsymbol{sm} \otimes \boldsymbol{y} \right\rangle,
\end{aligned} \tag{31}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are holographic representations of single objects (***tr***, ***sq***, ***ci***, ***st***) and the bracket $\langle \boldsymbol{u} \rangle$ indicates, that the vector $\boldsymbol{u}$ is normalized. The similarity between single figures is determined by the cosines of their holographic representations

$$similarity(X, X') = \cos(\boldsymbol{t}, \boldsymbol{t}'). \tag{32}$$

The obtained results are shown in Fig. 7. The dominant feature controlling similarity value is the horizontal or vertical arrangement of objects. The cosines between vectors (i.e. also the similarity) between two figures, which have different arrangement is usually smaller than 0.2.

In general, holographic reduced representation allows fairly simple determination of similarity of objects specified by a predicate structure (24) or by its generalization through further nested predicates (see (30)). This possibility opens new horizons for future developments in fundamental methods of finding similar objects or analogies. Similarity search is very difficult for artificial intelligence requiring special symbolic techniques [19].

6. Reasoning by modus ponens and modus tollens

Simulation of reasoning processes (inference) belongs to the basic problems, which are repeatedly solved in artificial intelligence and cognitive science [24]. Fodor's and Pylyshin's critique of connectionism [2] was based on the conclusion, that artificial neural networks can not adequately simulate the property of systematicity [5,20,28], which is an integral part of higher cognitive activities. The term systematicity means, that the ability to represent some states of affairs ("John loves Mary") is closely linked to the ability to represent other states of affairs (e.g., "Mary loves John"). The critique was certainly correct at the time they wrote it, but they also claimed, that even if a connectionist system would simulate systematicity, then it would be a mere implementation of a symbolic system, and hence uninteresting.

Further development of neural network theory showed, that connectionism is a universal computational tool, which does not have limits of applicability. In some domains the application of connectionism is cumbersome and complicated and other approaches for the given domain provide substantially simpler and more direct solution. The discussion concerning the Fodor and Pylyshin critique still continues; however, new approaches should at least demonstrate that they are able to simulate basic processes of logical thinking.

Similarly as for neural networks, in our paper we intend to show, that also holographic reduced representation might be used to simulate higher cognitive processes. We also want to demonstrate, that at least for simple cases a special representation of implication can link together basic logical operations of modus ponens and modus tollens. The result of these operations can be obtained from convolution of two vectors, the first representing an input (antecedent, resp. inverse of consequent) and the second representing the implication. We believe that the effortless use of one vector for two logical operations brings HRR closer to the property of systematicity.

A holistic transformation of hierarchical structures containing logical expressions with operations $\wedge, \vee$, and $\Rightarrow$ represented by HRR was already studied by Neumann [14–16], who shown, that a gradient descent technique can be used to learn such transformation. However, she did not use modus ponens or modus tollens. A possibility to represent rules of deductive logic was shortly discussed also by Plate [19].

In this section we shall show a possibility of representation of two basic modes of deductive reasoning of propositional logic,

$$\begin{array}{l} p \Rightarrow q \\ \underline{p} \\ q \end{array} \quad \text{and} \quad \begin{array}{l} p \Rightarrow q \\ \underline{\bar{q}} \\ \bar{p} \end{array} \tag{33}$$

which are called *modus ponens* resp. *modus tollens*. These modes of reasoning are equivalent to the following tautologies of the predicate logic

$$((p \Rightarrow q) \wedge p) \Rightarrow q, \tag{34a}$$

$$((p \Rightarrow q) \wedge \bar{q}) \Rightarrow \bar{p}. \tag{34b}$$

Implication can be understood as a binary relation, which can be represented in holographic distribution like this (see formula (29)):

$$\boldsymbol{t}_{p \Rightarrow q} = \boldsymbol{op} \otimes \boldsymbol{impl} + \boldsymbol{var}_1 \otimes \boldsymbol{p} + \boldsymbol{var}_2 \otimes \boldsymbol{q} \tag{35}$$

The formula contains a sum of three parts, the first part specifies the type of relation (implication), the second and third parts specify the first (antecedent) resp. the second (consequent) variable of the relation of implication.

However, the "classical" form (35) of the distributed representation of the implication operator is rather cumbersome from the application point of view. To extract the result of the implication from this representation would not be straightforward. We need a representation, which would give us the result of the implication after a simple operation. It means, that we need such a representation of implication $p \Rightarrow q$, which would produce q after convolution with p, and the same representation should produce $\bar{\boldsymbol{p}}$ after convolution with $\bar{\boldsymbol{q}}$. Therefore, we introduce a specially transformed representation of implication $\tilde{\boldsymbol{t}}_{p \Rightarrow q}$, which satisfies these requirements.

The next few paragraphs up to Eqs. (38a,b) describe a way how to apply *modus ponens*, reps. *modus tollens* by using an implication represented by $\tilde{\boldsymbol{t}}_{p \Rightarrow q}$ and antecedent p, resp. $\tilde{\boldsymbol{t}}_{p \Rightarrow q}$ and consequent $\bar{\boldsymbol{q}}$.

