Hyperdimensional Biosignal Processing: A Case Study for EMG-based Hand Gesture Recognition

Abbas Rahimi*, Simone Benatti†, Pentti Kanerva‡, Luca Benini†§, Jan M. Rabaey*
*EECS Department, University of California, Berkeley. Email: {abbas, jan}@eecs.berkeley.edu
†DEI, University of Bologna, Italy. Email: {simone.benatti, luca.benini}@unibo.it
‡Redwood Center for Theoretical Neuroscience, University of California, Berkeley. Email: pkanerva@berkeley.edu
§D-ITET, Integrated System Laboratory, ETHZ, Zurich, Switzerland. Email: luca.benini@iis.ee.ethz.ch

***Abstract*—The mathematical properties of high-dimensional spaces seem remarkably suited for describing behaviors produces by brains. Brain-inspired hyperdimensional computing (HDC) explores the emulation of cognition by computing with hypervectors as an alternative to computing with numbers. Hypervectors are high-dimensional, holographic, and (pseudo)random with independent and identically distributed (i.i.d.) components. These features provide an opportunity for energy-efficient computing applied to cyberbiological and cybernetic systems.**

We describe the use of HDC in a smart prosthetic application, namely hand gesture recognition from a stream of Electromyography (EMG) signals. Our algorithm encodes a stream of analog EMG signals that are simultaneously generated from four channels to a single hypervector. The proposed encoding effectively captures spatial and temporal relations across and within the channels to represent a gesture. This HDC encoder achieves a high level of classification accuracy (97.8%) with only $1/3$ the training data required by state-of-the-art SVM on the same task. HDC exhibits fast and accurate learning explicitly allowing online and continuous learning. We further enhance the encoder to adaptively mitigate the effect of gesture-timing uncertainties across different subjects endogenously; further, the encoder inherently maintains the same accuracy when there is up to 30% overlapping between two consecutive gestures in a classification window.

I. Introduction

Over the past decades, the semiconductor industry has been immensely successful at increasing computing power while reducing cost and energy consumption. This has been achieved thanks to availability of efficient predictable CMOS devices supporting deterministic operations. Unfortunately energy efficiency of CMOS devices is challenged when scaling down to nanometer dimensions. Maintaining a deterministic model of computing ultimately puts a lower bound on the amount of energy scaling that can be obtained. This bound is primarily set by the variability and reliability of the devices.

It is therefore worth exploring alternative computational models that enable further size and energy scaling by abandoning the deterministic requirement. Brain-inspired information processing architectures provide significant increase in energy efficiency, asymptotically approaching the efficiency of brain computation, while aligning well with the variability of nanoscale devices [1], [2]. Our approach is focused on a computational theory called hyperdimensional computing (HDC) [5]. In this formalism, information is represented in high-dimensional vectors called hypervectors. The mathematical properties of high-dimensional spaces correlate strongly with behaviors controlled by the brain [3], [4], [5], [6], hence HDC explores the emulation of cognition by computing with hypervectors as an alternative to computing with numbers. Hypervectors are high-dimensional, e.g., 10,000 dimensions (D = 10,000), (pseudo)random with i.i.d. components, and holographic. It means that every piece of information contained in the hypervector is distributed equally over all the components. Such hypervectors can then be mathematically manipulated to not only classify but also to bind, associate, and perform other types of cognitive operations in a straightforward manner [7]. In addition, these mathematical operations also ensure that the resulting hypervector is unique and thus learning can take place in a single shot.

By requiring very low energy, hyperdimensional computing is a prime candidate for applications such as wearable biosignal processing and cybernetic systems. In addition, HDC has some unique properties and features that make it extremely well matched to emerging 3D nanoscale technology. Key properties include: (1) HDC paradigm is universal and complete. (2) In contrast to other neuro-inspired approaches, in which learning is separate from execution, learning in HDC shares its constructs with execution, is relatively lightweight, and can be realized in an online fashion on a small low-energy device. (3) By its very nature, HDC is extremely robust in the presence of component variation, defects and failure, and it tolerates noise in the signal leading to ultra low-energy computation. (4) HDC is memory-centric by manipulating and comparing large patterns, within the memory; operations are either local or can be performed in a distributed fashion.

HDC has been used for language recognition as well as text classification solely from a stream of input letters. More specifically, HDC can identify the language of unknown sentences from 21 European languages [8], [2], and classify Reuters news articles to eight topics [9] with very high accuracy. In this paper, we show how HDC can be used for biosignal processing given a set of parallel and analog streaming inputs. Accordingly, we develop an encoding algorithm using HDC for hand gesture recognition from a stream of EMG signals. Our algorithm encodes a stream of analog EMG signals that are simultaneously generated from four channels to a single hypervector representing a hand gesture. The proposed encoding effectively captures spatial and temporal correlations across and within the channels to describe a gesture. Our proposed HDC surpasses the state-of-the-art support vector machine (SVM) [10] for the hand gesture recognition in four

978-1-5090-1370-8/16/$31.00

aspects: (1) HDC exhibits an average recognition accuracy of 90.8% (2% higher than SVM) by only encoding spatial correlation across the four channels. (2) Encoding also the temporal correlation by considering consecutive samples over time, boosts the accuracy of HDC to 100%, and on average to 97.8% (8.1% higher than SVM). We further enhance the encoder to adaptively mitigate the effect of gesture-timing uncertainties by endogenously observing the distance measures between the encoded input patterns and learned patterns. (3) HDC maintains the aforementioned accuracy when there is up to 30% overlapping between two gestures in a window of classification, even with reduced dimensionality of $D = 6{,}000$. (4) For the aforementioned comparisons, 25% of dataset is used for training both classifiers. HDC learns quickly, making it a prime candidate for online and continuous learning. For accuracy on par with HDC, SVM requires 3.2× as much training data. The algorithms and techniques described in this paper are all publicly released.[1]

This paper is organized as follows. In Section II, we describe EMG signal processing and its background, including data acquisition, preprocessing, system description, and finally the state-of-the-art SVM gesture classification. In Section III, we introduce hyperdimensional computing and discuss how its operations can be used to form a new classifier. In Section IV, we present our algorithm for EMG-based hand gesture recognition using hyperdimensional computing. In Section V, we provide more experimental results. Section VI concludes this paper.

