Generating Hyperdimensional Distributed Representations from Continuous-Valued Multivariate Sensory Input

Okko Räsänen (okko.rasanen@aalto.fi)
Department of Signal Processing and Acoustics. Aalto University
PO Box 00076 AALTO, Finland

Abstract

Hyperdimensional computing (HDC) refers to the representation and manipulation of data in a very high dimensional space using random vectors. Due to the high dimensionality, vectors of the space can code large amounts of information in a distributed manner, are robust to variation, and are easily distinguished from random noise. More importantly, HDC can be used to represent compositional and hierarchical relationships and recursive operations between entities using fixed-size representations, making it intriguing from a cognitive modeling point of view. However, the majority of the existing work in this area has focused on modeling discrete categorical data. This paper presents a new method for mapping continuous-valued multivariate data into hypervectors, enabling construction of compositional representations from non-categorical data. The mapping is studied in a word classification task, showing how rich distributed representations of spoken words can be encoded using HDC-based representations.

Keywords: hyperdimensional computing; distributed representations; speech recognition; memory

Introduction

Hyperdimensional computing (HDC) was first introduced by Kanerva (1988) in the context of his neurally inspired memory model called sparse distributed memory (SDM). HDC is based on the idea that the distances between concepts in our minds correspond to distances between points in a very high-dimensional space (Kanerva, 2009). Since its introduction, HDC has been used successfully in modeling of analogical processing (Plate, 1995; see also Eliasmith & Thagard, 2001), latent semantic analysis (Kanerva et al., 2000), multimodal data fusion and prediction (Räsänen & Kakouros, 2014), robotics (Jockel, 2010), and, e.g., cognitive architectures (Rachkovskij et al., 2013; see also Levy & Gayler, 2008; Kelly & West, 2012) as it successfully bridges the gap between symbolic processing and connectionist systems.

In typical systems using HDC, discrete entities w_i (e.g., symbols, states or words) are represented with randomly generated binary, ternary, or continuous-valued vectors $\mathbf{y}_i$ of huge dimensionality h, typically counted in thousands (e.g., Kanerva, 1988; Kanerva et al., 2000). These vectors can have only a small number of non-zero elements (as in SDM), or they can be fully dense. In all cases, the large dimensionality of such vectors leads to a number of interesting properties (see Gallant & Okaywe, 2013, for a recent overview). Firstly, the representations are highly robust against distortions, noise, and degradation due to the distribution of information across numerous dimensions.

Secondly, the distribution of the mutual distances between all possible random vectors is tightly packed around the mean of the distances. In the case of random hypervectors with zero mean, the pair-wise linear correlation $\rho(\mathbf{y}_a,\mathbf{y}_b) \in [-1, 1]$ between almost any two randomly drawn vectors $\mathbf{y}_a$ and $\mathbf{y}_b$ is very close to zero (Kanerva, 2009). This *quasi-orthogonality* of random vectors leads to the practical property that a set of unrelated items can be represented as the sum of the hypervectors corresponding to the items in the set. For example, a set $\{w_1, w_2, w_3\}$ can be coded as $\mathbf{y}_{set} = \mathbf{y}_1+\mathbf{y}_2+\mathbf{y}_3$, and this process is usually referred to as *chunking*. The obtained representation $\mathbf{y}_{set}$ is much more similar to its components than any unrelated vectors in the hyperspace, and therefore the individual items can still be recovered from the holistic representation if the codes of all possible items are known (see Gallant & Okaywe, 2013, for a capacity analysis). In addition, HDC can overcome the superposition catastrophe of distributed representations by using invertible vector operations such as circular convolution to *bind* vectors together (Plate, 1995). For instance, correct attribute encoding of a sentence "*black cats and red balls*" could be represented with $\mathbf{y} = \mathbf{y}_{black}\otimes\mathbf{y}_{cats}+ \mathbf{y}_{red}\otimes\mathbf{y}_{balls} + \mathbf{y}_{and}$ if each unique word is assigned with a random vector and where $\otimes$ denotes the binding operation. Importantly, the dimension of the representations always stays fixed during the chunking and binding operations, ensuring that distance metrics between representations of different granularity and combinatorial complexity are always defined.