The conceptual vector representing a particular implication relation can be transformed as follows:

$$\tilde{\boldsymbol{t}}_{p \Rightarrow q} = \boldsymbol{t}_{p \Rightarrow q} \otimes \boldsymbol{T}, \tag{36a}$$

where

$$\boldsymbol{T} = \boldsymbol{var}_1^* \otimes \boldsymbol{p}^* \otimes \boldsymbol{p}^* \otimes \mathbf{q} + \boldsymbol{var}_2^* \otimes \boldsymbol{q}^* \otimes \bar{\boldsymbol{q}}^* \otimes \bar{\boldsymbol{p}}. \tag{36b}$$

An analogue of the mapping vector $\boldsymbol{T}$ was initially used by Kanerva [8] in his theory of learning from examples, its main advantage consists in its ability to incorporate several substitutions at once. In particular, it transforms the original implication vector $\boldsymbol{t}_{p \Rightarrow q}$ onto a new implication vector $\tilde{\boldsymbol{t}}_{p \Rightarrow q}$, which is more appropriate for simulations of *modus ponens* and *modus tollens* rules. The vectors $\bar{\boldsymbol{p}}$ and $\bar{\boldsymbol{q}}$ are randomly generated vectors representing negations of propositions p and q, respectively.

The construction of $\tilde{\boldsymbol{t}}_{p \Rightarrow q}$, described by Eq. (36), from the original representation of $\boldsymbol{t}_{p \Rightarrow q}$ in (35) is rather complicated. However, we do not need to proceed from $\boldsymbol{t}_{p \Rightarrow q}$ as in (36) to construct a representation of a particular representation of $\tilde{\boldsymbol{t}}_{p \Rightarrow q}$, we can directly use further described formula (37) derived from Eqs. (36).

There are two important components in the transformed representation of implication:

$$\tilde{\boldsymbol{t}}_{p \Rightarrow q} \approx \boldsymbol{p}^* \otimes \boldsymbol{q} + \bar{\boldsymbol{q}}^* \otimes \bar{\boldsymbol{p}} \tag{37}$$

which gives the holographic representation of the rules *modus ponens* and *modus tollens*. This new representation is a consequence of Eqs. (35) and (36a),(36b), it is an approximate version of $\tilde{\boldsymbol{t}}_{p \Rightarrow q}$ defined originally by (36a),(36b). The vector $\tilde{\boldsymbol{t}}_{p \Rightarrow q}$ is an encoded form of the implication $p \Rightarrow q$ that allows the logical inference *modus ponens* and *modus tollens* to be computed via simple vector operations,

$$\boldsymbol{p} \otimes \tilde{\mathbf{t}}_{p \Rightarrow q} \approx \boldsymbol{q}, \tag{38a}$$

$$\bar{\boldsymbol{q}} \otimes \tilde{\boldsymbol{t}}_{p \Rightarrow q} \approx \bar{\boldsymbol{p}}. \tag{38b}$$

The first formula (38a) can be understood as a holographic representation of modus ponens (see (33) and (34a)), while the other formula (38b) is a holographic representation of modus tollens (see (33) and (34b)).

The pseudo-code for an algorithm implementing the implication for distributed representation is given in Algorithm 1, where we use the approximate version of $\tilde{\boldsymbol{t}}_{p \Rightarrow q}$ defined by (37) instead of its exact, but cumbersome definition in (36a),(36b).

```
procedure Construction_of_implication (input : conceptual vectors $\boldsymbol{p}$, $\boldsymbol{q}$;
                                        output : $\tilde{\boldsymbol{t}}_{p \Rightarrow q} := \boldsymbol{p}^* \otimes \boldsymbol{q} + \bar{\boldsymbol{q}}^* \otimes \bar{\boldsymbol{p}}$);
procedure Implication (input : conceptual vectors implication $\tilde{\boldsymbol{t}}_{p \Rightarrow q}, \boldsymbol{x}$;
                       output : $\boldsymbol{x} \otimes \tilde{\boldsymbol{t}}_{p \Rightarrow q}$);
```

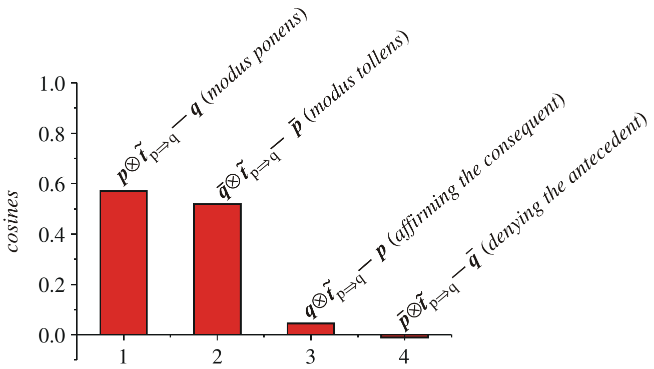

Fig. 8. Four different cosines for modus ponens, modus tollens and for two fallacies. Conceptual vectors dimension is $n = 1000$, the displayed results are for one typical trial.

Algorithm 1. A simplified version of implication procedure using implicitly vector $\tilde{\boldsymbol{t}}_{p\Rightarrow q}$ for modus ponens and tollens. Procedure convolution $\otimes$ is defined by Eqs. (3)–(5), procedure inverse $(\ )^*$ is defined by Eqs. (7),(8). The vector $\boldsymbol{x}$ in the procedure Implication can be substituted by $\boldsymbol{p}$, with a return value approximately equal to $\boldsymbol{q}$, as well as by $\bar{\boldsymbol{q}}$, with a return value approximately equal to $\bar{\boldsymbol{p}}$.

