Noname manuscript No.
(will be inserted by the editor)

Analogical mapping with sparse distributed memory: a simple model that learns to generalize from examples

Blerim Emruli · Fredrik Sandin

Received: date / Accepted: date

Abstract We present a computational model for the analogical mapping of compositional structures that combines two existing ideas known as holistic mapping vectors and sparse distributed memory. The model enables integration of structural and semantic constraints when learning mappings of the type $x_i \to y_i$ and computing analogies $x_j \to y_j$ for novel inputs x_j. The model has a one-shot learning process, is randomly initialized and has three exogenous parameters: the dimensionality $\mathcal{D}$ of representations, the memory size S and the probability χ for activation of the memory. After learning three examples the model generalizes correctly to novel examples. We find minima in the probability of generalization error for certain values of χ, S and the number of different mapping examples learned. These results indicate that the optimal size of the memory scales with the number of different mapping examples learned and that the sparseness of the memory is important. The optimal dimensionality of binary representations is of the order 10^4, which is consistent with a known analytical estimate and the synapse count for most cortical neurons. We demonstrate that the model can learn analogical mappings of generic two-place relationships and we calculate the error probabilities for recall and generalization.

Keywords analogy · analogical mapping · compositional structures · cognitive modeling · distributed representations · generalization · holistic processing · one-shot learning · sparse distributed memory

B. Emruli (✉) · F. Sandin
EISLAB
Luleå University of Technology, S-97187 Luleå, Sweden
E-mail: blerim.emruli@ltu.se

F. Sandin
E-mail: fredrik.sandin@ltu.se

1 Introduction

Computers have excellent quantitative information processing mechanisms, which can be programmed to execute algorithms on numbers at a high rate. Technology is less evolved in terms of qualitative reasoning and learning by experience. In that context biology excels. Intelligent systems that interact with humans and the environment need to deal with imprecise information and learn from examples. A key function is *analogy* [14, 22, 23, 24, 38, 39], the process of using knowledge from similar experiences in the past to solve a new problem without a known solution. Analogy-making is a high-level cognitive function that is well developed in humans [7, 24]. More formally, analogical mapping is the process of mapping relations and objects from one situation (a source), x, to another (a target), y; $M : x \to y$ [7, 55]. The source is familiar or known, whereas the target is a novel composition of relations or objects that is not among the learned examples.

Present theories of analogy-making usually divide this process into three or four stages. In this work we follow [7] and describe analogy-making as a three-step process consisting of: *retrieval, mapping* and *application*. As in the previous computational models of analogy-making, see Section 1.1, this paper focuses mainly on the challenging mapping stage. We present a model that is based on a well-known mechanism for analogical mapping with high-dimensional vectors, which are commonly referred to as mapping vectors. By integrating this mechanism with a model of associative memory, known as sparse distributed memory (SDM), we obtain an improved model that can learn multiple mapping vectors. The proposed model can learn mappings $x_{k,z} \to y_{k,z}$, where k denotes different examples of one particular relationship z. If $x_{k,z}$ is in the training set

the model retrieves a mapping vector that transforms $x_{k,z}$ to an approximation of the learned $y_{k,z}$, otherwise $x_{k,z}$ is transformed to an approximate generalization (an analogue), $y_{k,z}$, of learned examples corresponding to the relationship z. The mechanism that enables the possibility to learn multiple mapping vectors that can be used for analogy making is the major contribution of this paper.

An SDM operates on high-dimensional binary vectors so that associative and episodic mappings can be learned and retrieved [2, 6, 26, 27, 33]. Such high-dimensonal vectors are useful representations of compositional structures [28, 29, 30, 42, 45]. For further details and references, see [31, 41, 44, 46]. With an appropriate choice of operators it is possible to create, combine and extract compositional structures represented by such vectors. It is also possible to create *mapping vectors* that transform one compositional structure into another compositional structure [30, 42, 45]. Such mapping vectors can generalize to structures composed of novel elements [30, 45] and to structures of higher complexity than those in the training set [42]. The idea of holistic mapping vectors and the results referenced above are interesting because they describe a simple mechanism that enables computers to integrate the structural and semantic constraints that are required to perform analogical mapping. In this paper we integrate the idea of holistic mapping vectors with the SDM model of associative memory into one "analogical mapping unit (AMU), which enables learning and application of mappings in a simple way. The ability to learn unrelated mapping examples with an associative memory is a novel feature of the AMU model. The AMU also provides an abstraction that, in principle, enables visualization of the higher-level cognitive system architecture.

In the following subsection we briefly summarize the related work on analogical mapping. In Section 2 we introduce the theoretical background and key references needed to understand and implement the AMU model. The AMU model is presented in Section 3. In Section 4 we present numerical results that characterize some properties of the AMU model. Finally, we discuss our results in the context of the related work and raise some questions for further research.

1.1 Related work

Research on computational modelling of analogy-making traces back to the classical works in the 60s by Evans [8] and Reitman [51] and since then many approaches have been developed. In this section we introduce some models that either have been influential or are closely related to the work presented here. For comprehensive surveys see [11, 15], where computational models of analogy are categorize as symbolic, connectionist, or symbolic-connectionist hybrids.

A well-known symbolic model is the Structure Mapping Engine (SME) [9], which implements a well-known theory of analogy-making called Structure Mapping Theory (SMT) [14]. With SMT the emphasis in analogy-making shifted from attributes to structural similarity between the source and target domains. The AMU, like most present models, incorporates the two major principles underlying SMT: relation-matching and systematicity (see Section 4). However, in contrast to SMT and other symbolic models, the AMU satisfies semantic constraints, thereby allowing the model to handle the problem of similar but not identical compositional structures. Satisfying semantic constraints reduces the number of potential correspondence mappings significantly, to a level that is psychologically more plausible [7, 16] and computationally feasible [15].

Connectionist models of analogy-making include the Analogical Constraint Mapping Engine (ACME) [23] and Learning and Inference with Schemas and Analogies (LISA) [25]. The Distributed Representation Analogy MApper (Drama) [7] is a more recent connectionist model that is based on holographic reduced representations (HRR). This makes Drama similar to the AMU in terms of the theoretical basis. A detailed comparison between ACME, LISA and Drama in terms of performance and neural and psychological plausibility is made by [7]. Here, we briefly summarize results in [7], add some points that were not mentioned in that work, and comment on the differences between the AMU and Drama.

ACME is sometimes referred to as a connectionist model, but it is more similar to symbolic models. Like the SME it uses localist representations, while the search mechanism for mappings is based on a connectionist approach. In ACME semantics is considered after structural constraints have been satisfied. In contrast to ACME, the AMU and Drama implement semantic and structural constraints in parallel, thereby allowing both aspects to influence the mapping process. In comparison to the previous models Drama integrates structure and semantics to a degree that is more in accordance with human cognition [7]. Semantics are not decomposable in ACME. For example, ACME would be unable to determine whether the "Dollar of Sweden" (Krona) is similar to the "Dollar of Mexico" (Peso). Whereas, that is possible with the AMU and Drama because the distributed representations allow compositional concepts to be encoded and decomposed [7, 30, 31, 42, 45, 46]. LISA is a connectionist model that is based on distributed representations and dynamic bind-

ing to associate relevant structures. Like the AMU and Drama, LISA is semantically driven, stochastic, and designed with connectionist principles. However, LISA has a complex architecture that represents propositions in working memory by dynamically binding roles to their fillers, and encoding those bindings in long-term memory. It is unclear whether this model scales well and is able to handle complex analogies [7, 15, 54].

The third category of computational models for analogy making that is the hybrid approach, where both symbolic and connectionist parts are incorporated [11, 15]. Well-known examples of such models include Copycat [40], Tabletop [10], Metacat [36] and AMBR [32].

Other related work that should be mentioned here includes models based on Recursive Auto-Associative Memory (RAAM) [47] and models based on HRR [44]. RAAM is a connectionist network architecture that uses backpropagation to learn representations of compositional structures in a fixed-length vector. HRR is a convolution-based distributed representation scheme for compositional structures. In [47] a RAAM network that maps simple propositions like (LOVED X Y) to (LOVE Y X) in a holistic way (without decomposition of the representations) is presented. In [5] a feed-forward neural network that maps reduced representations of simple passive sentences to reduced representations of active sentences using a RAAM network is presented. In [43] a feedforward network is trained to perform inference on logical relationships, for example, the mapping of reduced representations of expressions of the form $(x \rightarrow y)$ to corresponding reduced representations of the form $(\neg x \vee y)$. Similar tasks have been solved with HRR [42, 44]. In particular, Neumann replicates the results by Niklasson and van Gelder using HRR and she demonstrates that HRR mappings generalize to novel compositional structures that are more complex than those in the training set. The significance of relation-matching in human evaluation of similarity is demonstrated in [35] by asking people to evaluate the relative similarity of pairs of geometric shapes. In [44] distributed representations (HRR) for each of these pairs of geometric shapes are constructed and it is found that the similarity of the representations is consistent with the judgement by human test subjects.