However, a major challenge in applying HDC to many real world problems has been that the world, as sensed by a number of senses (or sensors), does not give rise to inherently categorical (discrete) representations before some learning takes place. The idea of using random vectors for different inputs is only applicable after the data has been clustered or quantized into a finite number of representative states or receptive fields. Given the theoretically interesting properties of HDC, it would be useful to be able to represent non-categorical multivariate inputs such as speech in a HDC-compatible form without imposing hard quantization on the input features before further processing.

In order to address this issue, the present paper describes a method for transforming continuous multivariate data into quasi-orthogonal random vectors. The transformation maintains local distance metrics of the original feature space, allowing generalization across similar tokens, while simultaneously mapping more distant inputs into nearly orthogonal random vectors that is a requirement for the efficient use of chunking and binding operations. In comparison to the previously suggested scatter code (Stanford & Smith, 1994), the present method is not limited to binary vectors, enabling higher representational capacity in vector spaces of the same dimension. The proposed method is evaluated in a spoken word classification task using a simple prototype memory for acoustic modeling.

S-WARP mapping for multivariate data

The core of the mapping problem is that many types of data such as spectral features of speech do not come in discrete and mutually exclusive categories $w_i \neq w_j$ $(i \neq j)$ that can be assigned with unique random vectors but as multivariate observations $\mathbf{x}_t$ with varying degrees of similarity (correlation) between the vectors. The correlation is a problem because it significantly affects the coding capacity of the hyperspace as the entire idea of HDC is to operate on quasi-orthogonal representations. However, in order to generalize between different tokens of the same category, the correlation between the original features should be also reflected in the derived hyperdimensional representations, and therefore arbitrarily small differences in the input cannot lead to orthogonal codes in the hyperspace.

Given this, the minimal set of desired properties in the mapping $\mathbf{y} = f(\mathbf{x})$ from a low-dimensional space F to a high-dimensional space H can be listed as follows:

1) Local similarities between input vectors must be approximately maintained, enabling generalization towards new input tokens.
2) Distant inputs should be coded with quasi-orthogonal vectors, maximizing coding capacity of the hyperspace.
3) A continuous distance metric between original vectors should be also continuous and smooth in the hyperspace.
4) The local/distant trade-off in the requirements in 1) and 2) should be controllable.

The desired properties are illustrated in Fig. 1.

In order to approach a solution to the mapping problem, let $\mathbf{x}$ denote a feature vector of dimension $d = |\mathbf{x}|$ with feature values x_i, $i = [1, 2, \ldots, d]$ from the feature space F. In addition, let $\mathbf{M}$ denote a mapping matrix of size h x d where h is the dimension of the hyperspace H ($h >> d$). In the case of a trivial random mapping from F to H, one can initialize $\mathbf{M}$ as a randomly generated binary matrix (all values randomly set to +1 or -1) and then linearly expand the original feature vector $\mathbf{x}$ as:

$$\mathbf{y} = \mathbf{M}\mathbf{x} \tag{1}$$

This type of random mapping approximately preserves the relative distances in F (the Johnson-Lindenstrauss Lemma). However, this only makes use of a subspace $S \in H$ of the entire hyperspace due to the fixed mapping from each x_i to a set of y_j, $j \in [1, h]$. In other words, $\mathbf{M}$ acts as a single basis in H, and the distance metrics are linearly preserved. In

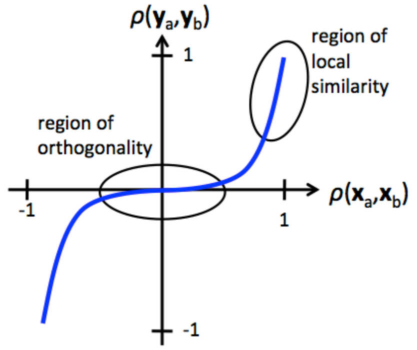

Fig. 1. An example of desired hyperspace mapping properties in terms of distance metrics. The x-axis shows the correlation $\rho(\mathbf{x}_a,\mathbf{x}_b) \in [-1, 1]$ between two data points in the low-dimensional input space and the y-axis shows the cross-correlation $\rho(\mathbf{y}_a,\mathbf{y}_b)$ between the corresponding hypervectors. Ideally, local similarity (high ρ) is carried over to the hyperspace while the hypervectors of distant inputs are independent of each other. Preservation of anti-correlations ($\rho \approx -1$) can also be beneficial for some tasks.