If we allow a formal modification of (38a),(38b) in such a way that the vector $\tilde{\boldsymbol{t}}_{p\Rightarrow q}$ is multiplied by $\bar{\boldsymbol{p}}^*$ or $\boldsymbol{q}^*$, then we shall call the results $\hat{\boldsymbol{p}}$, resp. $\hat{\bar{\boldsymbol{q}}}$, see (38c),(38d). The hats on the symbols in (38c),(38d) are used to distinguish the results of multiplication from the vectors $\boldsymbol{p}$ and $\bar{\boldsymbol{q}}$. If the values of vectors with hats were similar to the values of vectors without hats, then we obtained the logically incorrect results which correspond to reasoning fallacies known as "*affirming the consequent*" and "*denying the antecedent*", respectively,

$$\boldsymbol{q}^* \otimes \tilde{\boldsymbol{t}}_{p\Rightarrow q} \approx \hat{\boldsymbol{p}}, \tag{38c}$$

$$\bar{\boldsymbol{p}}^* \otimes \tilde{\boldsymbol{t}}_{p\Rightarrow q} \approx \hat{\bar{\boldsymbol{q}}}. \tag{38d}$$

In fact, the fallacies typical for human reasoning do not appear in this model, the vector $\boldsymbol{p}$ differs much from $\hat{\boldsymbol{p}}$, as well as the vector $\bar{\boldsymbol{q}}$ differs from $\hat{\bar{\boldsymbol{q}}}$. In Fig. 8 we show that the cosines between $\boldsymbol{p} \otimes \tilde{\boldsymbol{t}}_{p\Rightarrow q}$ and $\boldsymbol{q}$ and also between $\bar{\boldsymbol{q}} \otimes \tilde{\boldsymbol{t}}_{p\Rightarrow q}$ and $\bar{\boldsymbol{p}}$ are much greater than fallacious cosines between $\boldsymbol{q} \otimes \tilde{\boldsymbol{t}}_{p\Rightarrow q}$ and **p**, and between $\bar{\boldsymbol{p}} \otimes \tilde{\boldsymbol{t}}_{p\Rightarrow q}$ and $\bar{\boldsymbol{q}}$, respectively.

7. Predicate logic

We shall further deal with a simple form of predicate logic, which is based on predicate symbols $P(x)$, $Q(x,y)$, and $R(x,y,z)$, where their distributed representations have forms (see Section 6)

$$\boldsymbol{t}_{P(x)} = \boldsymbol{pred} \otimes \boldsymbol{P} + \boldsymbol{pred\_var}_1 \otimes \boldsymbol{x}, \tag{39a}$$

$$\boldsymbol{t}_{Q(x,y)} = \boldsymbol{pred} \otimes \boldsymbol{Q} + \boldsymbol{pred\_var}_1 \otimes \boldsymbol{x} + \boldsymbol{pred\_var}_2 \otimes \boldsymbol{y}, \tag{39b}$$

$$\boldsymbol{t}_{R(x,y,z)} = \boldsymbol{pred} \otimes \boldsymbol{R} + \boldsymbol{pred\_var}_1 \otimes \boldsymbol{x} + \boldsymbol{pred\_var}_2 \otimes \boldsymbol{y} + \boldsymbol{pred\_var}_3 \otimes \boldsymbol{z}. \tag{39c}$$

Of course, in standard predicate logic, these predicates may be accompanied by quantifiers. For example, a universally quantified predicate $(\forall x)P(x)$ can be represented in the following way:

$$\boldsymbol{t}_{(\forall x)} = \boldsymbol{uni\_quant} \otimes \boldsymbol{uni} + \boldsymbol{uni\_quant\_var} \otimes \boldsymbol{x}, \tag{40a}$$

$$\boldsymbol{t}_{(\forall x)P(x)} = \boldsymbol{t}_{(\forall x)} + \boldsymbol{t}_{(\forall x)} \otimes \boldsymbol{t}_{P(x)} \tag{40b}$$

All conceptual vectors $\boldsymbol{t}_{P(x)}$, $\boldsymbol{t}_{Q(x,y)}$, $\boldsymbol{t}_{R(x,y,z)}$, and $\boldsymbol{t}_{(\forall x)P(x)}$ can be recognized and extracted by a decoding and clean up procedure described in Section 2.

The above process is slightly complicated for further study of reasoning processes based on predicate logic and using distributed representation; the application of the conceptual vector $\boldsymbol{t}_{(\forall x)}$ for the representation of the symbol $(\forall x)$ basically only unnecessarily complicates the analysis of composed conceptual vectors containing $\boldsymbol{t}_{(\forall x)}$. We shall therefore stop using the symbol $(\forall x)$ explicitly, its meaning will be substituted by usage of a "*universal variable*" x. Predicate $P(x)$ containing the universal variable x is then interpreted as $(\forall x)P(x)$. We can therefore with a certain caution use a "formula" $(\forall x)P(x) \equiv P(x)$, and similarly simplify binary and ternary predicates.

In predicate logic, a rule of universal instantiation concretizes a predicate with a universal quantifier onto a predicate with a concrete value a, $(\forall x)P(x) \Rightarrow P(a)$, which is a result of a simple tautology of propositional logic $p \wedge q \Rightarrow q$. Both variables and their concrete values are represented by conceptual vectors. With the substitution $(\forall x)P(x) \equiv P(x)$ we shall rewrite this concretization into a simpler form

$$P(x) \Rightarrow P(a), \tag{41}$$

We shall construct a distributed representation of this universal instantiation of a simple unary predicate by a transformation vector $\boldsymbol{T}$, in a similar manner to Eqs. (36)–(38). When the conceptual vector $\boldsymbol{x}$ is substituted by a conceptual vector $\boldsymbol{a}$ in $\boldsymbol{t}_{P(x)}$ specified by (39a), we arrive at the conceptual vector $\boldsymbol{t}_{P(a)}$. The distributed representation (41) then looks as follows

$$\boldsymbol{t}_{P(a)} = \boldsymbol{pred} \otimes \boldsymbol{P} + \boldsymbol{pred\_var}_1 \otimes \boldsymbol{a} \approx \boldsymbol{t}_{P(x)} \otimes \boldsymbol{T}. \tag{42}$$