From our point of view there are two issues that have limited the further development of these analogical mapping techniques. One, is the "encoding problem", see for example [4, 18]. In this particular context it is the problem of how to encode compositional structures from low-level (sensor) information without the mediation of an external interpreter. This problem includes the extraction of elementary representations of objects and events, and the construction of representations of invariant features of object and event categories. Examples demonstrating some specific technique are typically based on hand-constructed structures, which makes the step to real-world applications non trivial, because it is not known whether the success of the technique is due to the hand-crafting of the representations and there is no demonstration of the feasibility of mechanically generating the representations. The second issue is that mapping vectors are explicitly created and that there is no framework to learn and organize multiple analogical mappings. The construction of an explicit vector for each analogical mapping in former studies is excellent for the purpose of demonstrating the properties of such mappings. However, in practical applications the method needs to be generalized so that a system can learn and use multiple mappings in a simple way.

Learning of multiple mappings from examples is made possible with the AMU model that is presented in this paper. The proposed model automates the process of creating, storing and retrieving mapping vectors. The basic idea is that mapping examples are fed to an SDM so that mapping vectors are formed successively in the storage locations of the SDM. After learning one or two examples the system is able to recall, but it typically cannot generalize. With additional related mapping examples stored in the SDM the ability of the AMU to generalize to novel inputs increases, which means that the probability of generalization error decreases with the number of examples learned.

2 Background

This section introduces the theoretical background and key references needed to understand and implement the AMU model, which is presented in Section 3.

2.1 Vector symbolic architectures

The AMU is an example of a vector symbolic architecture (VSA) [13], or perhaps more appropriately named a VSA "component" in the spirit of component-based software design. In general, a VSA is based on a set of operators on high-dimensonal vectors of fixed dimensionality, so-called reduced descriptions / representations of a full concept [21]. The fixed length of vectors for all representations implies that new compositional structures can be formed from simpler structures without increasing the size of the representations, at the cost of increasing the noise level. In a VSA all representations of conceptual entities such as ontological *roles, fillers* and *relations* have the same fixed dimensionality.

These vectors are sometimes referred to as holistic vectors [29] and operations on them are a form of holistic processing [17, 42]. Reduced representations in cognitive models are essentially manipulated with two operators, named *binding* and *bundling*. Binding is similar to the idea of neural binding in that it creates a representation of the structural combination of component representations. It combines two vectors into a new vector, which is indifferent (approximately orthogonal) to the two original vectors. The defining property of the binding operation is that given the bound representation and one of the component representations it is possible to recover the other component representation. The implementation of the binding operator and the assumptions about the nature of the vector elements are model specific. The bundling operator is analogous to superposition in that it creates a representation of the simultaneous presence of multiple component representations without them being structurally combined. It typically is the algebraic sum of vectors, which may or may not be normalized. Bundling and binding are used to create compositional structures and mapping vectors. Well-known examples of VSAs are the Holographic Reduced Representation (HRR) [44, 45, 46] and the Binary Spatter Code (BSC) [28, 29, 30]. The term "holographic" in this context refers to a convolution-based binding operator, which resembles the mathematics of holography. Historical developments in this direction include the early holography-inspired models of associative memory [12, 34, 50, 56]. Note that the BSC is mathematically related to frequency-domain HRR [1], because the convolution-style operators of HRR are equivalent to element-wise operations in frequency space.

The work in VSA has been inspired by cognitive behavior in humans and the approximate structure of certain brain circuits in the cerebellum and cortex, but these models are not intended to be accurate models of neurobiology. In particular, these VSA models typically discard the temporal dynamics of neural systems and instead use sequential processing of high-dimensional representations. The recent implementation of HRR in a network of integrate-and-fire neurons [48] is, however, one example of how these cognitive models eventually may be unified with more realistic dynamical models of neural circuits.

In the following description of the AMU we use the Binary Spatter Code (BSC), because it is straightforward to store BSC representations in an SDM and it is more simple than the HRR which is based on real- or complex-valued vectors. It is not clear how to construct an associative memory that enables a similar approach with HRR, but we see no reason why that should be impossible. In a BSC, roles, fillers, relations and compositional structures, are represented by a binary vector, x_k, of dimensionality $\mathcal{D}$

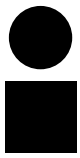

Fig. 1: The circle is above the square. This implies that the square is below the circle. The analogical mapping unit (AMU) can learn this simple "above–below" relation from examples and successfully apply it to novel representations, see Section 3 and Section 4.

$$x_k \in B^{\mathcal{D}}, \quad x_k = (x_{k,1},\ x_{k,2},\ x_{k,3}, \ldots,\ x_{k,\mathcal{D}}). \tag{1}$$

The binding operator, $\otimes$, is defined as the element-wise binary XOR operation. Bundling of multiple vectors x_k is defined as an element-wise binary average

$$\langle \sum_{k=1}^{n} x_{k,i} \rangle = \Theta \left(\frac{1}{n} \sum_{k=1}^{n} x_{k,i} \right), \tag{2}$$

where $\Theta(x)$ is a binary threshold function

$$\Theta(x) = \begin{cases} 1 \text{ for } x > 0.5, \\ 0 \text{ for } x < 0.5, \\ \text{random otherwise.} \end{cases} \tag{3}$$

This is an element-wise majority rule. When an even number of vectors are bundled there may be ties. These elements are populated randomly with an equal probability of zeros and ones. Structures are typically constructed from randomly generated names, roles and fillers by applying the binding and bundling operators. For example, the concept in Figure 1 can be encoded by the two-place relation $\langle a + a_1 \otimes \bullet + a_2 \otimes \blacksquare \rangle$, where a is the relation *name* ("above"), a_1 and a_2 are *roles* of the relation, and the geometric shapes are *fillers* indicating what is related.

All terms and factors in this representation are high-dimensional binary vectors, as defined in (1). Vectors are typically initialized randomly, or they may be compositions of other random vectors. In a full-featured VSA application this encoding step is to be automated with other methods, for example by feature extraction using deep learning networks or receptive fields in combination with randomized fan-out projections and superpositions of patterns. A key thing to realize in this context is that the AMU, and VSAs in general, operate on high-dimensional random distributed representations. This makes the VSA approach robust to noise and in principle and practice it enables operation with approximate compositional structures.

2.1.1 Holistic mapping vectors

The starting point for the development of the AMU is the idea of holistic mapping vectors, see [30, 42, 44]. In [30] a BSC mapping of the form "X is the mother of Y" → "X is the parent of Y" is presented. This mapping is mathematically similar to the above–below relation illustrated in Figure 1, with the exception that the mother–parent mapping is unidirectional because a parent is not necessarily a mother. The above–below relationship is bidirectional because a mapping between "the circle is above the square" and "the square is below the circle" is true in both directions. We think that the geometric example illustrated here is somewhat simpler to explain and we therefore use that. The key idea presented in [30] is that a mapping vector, M, can be constructed so that it performs a mapping of the type: "If the circle is above the square, then the square is below the circle". If the mapping vector is defined in this way

$$M = \bullet_{\uparrow}\blacksquare_{\downarrow} \otimes \blacksquare_{\downarrow}\bullet_{\uparrow}, \tag{4}$$

then it follows that

$$M \otimes \bullet_{\uparrow}\blacksquare_{\downarrow} = \blacksquare_{\downarrow}\bullet_{\uparrow}, \tag{5}$$

because the XOR-based binding operator is an involutary (self-inverse) function. The representations of these particular descriptions are illustrated in Table 1 and are chosen to be mathematically identical to the mother-parent representations in [30].

Table 1: Two different representations of the geometric composition presented in Figure 1. All terms and factors in these expressions are high-dimensional binary vectors, and the two states of each element in these vectors are equally likely. See the text for definitions of the operators. A mapping vector, M, can be constructed that maps one of the representations into the other [30]. The AMU that is presented below can learn the mapping vectors between many different representations like these.

