Modeling Dependencies in Multiple Parallel Data Streams with Hyperdimensional Computing

Okko Räsänen & Sofoklis Kakouros

Abstract—**This work presents an approach for modeling statistical dependencies in multivariate discrete sequences by using hyperdimensional random vectors. The system takes any number of parallel sequences as inputs and learns to predict the future states of these streams using the mutual dependencies between the inputs. Performance of the system is tested in an activity recognition task with data from multiple worn sensors. The results show that the approach outperforms the existing baseline results in the task and demonstrate that the system is capable to account for the varying reliability of different input streams.**

Index Terms—**activity recognition, hyperdimensional computing, machine learning, multimodal processing**

I. Introduction

MODELING statistical dependencies between multiple streams of data is relevant to many areas of signal processing and machine learning such as multisensory processing. In contrast to traditional sensor or classifier fusion (e.g., [1]), this work takes a novel perspective to the task of generic associative learning between multiple data streams by utilizing the benefits of so-called hyperdimensional computing (HDC) [2]. In the proposed HDC-based predictor (HDCP), the cross-modal context of each individual state $w_{s,t+1}$ in stream s at time t is represented as a single high-dimensional context vector $\mathbf{c}_{s,t}$ that encodes the current and preceding states of all concurrent input streams $s \in \{1, \ldots, S\}$. These context vectors are associated to the states $w_{s,t+1}$ by using a simple matrix memory that is learned from a set of training samples. As a result, the HDCP enables the estimation of the probability distribution $P(w_{s,t+1}|\mathbf{X})$ for any s given the currently observed multivariate discrete input $\mathbf{X}$.

Instead of performing matrix or tensor decomposition [3], [4], or using quaternion-based statistical analysis [5] across the multiple input dimensions, HDC makes use of the quasi-orthogonality of high-dimensional random vectors and the associated encoding operations that enable feed-forward encoding of structural relations between signal states within a fixed-dimensional vector space [2], [6]–[8]. The main advantage of the present approach is that it can automatically account for the varying reliability of different input dimensions at different temporal delays, utilizing signal states only when they have predictive power

Manuscript received March 11, 2014; revised April 14, 2014; accepted April 23, 2014; date of current version April 24, 2014. This work was supported by the Data to Intelligence (D2I) project funded by Tekes, Finland.

Both authors are with the Department of Signal Processing and Acoustics, Aalto University, P.O. Box 00076, Aalto, Finland (email: okko.rasanen@aalto.fi, sofoklis.kakouros@aalto.fi).

with respect to the presently predicted state in the dimension of interest. This allows the inclusion of multiple potentially beneficial inputs to the prediction task without risking a loss of performance if some of the inputs turn out to be uninformative. In addition, the system learns on-line. This makes it suitable for computational devices with limited storage and processing capability. For example, HDCP can be used as an adaptive recommendation engine in a mobile phone, enabling prediction of desired user actions in advance based on the available sensor and system inputs.

A. Hyperdimensional computing

HDC refers to computing with very large dimensional (d > 1000) random vectors, sometimes also referred to as Vector Symbolic Architectures (VSAs) [2], [6]-[11]. The basic idea behind HDC is that distances between concepts in our minds correspond to distances between points in a high-dimensional space [2]. Due to the high dimensionality, basically all randomly drawn hypervectors of the hyperspace are quasi-orthogonal with each other as the pair-wise distances of the vectors are tightly concentrated around the expected value of the chosen distance metric [2], [9], [10]. This means that the representations encoded by the vectors are highly tolerant against noise and gradual degradation.