order to achieve orthogonalization between distant inputs, different mapping matrices $\mathbf{M}_a$ and $\mathbf{M}_b$ should be used for feature vectors $\mathbf{x}_a$ and $\mathbf{x}_b$ that are far apart in the original feature space. Simultaneously, the same mapping matrix $\mathbf{M}_{cd}$ should be used for two vectors $\mathbf{x}_c$ and $\mathbf{x}_d$ that are similar to each other in F in order to maintain similarity in H also. The problem then is the selection of the best possible mapping matrix $\mathbf{M}_i$ for each input vector $\mathbf{x}_i$. In addition, the transition between matrices $\mathbf{M}_i$ and $\mathbf{M}_j$ should be smooth so that the mapping does not introduce points of discontinuity in the distance metrics between inputs $\mathbf{x}_i$ and $\mathbf{x}_j$.

We propose that a deterministic mapping with efficient use of the entire hyperspace can be achieved as a linear combination of individual mappings. More specifically, let us define $\mathbf{M}_i$ ($i = [1, 2, \ldots, v]$) as a set of v random mapping matrices. Then the mapping $\mathbf{x} \rightarrow \mathbf{y}$ can be written as

$$\mathbf{y} = \sum_{i=1}^{v} \lambda_i \mathbf{M}_i \mathbf{x} \tag{2}$$

Since each individual mapping with a random matrix $\mathbf{M}_i$ approximately preserves the distance metrics in a linear manner, the weights λ_i can be used to control the rate of change from one basis to another (Fig. 2). From now on, we will refer to the formulation in Eq. (2) as Weighted Accumulation of Random Projections (WARP). In the absence of any external knowledge of the input, the weights λ_i of each single mapping are determined by the input vector itself:

$$\lambda_i = f(\mathbf{x}) \tag{3}$$

In other words, the hypervector is a result of v random mappings $i = [1, 2, \ldots, v]$ into the hyperspace H with each individual mapping i weighted by a value that is derived from the vector itself that is being mapped. We will refer to this self-regulated mapping as S-WARP.

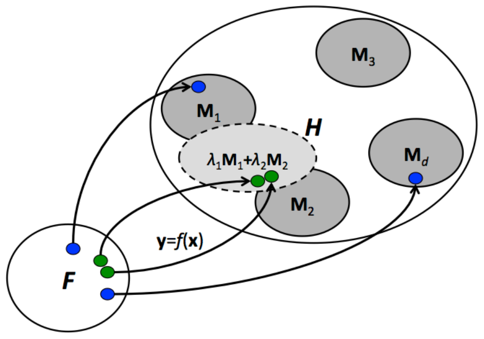

Fig. 2. A schematic view of the mapping in Eq. (2) utilizing a linear combination of multiple individual mappings. Each individual mapping matrix $\mathbf{M}_i$ acts as a pointer to a sub-space of H, and pairs of data points ending up in different sub-spaces become quasi-orthogonal with a high probability. Smooth transitions between sub-spaces are ensured by a weighting function that is a continuous function of the input itself.

Possibly the simplest way to implement S-WARP would be to use the elements of the input vector $\mathbf{x}$ directly as the weighting coefficients but then the mapping would be indifferent to the sign of the input vector, i.e., $\mathbf{y} = f(\mathbf{x}) = f(-\mathbf{x})$. This problem can be avoided by using the absolute value of the coefficients instead:

$$\lambda_i = (|x_i|^\alpha / \sum_j |x_j|^\alpha) \tag{4}$$

The additional parameter α in Eq. (4) controls the amount of orthogonalization by controlling the rate at which the hyperspace basis matrices $\mathbf{M}_i$ change when values of $\mathbf{x}$ change. When α has a high value, two vectors have to be very close in the original space F in order to end up close in the hyperspace whereas more distant vectors tend towards quasi-orthogonal representations (cf. Fig 1).

When the weights of Eq. (4) are used in the mapping described in Eq. (2), all previously listed requirements are satisfied. The mapping is also scale invariant with respect to the resulting hypervector direction, i.e., $\rho(f(\mathbf{x}_a), f(\mathbf{x}_b)) = \rho(f(\alpha\mathbf{x}_a), f(\beta\mathbf{x}_b))$, where $f(\mathbf{x})$ denotes the mapping operation and α and β are constants, while the magnitude of the resulting hypervector will be affected. This is not the case for the previously introduced scatter code (Stanford & Smith, 1994) where the direction of the vector changes if the input vector is multiplied by a constant.