Relation	Representation
Circle is above the square	$\bullet_{\uparrow}\blacksquare_{\downarrow} = \langle a + a_1 \otimes \bullet + a_2 \otimes \blacksquare \rangle$
Square is below the circle	$\blacksquare_{\downarrow}\bullet_{\uparrow} = \langle b + b_1 \otimes \blacksquare + b_2 \otimes \bullet \rangle$

2.1.2 Making analogies with mapping vectors

A more remarkable property appears when bundling several mapping examples

$$M = \langle \bullet_{\uparrow}\blacksquare_{\downarrow} \otimes \blacksquare_{\downarrow}\bullet_{\uparrow} + \blacksquare_{\uparrow}\blacktriangle_{\downarrow} \otimes \blacktriangle_{\downarrow}\blacksquare_{\uparrow} + \ldots \rangle, \tag{6}$$

because in this case the mapping vector *generalizes* correctly to novel representations

$$M \otimes \bigstar_{\uparrow}\blacklozenge_{\downarrow} \approx \blacklozenge_{\downarrow}\bigstar_{\uparrow}. \tag{7}$$

The symbols ★ and ◆ have not been involved in the construction of M, but the mapping results in an analogically correct compositional structure. This is an example of analogy making because the information about above-below relations can be applied to novel representations.

The left- and right-hand side of (7) are similar compositional structures, but they are not identical. The result is correct in the sense that it is close to the expected result, for example in terms of the Hamming distance or correlation between the actual output vector and expected result, see [30] for details. In real-world applications the expected mapping results could well be unknown. It is still possible to interpret the result of an analogical mapping if parts of the resulting compositional structure are known. For example, the interpretation of an analogical mapping result can be made using a VSA operation called *probing*, which extracts known parts of a compositional structure (which by themselves may be compositional structures). This process can be realized with a "clean-up memory", see [45, p. 102]. A mapping result that does not contain any previously known structure is interpreted as noise.

Observe that the analogical mapping (7) is not simply a matter of reordering the vectors representing the five-pointed star and diamond symbols. The source and target of this mapping are two nearly uncorrelated vectors, with different but analogical interpretations. In principle this mechanism can also generalize to compositional structures of higher complexity than those in the training set, see [42] for examples. These interesting results motivated the development of the AMU.

2.1.3 Making inferences with mapping vectors

Note that (4) is symmetric in the sense that the mapping is bidirectional. It can perform the two mappings $M \otimes \bullet_{\uparrow}\blacksquare_{\downarrow} = \blacksquare_{\downarrow}\bullet_{\uparrow}$ and $M \otimes \blacksquare_{\downarrow}\bullet_{\uparrow} = \bullet_{\uparrow}\blacksquare_{\downarrow}$ equally well. This is a consequence of the commutative property of the binding operator. In this particular example that does not pose a problem, because both mappings are true. In the parent–mother example in [30] it implies that "parent" is mapped to "mother", which is not necessarily a good thing. A more severe problem with bidirectional mappings appears in the context of inference and learning of sequences, which requires strictly unidirectional mappings. To enable robust analogical mapping, and inference in general, the mapping direction of BSC mapping vectors must be controlled. This

problem is solved by the integration of an associative memory in the AMU model, see Section 3.1.

2.2 Sparse distributed memory

To make practical use of mapping vectors we need a method to create, store and query them. In particular, this involves the difficulty of knowing how to bundle new mapping vectors with the historical record. How should the system keep the examples organized without involving a homunculus (which could just as well do the mappings for us)? Fortunately, a suitable framework has already been developed for another purpose. The development presented here rests on the idea that an SDM used in an appropriate way is a suitable memory for storage of analogical mapping vectors, which automatically bundles similar mapping examples into well-organized mapping vectors. The SDM model of associative and episodic memory is well described in the seminal book by Kanerva. Various modifications of the SDM model have been proposed [3, 19, 37, 49, 53] but here we use the original SDM model [26].

2.2.1 Architecture and initialization

An SDM essentially consists of two parts, which can be thought of as two matrices: a binary *address matrix*, A, and an integer *content matrix*, C. These two matrices are initialized as follows; The matrix A is populated randomly with zeros and ones (equiprobably), and the matrix C is initialized with zeros. The rows of the matrix A are so-called *address vectors*, and the rows of the matrix C are *counter vectors*. There is a one-to-one link between address vectors and counter vectors, so that an *activated* address vector is always accompanied by one particular activated counter vector. The address and counter vectors have dimensionality $\mathcal{D}$. The number of address vectors and counter vectors defines the size of the memory, S.

The SDM can be described algebraically as follows. If the address and counter vectors have dimensionality $\mathcal{D}$, the address matrix, A, can be defined as

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1\mathcal{D}} \\ a_{21} & a_{22} & \dots & a_{2\mathcal{D}} \\ \dots \\ a_{S1} & a_{S2} & \dots & a_{S\mathcal{D}} \end{bmatrix}, \tag{8}$$

where $a_{i,j} \in \{i = 1, 2, \dots, S, j = 1, 2, \dots, \mathcal{D}\}$ are random bits (short for binary digits) with equal probability for the two states. The content matrix, C, is defined as

$$C = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1\mathcal{D}} \\ c_{21} & c_{22} & \dots & c_{2\mathcal{D}} \\ \dots \\ c_{S1} & c_{S2} & \dots & c_{S\mathcal{D}} \end{bmatrix}, \tag{9}$$

where $a_{i,j} \in \{i = 1, 2, \dots, S, j = 1, 2, \dots, \mathcal{D}\}$ are integer numbers that are initialized to zero in an empty memory.

2.2.2 Storage operation

When a vector is stored in the SDM the content matrix C is updated, but the address matrix A is static. It is typically sufficient to have six or seven bits of precision in the elements of the content matrix [26]. The random nature of the compositional structures make saturation of counter vectors unlikely with that precision.

Two input vectors are needed to store a new vector in the SDM: a *query vector*, $x = (x_1, x_2 \dots, x_{\mathcal{D}})$, and a *data vector*, $d = \{d_1, d_2 \dots, d_{\mathcal{D}}\}$. The query vector, x, is compared with all address vectors, a_i, using the Hamming distance, δ. The Hamming distance between two binary vectors x and a is defined as

$$\delta(x, a) = \sum_{i=1}^{\mathcal{D}} I(x_i, a_i) \tag{10}$$

where I is a bit-wise XOR operation

$$I(x_i, a_i) = \begin{cases} 1, & x_i \neq a_i \\ 0, & x_i = a_i. \end{cases} \tag{11}$$

The Hamming distance between two vectors is related to the Pearson correlation coefficient, ρ, by $\rho = 1 - 2\delta$ [29]. A predefined threshold value, T, is used to calculate a new vector $s = \{s_1, s_2 \dots, s_S\}$ so that

$$s_i = \begin{cases} 1, & \delta(s, a_i) < T \\ 0, & \delta(s, a_i) \geq T. \end{cases} \tag{12}$$

The non-zero elements of s are used to activate a sparse subset of the counter vectors in C. In other words, the indices of non-zero elements in s are the row-indices of activated counter vectors in the matrix C. With this definition of s the activated counter vectors correspond to address vectors that are close to the query vector, x.

In a writing operation the activated counter vectors are updated using the data vector, d. For every bit that is 1 in the data vector the corresponding elements in the activated counter vectors are increased by one, and for every bit that is 0 the corresponding counters are

decreased by one. This means that the elements, $c_{i,j}$, of the matrix C are updated so that

$$c_{i,j} \leftarrow \begin{cases} c_{i,j}+1 & s_i = 1, d_j = 1 \\ c_{i,j}-1 & s_i = 0, d_j = 1 \\ c_{i,j} & s_i = 0 \end{cases} \tag{13}$$

The details of why this is a reasonable storage operation in a binary model of associative memory are well described in [26].

Various modifications of the SDM model have been presented in the literature that address the problem of storing non-random patterns and localist representations in an SDM without saturating the counter vectors. In our perspective, low-level structured inputs from sensors need to be preprocessed with other methods, such as methods for invariant feature extraction. There are good reasons to operate with random representations at the higher cognitive level [20, 26], and this is a typical prerequisite of VSAs.

2.2.3 Retrieval operation

In a retrieval operation, the SDM functions in a way that is similar to the storage operation except that an approximate data vector, d, is retrieved from the SDM rather than supplied to it as input. For a given query vector, $x = (x_1, x_2 \ldots, x_{\mathcal{D}})$, the Hamming distances, $\delta(x, a_i)$, between x and all address vectors, a_i, are calculated. A threshold condition (12) is used to activate counter vectors in C with addresses that are sufficiently close to the query vector. The activated counter vectors in C are summed,

$$h_j = \sum_{i=1}^{S} s_i c_{ij}, \tag{14}$$

and the resulting integer vector, h_j, is converted to a binary output vector, d, with the rule

$$d_j = \begin{cases} 1, & h_j > 0 \\ 0, & h_j < 0 \\ \text{random otherwise.} \end{cases} \tag{15}$$

In this paper we use the term *recall* for the process of retrieving a data vector from the SDM that previously has been stored. An SDM retrieval operation is analogous to bundling (2) of all vectors stored with an address vector that is similar to the query vector.