Importantly, HDC provides a link between distributed and symbolic computing. For any two hyperdimensional random vectors $\mathbf{v}_i$ and $\mathbf{v}_j$ with zero mean and unit variance (or alternatively, ternary -1, 0, 1 vectors), their dot product is likely to be approximately zero, i.e., $\mathbf{v}_i^T\mathbf{v}_j \approx 0$ ($i \neq j$) [2]. This means that a set of discrete items {A, B, C} can be encoded as the sum $\mathbf{v}_{ABC} = \mathbf{v}_A+\mathbf{v}_B+\mathbf{v}_C$ so that the codes of the individual items still correlate with the set code ($\mathbf{v}_A^T\mathbf{v}_{ABC} \approx \mathbf{v}_B^T\mathbf{v}_{ABC} \approx \mathbf{v}_C^T\mathbf{v}_{ABC} >> 0$) while all other codes are orthogonal to the code (e.g., $\mathbf{v}_D^T\mathbf{v}_{ABC} \approx 0$). This type of sum-coding is referred to as "chunking" and it can be used to store tens or hundreds of items with gradual degradation of recall fidelity. As the vector dimension d increases, the number of vectors that can be discriminated from the sum increases exponentially while the number of vectors that can be simultaneously stored in the sum increases linearly [6]. All subsets such as $\mathbf{v}_{AB}$ will also have non-zero positive correlation with $\mathbf{v}_{ABC}$, enabling decoding of the constituent vectors from the set code. Due to

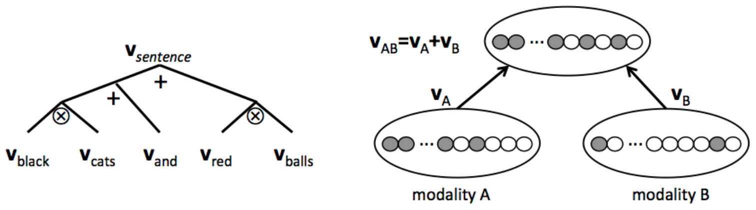

Fig.1. Left: An example encoding of the sentence "*black cats and red balls*" into a single hypervector ($\mathbf{v}_{sent.} = \mathbf{v}_{black}\otimes\mathbf{v}_{cats}+\mathbf{v}_{red}\otimes\mathbf{v}_{balls}+\mathbf{v}_{and}$) so that the adjectives become bound to the correct nouns (⊗ takes priority over +). Right: Creating a composite multimodal vector $\mathbf{v}_{AB}$ by chunking the hypervector representations $\mathbf{v}_A$ and $\mathbf{v}_B$ of the individual modalities (adapted from [11]).

the additive processing of zero-mean random vectors, the chunking operation also approximates the frequency distribution of the component vectors, the dot product $\mathbf{v}_A^T\mathbf{v}_{ABC}$ being proportional to the number of $\mathbf{v}_A$ vectors added to the set $\mathbf{v}_{ABC}$ (e.g., $0 < \mathbf{v}_A^T(\mathbf{v}_A+\mathbf{v}_B+\mathbf{v}_C) < \mathbf{v}_A^T(2\mathbf{v}_A+\mathbf{v}_B+\mathbf{v}_C)$).

Since the chunking operation is commutative ($\mathbf{v}_{AB}+\mathbf{v}_C = \mathbf{v}_{CB}+\mathbf{v}_A$), it cannot represent sequential or hierarchical structure. This problem is solved with the "binding" operation $\otimes$ that can be realized as component-wise logical operation, matrix multiplication, circular convolution, or permutation of the operand vectors [6]–[8], [11]. For example, circular convolution of two hypervectors leads to a new quasi-orthogonal vector $\mathbf{v}_{AB} = \mathbf{v}_A \otimes \mathbf{v}_B$ with $\mathbf{v}_{AB}^T\mathbf{v}_A \approx \mathbf{v}_{AB}^T\mathbf{v}_B \approx 0$. The resulting vector is of the same dimension as the inputs and the result is invertible [7], [8]. Importantly, both the chunking and binding are similarity preserving, so the code $\mathbf{v}_A+\mathbf{v}_B+\mathbf{v}_C$ (or $\mathbf{v}_A \otimes \mathbf{v}_B \otimes \mathbf{v}_C$) is similar to $\mathbf{v}'_A+\mathbf{v}'_B+\mathbf{v}'_C$ (or $\mathbf{v}'_A \otimes \mathbf{v}'_B \otimes \mathbf{v}'_C$) where $\mathbf{v}'$ refers to a noisy version of $\mathbf{v}$. When used together, binding and chunking allow recursive construction of similarity preserving compositional representations from input data (Fig. 1) so that the distances between representations and their parts are always defined within the hyperspace [6]. This also provides natural means of combining information not only across time, but also across input modalities.