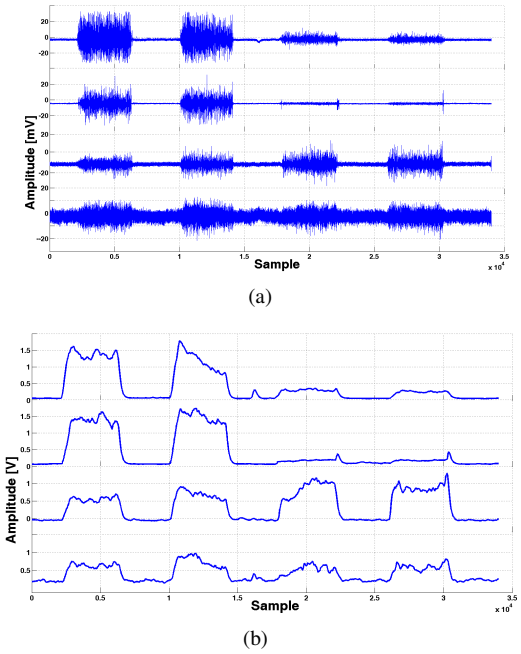
II. Electromyography (EMG)

A. EMG Signal and Acquisition

The muscular contractions are generated by the electrical activity of nerve cells called motoneurons. Their cell bodies are located in the spinal cord and their axons are directly connected to the target muscles. The stimulus that generates a muscular contraction propagates from the brain cortex to the target muscles as an electrical potential, named action potential (AP). APs are generated by the passage of Na+ and K+ ions along nerve cell membranes. As a result of this ion flow, the nerve impulses propagate towards the muscle cells and start the contractions [11]. The EMG signal represents the monitoring of this electrophysiological activity. The signal results from the superposition of all the APs of the cells underlying a couple of metal electrodes, placed on the skin and aligned parallel to the muscle fibers. The surface EMG electrodes are made by two conductive plates each one connected to the inputs of a differential amplifier that sense the action potential of muscular cells.

The maximum amplitude of this signal is 20 mV (-10 to +10) depending on the dimension of the muscle fibers, on the distance between the muscle and the electrodes and on the electrode properties. Signals of this kind are also very noisy and difficult to manage even if the maximum bandwidth does not exceed 2 kHz. The main causes are noise from motion artifacts, fiber crosstalk, electrical equipment and the floating ground of the human body.

Hence, the typical EMG acquisition in high-end gesture recognition applications is based on active analog sensors that provide a high-quality signal [12]. Such sensors represent the commercial solution for EMG acquisitions, both in research and industrial applications. These sensors perform a full-analog signal conditioning based on a bandpass discrete filter, an instrumentation amplifier with a high gain stage, and an offset cancellation feedback circuit. The bandwidth of the Ottobock [12] sensor is 90–450 Hz with a further notch filter for the 50 Hz. This is because the sensors for the classification of the gestures do not need extensive frequency information but a clear low-noise signal is preferable. The EMG raw signal is a zero-mean differential signal and the preprocessing is done in hardware by the internal circuitry of the active sensors. The EMG differential signal is integrated and a low pass filter is applied to extract the envelope of the signal. Furthermore a notch filter is applied to remove the residual power-line interference. Fig. 1 shows the raw signal acquired from the electrodes (a) and the enveloped output of the active EMG sensor (b).



(a)

(b)

Fig. 1. EMG Signal: (a) Raw acquired data, (b) Enveloped data.

B. Background and Related Work

The robustness and the reliability are major requirements in the design of EMG gesture recognition systems. In the commercial devices, used in prosthetics [13] or telesurgery [14], the recognition of the muscular activity is based on threshold detections. The recognition of the users' intended movements are encoded in predefined sequences of muscular contractions related to the wrist flexion and extension. Despite its robustness, this approach suffers from several drawbacks, since it is a unnatural way of interaction, requiring a high-level of concentration and a long time to learn. Instead, pattern recognition approaches aim to address these limits by proposing a natural way to recognize hand gestures based on EMG.

[1] https://github.com/abbas-rahimi/HDC-EMG

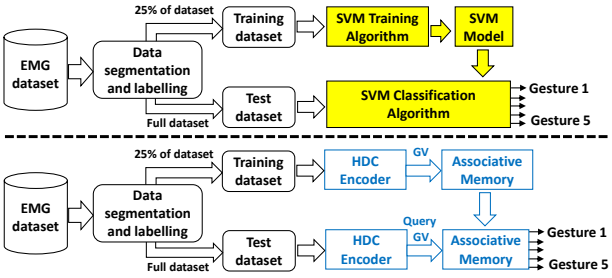


Fig. 2. Training and classification scheme for SVM and HDC.

Several studies focused on both acquisition setup and signal processing algorithms required in this application. The number of electrodes and their placement [15] has a high impact on the design of an EMG hand gesture recognition system. Moreover, many studies have focused on machine learning techniques for EMG-based gesture recognition [16]. These techniques include both linear and nonlinear methods, such as LDA [17], SVM [18], [10], ANN [19], and a combination of two heterogeneous classifiers to obtain a better classification accuracy [20], [21]. Results show accuracies around and beyond 90% with the use of feature extraction techniques to improve the performance. The main limit of these typical machine learning approaches is the high number of training data needed to achieve such accuracies. We will show that HDC can reach the same accuracy with much less training data.