2.2.4 Implementation

An SDM can be visualized as suggested in Figure 2, with two inputs and one output. The open circle denotes the query vector, which is supplied both when storing and retrieving information. The open square denotes the input data vector, which is supplied when storing information in the SDM. The solid square denotes the output data vector, which is generated when retrieving information. For software simulation purposes,

Fig. 2: Schematic illustration of a sparse distributed memory (SDM), which is an associative memory for random high-dimensional binary vectors [26].

for example in Matlab, the VSA operators and the SDM can be implemented in an alternative way if binary vectors are replaced with bipolar vectors according to the mapping $\{0 \rightarrow 1,\ 1 \rightarrow -1\}$. In that case the XOR binding operator is replaced with element-wise multiplication.

The number of counter vectors that are activated during one storage or retrieval operation depends on the threshold value, T. In this work we choose to calculate this parameter so that a given fraction, χ, of the storage locations of the SDM is activated. Therefore, χ is the average probability of activating a row in the matrices A and C, which we also refer to as one storage *location* of the SDM. The number of activated locations in each storage or retrieval operation is exactly χS. When storing multiple mapping vectors in an SDM it is possible that a subset of the counter vectors are updated with several mapping vectors, which can correspond to different mapping examples. On average the probability for *overlap* of two different mapping vectors in a location of the SDM is of the order χ^2.

Next we present the AMU model, which incorporates an SDM for the storage and retrieval of analogical mapping vectors.

3 Model

The AMU consists of one SDM with an additional input-output circuit. First we introduce the learning and mapping parts of this circuit, and then we combine these parts into a complete circuit that represents the AMU.

3.1 Learning circuit

The AMU stores vectors in a process that is similar to the bundling of examples in (6), provided that the addresses of the mapping examples are similar. If the ad-

dresses are uncorrelated the individual mapping examples will be stored in different locations of the SDM and no bundling takes place, which prevents generalization and analogical mapping. A simple approach is therefore to define the query vector of a mapping, $x_k \rightarrow y_k$, as the variable x_k. This implies that mappings with similar x_k are, qualitatively speaking, bundled together within the SDM. The mapping vector $x_k \otimes y_k$ is the data vector supplied to the SDM, and it is bundled in counter vectors with addresses that are close to x_k. A schematic diagram illustrating this learning circuit is shown in Figure 3.

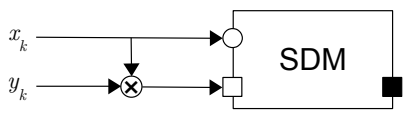

Fig. 3: Schematic diagram of learning circuit for mappings of type $x_k \rightarrow y_k$. Learning can be supervised or unsupervised, for example in the form of coincidence (Hebbian) learning.

This circuit avoids the problem of bidirectionality discussed in Section 2.1.3 because the SDM locations that are activated by the query vector x_k are different from the locations that are activated by the query vector y_k of the reversed mapping. The forward and reversed mappings are therefore stored in different SDM locations. Therefore, the output of a query with y_k is nonsense (noise) if the reversed mapping is not explicitly stored in the SDM. In other words, the reversed mapping $y_k \rightarrow x_k$ is not implicitly learned.

Note that different types of mappings have different x_k. Queries with different x_k activate different locations in the SDM. Therefore, given a sufficiently large memory it is possible to store multiple types of mappings in one SDM. This is illustrated with simulations in Section 4.

3.2 Mapping circuit

The learning mechanism in combination with the basic idea of analogical mapping (7) suggests that the mapping circuit should be defined as illustrated in Figure 4. This circuit binds the input, x_k, with the bundled mapping vector of similar mapping examples that is stored in the SDM. The result is an output vector y'_k. If the mapping $x_k \rightarrow y_k$ is stored in the SDM then $y'_k \approx y_k$. When a number of similar mapping examples are stored in the SDM, this circuit can generalize correctly to novel compositional structures. In such cases y'_k is an approximate analogical mapping of x_k. This is illustrated with simulations in the next section.

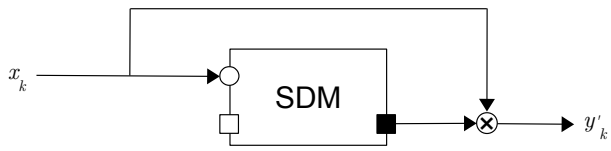

Fig. 4: Schematic diagram of circuit for mappings of type $x_k \rightarrow y'_k$. Here $y'_k \approx y_k$ when $x_k \rightarrow y_k$ is in the training set. Otherwise y'_k is an approximate analogical mapping of x_k, or noise if x_k is unrelated to the learned mappings.

3.3 The analogical mapping unit

The AMU includes one SDM and a combination of the learning and mapping circuits that are presented in the former two subsections, see Figure 5. It is a computational unit for distributed representations of compositional structures that takes two binary input vectors and provides one binary output vector, much like the SDM does, but with a different result and interpretation. If the input to the AMU is x_k the corresponding output vector y'_k is calculated (mapping mode). With two input vectors, x_k and y_k, the AMU stores the corresponding mapping vector in the SDM (learning mode). In total the AMU has three exogenous parameters: S, χ and the dimensionality $\mathcal{D}$ of the VSA, see Table 2. In principle the SDM of the AMU can be shared with other VSA components, provided that the mappings are

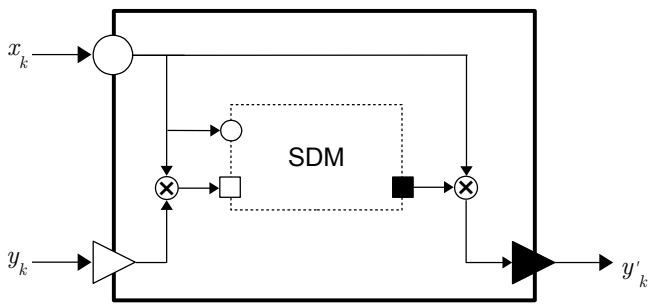

Fig. 5: The analogical mapping unit (AMU), which includes the learning and mapping circuits introduced above. This unit learns mappings of the type $x_k \rightarrow y_k$ from examples and uses bundled mapping vectors stored in the SDM to calculate the output vector y'_k.

encoded in a suitable way. Next we present simulation results that characterize some important and interesting properties of the AMU.

Table 2: Summary of exogenous parameters of the AMU.

Expression	Description
S	Memory size, number of address and counter vectors.
$\mathcal{D}$	Dimensionality of vector symbolic representations.
χ	Average probability for activating an SDM location.

4 Simulation experiments

An important measure for the quality of an intelligent system is its ability to generalize with the acquired knowledge [52]. To test this aspect, we generate previously unseen examples of the "above-below relation using the sequence shown in Figure 6. The relations between the items in this sequence are encoded in a similar way to those outlined in Table 1, with the only difference being that the fillers are different, see Table 3.

A
B
C
⋮

Fig. 6: A sequence of novel "above-below" relations that is used to test the ability of the AMU to generalize.

4.1 Comparison with other results

Our mapping approach is similar to that in [30], but differs from that in three ways. First, mapping examples are fed to an SDM so that multiple mapping vectors are stored in the SDM, instead of defining explicit mapping vectors for each mapping. Second, the model uses binary spatter codes to represent mappings, which means that the mapping vectors generated by the AMU are binary vectors and not integer vectors. Third, the dimensionality $\mathcal{D}$ is set to a somewhat low value of 1000 in most simulations presented here, and we illustrate the effect of higher dimensionality at the end of this section. We adopt the method in [30] to calculate the similarity between the output of the AMU and alternative mappings with the correlation coefficient, ρ, see Section 2.2.2. Randomly selected structures, such as: a, b, a_1, a_2, b_1, b_2, ●, ■ $\ldots A, B$ are uncorrelated, $\rho \approx 0$. That is true also for x_i and y_i, such as: $●_{\uparrow}■_{\downarrow}$ and $■_{\downarrow}●_{\uparrow}$. However, $●_{\uparrow}■_{\downarrow}$ and $\mathrm{A}_{\uparrow}\mathrm{B}_{\downarrow}$ include the same relation name, a, in the composition, and these vectors are therefore correlated, $\rho \approx 0.25$.

Table 3: Representation of "above-below" relations between novel structures.