However, the weighting scheme in Eq. (4) still has a shortcoming. Consider two vectors $\mathbf{x}_a$ and $\mathbf{x}_b$ with possibly different signs and scale but similar relative order of magnitudes within the set of largest elements. After applying Eq. (4), the weights λ_i become similar for the two vectors, and they are mapped using a similar set of mapping matrices $\mathbf{M}$. Since the non-linearity of the distances in the hyperspace is caused by the use of *different* weights for different vectors, the distance between the two different vectors $\mathbf{x}_a$ and $\mathbf{x}_b$ using the *similar* weights $\boldsymbol{\lambda}$ becomes linearized. With large values of α, the mapping becomes

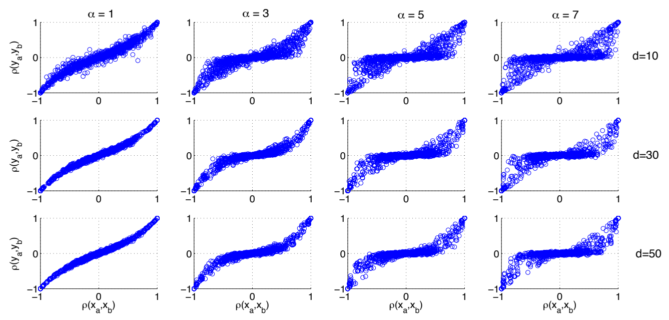

Fig. 3. Correlation $\rho(\mathbf{x}_a,\mathbf{x}_b)$ of random vectors in the original space F (x axis) and the corresponding correlation $\rho(\mathbf{y}_a,\mathbf{y}_b)$ in the hyperspace H (y axis) as a function of α in Eq. (6) (columns) and dimension d of the input vectors (rows).

dependent only on the largest magnitude elements in the input and thus the probability of linearization increases. In practice, this phenomenon limits the maximum value of α that still leads to a consistent mapping with a sufficiently high probability. The risk of linearization is also dependent on the dimension d of the original vectors.

The effects of α and d on the mapping of Eq. (2) and (4) are illustrated in Fig. 3, where correlations between pairs of randomly generated original low-dimensional vectors and the corresponding hypervectors are plotted. As can be observed, α successfully controls the non-linearity of the distances in the hyperspace, but the non-linearity breaks down for a large ratio of α/d. For increasing α, increasingly many vector pairs maintain linear or near-linear mutual distance across the mapping.

As the largest useful non-linearity factor α of a single hyperspace mapping is determined by the dimension d of the input vectors, the problem can be easily solved by simply first expanding the original d-dimensional input data into a higher dimension $h_1 > d$ using linear random mapping in Eq. (1) before applying S-WARP in Eq. (2). Another option is to recursively apply S-WARP mapping with a smaller value of α, in which case the non-linearity will gradually increase towards a desired level.

The linear expansion approach is demonstrated in Fig. 4, where random $\mathbf{x}$ of original dimension d = 10 are first mapped into a 300-dimensional space with a randomly generated fully dense expansion matrix $\mathbf{E}$ (all elements +1 or -1) according to $\mathbf{y}' = \mathbf{E}\mathbf{x}$, and then the resulting $\mathbf{y}'$ are mapped into hypervectors $\mathbf{y}$ according to Eq. (2) with weights according to Eq. (4). As can be observed, the linearization problem is now absent, confirming that the linear expansion is sufficient for avoiding the linearization artifacts occurring with small d and/or large α. In general, the method is successful at generating quasi-orthogonal representations for weakly-correlated inputs.

Spoken word classification with HDC

S-WARP was studied in word recognition from speech. Since the current goal was to study hypervectors' capability to code structural information of time-varying signals, the experiment was limited to the classification of a small vocabulary of words that had been segmented from continuous speech using the available word-level annotation of the data.