The earlier work on HDC (e.g., [2], [6], [9], [10], [12]–[19]) has mainly focused on the study and application of Kanerva's Sparse Distributed Memory (SDM) [9], [10], that is a special case of an autoassociative or heteroassociative neural network. Beyond the SDM-specific studies, HDC has mostly received interest for its theoretical possibilities [6], [11], [20], in the well-known random indexing (RI) algorithm [15], [21], used for on-line latent semantic analysis, and recently in univariate discrete sequence prediction [22].

The current paper generalizes the initial idea of HDC-based sequence processing in [22] to account for prediction from any number of parallel data streams. Section 2 describes a mutual information-based time- and stream-dependent weighting scheme for HDC chunking. Performance of the HDCP is tested in an activity recognition task using data from worn sensors (section 3), showing that the proposed system is able to deal with inputs of varying reliability in the prediction task.

II. Methods

The proposed HDCP system is based on modeling the state space of each input stream in the context of the concurrent and preceding states in the other input streams. Specifically, each time-varying input stream s is assumed to be quantized into a discrete state space (sequence) $X_s = \{w_{s,1}, w_{s,2}, \ldots, w_{s,t}\}$ with each $w_s \in \{1, \ldots, N_s\}$. The overall goal of the HDCP is to derive the distribution $P(w_{s,t+1}|\mathbf{X})$ for any s of interest given the history of multivariate discrete observations $\mathbf{X} = [\mathbf{w}_{t-K}, \ldots, \mathbf{w}_{t-1}, \mathbf{w}_t]$ across the S streams.

The idea in HDCP is to represent each possible input state $w_{s,t}$ as a random unique hypervector $\mathbf{v}_s$ of dimension d and then to combine these state-specific hypervectors across time and across modalities into an overall context-vector $\mathbf{c}_{s,t}$ using the HDC chunking operation. Before the chunking, HDC binding operation is used to encode the temporal structure (order) of the states, leading to a unique quasi-orthogonal code for each unique state at each unique position. Also, the basic chunking operation of HDC is replaced by a weighted chunking of the form

$$\mathbf{v}_{\text{chunk}} = \lambda_1 \mathbf{v}_1 + \lambda_2 \mathbf{v}_2 + \ldots + \lambda_K \mathbf{v}_K \tag{1}$$

with coefficients $\boldsymbol{\lambda}$ corresponding to the statistical dependency (predictive power) of each component in the sum code. With proper setting of $\boldsymbol{\lambda}$, the relative importance of each input at each temporal distance can be encoded to the final cross-modal representation. Once the current cross-modal and temporal context is represented using the single hypervector $\mathbf{c}_{s,t}$, the vector is used to update a model $\mathbf{m}_{s,w}$ of the next state $w_{s,t+1}$ in the training data. During the prediction stage, the distribution of the forthcoming states in any s can be estimated by measuring the similarity of the current context vector $\mathbf{c}_{s,t}$ to the learned models $\mathbf{m}_{s,w}$.

In HDC, the hypervectors $\mathbf{v}$ can be binary-, ternary-, or continuous-valued with zero mean with the proportion of $\rho \in [0,1]$ values being non-zero. In the HDCP, fully dense binary representations (all values being ± 1, $\rho = 1$) are used as the experience from several different experiments suggests that the density is not critical as long as it is clearly above zero (e.g., $\rho > 0.05$ for $d > 1000$). However, the vector density should be taken into consideration if some special memory architectures such as SDM [9], [10], are used instead of linear additive memory. Also, in practical implementations of the algorithm, the use of sparse vectors can provide significant advantage in computational speed.