C. System Description and Experimental Setup

EMG data acquisition is based on a four sensors that cover the muscles involved in hand movement from a physiological point of view. The muscles of the forearm are divided in four groups, and the use of surface EMG sensors requires that muscle near the skin surface must be targeted for the classification. Functionally, we can separate the forearm muscles in two classes: muscles in the internal part of the forearm (flexor radialis carpi, palmares longus, flexor carpi ulnaris, flexor superioris digitalis), involved in flexion movements, and muscles placed in the external part of the forearm (externsor comunis digitorum, extensor digiti minimi, externsor carpi ulnaris), involved mainly in extension movements.

By placing sensors on the flexor carpi radialis, flexor carpi ulnaris, extensor digitorum communis and extensor carpi ulnaris we obtained a good differentiation in classification patterns. The assumption for a multi-gesture control system is that the set of signals and features describing a given state of muscular activation are different from one state of activation to another.

The dataset used in this paper is based on the collection of the EMG signals of the most common hand gesture used in daily life. The selected gestures are: closed hand, open hand, 2-finger pinch, and point index. The classification includes also the rest position of the hand, recorded between two subsequent gestures. The data were collected from nine subjects [22]. Here, we use a subset of the data for five subjects. The collected sequences are composed of 10 repetitions of muscular contraction three seconds each. Between each contraction there are three seconds of rest. The gestures are sampled at 500 Hz. Subjects wear an elastic strip with the four EMG sensors [12]. See Fig. 1.

The theoretical framework of the SVM is the evolution of the logistic regression methods, through the application of the large margin classification. The goal of training is to define the optimal separation hyperplanes between classes by minimizing a cost function. Such hyperplanes are made up of a subset of the input data, and their vectors are called support vectors (SVs). The SVs are calculated through the the solution of a convex optimization problem [23] that produces the global minimum of the cost function.

Due to its robustness and the good results reported in the literature, SVM has become the classification technique of choice for EMG-based gesture recognition [24], [25], [26]. The classification algorithm calculates the distance of the input data from this decision boundary. When the two classes are not linearly separable the data space is mapped to an higher-dimensional space through a *kernel trick* to define a similarity function between the margin hyperplane and the classification data. The classification algorithm is based on a decision function that calculates the distances of an input vector with all the SVs. In particular, the formula of the decision function to classify a new input instance is:

$$f(\mathbf{x}) = \sum_{i=1}^{N} y_i \alpha_i K\langle \mathbf{x}, \mathbf{s}_i \rangle - \rho \quad \begin{cases} f(\mathbf{x}) > 0, \mathbf{x} \in Cl_1 \\ f(\mathbf{x}) < 0, \mathbf{x} \in Cl_2 \end{cases} \tag{1}$$

where Cl_1 and Cl_2, are the two classes, $\mathbf{x} \in \mathbb{R}^k$ is the input features vector, $\mathbf{s}_i \in \mathbb{R}^k$, $i = 1, ..., N$ are the support vectors, α_i are the support values, while y_i denotes the class they reference ($y_i = +1$ for Cl_1, $y_i = -1$ for Cl_2) and $K\langle \cdot, \cdot \rangle$ denotes the kernel function.

Fig. 2 shows training and testing (classification) flows for SVM. The EMG dataset is labeled using a threshold assigning the gesture labels to the input EMG data. We use 25% of this dataset to generate training data using a uniform random sampling. This training session can be performed offline. In Section V-B, we explore the trade-offs with increasing the fraction of training data. These labeled data are the input of the training algorithm that builds the SVM model (i.e., the list of the SVs). The decision function (1) calculates the label of an input vector comparing its distance from the SVs that represent the decision boundary The output is the label associated with a decoded gesture. The SVM and the HDC are trained and tested on the same data as shown in Fig. 2.

III. Hyperdimensional Computing Background

The brain's circuits are massive in terms of numbers of neurons and synapses, suggesting that large circuits are fundamental to the brain's computing. Hyperdimensional computing [5], [6] explores this idea by looking at computing with ultra-wide words – that is, with very high-dimensional vectors, or hypervectors. There exist a huge number of different, nearly orthogonal hypervectors with the dimensionality in the thousands (D = 10,000) [27]. This lets us combine two such hypervectors into a new hypervector using well-defined vector space operations, while keeping the information of the two with high probability.

Hypervectors are made using random indexing [3], [4] with operations akin to multiplication, addition, and permutation that form an algebra over the vector space (e.g., a field). Random indexing represents information by projecting data onto hypervectors. It is incremental, scalable, and computes hypervectors in a single pass over the input data. Random indexing generates hypervectors that are initially taken from a 10,000-dimensional space and have an equal number of randomly placed $+1$s and -1s, i.e., $\{-1,+1\}^{10,000}$. Such hypervectors are used to represent the basic elements, e.g., the 26 letters of the Latin alphabet and the (ASCII) space for text inputs. These seed letter hypervectors are generated (pseudo)randomly with i.i.d. components. Hypervectors are holographic, too; a hypervector contains all the information combined and spread across all its bits in a full holistic representation so that no bit is more responsible to store any piece of information than another. Hypervectors can be compared for similarity using a distance metric over the vector space.

Hyperdimensional computing has been used for identifying the source language of text samples from a sequence of N consecutive letters (N-grams) [8], [2]. The letter trigrams of a text sample are encoded into a hypervector by the random indexing and vector space operations to represent a language. In the same vein, pentagrams of letters have been used for classifying news articles [9].