Relation	Representation
A is above B	$\mathrm{A}_{\uparrow}\mathrm{B}_{\downarrow} = \langle a \ + \ a_1 \otimes A \ + \ a_2 \otimes B \rangle$
B is below A	$\mathrm{B}_{\downarrow}\mathrm{A}_{\uparrow} = \langle b \ + \ b_1 \otimes B \ + \ b_2 \otimes A \rangle$

The AMU learns mappings between compositional structures according to the circuit shown in Figure 3, and the output y'_k of the AMU is determined according to Figure 4 and related text. The output, y'_k, is compared with a set of alternative mapping results in terms of the correlation coefficient, ρ, both for recall and generalization. The number of training examples is denoted with N_e. To estimate the average performance of the AMU we repeat each simulation 5000 times with independently initialized AMUs. This is sufficient to estimate the correlation coefficients with a relative error of about $\pm 10^{-2}$.

In order to compare our model with [30], we first set the parameters of the AMU to $S = 1$, $\mathcal{D} = 1000$ and $\chi = 1$. This implies that the AMU has one location that is activated in each storage and retrieval operation. The result of this simulation is presented in Figure 7. Figure 7a shows the average correlation between the output y'_k resulting from the input, $x_k = ●_{\uparrow}■_{\downarrow}$, which is in the training set, and four alternative compositional structures. Figure 7b shows the average correlation between four alternative compositional structures and the AMU output resulting from a novel input, $x_k = \mathrm{A}_{\uparrow}\mathrm{B}_{\downarrow}$. The alternative with the highest correlation is selected as the correct answer. With more training examples the ability of the AMU to generalize increases, because the correlations with wrong alternatives decrease with increasing N_e. The alternative with the highest correlation always corresponds to the correct result in this simulation, even in the case of generalization from three training examples only.

These results demonstrate that the number of training examples affects the ability of the AMU to generalize. This conclusion is consistent with the results (see Figure 1) and the related discussion in [30]. The constant correlation with the correct mapping alternative in Figure 7b is one quantitative difference between our result and the result in [30]. In our model the correlation of the correct generalization alternative does not increase with N_e, but the correlations with incorrect

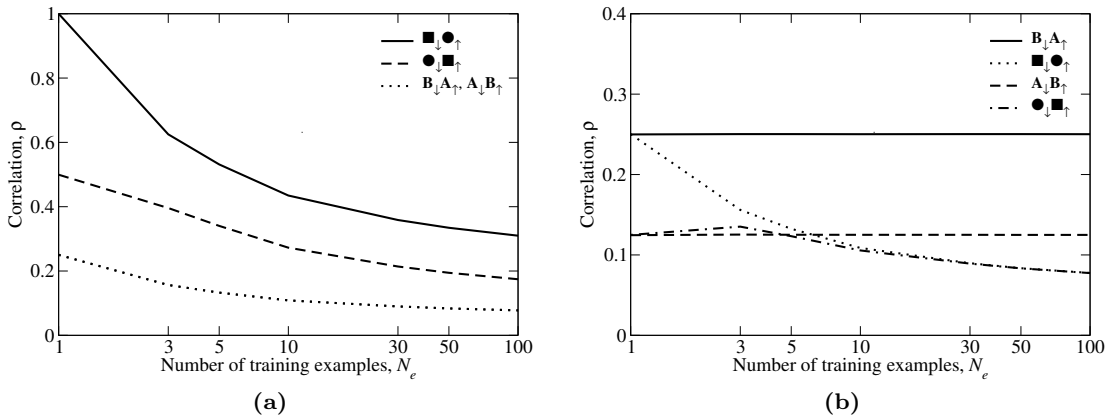

Fig. 7: The correlation, ρ, between the output of the analogical mapping unit (AMU) and alternative compositional structures versus the number of training examples, N_e, for (a) recall of learned mappings; (b) generalization from novel inputs. The parameters of the AMU are $S = 1$, $\chi = 1$ and $\mathcal{D} = 1000$.

alternatives decrease with increasing N_e. This difference is caused by the use of binary mapping vectors within the AMU, instead of integer mapping vectors. If we use the integer mapping vectors (14) that are generated within the SDM of the AMU we reproduce the quantitative results obtained in [30]. Integer mapping vectors can be implemented in our model by modifying the normalization condition (15). The correlation with correct mappings can be improved in some cases with the use of integer mapping vectors, but the use of binary representations of both compositional structures and mapping vectors is more simple, and it makes it possible to use an ordinary SDM for the AMU. By explicitly calculating the probability of error from the simulation results we conclude that the use of binary mapping vectors is sufficient. (We return to this point below.)

In principle there are many other "wrong" mapping alternatives that are highly correlated with the AMU output, y'_k, in addition to the four alternatives that are considered above. However, the probability that such incorrect alternatives or interpretations would emerge spontaneously is practically zero due to the high dimensionality and random nature of the compositional structures [26]. This is a key feature and design principle of VSAs.

4.2 Storage of mapping vectors in a sparse distributed memory

Next, we increase the size of the SDM to $S = 100$, which implies that there are one hundred storage locations. The possibility to have $S > 1$ is a novel feature of the AMU model, which was not considered in former studies [30, 31, 42, 45, 46]. We investigate the effect of different values of χ and present results for $\chi = 0.05$ and $\chi = 0.25$. That is, when the AMU activates 5% and 25% of the SDM to learn each mapping vector. Figure 8 shows that the effect of increasing χ is similar to training with more examples, in the sense that it improves generalization and reduces the accuracy of recall. This figure is analogous to Figure 7, with the only difference being that here the AMU parameters are $S = 100$ and $\chi = 0.05$ or $\chi = 0.25$. Figure 8a and Figure 8c show the average correlation between the output y'_k resulting from the input, $x_k = \bullet_\uparrow\blacksquare_\downarrow$, which is in the training set, and four alternative compositional structures for, respectively, $\chi = 0.05$ and $\chi = 0.25$. Figure 8b and Figure 8d show the average correlation between four alternative compositional structures and the AMU output resulting from a novel input, $x_k = \mathrm{A}_\uparrow\mathrm{B}_\downarrow$. When retrieving learned examples for $\chi = 0.05$ and $\chi = 0.25$ the alternative with the highest correlation always corresponds to the correct result in this simulation experiment. See Figure 7 and related text for details of how the average correlations are calculated. A high value of χ enables generalization with fewer training examples, but there is a trade-off with the number of different two-place relations that are learned by the AMU. We return to that below.

4.3 Probability of error

The correlations between the output of the AMU and the alternative compositional structures varies from one simulation experiment to the next because the AMU is randomly initialized. The distribution functions for these variations have a non-trivial structure. It is difficult to translate average correlation coefficients and variances into tail probabilities of error. Therefore, we

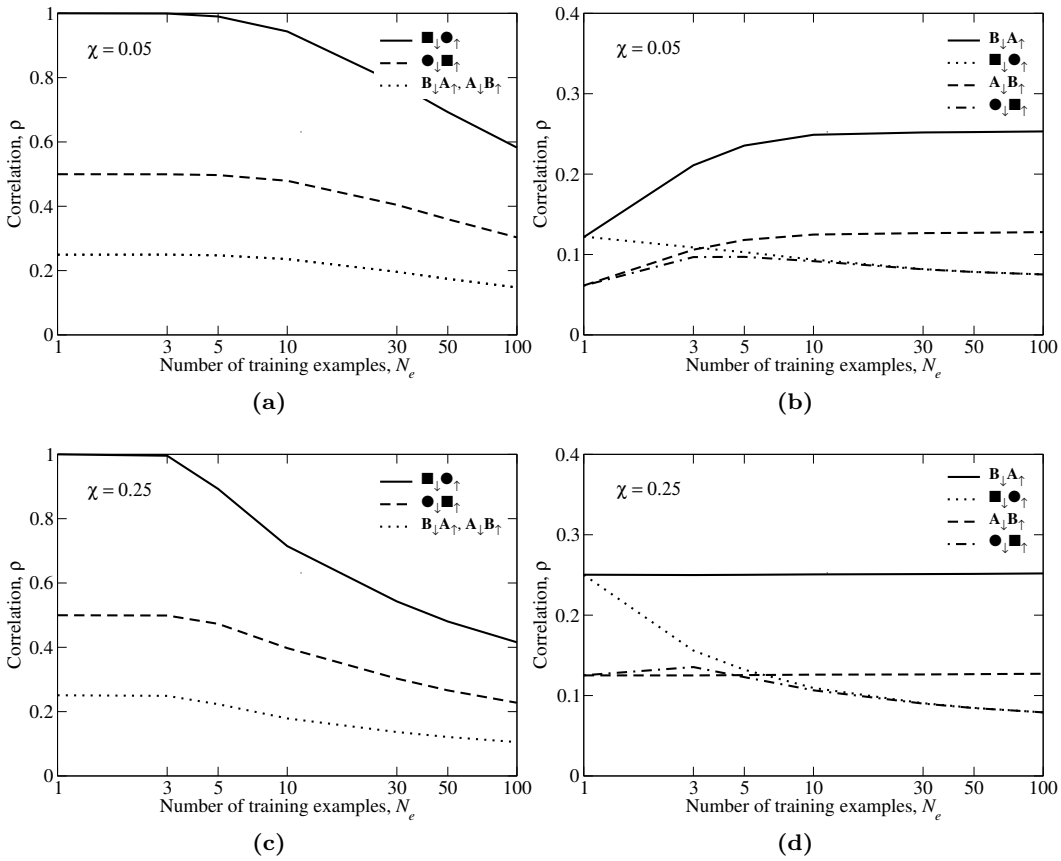

Fig. 8: Correlations between alternative compositional structures and the output of an AMU versus the number of examples in the training set, N_e, for (a) recall with $\chi = 0.05$; (b) Generalization with $\chi = 0.05$; (c) Recall with $\chi = 0.25$; (d) Generalization with $\chi = 0.25$. The size of the SDM is $S = 100$ in all four cases.