If we put $\boldsymbol{T} = \boldsymbol{x}^* \otimes \boldsymbol{a}$, then the conceptual vector $\boldsymbol{t}_{P(a)}$ is approximately equal to

$$\boldsymbol{t}_{P(a)} \approx \boldsymbol{t}_{P(x)} \otimes \boldsymbol{x}^* \otimes \boldsymbol{a} \tag{43}$$

We see that a distributed representation of the universal instantiation can be concretized by a transformation vector $\boldsymbol{T}$, which substitutes a universal variable $\boldsymbol{x}$ by a particular value $\boldsymbol{a}$. A similar approach was initially used by Plate [13].

We shall use this simplified representation of quantified predicates to study so called *generalized modus ponens* and *generalized modus tollens* in a simplified form using implied universal quantification of x

$$\begin{array}{lcl} P(x) \Rightarrow Q(x) & & P(x) \Rightarrow Q(x) \\ \underline{P(a)} & \text{and} & \underline{\bar{Q}(a)} \\ Q(a) & & \bar{P}(a) \end{array} \quad . \tag{44}$$

These generalized schemes of deductive reasoning are isomorphic to the corresponding propositional schemes (33). The distributed representation of the main (top) premise of these rules has a form

$$\boldsymbol{t}_{P(x)\Rightarrow Q(x)} = \boldsymbol{op} \otimes \boldsymbol{impl} + \boldsymbol{impl\_var}_1 \otimes \boldsymbol{t}_{P(x)} + \boldsymbol{impl\_var}_2 \otimes \boldsymbol{t}_{Q(x)}. \tag{45}$$

The concretization of implication $P(x) \Rightarrow Q(x)$ onto $P(a) \Rightarrow Q(a)$, can be formally expressed by an implication (see (41))

$$(P(x) \Rightarrow Q(x)) \Rightarrow (P(a) \Rightarrow Q(a)), \tag{46}$$

where the right hand side has the following distributed representation:

$$\begin{aligned} \boldsymbol{t}_{P(a)\Rightarrow Q(a)} &= \boldsymbol{op} \otimes \boldsymbol{impl} + \boldsymbol{impl\_var}_1 \otimes \boldsymbol{t}_{P(a)} + \boldsymbol{impl\_var}_2 \otimes \boldsymbol{t}_{Q(a)} \\ &\approx \boldsymbol{t}_{P(x)\Rightarrow Q(x)} \otimes \boldsymbol{x}^* \otimes \boldsymbol{a}. \end{aligned} \tag{47}$$

Similarly as in the previous section (see (36)–(38)), this concretization of an implication (47) can be expressed by using a transformation operator $\tilde{T}$ in vector form as follows:

$$\tilde{\boldsymbol{t}}_{P(x)\Rightarrow Q(x)} = \boldsymbol{t}_{P(x)\Rightarrow Q(x)} \otimes \tilde{\boldsymbol{T}}, \tag{48}$$

where the new transformed distributed representation $\tilde{\boldsymbol{t}}_{P(x)\Rightarrow Q(x)}$ satisfies the formulas, which represent the rules (44) modus ponens and modus tollens

$$\boldsymbol{t}_{P(a)} \otimes \tilde{\boldsymbol{t}}_{P(x)\Rightarrow Q(x)} \approx \boldsymbol{t}_{Q(a)}, \tag{49a}$$

$$\boldsymbol{t}_{\bar{Q}(a)} \otimes \tilde{\boldsymbol{t}}_{P(x)\Rightarrow Q(x)} \approx \boldsymbol{t}_{\bar{P}(a)}, \tag{49b}$$

where $\boldsymbol{t}_{\bar{P}(a)}$ and $\boldsymbol{t}_{\bar{Q}(a)}$ are conceptual vectors assigned to negated predicates $\neg P(x)$ and $\neg Q(x)$, respectively, concretized for $x = a$. These predicates have a conceptual vector representation as follows:

$$\begin{aligned} &\boldsymbol{t}_{\bar{P}(x)} = \boldsymbol{pred} \otimes \bar{\boldsymbol{P}} + \boldsymbol{pred\_var}_1 \otimes \boldsymbol{x},\ \boldsymbol{t}_{\bar{P}(a)} \\ &\boldsymbol{t}_{\bar{P}(a)} = \boldsymbol{pred} \otimes \bar{\boldsymbol{P}} + \boldsymbol{pred\_var}_1 \otimes \boldsymbol{a}, \end{aligned} \tag{50a}$$

$$\begin{aligned} &\boldsymbol{t}_{\bar{Q}(x)} = \boldsymbol{pred} \otimes \bar{\boldsymbol{Q}} + \boldsymbol{pred\_var}_1 \otimes \boldsymbol{x},\ \boldsymbol{t}_{\bar{Q}(a)} \\ &\boldsymbol{t}_{\bar{Q}(a)} = \boldsymbol{pred} \otimes \bar{\boldsymbol{Q}} + \boldsymbol{pred\_var}_1 \otimes \boldsymbol{a}, \end{aligned} \tag{50b}$$

where $\bar{\boldsymbol{P}}$ and $\bar{\boldsymbol{Q}}$ are new randomly generated conceptual vectors. The composed conceptual vector $\tilde{\boldsymbol{t}}_{P(x)\Rightarrow Q(x)}$ from (49) can be simply specified by

$$\tilde{\boldsymbol{t}}_{P(x)\Rightarrow Q(x)} = \boldsymbol{t}^*_{P(a)} \otimes \boldsymbol{t}_{Q(a)} + \boldsymbol{t}^*_{\bar{Q}(a)} \otimes \boldsymbol{t}_{\bar{P}(a)}. \tag{51}$$

If we introduce this expression into (48) and multiply from the left by $\boldsymbol{t}^*_{P(x)\Rightarrow Q(x)}$, we arrive at an explicit formula for the transformation operator $\tilde{\boldsymbol{T}}$.