Data

The speech data consisted of 2397 utterances from the four main talkers of the CAREGIVER Y2 UK corpus (Altosaar et al., 2010). The material consists of child directed speech with an overall vocabulary size of 79 unique words (silences excluded). Each signal corresponding to an utterance was segmented into individual words using the associated word-level annotation. Separate training and testing sets were created for each of the four talkers by choosing 80% of the first words as the training samples ($N = 10423 \pm 4.6$ for each talker) and the remaining 20% as the test samples ($N = 2606 \pm 1.1$) in the order of appearance in the corpus. A total of 79 unique words occurred in the training data of which 71 also occurred in the test set. All models were always trained on the full set of 79 words.

Experimental setup

The entire word classification architecture is based on learning a hypervector prototype $\mathbf{m}_w$ for each word w in the training data, where the prototype is constructed incrementally from the short-term spectral features extracted from the acoustic realizations of the word (Fig. 5).

The audio signal corresponding to a spoken word is first fed to a pre-processing block where standard Mel-frequency cepstral coefficient (MFCC) features, including delta and delta-delta, are extracted using a 32-ms Hamming window with a step size of 10 ms (a total of 39 coefficients including energy). Each MFCC vector $\mathbf{x}_t$ is then used as an input to the hyperdimensional mapping processs (S-WARP or scatter code), yielding a hypervector $\mathbf{y}_t$ for the corresponding time frame. The temporal structure of the words is encoded with the binding operation by computing pair-wise circular convolutions $\mathbf{z}_{t,k} = \mathbf{y}_t \otimes \mathbf{y}_{t-k}^{\mathrm{P}}$ between all vectors within 250 ms from each other ($k \in [1, 2, \ldots, 25]$) (cf., Plate, 1995). In the convolution, the preceding vector is always permuted with a fixed permutation (denoted with $\mathbf{y}^{\mathrm{P}}$) in order to encode temporal order information, since otherwise $\mathbf{y}_t \otimes \mathbf{y}_{t-k} = \mathbf{y}_{t-k} \otimes \mathbf{y}_t$, i.e., making the representation invariant with respect to the direction of time.

Finally, all the obtained hypervectors $\mathbf{y}_t$ and $\mathbf{z}_{t,k}$ are additively combined to form a single hypervector $\mathbf{y}_{\text{input}}$ for the current input, and the result is summed to the existing hypervector model $\mathbf{m}_w$ for the word w in order to have an updated model $\mathbf{m}'_w$.

$$\begin{aligned} \mathbf{y}_{\text{input}} &= \sum_{t,k} \mathbf{z}_{t,k} + \sum_{t} \mathbf{y}_t \\ \mathbf{m}'_w &\leftarrow \mathbf{m}_w + \mathbf{y}_{\text{input}} \end{aligned} \tag{7}$$

As a result of processing the training data, a word model $\mathbf{m}_w$ is the sum of all word w realizations, where each realization is the sum of all frame-based hypervectors and their pair-

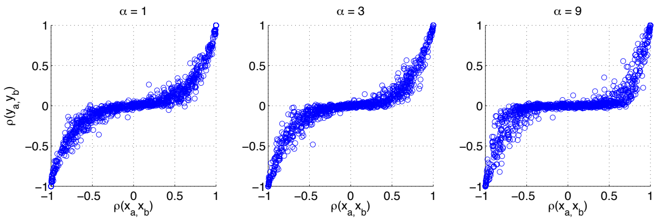

Fig. 4. Examples of cross-correlation plots for a two-stage process where the low-dimensional input vectors are first expanded to a larger dimension with a linear random mapping and then used as an input to the non-linear mapping in Eq. (2). Results for three different values of non-linearity, namely $\alpha = 1$, 3, and 9, are shown from left to right, respectively.

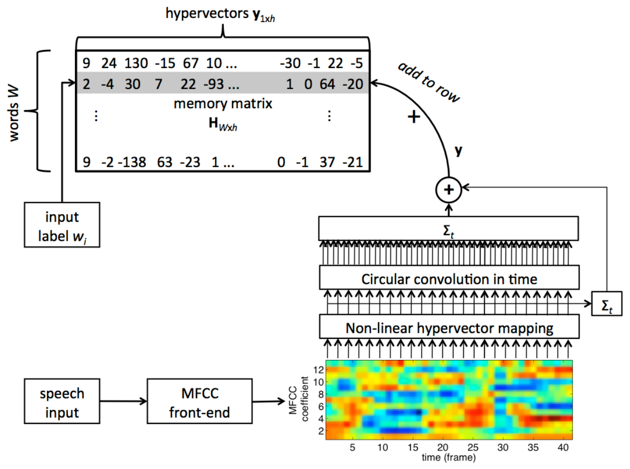

Fig. 5. A schematic view of the word recognition system used in the current experiment (training stage).