A. MAP Operations

We consider a variant of the multiplication, addition, and permutation (MAP) coding described in [28] to define the hyperdimensional vector space. The MAP operations on the hypervectors are defined as follows. Point-wise multiplication of two hypervectors A and B, is denoted by $A*B$. It produces a vector that is dissimilar to its constituent vectors; hence multiplication is well suited for binding two hypervectors. Point-wise addition is denoted by $A+B$. Information from a pair of hypervectors A and B is stored and utilized in a single hypervector by exploiting the addition operation. That is, the sum of two separate hypervectors naturally preserves unique information from each hypervector because of the mathematical properties of vector addition. This addition is well suited for representing sets. Multiplication, or binding, takes two vectors and yields a third, $A*B$, that is dissimilar (orthogonal) to the two; and addition, or bundling, takes several vectors and yields their mean vector $[A+B+...+X]$ that is maximally similar to them. In the following, we describe how these two operations can holistically encode a data record composed of various fields [7].

A data record consists of a set of variables (attributes, fields) and their values (fillers); for example, the variables x, y, z with values a, b, c, respectively. The holistic encoding is done as follows. The variable-value pair $x = a$ is encoded by the hypervector $X*A$ that binds the corresponding hypervectors, and the entire record is encoded by the hypervector $R = [(X*A)+(Y*B)+(Z*C)]$ which includes both the variables and the values, and each of them spans the entire 10,000-bit hypervector.

Finally, the third operation is a permutation, ρ, that rotates the hypervector coordinates. It is implemented as a cyclic right-shift by one position. The permutation operation generates a dissimilar pseudo-orthogonal hypervector that is good for storing a sequence. In geometry sense, the permutation rotates the hypervector in the space. For example, the sequence trigram of a-b-c, is stored as the hypervector $\rho(\rho A*B)*C = \rho\rho A*\rho B*C$. This efficiently distinguishes the sequence a-b-c from a-c-b, since a rotated hypervector is uncorrelated to all the other hypervectors.

Cosine similarity is used to measure similarity between two hypervectors by measuring the cosine of the angle between them using a dot product. It is defined as $cos(A,B) = |A'*B'|$, where A' and B' are the length-normalized vectors of A and B, respectively, and $|C|$ denotes the sum of the elements in C. It is thus a measure of orientation and not magnitude: two hypervectors with the same orientation have a cosine similarity of 1, two orthogonal hypervectors have a similarity of 0, and two hypervectors diametrically opposed have a similarity of -1.

IV. HDC Encoding for EMG Signals

In this section, we describe how MAP operations can be used to encode the enveloped EMG signals. As described in Section II-A, there are four channels and every channel produces an analog signal with an amplitude of 0 mV to 20 mV. To have a linear quantization with a resolution of 1 mV, we discretize the channel's signal to 21 discrete levels as shown in Fig. 4. We design an encoder that accepts a stream of such discretized levels from the channels and computes a hypervector that represents a gesture. We first describe how spatial correlation across the channels can be encoded.

A. Encoding Spatial Correlation into a Holistic Record

We draw an analogy from [7] to generate a holistic record to bind information across the channels together. Each channel is treated as a separate field, and its signal level is interpreted as a value for the field. Hence, there are four fields in the record, namely, CH1, CH2, CH3, and CH4. To represent these fields into hyperspace, we use an item memory (iM) that assigns a unique but random hypervector to every field. This assignment is fixed throughout the computation, and formed four unique orthogonal hypervectors, {iM(CH1) $\perp$ iM(CH2) $\perp$ iM(CH3) $\perp$ iM(CH4)}, as the basic fields. Fig. 3(left) illustrates the cosine similarity measures between these hypervectors in iM which is implemented using a lookup table with four symbols.

Every field has to be assigned its value which is a signal level ranging from 0 to 20. To represent these 21 discrete levels, we use a method of mapping quantities and dates "continuously" to hypervectors [29]. In this continuous vector space, orthogonal endpoint hypervectors are generated for the minimum and maximum levels in the range. Hypervectors for intermediate levels are then generated by weighted interpolation between these endpoints. To perform such mapping we consider a continuous item memory (CiM). CiM assigns an orthogonal hypervector for the minimum signal level (0 mV). For the remaining 20 levels, their corresponding hypervectors are generated by getting gradually further from the minimum hypervector such that the hypervector for maximum signal level (20 mV) is orthogonal to the minimum one, i.e., CiM(0

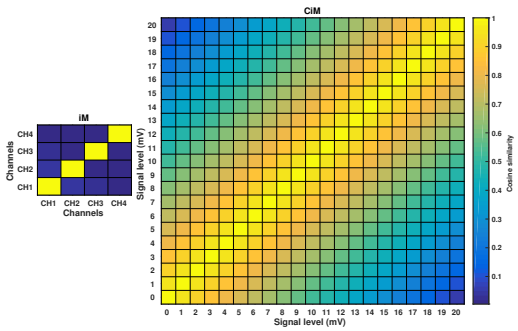


Fig. 3. Cosine similarity between hypervectors stored in iM and CiM.

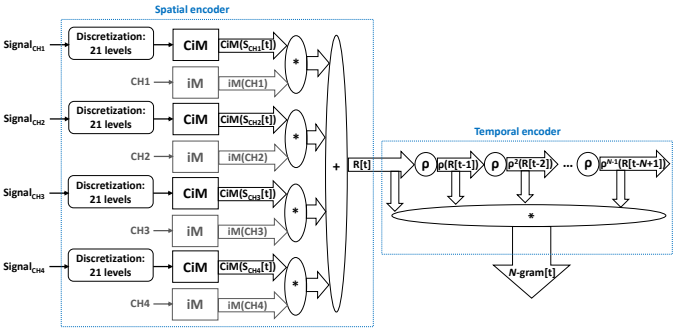


Fig. 4. Spatiotemporal HDC encoder: 1) Spatial encoder to capture correlation across the channels by forming a holistic record; 2) Temporal encoder to capture correlations within channels by rotating records to generate N-gram.

mV) $\perp$ CiM(20 mV). If two hypervectors are dissimilar in $D/2$ of their components, they are orthogonal to each other. Hence, in each step of generating a new hypervector for the next intermediate levels, we flip $D/2/20$ components of the hypervector assigned to the previous level. Such continuous mapping better represents the adjacent levels since their corresponding hypervectors are similar to each other. Fig. 3(right) illustrates the cosine similarity measures between each pair of hypervectors in CiM. As opposed to iM, the similarity between hypervectors in CiM is smooth, hence continuous. CiM is implemented using a lookup table as catalogs of meaningful levels with 21 symbols.