7.1. Algorithmic description of reasoning by modus ponens and modus tollens

In this subsection we give a precise algorithmic analysis of our approach presented in the previous subsection for the distributed holographic implementation of generalized logical rules modus ponens and modus tollens (44):

Step 1: Constant conceptual vectors $\boldsymbol{pred}$, $\boldsymbol{P}$, $\boldsymbol{Q}$, $\bar{\boldsymbol{P}}$, $\bar{\boldsymbol{Q}}$, $\boldsymbol{pred\_var}_1$, $\boldsymbol{pred\_var}_2$, $\boldsymbol{x}$, $\boldsymbol{a}$, and $\boldsymbol{b}$ are randomly generated.

Step 2: Conceptual vectors $\boldsymbol{t}_{P(x)}$ and $\boldsymbol{t}_{Q(x)}$ are constructed, see Eq. (39a).

Step 3: Conceptual vector $\boldsymbol{t}_{P(x)\Rightarrow Q(x)}$ is constructed, see Eq. (45), we omitted the first term on the left-hand side as irrelevant for the present purposes.

Step 4: Construction of conceptual vectors $\boldsymbol{t}_{P(\boldsymbol{a})}$, $\boldsymbol{t}_{P(\boldsymbol{b})}$, $\boldsymbol{t}_{Q(\boldsymbol{a})}$, and $\boldsymbol{t}_{Q(\boldsymbol{b})}$ by making use of (43) or its analogs for $\boldsymbol{t}_{P(\boldsymbol{b})}$, $\boldsymbol{t}_{Q(\boldsymbol{a})}$, and $\boldsymbol{t}_{Q(\boldsymbol{b})}$, respectively. Of course, we might use immediately the equation (42) or its analogs, which would give us an exact construction. However, using an approximate equality like (43) brings us closer to a course of natural deductive process.

Step 5: Construction of conceptual vectors $\boldsymbol{t}_{\bar{P}(\boldsymbol{a})}$, $\boldsymbol{t}_{\bar{P}(\boldsymbol{b})}$, $\boldsymbol{t}_{\bar{Q}(\boldsymbol{a})}$, and $\boldsymbol{t}_{\bar{Q}(\boldsymbol{b})}$ by making use of (50).

Step 6: Construction of a transformation conceptual vector $\tilde{\boldsymbol{T}}$ from (48) such that a generalized form of (36b) is used

$$\begin{aligned} \tilde{\boldsymbol{T}} = {} & \boldsymbol{impl\_var}^*_1 \otimes \boldsymbol{t}^*_{\boldsymbol{P}(\boldsymbol{x})} \otimes \boldsymbol{t}^*_{\boldsymbol{P}(\boldsymbol{a})} \otimes \boldsymbol{t}_{\boldsymbol{Q}(\boldsymbol{a})} \\ & + \boldsymbol{impl\_var}^*_2 \otimes \boldsymbol{t}^*_{\boldsymbol{Q}(\boldsymbol{x})} \otimes \boldsymbol{t}^*_{\bar{\boldsymbol{Q}}(\boldsymbol{a})} \otimes \boldsymbol{t}_{\boldsymbol{P}(\boldsymbol{a})}, \end{aligned}$$

Step 7: Construction of $\tilde{\boldsymbol{t}}_{P(x)\Rightarrow Q(x)}$, see Eq. (48).

Step 8. Applying (49a),(49b) we construct from $\tilde{\boldsymbol{t}}_{P(x)\Rightarrow Q(x)}$ consequences specified by modus ponens and modus tollens. Results of this application may be diagrammatically visualized as in Fig. 8, where the first two columns correspond to correct rules *mp* and *mt*, whereas the third and fourth results correspond to fallacies.

7.1.1. Illustrative example – modeling of reflexive reasoning

Applying the algorithm from the previous Section 7.1 we may outline a slightly generalized case, when predicates contain more than one argument. This illustrative example demonstrates that the holographic reduced representation provides formal tools, which can be used to simulate the reasoning process based on generalized modus ponens (44). This process was widely studied by Shastri and Ajjanagadde [24] by the connectionist system called SHRUTI, which was able to simulate reflexive reasoning based on predicate logic. The process of reflexive reasoning on a formal level can be modeled by the well-known process of forward chaining, which belongs to basic deductive processes in computational logic. Similar results are achieved also by a formalism of holographic distributed representation.

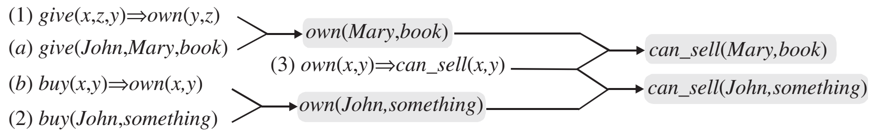

Fig. 9. Illustration of an application of the generalized rule modus ponens (44) for deduction or knowledge discovery (marked by gray shading and also by incoming arrows) from implications (1-3) and from input facts (a-b), specified further in the text and marked in the figure by outgoing arrows.