The next subsections will first describe how the temporal information from a single data sequence is encoded using the linear weighting scheme of Eq. (1). Then the formulation is extended to two or more parallel data streams. Finally, the use of a simple memory matrix for learning and prediction is described in 2.3

A. Encoding sequences with hyperdimensional vectors

Consider a univariate discrete sequence X = $[w_1, w_2, \ldots, w_L]$. In order to encode its structure, the first step is to represent the preceding context $[w_{t-K}, \ldots, w_{t-1}, w_t]$ of each state w_{t+1} for each t using a single context vector $\mathbf{c}_t$ of dimension d. First, a codebook of size N x d containing a hypervector $\mathbf{v}$ for each state w is generated by randomly assigning each element of each vector as +1 or -1. Given the codevectors, all the states up to a maximum lag K can be represented as a sum of the hypervectors corresponding to the states (c.f., chunking). However, this does not encode the temporal ordering of the states. Therefore, another set of K random vectors of dimension d is introduced to represent the relative temporal position attribute of each state with respect to current time t. These position vectors are referred to as $\mathbf{pos}_1$, $\mathbf{pos}_2, \ldots$, $\mathbf{pos}_k$. Encoding of the sequential order can now be performed as

$$\begin{aligned} X &= \{\ldots, w_{t-K}, w_{t-K+1}, \ldots, w_{t-1}, w_t\} \rightarrow \\ \mathbf{c}_t &= (\mathbf{v}_{t-K} \otimes \mathbf{pos}_K) + (\mathbf{v}_{t-K+1} \otimes \mathbf{pos}_{K-1}), \ldots, (\mathbf{v}_{t-1} \otimes \mathbf{pos}_1) + \mathbf{v}_t \end{aligned} \tag{2}$$

where ⊗ denotes circular convolution, i.e., by binding each state with its positional attribute [7], [8]. What the binding effectively does is that it generates a unique code vector for each state w at each temporal lag k, leading to a unique $\mathbf{c}_t$ for each possible n-tuple of states. As a result, the $\mathbf{c}_t$ will correlate with other encodings that share a subset or all of the same states in the same relative positions.

However, the encoding in Eq. (2) is still not optimal as it will give equal representational power to the states at all temporal distances. Therefore, the Eq. (2) is revised as

$$\mathbf{c}_t = \lambda_0 \mathbf{v}_t + \textstyle\sum_{k=1}^{K} \lambda_k (\mathbf{v}_{t-k} \otimes \mathbf{pos}) \tag{3}$$

so that each lag k becomes associated with a specific weighting coefficient λ_k. In principle, the weights $\boldsymbol{\lambda}$ could be iteratively estimated using the performance of the entire HDCP-system as the target criterion. However, we propose a computationally efficient one-pass approximation to the problem, namely the degree that the joint probability of states w_{t-k} and w_t differs from total randomness (or statistical independence) at each lag (see also [22]). For sequential data, such measure is known as the mutual information function (MIF; [23]):

$$\lambda_k = \sum_{w,w'} P_k(w,w') \log_2 \frac{P_k(w,w')}{P(w)P(w')} \tag{4}$$

In the equation, $P_k(w,w')$ is the joint probability for an ordered state pair $\{w, w'\}$ with the states being k time frames apart from each other. The use of MI as the weighting coefficient is natural since it will decay to zero as the temporal distance increases to a point where no dependencies exist anymore. As a special case, the decay of MI will correspond to a geometric decay $\lambda_k = \lambda_0^k$ ($\lambda_0 < 1$) for a first order Markov process. In general, the more there is predictive power at the given distance, the more the corresponding states will contribute to the overall $\mathbf{c}_t$.