estimate the *probability of error* numerically from the error rate in the simulation experiments. An error occurs when the output of the AMU has the highest correlation with a compositional structure that represents an incorrect mapping. The probability of error is defined as the number of errors divided by the total number of simulated mappings, $N_e N_r N_s$, where N_e is the number of training examples for each particular two-place relation, N_r is the number of different two-place relations (unrelated sets of mapping examples), and $N_s = 5000$ is the number of simulation experiments performed with independent AMUs. We test all N_e mappings of each two-place relation in the training set, and we test equally many generalization mappings in each simulation experiment. This is why the factors N_e and N_r enter the expression for the total number of simulated mappings. The N_e different x_k and y_k are generated in the same way for the training and generalization sets, by using a common set of names and roles but different fillers for the training and generalization sets. Different two-place relations are generated by different sets of names and roles, see Section 4.4.

When the probability of error is low we have verified the estimate of the probability by increasing the number of simulation experiments, N_s. For example, this is sometimes the case when estimating the probability of error for mappings that are in the training set. As mentioned in Section 4.1 this is why we choose a suboptimal and low value for the dimensionality of the representations, $\mathcal{D}$. The probability of error is lower with high-dimensional representations, which is good for applications of the model but makes estimation of the probability of error costly. Figure 9 presents an interesting and expected effect of the probability of error during generalization versus the number of training examples. The probability of activating a storage location, χ, affects the probability of generalization error significantly. For $\chi = 0.25$ and $N_e = 30$ the AMU provides more than 99.9% correct mappings. Therefore, that point has poor precision and is excluded from the figure.

In Figure 7 all mapping vectors are bundled in one single storage location, while Figure 9 is generated with AMUs that have an SDM of size $S = 100$ so that each mapping vector is stored in multiple locations. Figure 8

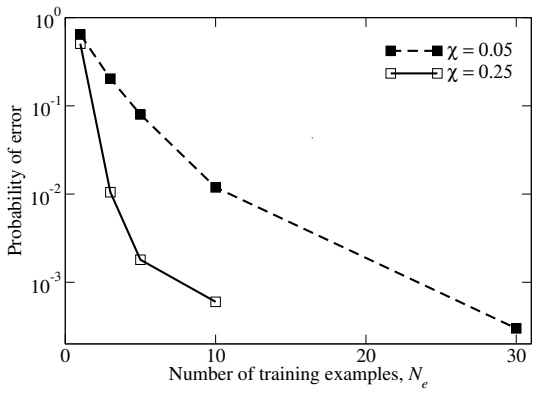

Fig. 9: The probability of generalization error versus the number of training examples, N_e, for $\chi = 0.05$ and $\chi = 0.25$.

and Figure 9 are complementary because the parameters of the AMUs used to generate these figures are identical. For $\chi = 0.25$ the AMU generalizes correctly with fewer training examples compared to the results for the lower value of $\chi = 0.05$. This result is consistent with Figure 8, which suggests that the AMU with $\chi = 0.25$ generalizes with fewer training examples. The rationale of this result is that a higher value of χ gives more overlap between different mapping vectors stored in the AMU, and multiple overlapping (bundled) mapping vectors are required for generalization, see (7). This effect is visible also in Figure 7, where the AMU provides the correct generalization after learning three examples.

4.4 Storing multiple relations

Since mapping vectors are bundled in a fraction of the storage locations of the SDM it is reasonable to expect that the AMU can store additional mapping vectors, which could be something else than "above-below" relations. To test this idea we introduce a number, N_r, of different *two-place relations*. The above-below relation considered above is one example of such a two-place relation, see Table 1 and Table 3. In general, names and roles can be generated randomly for each particular two-place relation, see Table 4, in the same way as the names and roles are generated for the above-below relations. For each two-place relation we generate N_e training examples and N_e test (generalization) examples, see the discussion in Section 4.3 for further details. This way we can simulate the effect of storing multiple two-place relations in one AMU.

An interesting effect appears when varying the number of two-place relations, N_r, and the size of the SDM, see Figure 10a. The size of the SDM, S, and the average

Table 4: Representation of generic two-place relations, labeled by z, between two compositional structures represented by the fillers $f_{1,k,z}$ and $f_{2,k,z}$. Training examples are indexed by $k \in [1, N_e]$ and there are equally many test (generalization) examples $k \in [N_e + 1, 2N_e]$, which implies that $2N_e$ mapping examples are created for each two-place relation. The names, $n_{i,z}$, and roles, $r_{ij,z}$, are unique for each two-place relation. By definition $z \in [1, N_r]$.

Relation	Representation
Source, $x_{k,z}$	$\langle n_{1,z} + r_{11,z} \otimes f_{1,k,z} + r_{12,z} \otimes f_{2,k,z} \rangle$
Target, $y_{k,z}$	$\langle n_{2,z} + r_{21,z} \otimes f_{1,k,z} + r_{22,z} \otimes f_{2,k,z} \rangle$

probability of activating a storage location, χ, affects the probability of generalization error. Provided that the SDM is sufficiently large and that N_e is not too high, there is a minimum in the probability of error for some value of N_r. A minimum appears at $N_r = 3-4$ for $\chi = 0.05$ and $S = 1000$ in Figure 10a. Figure 10b has a minimum at $N_r = 2$ for $chi = 0.25$ and $S = 100$. Generalization with few examples requires that the number of storage locations is matched to the number of two-place relations that are learned. The existence of a minimum in the probability of error can be interpreted qualitatively in the following way. The results in Figures 7–9 illustrate that a low probability of generalization error requires that many mapping vectors are bundled. The probability of error decreases with increasing N_e because the correlation between the AMU output and the incorrect compositional structures decreases with N_e. This is not so for the correlation between the output and the correct compositional structure, which is constant. This suggests that one effect of bundling additional mapping vectors, which adds noise to any particular mapping vector that has been stored in the AMU, is a reduction of the correlation between the AMU output and incorrect compositional structures. A similar effect apparently exists when bundling mapping vectors of unrelated two-place relations; The noise introduced mainly reduces the correlation between the AMU output and compositional structures representing incorrect alternatives.

Another minimum in the probability of error appears when we vary the average probability of activating a storage location, χ, see Figure 11. In this figure we illustrate the worst-case scenario in Figure 10, $S = 100$ and $N_e = N_r = 10$, for different values of χ. A minimum appears in the probability of error for $\chi \approx 0.2$. This result shows that the (partial) bundling of mapping vectors, including bundling of unrelated mapping vectors, can reduce the probability of generalization error.

The interpretation of these results in a neural-network perspective on SDM and binary VSAs is that the sparse-

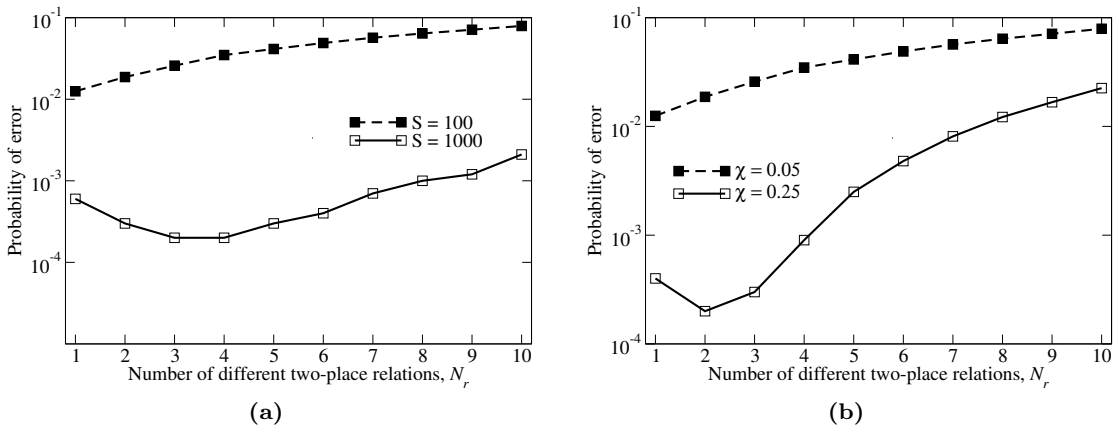

Fig. 10: The probability of generalization error versus the number of different two-place relations, N_r, for (a) $S = 100$ and $S = 1000$ when $\chi = 0.05$; (b) $\chi = 0.05$ and $\chi = 0.25$ when $S = 100$. Other parameters are $\mathcal{D} = 1000$ and $N_e = 10$.