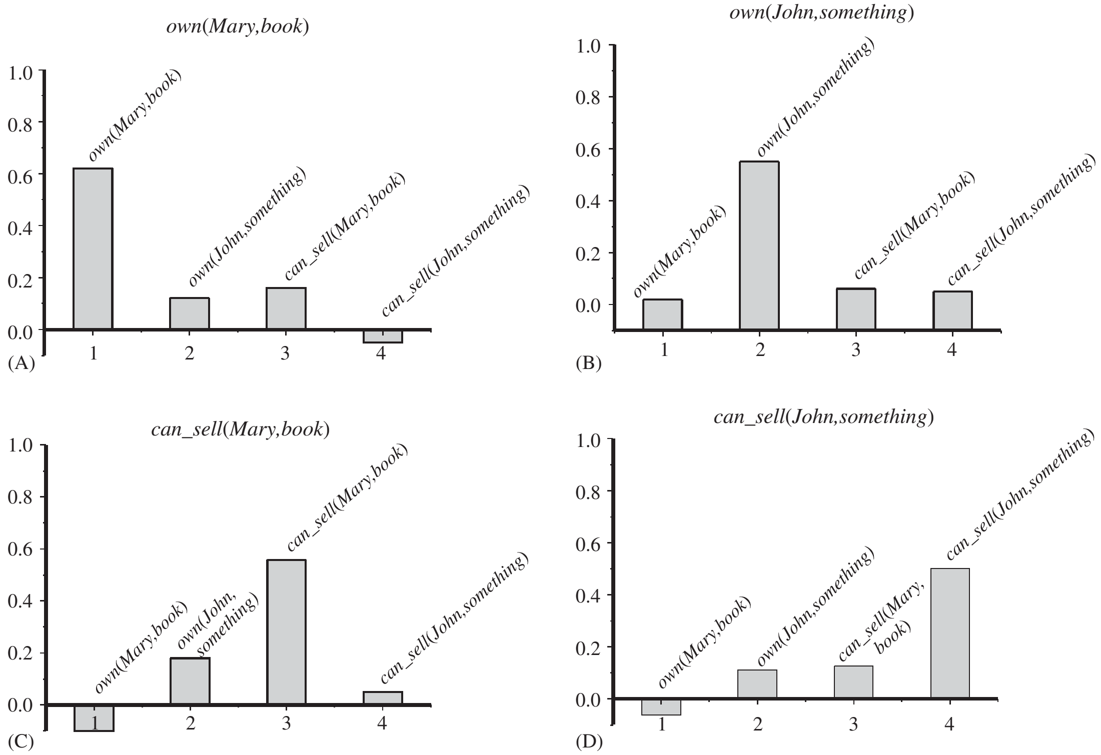

Fig. 10. Graphical representation of modus ponens for all four cases presented in Fig. 9.

Let us have a formal system containing three general rules (see Fig. 9):

(1) $give(x, y, z) \Rightarrow own(y, z)$, *type* x : donor; *type* y : acceptor; *type* z : object,
(2) $buy(y, z) \Rightarrow own(y, z)$, type y : buyer; type z : object,
(3) $own(y, z) \Rightarrow can\_sell(y, z)$, type y : owner; type z : object,

and two observations (facts)

(a) *give (John, Mary, book),*
(b) *buy (John, something).*

What are the deductive conclusions of this system? The results are shown in Fig. 9, we shall now deduce them with an application of distributed representation based on conceptual vectors and operations over them.

Let us analyze the first generalized modus ponens from Fig. 9

$$\frac{\begin{array}{c} give(x, y, z) \Rightarrow own(y, z) \\ \underline{give(John, Mary, book)} \end{array}}{own(Mary, book)}. \tag{52}$$

With an application of the approach described by (45)–(51) we can realize this scheme of reasoning by a representation of conceptual vectors, its single items (going top down) are represented as follows:

$$\boldsymbol{t}_1 = \boldsymbol{op} \otimes \boldsymbol{impl} + \boldsymbol{var}_1 \otimes \boldsymbol{t}_{give(x,y,z)} + \boldsymbol{var}_2 \otimes \boldsymbol{t}_{own(y,z)}, \tag{53a}$$

$$\boldsymbol{t}_2 = \boldsymbol{t}_{give(John,Mary,book)} = \boldsymbol{give} + \boldsymbol{give\_var}_1 \otimes \boldsymbol{John} + \boldsymbol{give\_var}_2 \otimes \boldsymbol{Mary} + \boldsymbol{give\_var}_3 \otimes \boldsymbol{book}, \tag{53b}$$

$$\boldsymbol{t}_3 = \boldsymbol{t}_{own(Mary,book)} = \boldsymbol{own} + \boldsymbol{own\_var}_1 \otimes \boldsymbol{Mary} + \boldsymbol{own\_var}_2 \otimes \boldsymbol{book} \tag{53c}$$

where conceptual vectors $\boldsymbol{t}_{give(x,y,z)}$ and $\boldsymbol{t}_{own(y,z)}$ are constructed analogically to (39). In the first step we must carry out a concretization of the implication $give(x, y, z) \Rightarrow own(y, z)$, so that the general variables x, y, z are substituted by concrete values *John*, *Mary*, *book*. This concretization is carried out by a transformation $\boldsymbol{T} = \boldsymbol{x}^* \otimes \boldsymbol{a} + \boldsymbol{y}^* \otimes \boldsymbol{b} + \cdots$, where $\boldsymbol{x}$, $\boldsymbol{y}$, ... are general objects and $\boldsymbol{a}$, $\boldsymbol{b}$, … are their concretizations (e. g. $\boldsymbol{x}$, $\boldsymbol{y}$, … and *John*, *Mary*,…)