B. Representing crossmodal contexts

In the case of multiple data streams $s \in \{1, 2, \ldots, S\}$ and the associated discrete multivariate input $\mathbf{X} = [X_1^T, X_2^T, \ldots, X_S^T]$, the goal is to represent the temporal and crossmodal context of each state $w_{s,t}$ in each stream s at time t by taking into account the recent history of the states in all of the input streams. By performing chunking also across the parallel input streams, Eq. (3) can be generalized as

$$\mathbf{c}_{s_2,t} = \sum_{m=1}^{M} \left(\lambda_0^{s_1|s_2} \mathbf{v}_{s_1,t} \right) + \sum_{s_1=1}^{S} \sum_{k=1}^{K} \left(\lambda_k^{s_1|s_2} (\mathbf{v}_{s_1,t-k} \otimes \mathbf{pos}_k) \right) \tag{5}$$

i.e., as the sum of all states at all temporal lags up to the maximum lag K and across all parallel streams S. Since the statistical dependencies between any two data streams depend on the stream contents, the lag-specific weights λ_k have been generalized to a form $\lambda_k^{s1|s2}$ where k is the number of time steps that s_2 is delayed with respect to s_1 (note that $\lambda_k^{s1|s2} \neq \lambda_k^{s2|s1}$ if $s_1 \neq s_2$). Also, $\mathbf{c}_{s,t}$ is now also dependent on s as different streams have different predictive power with respect to each other.

In the multi-stream case, the MIF-based weights can be computed according to

$$\lambda_k^{s_1|s_2} = \sum_{w_{s_1}, w'_{s_2}} P_k(w_{s_1}, w'_{s_2}) \log_2 \frac{P_k(w_{s_1}, w'_{s_2})}{P(w_{s_1})P(w'_{s_2})} / H_{s_1} \tag{6}$$

with $P_k(w_{s1}, w'_{s2})$ denoting the joint probability for a cross-stream ordered state pair $\{w_{s1}, w'_{s2}\}$ with the states being k time frames apart from each other. In Eq. (6), the normalization factor H_{s_1} is the entropy of stream s_1:

$$H_{s_1} = -\sum_{w_{s_1}} P(w_{s_1}) \log_2 P(w_{s_1}) \tag{7}$$

The use of entropy ensures that $\lambda_k^{s_1|s_2}$ always scales between 0 and 1 despite the type of quantization and the number of states in each stream, making the weights comparable across different pairs of data streams.

C. Predicting with context hypervectors

The goal of the learning stage is to associate the context vectors $\mathbf{c}_{s,t}$ to the following states $w_{s,t+1}$ in order to enable prediction. Instead of applying a parametric model to the conditional distribution $P(w_{s,t+1}|\, \mathbf{c}_{s,t})$, the chunking principle is again utilized. More specifically, a prototype model vector $\mathbf{m}_{s,w}$ of dimension d is constructed for each state $w_s \in \{1, \ldots, N_s\}$ as the sum of the context vectors $\mathbf{c}_{s,t}$ preceding the w_s in the data:

$$\mathbf{m}_{s,w} = \sum_{t=1}^{T} \mathbf{c}_{s,t} \delta(w_{s,t+1} = w_s) \tag{8}$$

In Eq. (8), t runs across the entire training data and $\delta(w_{s,t+1} = w_s) = 1$ iff $w_{s,t+1} = w_s$, . Due to the quasi-orthogonality of the vectors $\mathbf{c}_{s,t}$, the model $\mathbf{m}_{s,w}$ approximates the distribution of the context vectors $\mathbf{c}_{s,t}$ preceding w_s similarly to the word-context modeling in RI [15], [21] (see also [6]).

Finally, a pseudo-probability distribution ($p \in [-1, 1]$) for the state $w_{s,t+1}$ can now be obtained by comparing the currently observed $\mathbf{c}_{s,t}$ with all $\mathbf{m}_{s,w}$ corresponding to the possible states w_s. If the $\mathbf{m}_{s,w}$ are stored as rows of a memory matrix $\mathbf{M}_s$ of size N_s x d and the rows and $\mathbf{c}_{s,t}$ are normalized to unit vectors (denoted with $\langle \mathbf{x} \rangle$), the distribution for the next state can be obtained by taking the inner product

$$\mathbf{p}_{s,t} = \langle \mathbf{M}_s \rangle \langle \mathbf{c}_{s,t} \rangle \tag{9}$$

with the most likely next state $w_{s,t+1}$ defined as

$$\arg_{w_s} \max(p_{s,t,w_s} \in \mathbf{p}_{s,t}, w_s \in \{1,2,...,N_s\}). \tag{10}$$