The projection of a channel to the hyperspace is done by pairing a vector from iM with a vector from CiM. We quadruple iM and CiM, with the same contents, for all four channels to allow simultaneous mapping of the EMG inputs to the hyperspace as shown in the left side of Fig. 4. After projecting into the hyperspace, the multiplication operation is used to bind each channel to its signal level for every timestamp t, for instance, for the first channel, iM(CH1) $*$ CiM(S_{CH1}[t]). Finally, to form a record (R[t]) all the four bound fields are bundled by addition operations, i.e., R[t] = iM(CH1) $*$ CiM(S_{CH1}[t]) + iM(CH2) $*$ CiM(S_{CH2}[t]) + iM(CH3) $*$ CiM(S_{CH3}[t]) + iM(CH4) $*$ CiM(S_{CH4}[t]). This record captures the spatial correlation between the four channels for a given time-aligned sample of EMG signals.

Next step is to generate a gesture hypervector, GV, to represent a known gesture. We generate five such vectors, each of which representing one of the five gestures described in Section II-C. As mentioned, in our training dataset each timestamp t is labeled with a gesture tag Label[t]$\in \{1, 2, 3, 4, 5\}$, including the rest position. For every gesture, its corresponding GV acts as a set that contains all the records that are labeled with that specific gesture. Hence, the addition operation bundles R[t] observed in every timestamp to a single sum hypervector GV as follows: GV(Label[t]) += R[t]. Before adding a new record R[t] to GV, we check whether this record is already in GV. This checking forms a conditional addition that adds R[t] to GV when cos(GV(Label[t]), R[t]) < 0.9. If GV has a high cosine similarity (≥ 0.9) with R[t], it means that the record is already in GV, hence there is no need to add the redundant record.

After training, these five GVs are stored in an associative memory as the learned patterns. The same encoding is used for both training and testing (i.e., classification) as shown in Fig. 2. When testing, we call the output of the encoder a "query hypervector" since its label is unknown. In the spatial encoder, the query hypervector is a record R[t] because we are performing sample-by-sample classification. The query hypervector is then sent to the associative memory to identify its source gesture. Determining the gesture of an unknown sample is done by comparing its query to all stored GVs using the cosine similarity. Finally, the associative memory selects the highest similarity among the five measures and returns its associated label as the gesture that the query hypervector has been generated from. Fig. 5 compares the classification accuracy of HDC using this spatial encoding with SVM described in Section II-C. Across five subjects, HDC achieved on average 90.8% classification accuracy, 2% higher than SVM. We have observed that most of misclassifications take place during transitions between two consecutive gestures. This is due to purely vertical slicing of EMG signals. Moreover, a gesture has time-dependent components that require considering a set of samples over time. To address this issue, we develop a temporal encoder applied in cascade after the aforementioned spatial encoder, described in the following section.

B. Encoding Spatiotemporal Correlations by Rotating Records

HDC can encode sequences by using the permutation operation, ρ. As shown in Section III-A, permutation has been used to encode a sequence of N letters to form an N-gram

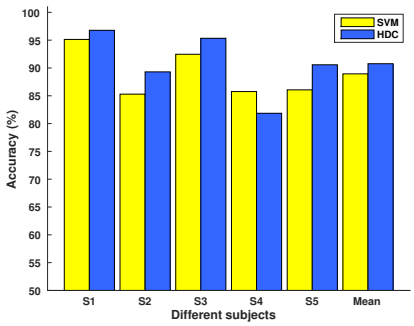


Fig. 5. Sample-by-sample gesture classification using spatial encoder.

hypervector. By analogy, a sequence of four records with time stamps t−3, t−2, t−1, t is encoded as follows: The first hypervector R[t−3] is rotated thrice ρ^3 R[t−3], the second hypervector is rotated twice ρ^2 R[t−2], the third is rotated once ρ R[t−1], and finally there is no rotation for the last hypervector R[t]. The four hypervectors are then combined with point-wise multiplication into a single hypervector for the tetragram, as shown in Fig. 4. For N-grams at large this becomes N-gram[t]= $\prod_{i=0}^{N-1} \rho^i$R[t−i]. As done in the spatial encoder, a single sum hypervector GV is computed for every gesture using the conditional addition: GV(Label[t]) += N-gram[t]. These five GVs are stored into the associative memory as the spatiotemporal learned patterns.

With this encoding algorithm, one important step is to determine the proper size of an N-gram to be able to capture the entire gesture. In this regard, we measured the number of samples available in a gesture. The first two columns of Table I show the mean (i.e., duration) and standard deviation for the number of samples during various gestures of every subject. As mentioned in Section II-C, every gesture is three seconds long sampled at 500 Hz. To fit the samples of a gesture in an N-gram, we applied a downsampling by integer factors shown in the third column. Hence, the gestures can be represented by N-grams where $N \in [1, 10]$. However, for testing and classifying a gesture, there is typically a window of W samples where $W > N$. Hence, we slide the N-gram through the window one step at a time and generate $W - N + 1$ N-grams as query hypervectors, among which we choose the one that has the highest similarity with the stored GVs in the associative memory. Fig 7 shows accuracy results of this classification when using different N-grams. Using N-grams with $N \geq 2$ significantly improves the classification accuracy, while there is a saturation or drop in accuracy for N-gram sizes larger than 6. The last column in Table I lists the sizes for N that maximize accuracy for each subject. Using these N-grams in HDC leads to 97.8% classification accuracy averaged across the five subjects. HDC with spatiotemporal encoding shows 7% higher accuracy compared to solely spatial encoding.