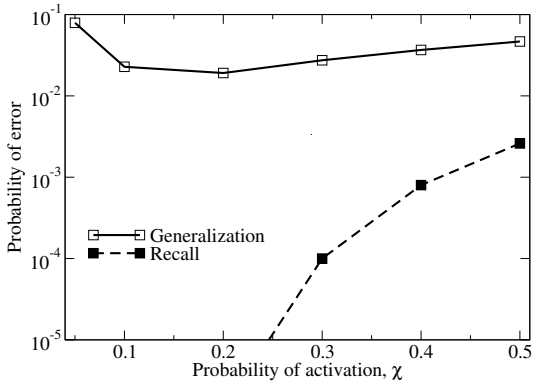

Fig. 11: The probability of error of the AMU versus the average probability of activating a storage location, χ, for recall and generalization.

ness of the neural codes for analogical mapping is important. In particular, the sparseness of the activated address decoder neurons in the SDM [26] is important for the probability of generalization error. A similar effect is visible in Figure 9, where $\chi = 0.25$ gives a lower probability of error than $\chi = 0.05$.

4.5 Effect of dimensionality

All simulations that are summarized above are based on a low dimensionality of compositional structures, $\mathcal{D} = 1000$, which is one order of magnitude lower than the preferred value. A higher dimensionality results in a lower probability of error. Therefore, it would be a tedious task to estimate the probability of error with simulations for higher values of $\mathcal{D}$. We illustrate the effect of varying dimensionality in Figure 12. By choosing a dimensionality of order 10^4 the probabilities of errors reported in this work can be lowered significantly, but that would make the task to estimate the probability of error for different values of the parameters S, χ, N_e and N_r more tedious. The analytical results obtained by [26, 30, 31] suggest that the optimal choice for the dimensionality of binary vector symbolic representations is of order 10^4, this conclusion remains true for the AMU model. A slightly higher dimensionality than 10^4 may be motivated, but 10^5 is too much because it does not improve the probability of error much and requires more storage space and computation. Kanerva derived this result from basic statistical properties of hyperdimensional binary spaces, which is an interesting result in a cognitive computation context because that number matches the number of excitatory synapses observed on pyramidal cells in cortex.

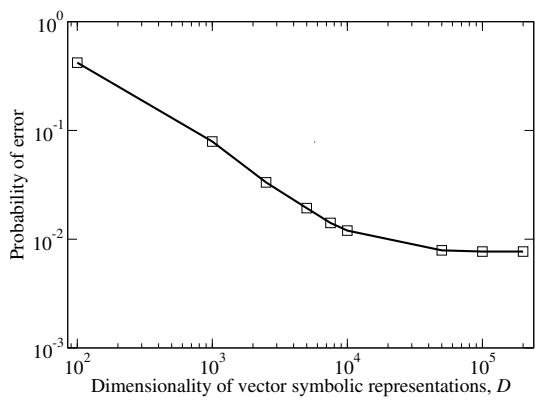

Fig. 12: The probability of generalization error versus the dimensionality of binary compositional structures, $\mathcal{D}$. Other parameters are $S = 100$, $\chi = 0.05$, $N_e = 10$ and $N_r = 10$. The optimal dimensionality is of the order $\mathcal{D} = 10^4$.

5 Discussion

Earlier work on analogical mapping of compositional structures deals with isolated mapping vectors that are hand-coded or learned from examples [7, 30, 31, 42, 45, 46]. The aim of this work is to investigate whether such mapping vectors can be stored in an associative memory so that multiple mappings can be learnt from examples and applied to novel inputs, which in principle can be unlabeled. We show that this is possible and demonstrate the solution using compositional structures that are similar to those considered by others. The proposed model integrates two existing ideas into one novel computational unit, which we call the Analogical Mapping Unit (AMU). The AMU integrates a model of associative memory known as sparse distributed memory (SDM) [26] with the idea of holistic mapping vectors [30, 42, 44] for binary compositional structures that generalize to novel inputs. By extending the original SDM model with a novel input-output circuit the AMU can store multiple mapping vectors obtained from similar and unrelated training examples. The AMU is designed to separate the mapping vectors of unrelated mappings into different storage locations of the SDM.

The AMU has a one-shot learning process and it is able to recall mappings that are in the training set. After learning many mapping examples no specific mapping is recalled and the AMU provides the correct mapping by analogy. The ability of the AMU to recall specific mappings increases with the size of the SDM. We find that the ability of the AMU to generalize does not increase monotonically with the size of the memory; It is optimal when the number of storage locations is matched to the number of different mappings learnt. The relative number of storage locations that are activated when one mapping is stored or retrieved is also important for the ability of the AMU to generalize. This can be understood qualitatively by a thought experiment: If the SDM is too small there is much interference between the mapping vectors and the output of the AMU is essentially noise. If the SDM is infinitely large each mapping vector is stored in a unique subset of storage locations and the retrieved mapping vectors are exact copies of those learnt from the training examples. Generalization is possible when related mapping vectors are combined into new (bundled) mapping vectors, which integrate structural and semantic constraints from similar mapping examples (7). In other words, there can be no generalization when the retrieved mapping vectors are exact copies of examples learnt. Bundling of too many unrelated mapping vectors is also undesirable because it leads to a high level of noise in the output, which prevents application of the result. Therefore, a balance between the probability of activating and allocating storage locations is required to obtain a minimum probability of error when the AMU generalizes.

The probability of activating storage locations of the AMU, χ, is related to the "sparseness of the representation of mapping vectors. A qualitative interpretation of this parameter in terms of the sparseness of neural coding is discussed in [26]. We find that when the representations are too sparse the AMU makes perfect recall of known mappings but is unable to generalize. A less sparse representation results in more interference between the stored mapping vectors. This enables generalization and has a negative effect on recall. We find that the optimal sparseness for generalization depends in a non-trivial way on other parameters and details, such as the size of the SDM, the number of unrelated mappings learnt and the dimensionality of the representations. The optimal parameters for the AMU also depend on the complexity of the compositional structures, which is related to the encoding and grounding of compositional structures. This is an open problem that needs further research.

A final note concerns the representation of mapping vectors in former work versus the mapping vectors stored by an AMU. In [30] integer mapping vectors for binary compositional structures are used to improve the correlation between mapping results and the expected answers. An identical approach is possible with the AMU if the normalization in (15) is omitted. However, by calculating the probability of error in the simulation experiments we conclude that it is sufficient to use binary mapping vectors. This is appealing because it enables us to represent mapping vectors in the same form as other compositional structures. Kanervas estimate that the optimal dimensionality for the compositional structures is of order 10^4 remains true for the AMU. The probability of errors made by the AMU decreases with increasing dimensionality up to that order, and remains practically constant at higher dimensionality.

There is much that remains to understand concerning binary vector symbolic models, for example whether these discrete models are compatible with cortical networks and to what extent they can describe cognitive function in the brain. At a more pragmatic level the problem of how to encode and ground compositional structures automatically needs further work, for example in the form of receptive fields and deep learning networks. Given an encoding unit for binary compositional structures the AMU is ready to be used in applications. Therefore, this paper is a small but potentially impor-

tant step towards a future generation of digital devices that compute with qualitative relationships and adapt to the environment with minimal assistance from humans.

Acknowledgements We thank the anonymous reviewers for their constructive suggestions that helped us to improve this article, Serge Thill for reviewing and commenting on an early version of the manuscript, Ross Gayler for helping us to improve the final text, and Jerker Delsing for comments and encouraging support. This work was partially supported by the Swedish Foundation for International Cooperation in Research and Higher Education (STINT) under the grant number IG2011-2025, and the Kempe Foundations.