TABLE I

RESULTS FOR DIFFERENT DATA STREAM COMBINATIONS. UARN REFERS TO THE HDCP WITHOUT THE MI-BASED WEIGHTS OF EQ. (6). GREY BACKGROUND REFERS TO STATISTICALLY SIGNIFICANT DIFFERENCE BETWEEN UAR AND UARN ($P < 0.05$, PAIRED T-TEST).

acc_wrist	x				x	x	x	
acc_hip		x			x			
skintemp			x			x		
RIP				x			x	
UAR (%)	68.4	75.6	13.5	53.1	81.7	68.3	73.0	
UARn (%)	68.2	75.6	13.5	52.9	81.2	54.1	73.4	
acc_wrist					x	x	x	x
acc_hip	x	x		x	x		x	x
skintemp	x		x	x	x	x		x
RIP		x	x	x		x	x	x
UAR (%)	75.6	76.3	53.3	76.4	81.7	72.9	82.1	**82.2**
UARn (%)	60.1	75.8	40.5	65.1	72.6	62.6	81.8	74.9

TABLE II

THE CONFUSION MATRIX FOR ACTIVITY RECOGNITION ACCURACIES (UAR %) ACROSS ALL THE TWELVE TEST SUBJECTS. ROWS: GROUND TRUTH. COLUMNS: HYPOTHESIZED CLASS.

	bike	**ex-bike**	**foot-ball**	**lie**	**nordic walk**	**row**	**run**	**sit/ stand**	**walk**
bike	82.5	3.9	3.0	0.5	0.6	3.1	0.1	3.9	2.4
ex-bike	3.6	89.0	0.2	0.1	0.0	4.9	0.0	1.5	0.5
football	5.2	0.4	63.0	0.1	3.3	1.2	9.1	5.4	12.3
lie	0.4	0.8	0.1	87.7	0.0	0.4	0.0	10.3	0.1
nordic	0.2	0.0	3.4	0.0	89.2	0.1	0.3	1.0	5.8
row	0.8	0.3	0.4	0.0	0.0	96.4	0.1	1.7	0.3
run	0.2	0.0	6.3	0.0	1.4	0.0	90.3	0.5	1.3
sit/stand	2.3	1.9	0.8	7.7	0.2	1.4	0.1	84.7	1.0
walk	4.6	2.1	9.4	0.9	6.2	2.0	0.1	18.0	56.7

III. EXPERIMENTS

The performance of HDCP was tested in a context recognition task where the inputs consist of four parallel body-worn sensory signals and a fifth labeling stream related to the concurrent physical activity of the test subject. The task of the system is to learn the correspondence between sensory inputs and the physical activity.

A. *Material and evaluation*

Palantir Context Data Library 2004 [24], [25], was used in the experiments. The data consist of approximately 68 hours of recordings from 12 test subjects wearing a large number of sensors while performing a number of physical activities in controlled and uncontrolled conditions (see Table 2 for the activities). The following sensors were used in the current study: 3–D accelerometers from the hip and wrist, skin temperature, and two respiratory inductive plethysmogram (RIP) belts worn around the waist and thorax.

The data were divided into 12 folds for training and testing, always training HDCP with 11 subjects and testing with the remaining one. For each signal frame of the test set, HDCP was asked to provide the most likely activity using the concurrent sensor information. The overall performance was measured in terms of unweighted average recall (UAR). As in [24], *sitting* and *standing* activities were combined into a single class as the sensor placement used in the data recording does not differentiate between these two [24], [25].

B. *Pre-processing of sensor data*

The sensors were used to form four distinct input streams: $\boldsymbol{s_1}$**:** acceleration from the wrist (x-, y-, and z-directions), $\boldsymbol{s_2}$**:** acceleration from the hip (x, y, z), $\boldsymbol{s_3}$**:** skin temperature, and $\boldsymbol{s_4}$**:** a combined 2–D signal from the waist and thorax RIPs. All signals were downsampled to 50 Hz and the FFT spectrum was computed for each signal dimension of each stream using a sliding Hamming window of 1 s with 100-ms window shifts (10 Hz rate). The spectra across the signal dimensions in each stream (e.g., x, y, and z acceleration) were then concatenated into one long feature vector and PCA was used to compress the dimensionality of the vectors so that 95% of the variance was retained in the features.