We also compare the classification accuracy of downsampled EMG signals with SVM. In Section II-C, the four channels have been used as the input features for SVM compatible with [10]. To capture temporal correlations in SVM, we use all samples in an N-gram as the features. This forms a brute-force SVM with $4N$ input features. As shown in Fig 7, accuracy of SVM gets worse by using $N > 1$. Its best accuracy, on average 89.7%, is achieved with $N = 1$. By using the N-grams listed in Table I HDC achieves on average 8.1% higher accuracy compared to SVM with $N = 1$.

C. Adaptive Encoder Using Feedback

Although the temporal correlations are well-captured by the proper N-gram, its size varies from subject to subject. For instance, even with using the same downsampling rate of 250, the best N-gram sizes for subjects S3 and S4 are 3 and 5. To mitigate the effect of such dynamic gesture-timing uncertainties across subjects, we design an adaptive mechanism to adjust the size of N-grams during classification. To control the size of N-grams on-the-fly, we define a feedback to close a loop from the associative memory to the encoder as shown in Fig. 6. We use the cosine similarity metric produced in the associative memory as the criterion and maximize it by tuning the size of N-gram for the encoder. This controller starts with an individual spatial record ($N = 1$) and measures its corresponding cosine similarity through the feedback. If the measured similarity is low, the controller keeps increasing N in the available range until a maximum similarity is reached.

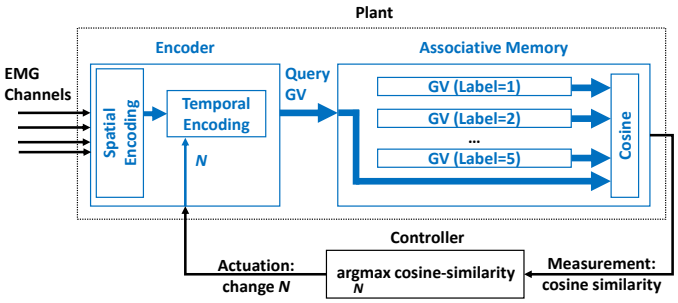


Fig. 6. Using feedback to adaptively tune N-grams based on stored patterns in associative memory.

Fig. 8 shows the cosine similarity tested for various N-grams, where $N \in [1, 10]$, when the associative memory is trained for a fixed N-gram. The graphs show average and standard deviation of cosine similarity for all available gestures in the dataset. As shown, the cosine similarity is maximized when the size of N-gram used for encoder is matched with the size of N-gram patterns that are stored in the associative memory. Such a distinction is extremely robust: when the tested N-gram (in the encoder) and the trained N-gram (in the associative memory) are matched, the cosine similarity is 13.1× larger on average, with a very large safety margin. This ensures that by looking at the output of the associative memory the proper size of N-gram for the encoder can be inferred. The presented feedback mechanism enables an adaptive encoder to be robustly reused across various subjects (independent of their chosen N) for the classification.

V. Experimental Results

In this section, we present more experimental results and sensitivity analyses for classification accuracy. For every experiment (Figs 9 and 10), we measure the mean and the standard deviation of classification accuracy across the five subjects.

A. Overlapping Gestures in a Classification Window

Although capturing temporal correlations of the EMG signal improves classification accuracy in principle, it poses two main challenges: (1) What is the proper size of an N-gram to represent the gestures? (2) What if the window of the signal to be classified is wider than the trained N-gram? Using the feedback presented in Section IV-C addresses the first issue; HDC can determine N-gram for encoding based on the stored patterns in the associative memory. Here, we address the second issue. Such an N-gram typically lies in a window with $W > N$ samples to be used for identifying the gesture. The results presented in the earlier sections contain only one gesture in the classification window. Here, we widen

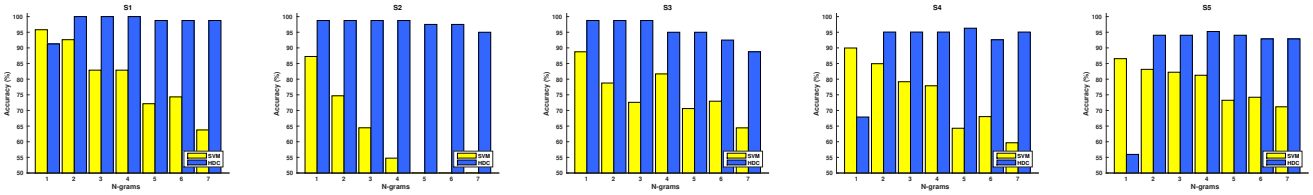


Fig. 7. Classification accuracy for various sizes of N-gram using spatiotemporal encoder.

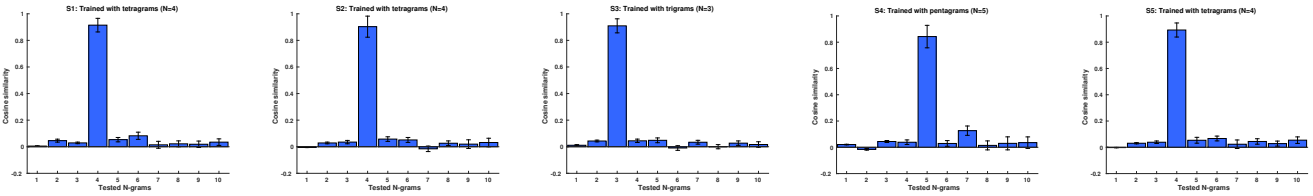


Fig. 8. Cosine similarity measures for various tested N-grams when the associative memory is trained with a fixed N-gram.

TABLE I
STATISTICS FOR THE NUMBER OF SAMPLES IN A GESTURE.