References

1. Diederik Aerts, Marek Czachor, and Bart De Moor. Geometric analogue of holographic reduced representation. *Journal of Mathematical Psychology*, 53(5):389–398, 2009.
2. James A. Anderson, Edward Rosenfeld, and Andras Pellionisz. *Neurocomputing*. MIT Press, 1993.
3. Ashraf Anwar and Stan Franklin. Sparse distributed memory for 'conscious' software agents. *Cognitive Systems Research*, 4(4):339–354, 2003.
4. Lawrence W. Barsalou. Grounded cognition. *Annual Review of Psychology*, 59:617–645, 2008.
5. David J. Chalmers. Syntactic transformations on distributed representations. *Connection Science*, 2(1-2):53–62, 1990.
6. Adam Claridge-Chang, Robert D. Roorda, Eleftheria Vrontou, Lucas Sjulson, Haiyan Li, Jay Hirsh, and Gero Miesenböck. Writing memories with Light-Addressable reinforcement circuitry. *Cell*, 139(2):405–415, 2009.
7. Chris Eliasmith and Paul Thagard. Integrating structure and meaning: a distributed model of analogical mapping. *Cognitive Science*, 25(2):245–286, 2001.
8. Thomas G. Evans. A heuristic program to solve geometric-analogy problems. In *Proceedings of the Spring Joint Computer Conference*, pages 327–338, 1964.
9. Brian Falkenhainer, Kenneth D. Forbus, and Dedre Gentner. The structure-mapping engine: Algorithm and examples. *Artificial Intelligence*, 41(1):1–63, 1989.
10. Robert Matthew French. *The Subtlety of Sameness: A Theory and Computer Model of Analogy-Making*. MIT Press, 1995.
11. Robert Matthew French. The computational modeling of analogy-making. *Trends in Cognitive Sciences*, 6(5):200–205, 2002.
12. D. Gabor. Improved holographic model of temporal recall. *Nature*, 217(5135):1288–1289, 1968.
13. Ross W. Gayler. Vector symbolic architectures answer Jackendoff's challenges for cognitive neuroscience. In *Proceedings of the Joint International Conference on Cognitive Science*, pages 133–138, 2003.
14. Dedre Gentner. Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7(2):155–170, 1983.
15. Dedre Gentner and Kenneth D. Forbus. Computational models of analogy. *Wiley Interdisciplinary Reviews: Cognitive Science*, 2(3):266–276, 2011.
16. Dedre Gentner and Arthur B. Markman. Defining structural similarity. *The Journal of Cognitive Science*, 6:1–20, 2006.
17. James Hammerton. Holistic computation: Reconstructing a muddled concept. *Connection Science*, 10(1):3–19, 1998.
18. Stevan Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1-3):335–346, 1990.
19. A. Tim Hely, J. David Willshaw, and Hayes M. Gillian. A new approach to kanerva's sparse distributed memory. *IEEE Transactions on Neural Networks*, 8(3):791–794, 1997.
20. Sean L. Hill, Yun Wang, Imad Riachi, Felix Schürmann, and Henry Markram. Statistical connectivity provides a sufficient foundation for specific functional connectivity in neocortical neural microcircuits. *Proceedings of the National Academy of Sciences*, 2012.
21. Geoffrey E. Hinton. Mapping part-whole hierarchies into connectionist networks. *Artificial Intelligence*, 46(1-2):47–75, 1990.
22. Douglas R Hofstadter. Epilogue: Analogy as the core of cognition. In Dedre Gentner, Keith J. Holyoak, and Boicho N. Kokinov, editors, *The Analogical Mind: Perspectives from Cognitive Science*, pages 499–538. MIT Press, 2001.
23. Keith J. Holyoak and Paul Thagard. Analogical mapping by constraint satisfaction. *Cognitive Science*, 13(3):295–355, 1989.
24. Keith J. Holyoak and Paul Thagard. *Mental Leaps: Analogy in Creative Thought*. MIT Press, 1996.
25. John E. Hummel and Keith J. Holyoak. Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review*, 104(427–466), 1997.
26. Pentti Kanerva. *Sparse Distributed Memory*. The MIT Press, 1988.
27. Pentti Kanerva. Sparse distributed memory and related models. In Mohamad H. Hassoun, editor,

Associative Neural Memories: Theory and Implementation, pages 50–76. Oxford University Press, 1993.

28. Pentti Kanerva. The spatter code for encoding concepts at many levels. In *Proceedings of the International Conference on Artificial Neural Networks*, volume 1, pages 226–229, 1994.
29. Pentti Kanerva. Fully distributed representation. In *Proceedings of the Real World Computing Symposium*, volume 97, pages 358–365, 1997.
30. Pentti Kanerva. Large patterns make great symbols: An example of learning from example. In Stefan Wermter and Ron Sun, editors, *Hybrid Neural Systems*, volume 1778, pages 194–203. Springer-Verlag, 2000.
31. Pentti Kanerva. Hyperdimensional computing: an introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive Computation*, 1(2):139–159, 2009.
32. Boicho N. Kokinov and Alex A. Petrov. Integrating memory and reasoning in analogy-making: the ambr model. In *The Analogical Mind: Perspectives Prom Cognitive Science*, pages 161–196. MIT Press, 2001.
33. Alexandre Linhares, Daniel M. Chada, and Christian N. Aranha. The emergence of miller's magic number on a sparse distributed memory. *PLoS ONE*, 6(1):e15592, 2011.
34. H. C. Longuet-Higgins. Holographic model of temporal recall. *Nature*, 217:104–104, 1968.
35. B. Arthur Markman, Dedre Gentner, and J. Edward Wisniewski. Comparison and cognition: Implications of structure-sensitive processing for connectionist models. Unpublished report, 1993.
36. James B. Marshall and Douglas R. Hofstadter. The metacat project: a self-watching model of analogy-making. *Cognitive Studies*, 4(4):57–71, 1997.
37. Hongying Meng, Kofi Appiah, Andrew Hunter, Shigang Yue, Mervyn Hobden, Nigel Priestley, Peter Hobden, and Cy Pettit. A modified sparse distributed memory model for extracting clean patterns from noisy inputs. In *Proceedings of the International Joint Conference on Neural Networks*, pages 2084–2089, 2009.
38. Marvin Minsky. *The Society of Mind*. Simon & Schuster, 1988.
39. Marvin Minsky. *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind*. Simon & Schuster, 2006.
40. Melanie Mitchell. *Analogy-Making as Perception: A Computer Model*. MIT Press, 1993.
41. Jane Neumann. *Holistic Processing of Hierarchical Structures in Connectionist Networks*. PhD thesis, School of Informatics, University of Edinburgh, Edinburgh, United Kingdom, 2001.
42. Jane Neumann. Learning the systematic transformation of holographic reduced representations. *Cognitive Systems Research*, 3(2):227–235, 2002.
43. Lars F Niklasson and Tim van Gelder. On being systematically connectionist. *Mind & Language*, 9(3):288–302, 1994.
44. Tony A. Plate. *Distributed Representations and Nested Compositional Structure*. PhD thesis, Department of Computer Science, University of Toronto, Toronto, Canada, 1994.
45. Tony A. Plate. Holographic reduced representations. *IEEE Transactions on Neural Networks*, 6(3):623–641, 1995.
46. Tony A. Plate. *Holographic Reduced Representation: Distributed Representation for Cognitive Structures*. Center for the Study of Language and Information (CSLI), 2003.
47. Jordan B. Pollack. Recursive distributed representations. *Artificial Intelligence*, 46:77–105, 1990.
48. Daniel Rasmussen and Chris Eliasmith. A neural model of rule generation in inductive reasoning. *Topics in Cognitive Science*, 3(1):140–153, 2011.
49. Bohdana Ratitch and Doina Precup. Sparse distributed memories for on-line value-based reinforcement learning. *Lecture Notes in Computer Science*, 3201:347—358, 2004.
50. W Reichardt. Autokorrelations auswertung als funktionsprinzip des zentral-nervensystems (Autocorrelations as a functional principle of the central nervous system). *Z. Naturforsch*, 12(b):448–457, 1957.
51. Walter R. Reitman. Cognition and thought. an information processing approach. *Psychology in the Schools*, 3(2):189–189, 1966.
52. Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3 edition, 2009.
53. Javier Snaider and Stan Franklin. Extended sparse distributed memory and sequence storage. *Cognitive Computation*, 4(2):172–180, 2012.
54. Terry Stewart and Chris Eliasmith. Compositionality and biologically plausible models. In Markus Werning, Wolfram Hinzen, and Edouard Machery, editors, *The Oxford Handbook of Compositionality*. Oxford University Press, 2012.
55. Peter D. Turney. The latent relation mapping engine: Algorithm and experiments. *Journal of Artificial Intelligence Research*, 33:615–655, 2008.
56. D. J. Willshaw, O. P. Buneman, and H. C. Longuet-Higgins. Non-holographic associative memory. *Nature*, 222:960–962, 1969.