wise circular convolutions (note that the model is of same dimension as each individual feature frame or each individual realization of the word). During the training, the word identity w is always known due to labeling, and the word models $\mathbf{m}_w$ for all W unique words are accumulated as row vectors of a memory matrix $\mathbf{H}$ of size W x h. During the recognition, the input segment is again coded into $\mathbf{y}_{\text{input}}$ and the most likely word label w is obtained by computing the activation distribution $\mathbf{p}$ with

$$\mathbf{p} = \langle \mathbf{H} \rangle \mathbf{y}_{\text{input}} \tag{8}$$

where $\langle \mathbf{H} \rangle$ denotes $\mathbf{H}$ with each row normalized into a unit vector. The hypothesis w_i for the current input is determined by finding the p_i ($i \in [1, W]$) with the largest value.

The experiment was conducted using both binary scatter code and the continuous-valued S-WARP proposed in the current paper. The test was repeated for different values of the non-linearity parameter s of the scatter code, for different values of α in the present S-WARP formulation, and with and without the linear expansion layer before the non-linear mapping. Based on preliminary tests, the size of the linear expansion layer in S-WARP was always fixed to $d_E = 300$ in order to ensure that no linearization occurs for the studied values of α.

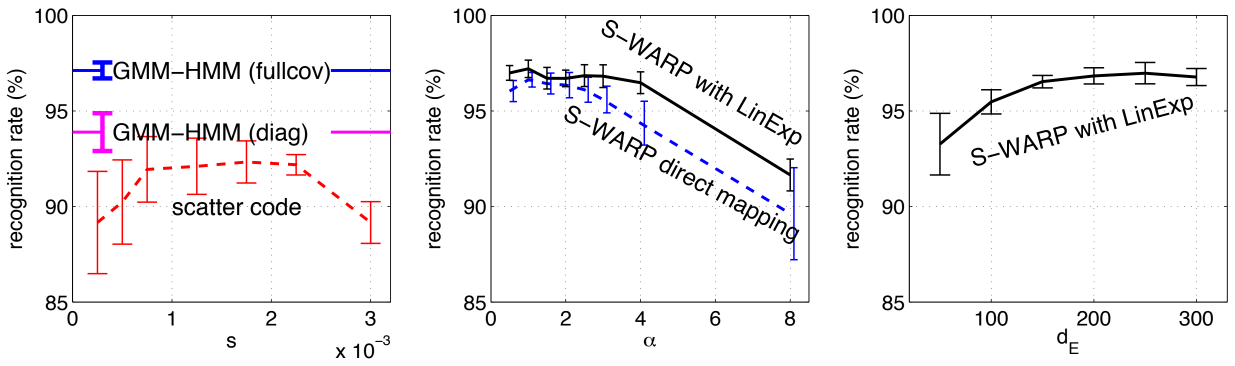

Fig. 6. Word classification accuracies (UAR %) with talker specific models (mean and one standard deviation of the results across the four talkers). Left: Scatter code (red dashed line) as a function of the non-linearity parameter s and the reference HMM systems with diagonal and full covariance matrices ("diag" and "fullcov", respectively). Center: Performance of the S-WARP with and without the linear expansion layer as a function of the α parameter and with d_E = 300. Right: S-WARP performance as a function of the linear expansion layer size d_E with fixed $\alpha = 1.5$. Hyperspace dimensionality is always set to $h = 4000$.

In the scatter code, the integer values of each dimension of the input space F are first sorted into numerical order and one of the integers is mapped into a randomly generated binary vector of dimension h. Then a code for a neighboring integer is generated by randomly choosing b locations of the first hypervector and flipping the corresponding bits. The new vector is then used as a starting point for the next integer, and the random flipping process is repeated until the entire range of the input space is covered. In this manner, the expected Hamming distance of two codes in the hyperspace is equal to $h/2*(1-(1-2/h)*(b*t/h))$, where h is the dimension of the hyperspace and t is the distance in the original space, i.e., the rate of orthogonalization is controlled by the proportion $s = b/h$ of flipped bits per iteration (Smith & Stanford, 1990). After the process is repeated for each input dimension separately, the resulting hypervectors are combined with the XOR operation (Stanford & Smith, 1994) in order to obtain the final hypervector describing the entire multivariate input vector.