Subjects	Mean	Std	Downsampling	N
S1	1809	503	250	4
S2	1678	814	250	4
S3	1666	941	250	3
S4	1563	590	250	5
S5	1148	542	50	4

the window so that it can include multiple gestures (up to 3). This experiment assesses the ability of HDC to identify the correct gesture when there is no precise partitioning between consecutive gestures but a "gray" region between them.

Fig. 9 shows the classification accuracy with increasing gesture-timing uncertainties as the number of gestures in the window is increased from 1 to 3. Our labeling considers the first gesture in the window as the true label, assuming the subject is moving very quickly. HDC maintains its original accuracy of 97.8% if there is up to 30% overlapping between two gestures in a single window of classification. The accuracy drops to 79.8% when there are two gestures in the window, and finally to 46.9% with three gestures.

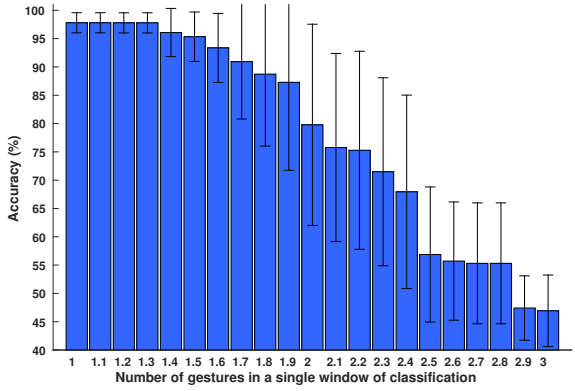


Fig. 9. Accuracy with overlapping gestures in a classification window.

B. Learning Rate

As mentioned in Section II-C, only 25% of the dataset is used for training in the results presented thus far. Fig. 10 explores the classification accuracy as a function of training set size. The classification accuracy is improved by increasing the fraction of training data for both SVM and HDC, but their learning slopes are different. HDC shows an average accuracy of 86.8% when 10% of total dataset is used for training. By increasing the training fraction to 25%, HDC reaches to 97.8% accuracy; after this point increasing the training fraction does not bring accuracy improvement. However, this is not the case for SVM since it requires 3.2× as much training data (i.e., 80%) to reach to the same accuracy as HDC does with 25%.

Increasing the fraction of training data increases the number of support vectors used in SVM: by increasing the training fraction from 10% to 80%, the number of support vectors increases from 30 to 155. This translates directly to higher execution time during classification. On the other hand, HDC adds more patterns into the GVs since increasing the training set size generates new N-grams that are not yet in the sum hypervectors. For instance, HDC adds 83 new patterns by moving from 10% to 25%. However, HDC uses the same hardware structure to store these extra patterns, hence increasing the number of learned patterns does not impact the classification time. In addition, the operations needed for learning and classification are similar in HDC, explicitly allowing continuous learning. In a nutshell, HDC learns quickly and its ability to exploit low-precision scalar operations within a "fixed" hardware structure makes it a prime candidate for online low-cost learning.

We further measure the accuracy of HDC while reducing dimensionality of hypervectors from D = 10,000 to D = 100. HDC is able to maintain its original accuracy by reducing D to 6,000. After this point, the accuracy drops to 96% until D

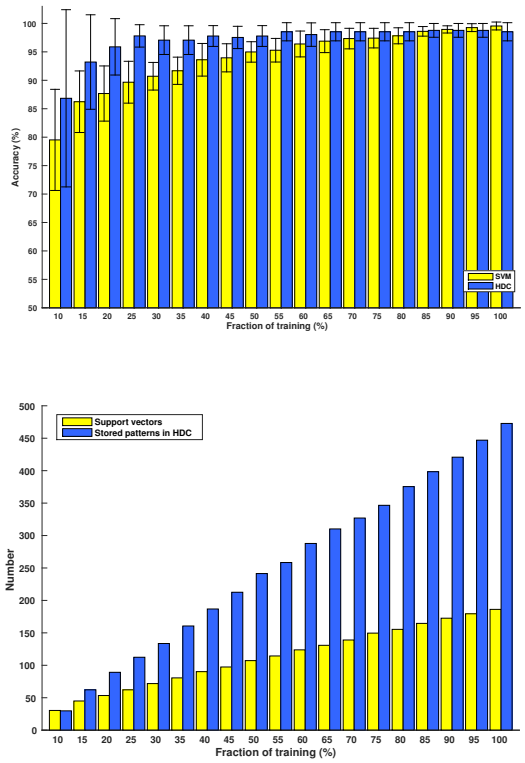


Fig. 10. The impact of training size on accuracy, number of support vectors and stored patterns.

= 300. Below this dimension, the accuracy drops significantly to 62%.

We should not that the implementation of HDC is hardware friendly, and exhibits energy efficiency and robustness benefits compared to the traditional machine learning methods [2]. Hyperdimensional computing is memory-centric and effectively merges computation and storage into a single fabric. Its implementation can be accomplished in a traditional 2D process, however a 3D approach where logic and memory are stacked on top of each other can lead to a far more efficient realization.

VI. Conclusion

This paper presents an application of hyperdimensional computing to the classification of hand gestures from Electromyography recordings. Very simple vector-space operations are used to encode analog input signals for classification. Our algorithm encodes spatiotemporal EMG signals from multiple channels into a hypervector representing a gesture and achieves a high level of accuracy (97.8%) with only $1/3$ the training data required by state-of-the-art support vector machines. Programming HDC is learning-based and uses the same algorithms as subsequent classification. HDC can be adaptive and the resulting classification accuracy is robust: our encoder can adjust to variations in gesture-timing and other uncertainties across different subjects, for each of which the classification can be done correctly even with 30% overlapping between two neighboring gestures.

VII. Acknowledgment

This work was supported by Systems on Nanoscale Information fabriCs (SONIC), one of the six SRC STARnet Centers, sponsored by MARCO and DARPA.

References

[1] T.-T. Liu and J. Rabaey, "A 0.25 V 460 nW asynchronous neural signal processor with inherent leakage suppression," *Solid-State Circuits, IEEE Journal of*, vol. 48, pp. 897–906, April 2013.