$$\hat{\boldsymbol{t}}_1 \approx \boldsymbol{t}_1 \otimes \boldsymbol{T} \tag{54a}$$

where

$$\begin{aligned}\hat{\boldsymbol{t}}_1 &= \boldsymbol{t}_{give(John,Mary,book)\Rightarrow own(Mary,book)}\\ &= \boldsymbol{op} \otimes \boldsymbol{impl} + \boldsymbol{var}_1 \otimes \boldsymbol{t}_{give(John,Mary,book)}\\ &\quad + \boldsymbol{var}_2 \otimes \boldsymbol{t}_{own(Mary,book)}\end{aligned} \tag{54b}$$

The concretized representation $\boldsymbol{t}_{give(John,Mary,book)\Rightarrow own(Mary,book)}$ is in the next step applied for modus ponens carried out by formulas (48),(49)

$$\tilde{\boldsymbol{t}}_1 = \boldsymbol{t}^*_{give(John,Mary,book)} \otimes \boldsymbol{t}_{own(Mary,book)} \tag{55}$$

where the resulting conceptual vector $\tilde{\boldsymbol{t}}_1$ is an actual concretization of (48) for an implication $give(John, Mary, book) \Rightarrow own(Mary, book)$; it already represents modus ponens, i.e.

$$\boldsymbol{t}_{give(John,Mary,book)} \otimes \tilde{\boldsymbol{t}}_1 \approx \boldsymbol{t}_{own(Mary,book)} \tag{56}$$

The other three instances of generalized modus ponens from Fig. 9 can be realized in a similar way. Numerical results for all four modus ponens in Fig. 9 are presented in Fig. 10, where are displayed cosines between results presented by (56) and expected vectors highlighted in Fig. 9. For instance, by the application of relation (56) we arrived at a vector $\boldsymbol{t}_{give(John,Mary,book)} \otimes \tilde{\boldsymbol{t}}_1 = \boldsymbol{t}'_{own(Mary,book)}$, which is similar to $\boldsymbol{t}_{own(Mary,book)}$. Diagram A contains all four cosines between this vector $\boldsymbol{t}'_{own(Mary,book)}$ and all four expected results represented by $\boldsymbol{t}_{own(Mary,book)}$, $\boldsymbol{t}_{own(John,something)}$, $\boldsymbol{t}_{can\_sell(Mary,book)}$, and $\boldsymbol{t}_{can\_sell(John,something)}$.

8. Conclusions

A suitable distributive coding of structured information (sequence of symbols, nested relational structures, etc.) is very important in artificial intelligence and cognitive science. Holographic reduced representation offers new unconventional approach to such a coding. The used distributed representation is based on two binary operations: unary operation "involution" and binary operation "convolution" over a domain of n-dimensional randomly generated conceptual vectors, which elements satisfy normal distribution $N(0,1/n)$. Application of this distributed representation allows us to model various types of associative memory, which are represented by a conceptual vector. It also allows us to decode a memory vector, i.e. to determine the conceptual (atomic) vectors it is composed of. Such an analysis of the memory vector is carried out by a clean-up procedure that determines from the cosine of the angle between the vectors, which of the vectors is the most similar to the memory vector. Holographic reduced representation allows measuring a similarity between two structured concepts by a simple algebraic operation — scalar product of their distributed representations. This product can be very useful for modeling of processes, which search through memory to find its similar (analogical) single components. Of course, the procedure, as well as any procedure based on randomly generated vectors, may occasionally fail to provide the correct answer, but this possibility decreases with the size of the used vectors.

The main contribution of this paper is to outline a possibility, how to use the holographic reduced representation to model an inference process based on the rules *modus ponens* and *modus tollens* in propositional calculus as well as in predicate logic. This possibility is very important for a proper assessment of distributed representation (including neural networks) in artificial intelligence and cognitive science. It enables to implement higher cognitive activities, which was retained in literature mainly by symbolic localist representations.

Acknowledgments

This work was supported by the grants APVT-20-002504 and # 1/1047/04 of the Scientific Grant Agency of Slovak Republic. Special thanks go to the anonymous referees for comments and suggestions that helped us considerably improve the paper.

References

[1] A. Borsellino, T. Poggio, Convolution and correlation algebras, Kybernetik 13 (1973) 113–122.

[2] J.A. Fodor, Z.W. Pylyshyn, Connectionism and cognitive architecture: a critical analysis, Cognition 28 (1988) 3–71.

[3] D. Gabor, Holographic model for temporal recall, Nature 217 (1968) 1288–1289.

[4] P. Gärdenfors, Conceptual Spaces: The Geometry of Thought, MIT Press, Cambridge, MA, 2000.

[5] R.F. Hadley, A. Rotaru-Varga, D.V. Arnold, V.C. Cardei, Syntactic systematicity arising from semantic predictions in a Hebbian-competitive network, Connection Science 13 (2001) 73–94.

[6] P. Kanerva, The spatter code for encoding concepts at many levels, in: M. Marinaro, P.G. Morasso (Eds.), ICANN '94, Proceedings of the International Conference on Artificial Neural Networks, Springer, London, 1994, pp. 226–229.