Finally, the features of each stream were vector quantized into 132 unique states $w_s \in \{1, 2,\ldots, 132\}$, $s \in \{1, 2, 3, 4\}$. The codebooks were created using the k-means algorithm, first making individual codebooks for each activity class using 3000 random feature vectors of training data and then combining the class-specific codebooks into a single large codebook. The fifth data stream $\boldsymbol{s_5}$ consisted of the physical activity labels (Table 2) synchronously sampled with the sensory data. The preceding activity states were never included in the context vector so that the activity prediction was always performed purely on the sensor data.

C. *Results*

The experiment was run with HDCP using hypervector dimension of $d = 2000$, a maximum lag of $K = 10$ (1 sec), and across 3 trials with new random codevectors. Table 1 shows the mean UAR (%) for different subsets of the four sensors with and without the weighting in Eq. (6). Table 2 shows the confusion matrix across all folds using all sensors. The mean UAR of 82.2% compares well against the previous state-of-the-art result of 75.4% reported in [24] using the same dataset and after comparing multiple standard classifiers in the task (the best result in [24] was obtained using a hybrid of a decision-tree and a multilayer perceptron systems). Importantly, the HDCP performance increases or stays the same with increasing number of sensors, confirming that the MI-based weighting is able to account for varying reliability of different inputs and ensuring that the performance does not degrade due to the addition of non-useful inputs (skin temperature in this case).

IV. Conclusions

An HDC-based generic framework for modeling dependencies across multiple parallel data streams was presented in this work. The system was tested in an activity recognition task using wearable sensors, achieving state-of-the-art recognition performance using multiple sensors channels and showing systematic improvement due to the inclusion of additional sensors. The result suggests that the system is successful in utilizing different inputs according to their relevance in the prediction task, making it robust for applications where a priori sensor selection may not be feasible.

References

[1] H. Wu, "*Sensor Data Fusion for Context-Aware Computing Using Dempster Shafer Theory*," Ph.D. dissertation, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 2003.

[2] P. Kanerva, "Hyperdimensional Computing: An Introduction to Computing in Distributed Representation with High-Dimensional Random Vectors," *Cognitive Computation*, vol. 1, no. 1, pp. 139–159, Jan. 2009.

[3] A. Cichocki, D. Mandic, A-H. Phan, C. Caiafa, G. Zhou, Q. Zhao, and L. De Lathauwer, "Tensor Decompositions for Signal Processing Applications. From Two-way to Multiway Component Analysis," to appear in *IEEE Signal Process. Mag.*, 2014.

[4] Q. Zhao, G. Zhou, T. Adali, L. Zhang, and A. Cichocki, "Kernelization of Tensor-Based Models for Multiway Data Analysis," *IEEE Signal Process. Mag.*, vol. 30, no. 4, pp. 137–148, July 2013.

[5] C. Took and D. Mandic, "Augmented second-order statistics of quaternion random signals," *Signal Processing*, vol. 91, no. 2, pp. 214–224, Feb. 2011.

[6] S. I. Gallant and T. W. Okaywe, "Representing Objects, Relations, and Sequences," *Neural Computation*, vol. 25, no. 8, pp. 2038–2078, Apr. 2013.

[7] T. Plate, "Holographic reduced representations," *IEEE Trans. Neural Netw.*, vol. 6, no. 3, pp. 623–641, May 1995.

[8] T. Plate, "Analogy retrieval and processing with distributed vector representations," *Expert Systems: The Int. Journal of Knowledge Eng. and Neural Networks*, vol. 17, no. 1, pp. 29–40, 2000.

[9] P. Kanerva, *Sparse Distributed Memory*. The MIT Press, Cambridge, MA, 1988.