[2] A. Rahimi, P. Kanerva, and J. M. Rabaey, "A robust and energy efficient classifier using brain-inspired hyperdimensional computing," in *Low Power Electronics and Design (ISLPED), 2016 IEEE/ACM International Symposium on*, August 2016.

[3] P. Kanerva, J. Kristoferson, and A. Holst, "Random indexing of text samples for latent semantic analysis," in *Proceedings of the 22nd Annual Conference of the Cognitive Science Society*, p. 1036, Erlbaum, 2000.

[4] M. Sahlgren, "An introduction to random indexing," in *Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE 2005*, 2005.

[5] P. Kanerva, "Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors," *Cognitive Computation*, vol. 1, no. 2, pp. 139–159, 2009.

[6] P. Kanerva, "Computing with 10,000-bitwords," in *Proc. 52nd Annual Allerton Conference on Communication, Control, and Computing*, 2014.

[7] P. Kanerva, "What we mean when we say 'What's the dollar of Mexico?': Prototypes and mapping in concept space," in *AAAI Fall Symposium: Quantum Informatics for Cognitive, Social, and Semantic Processes*, pp. 2–6, 2010.

[8] A. Joshi, J. Halseth, and P. Kanerva, "Language recognition using random indexing," in *Quantum Interaction 2016 Conference Proceedings*, in press.

[9] F. R. Najafabadi, A. Rahimi, P. Kanerva, and J. M. Rabaey, "Hyperdimensional computing for text classification," *Design, Automation Test in Europe Conference Exhibition (DATE), University Booth*, 2016.

[10] S. Benatti, F. Casamassima, B. Milosevic, E. Farella, P. Schnle, S. Fateh, T. Burger, Q. Huang, and L. Benini, "A versatile embedded platform for EMG acquisition and gesture recognition," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 9, pp. 620–630, Oct 2015.

[11] J. V. Basmajian and C. De Luca, "Muscles alive," *Muscles alive: their functions revealed by electromyography*, vol. 278, p. 126, 1985.

[12] "Ottobock Sensor 13E200." http://www.ottobock.com/.

[13] D. Yang, L. Jiang, Q. Huang, R. Liu, and H. Liu, "Experimental study of an EMG-controlled 5-dof anthropomorphic prosthetic hand for motion restoration," *J. Intell. Robotics Syst.*, vol. 76, pp. 427–441, 2014.

[14] P. K. Artemiadis and K. J. Kyriakopoulos, "Emg-based teleoperation of a robot arm using low-dimensional representation," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 489–495, Oct 2007.

[15] L. Hargrove, K. Englehart, and B. Hudgins, "A comparison of surface and intramuscular myoelectric signal classification," *Biomedical Engineering, IEEE Transactions on*, vol. 54, pp. 847–853, May 2007.

[16] "Myoelectric control systemsa survey," *Biomedical Signal Processing and Control*, vol. 2, no. 4, pp. 275 – 294, 2007.

[17] H. Zhang, Y. Zhao, F. Yao, L. Xu, P. Shang, and G. Li, "An adaptation strategy of using LDA classifier for EMG pattern recognition," in *International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 4267–4270, 2013.

[18] M. A. Oskoei, S. Member, and H. Hu, "Support vector machine-based classification scheme for myoelectric control applied to upper limb."

[19] M. Ahsan, M. Ibrahimy, and O. Khalifa, "Electromygraphy (EMG) signal based hand gesture recognition using artificial neural network (ANN)," in *Mechatronics (ICOM), 2011 4th International Conference On*, pp. 1–6, May 2011.

[20] H.-B. Xie, T. Guo, S. Bai, and S. Dokos, "Hybrid soft computing systems for electromyographic signals analysis: a review," *BioMedical Engineering OnLine*, vol. 13, no. 1, pp. 1–19, 2014.

[21] M. Rossi, S. Benatti, E. Farella, and L. Benini, "Hybrid EMG classifier based on HMM and SVM for hand gesture recognition in prosthetics," in *Industrial Technology, 2015 IEEE International Conference on*, pp. 1700–1705, IEEE, 2015.

[22] S. Benatti, E. Farella, E. Gruppioni, and L. Benini, "Analysis of robust implementation of an EMG pattern recognition based control," in *International Conference on Bio-inspired Systems and Signal Processing 2014*, pp. 45–54, 2014.

[23] F. Sha, L. K. Saul, and D. D. Lee, "Multiplicative updates for nonnegative quadratic programming in support vector machines," in *Advances in neural information processing systems*, pp. 1041–1048, 2002.

[24] S. Benatti, B. Milosevic, F. Casamassima, P. Schnle, P. Bunjaku, S. Fateh, Q. Huang, and L. Benini, "EMG-based hand gesture recognition with flexible analog front end," in *2014 IEEE Biomedical Circuits and Systems Conference (BioCAS) Proceedings*, pp. 57–60, Oct 2014.

[25] Z. O. Khokhar, *et al.*, "Surface EMG pattern recognition for real-time control of a wrist Exoskeleton," *Biomedical engineering online*, 9(1), 2010.

[26] D. Yang, J. Zhao, Y. Gu, L. Jiang, and H. Liu, "EMG pattern recognition and grasping force estimation: Improvement to the myocontrol of multi-dof prosthetic hands," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pp. 516–521, IEEE, 2009.

[27] P. Kanerva, *Sparse Distributed Memory*. Cambridge, MA, USA: MIT Press, 1988.

[28] R. W. Gayler, "Multiplicative binding, representation operators & analogy," *Advances in analogy research*, 1998.

[29] D. Widdows and T. Cohen, "Reasoning with vectors: A continuous model for fast robust inference," in *Logic Journal of the IGPL*, vol. 23, no. 2, pp. 141–173, 2015.