Two other reference systems were also used. The basic reference was exactly the same setup as the system in Fig. 5 except that the hypervectors $\mathbf{y}$ were replaced with the original low-dimensional MFCC feature vectors $\mathbf{x}$ before the convolution and accumulation. This provides a test for the benefits of hyperdimensionality in comparison to operating in low-dimensional spaces. The second reference system was a standard Gaussian mixture -based continuous-density hidden-Markov model (GMM-HMM), one HMM trained for each word. For benchmarking purposes, the number of states and Gaussians in the HMMs were optimized directly on the test data, leading to $Q = 3$ states for all words and $M = 3$ Gaussians per mixture. The Gaussians were initialized using the k-means algorithm, and parameters were estimated using the Baum-Welch algorithm with four iterations, as this was found to perform best on the test set.

Classification performance was evaluated in terms of unweighted average recall (UAR) computed across the words occurring at least once in the test data (the mean of word-specific classification accuracies).

Results

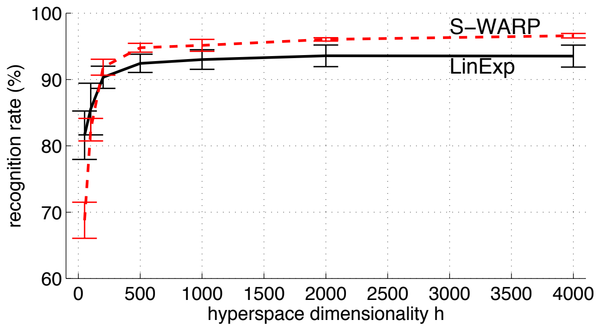

Fig. 7. The effect of hyperspace dimensionality h on the classification accuracy for linear (Eq. 1) and S-WARP (Eq. 2) mappings.

Fig. 6 shows the average results for the speaker-dependent models across all four speakers. The S-WARP template system performs at a level comparable with the HMM system using full covariance matrices, with S-WARP achieving an UAR of 97.2% ($\alpha = 1$, $d_E = 300$, $h = 4000$) while the HMM reaches on average an UAR of 97.1%. Without the linear expansion layer, S-WARP achieves an UAR of 96.6%. The scatter code achieves best performance of 92.3% correct recognitions at $s = 0.00175$ ($h = 4000$)

The word recognition accuracy using the original MFCCs is 67.9% with convolutional encoding of temporal dependencies. If only the sum of the individual MFCC vectors $\mathbf{x}_t$ is used as a model for each word (i.e., no temporal convolution), the performance drops to 31.4%. This means that the S-WARP and scatter code -based HDC representations are able to maintain information about not only the average spectrum of a word, but also the evolution of the spectrum across the word duration and a typical variability of this trajectory across different realizations of the word. The latter aspects are lost in a low-dimensional average MFCC template.

The results also reveal that the degree of distance metric non-linearity in the mapping has an effect on the overall results. This is revealed by the scatter code results (Fig. 6, left) and in the effects of α and d_E that control the orthogonalization in S-WARP (Fig. 6 middle and right). Note that the use of Eq. (2) with $\alpha = 1$ and the original data dimensionality of $d = 39$ (the MFCC coefficients) already leads to a relatively large degree of non-linearity, and therefore the performance of S-WARP without the linear expansion layer is already optimal at $\alpha = 1$. However, if α is fixed and d_E is gradually decreased from the default value of, the performance drops significantly (Fig. 6, right).

Finally, Fig. 7 shows the effect of hyperspace dimensionality on the results when using standard non-orthogonalizing random mapping and the proposed S-WARP mapping. As can be observed, the S-WARP outperforms the linear mapping with a clear margin but both benefit from an increasing hypervector dimensionality. This is expected as the chunking and binding operations assume that the data lies in a sufficiently high dimensional space.