[10] P. Kanerva, "Sparse Distributed Memory and Related Models," In M.H. Hassoun (Ed.): *Associative Neural Memories: Theory and Implementation*, pp. 50–76, New York: Oxford University press, 1993.

[11] D. A. Rachkovskij, E. M. Kussul, and T. N. Baidyk, "Building a world model with structure-sensitive sparse binary distributed representations," *Biologically Inspired Cognitive Architectures*, vol. 3, no. 1, pp. 64–86, Jan. 2013.

[12] J. T. Abbott, J. B. Hamrick, and T. L. Griffiths, "Approximating Bayesian inference with a sparse distributed memory system," in *Proc. 35th Annual Conference of the Cognitive Science Society*, Berlin, Germany, 2013, pp. 1686–1691.

[13] Y-S. Hong and S-S. Chen, “Character Recognition in a Sparse Distributed Memory,” *IEEE Trans. Syst., Man, Cybern.*, vol. 21, no. 3, pp. 674–678, May/June 1991.

[14] S. Jockel, *Crossmodal learning and prediction of autobiographical episodic experiences using a sparse distributed memory*. Doctoral Thesis, University of Hamburg, Department of Informatics, 2010.

[15] P. Kanerva, J. Kristoferson, and A. Holst, “Random Indexing of Text Samples for Latent Semantic Analysis,” *Proc. 22nd Annual Conference of the Cognitive Science Society*, Mahwah, New Jersey, 2000, pp. 1036.

[16] M. Mendes, M. Chrisostomo, and A.P. Coimbra, “Robot Navigation Using a Sparse Distributed Memory,” in *Proc. IEEE Int. Conf. Robotics and Automation*, Pasadena, CA, USA, 2008, pp. 53–58.

[17] R. W. Prager and F. Fallside, “The Modified Kanerva Model for Automatic Speech Recognition,” *Computer Speech and Language*, vol. 3, no. 1, pp. 61–81, Jan. 1989.

[18] U. Ramamurthy, S. K. D’Mello, and S. Franklin, “Modified sparse distributed memory as transient episodic memory for cognitive software agents,” in *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics,* The Hague, Netherlands, 2004, pp. 5858–5863.

[19] B. Ratitch and D. Precup, “Sparse distributed memories for on-line value-based reinforcement learning,” *Lecture Notes in Computer Science,* vol. 3201, pp. 347–358, 2004.

[20] P. H. Stanford and D. J. Smith, “The Multidimensional Scatter Code: A Data Fusion Technique with Exponential Capacity,” in *Proc. Int. Conf. on Artificial Neural Networks (ICANN’94)*, Sorrento, Italy, vol. 2, 1994, pp. 1432–1435.

[21] M. Sahlgren, “An introduction to random indexing,” in *Proc. Methods and Applications of Semantic Indexing Workshop, 7th Int. Conf. on Terminology of and Knowledge Engineering*, Copenhagen, Denmark, 2005, pp. 1–9.

[22] O. Räsänen and J .P. Saarinen, “Sequence Prediction with Sparse Distributed Hyperdimensional Coding Applied to the Analysis of Mobile Phone Use Patterns,” unpublished.

[23] W. Li, “Mutual information functions versus correlation functions,” *J. Statist. Physics*, vol. 60, no. 5/6, pp. 823–837, Sep. 1990.

[24] M. Ermes, J. Pärkkä, J. Mäntyjärvi, and I. Korhonen, “Detection of Daily Activities and Sports with Wearable Sensors in Controlled and Uncontrolled Conditions,” *IEEE Trans. Inf. Technol. Biomed.*, vol. 12, no. 1, pp. 20–26, Jan. 2008.

[25] J. Pärkkä, M. Ermes, P. Korpipää, J. Mäntyjärvi, J. Peltola, and I. Korhonen, "Activity Classification Using Realistic Data From Wearable Sensors," *IEEE Trans. Inf. Technol. Biomed.*, vol. 10, no. 1, pp. 119–128, Jan. 2006.