[7] P. Kanerva, Binary spatter-coding of ordered K-tuples, in: C. von der Malsburg, et al. (Eds.), ICANN'96, Proceedings of the International Conference on Artificial Neural Networks, Springer, Berlin, 1996, pp. 869–873.

[8] P. Kanerva, Analogy as a basis of computation, in: Y. Uesaka, P. Kanerva, H. Asoh (Eds.), Foundations of Real-World Intelligence, CSLI Publications, Stanford, CA, 2001.

[9] J.L.M. McClelland, D.E. Rumelhart, The PDP Research Group (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1,2, The MIT Press, Cambridge, MA, 1986.

[10] J. Metcalfe-Eich, Levels of processing, encoding specificity, elaboration, and charm, Psychol. Rev. 92 (1985) 1–38.

[11] B.B. Murdock, A theory for the storage and retrieval of item and associative information, Psychol. Rev. 89 (1982) 316–338.

[12] B.B. Murdock, A distributed memory model for serial-order information, Psychol. Rev. 90 (4) (1983) 316–338.

[13] B.B. Murdock, Convolution and matrix systems: a reply to pike, Psychol. Rev. 92 (1) (1985) 130–132.

[14] J. Neumann, Holistic transformation of holographic reduced representations, in: Foundations of Connectionist-Symbolic Integration, Workshop Notes, 14th European Conference on Artificial Intelligence, Berlin, August 2000.

[15] J. Neumann, Holistic processing of hierarchical structures in connectionist networks. Ph.D. Thesis, University of Edinburgh, 2001, http://www.cogsci.ed.ac.uk/~jne.

[16] J. Neumann, Learning the systematic transformation of holographic reduced representations, Cognitive Sys. Res. 3 (2002) 227–235.

[17] T.A. Plate, Distributed representations and nested compositional structure, Ph.D. Thesis, Department of Computer Science, University of Toronto, 1994.

[18] T.A. Plate, Holographic distributed representations, IEEE Trans. Neural Networks 6 (1995) 623–641.

[19] T.A. Plate, Holographic reduced representation: Distributed Representation for Cognitive Structures, CSLI Publications, Stanford, CA, 2003.

[20] J.B. Pollack, Recursive distributed representations, Artif. Intelli. 46 (1990) 77–105.

[21] W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, Numerical Recipes in C: The Art of Scientific Computing, second Ed., Cambridge University Press, Cambridge, 1993 (Chapter 12).

[22] D.A. Rachkovskij, Representation and processing of structures with binary sparse distributed codes, IEEE Trans. on Knowl. Data Eng. 13 (2) (2001) 261–276.

[23] D.A. Rachkovskij, E.M. Kussul, Binding and normalization of binary sparse distributed representations by context-dependent thinning, Neural Comput 13 (2) (2001) 411–452.

[24] L. Shastri, V. Ajjanagadde, From simple associations to systematic reasoning: connectionist representation of rules, variables, and dynamic bindings using temporal synchrony, Behav. Brain Sci. 16 (1993) 417–494.

[25] J.N. Slack, The role of distributed memory in natural language processing, in: T. O'Shea (Ed.), Advances in Artificial Intelligence: Proceedings of the Sixth European Conference on Artificial Intelligence, ECAI-84, Elsevier Science Publishers, New York, 1984.

[26] P. Smolensky, Neural and conceptual interpretations of parallel distributed processing models, in: J.L. McClelland, D.E. Rumelhart (Eds.), The PDP Research Group, Parallel distributed Processing: Explorations in the Microstructure of Cognition, vol. 1, 2, The MIT Press, Cambridge, MA, 1986, pp. 390–431.

[27] P. Smolensky, On the proper treatment of connectionism, Behav. Brain Sci. 11 (1988) 1–74.

[28] P. Smolensky, Tensor product variable binding and the representation of symbolic structures in connectionist networks, Artif. Intell. 46 (1990) 159–216.

[29] P. Smolensky, G. Legendre, Y. Miyata, Principles for an Integrated Connectionist/Symbolic Theory of Higher Cognition, technical report CU-CS-600-92, Department of Computer Science and 92-8, Institute of Cognitive Science, University of Colorado at Boulder, 1992 75 p.

[30] D.J. Willshaw, O.P. Buneman, H.C. Longuet-Higgins, Non-holographic associative memory, Nature 222 (1969) 960–962.

Vladimir Kvasnicka received a diploma degree in nuclear chemistry from Czech Technical University in Prague in 1964, and the PhD degree in chemistry from Heyrovsky Institute of Czech. Academy of Sciences in Prague in 1968. From 1973 to 2003 he held positions of assistant professor, associate professor, professor and head of Department of Mathematics, Faculty of Chemical and Food Technologies, Slovak University of Technology, Bratislava. Since 2004 he is a deputy head of Institute of Applied Informatics at Faculty of Informatics and Information Technology at the same university. His fields of research are artificial intelligence, cognitive science, neural networks, evolutionary algorithms, theoretical and mathematical chemistry.

Jiri Pospichal received a diploma degree in physical chemistry from University of Jan Evangelista Purkyne in Brno, Czech Republic in 1984, and the Ph.D. degree in chemistry from Faculty of Chemical and Food Technologies at Slovak University of Technology, Bratislava, in 1990. From 1990 to 2003, he held positions of assistant professor and associate professor of Applied Informatics at Department of Mathematics, Faculty of Chemical and Food Technologies, Slovak University of Technology, Bratislava. Since 2004 he works at Institute of Applied Informatics at Faculty of Informatics and Information Technology at the same university. His research interests are evolutionary algorithms, artificial intelligence, cognitive science, neural networks, mathematical chemistry and graph theory.