Conclusions

The current work describes a new method for converting multivariate inputs into continuous-valued hyper-dimensional random vectors. The method attempts to solve the conflicting requirements of similarity preservation and decorrelation by performing recursive application of self-modulated random mappings. This leads to orthogonalization of the input data in a manner that still allows detection of the degree of similarity in the inputs. This is a highly relevant property for any cognitive system utilizing distributed representations, since no learning can occur without a consistent mapping from sensory input to internal representations in the memory and without the ability to measure similarity between the representations. In contrast to standard (deep) neural networks, the proposed approach does not involve learning and therefore the quality of the distributed representations is not dependent on the amount of training data available for training of the mapping network. On the other hand, S-WARP does not learn abstracted features from the data, but simply makes non-categorical data available for use in HDC –based systems.

The present work also shows that both S-WARP and scatter code –based HDC representations can be used to encode the complex time-frequency structure of spoken words by utilizing the chunking and binding operations of HDC systems. However, a more systematical approach to encoding temporally evolving multivariate inputs should be investigated in future work. In addition, it should be noted that despite its mathematical simplicity, the large number of vector operations in S-WARP makes its computational complexity non-trivial for large-scale experiments.

Acknowledgement

This research was funded by the Academy of Finland and by Tekes D2I program. The author would like to thank Unto K. Laine, Sofoklis Kakouros and Jukka P. Saarinen for their useful comments on the manuscript.

References

Altosaar, T., ten Bosch, L., Aimetti, G., Koniaris, C., Demuynck, K., & van den Heuvel, H. (2010). A Speech Corpus for Modeling Language Acquisition: CAREGIVER. *Proc. LREC'2010*, Malta.

Eliasmith, C. & Thagard, P. (2001). Integrating Structure and Meaning: a distributed model of analogical mapping. *Cognitive Science*, 25, 245–286.

Gallant, S. I., & Okaywe, T. W. (2013). Representing Objects, Relations, and Sequences. *Neural Computation*, 25, 2038–2078.

Jockel, S (2010). *Crossmodal Learning and Prediction of Autobiographical Episodic Experiences using a Sparse Distributed Memory.* Ph.D. dissertation, Department of Informatics, University of Hamburg.

Kanerva, P. (1988). *Sparse distributed memory*. Cambridge, Mass.: Bradford/MIT Press.

Kanerva, P., (2009). Hyperdimensional Computing: An Introduction to Computing in Distributed Representation with High-Dimensional Random Vectors. *Cognitive Computation*, 1, 139–159.

Kanerva, P., Kristoferson, J., & Holst, A. (2000). Random Indexing of Text Samples for Latent Semantic Analysis. *Proc. 22nd Annual Conference of the Cognitive Science Society*, Mahwah, New Jersey, pp. 1036.

Kelly, M. A., & West, R. L. (2012). From Vectors to Symbols to Cognition: The Symbolic and Sub-Symbolic Aspects of Vector-Symbolic Cognitive Models. *Proc. 34th Annual Conference of the Cognitive Science Society*, Austin, TX, pp. 1768–1773.

Levy, S. D., & Gayler, R. (2008). Vector Symbolic Architectures: A New Building Material for Artificial General Intelligence. *Proc. Conf. on Artificial General Intelligence 2008*, IOS Press, Amsterdam, pp. 414–418.

Plate, T. (1995). Holographic reduced representations. *IEEE Trans. Neural Networks*, 6, 623–641.

Plate, T. (2000). Analogy retrieval and processing with distributed vector representations. *Expert Systems: Int. J. Knowledge Eng. and Neural Networks*, 17, 29–40.

Rachkovskij, D. A., Kussul, E. M., & Baidyk, T. N. (2013). Building a world model with structure-sensitive sparse binary distributed representations. *Biologically Inspired Cognitive Architectures*, 3, 64–86.

Räsänen, O., & Kakouros, S. (2014). Modeling Dependencies in Multiple Parallel Data Streams with Hyperdimensional Computing. *IEEE Signal Processing Letters*, 21 899–903.

Smith, D. J., & Stanford, P. H. (1990). A Random Walk in Hamming Space. *Proc. IEEE Intl. Conf. on Neural Networks*, San Diego, California, pp. 465–470.

Stanford, P. H., & Smith, D. J. (1994). The Multidimensional Scatter Code: A Data Fusion Technique with Exponential Capacity. *Proc. Int. Conf. on Artificial Neural Networks (ICANN'94)*, Sorrento, Italy, pp. 1432–1435.