Adaptive EMG-based hand gesture recognition using hyperdimensional computing

Ali Moin[1*], Andy Zhou[1*], Simone Benatti[2], Abbas Rahimi[3], George Alexandrov[1], Alisha Menon[1], Senam Tamakloe[1], Jonathan Ting[1], Natasha Yamamoto[1], Yasser Khan[1], Fred Burghardt[1], Ana C. Arias[1], Luca Benini[2,3], Jan M. Rabaey[1]

[1] Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA 94720 USA.

[2] Department of Electrical, Electronic and Information Engineering, University of Bologna, 40136 Bologna, Italy.

[3] Department of Information Technology and Electrical Engineering, ETH Zürich, 8092 Zürich, Switzerland.

[*] These authors contributed equally to this work. Correspondence should be addressed to A.M. (email: moin@berkeley.edu) or to J.M.R. (email: jan_rabaey@berkeley.edu).

Abstract

Accurate recognition of hand gestures is crucial to the functionality of smart prosthetics and other modern human-computer interfaces. Many machine learning-based classifiers use electromyography (EMG) signals as input features, but they often misclassify gestures performed in different situational contexts (changing arm position, reapplication of electrodes, etc.) or with different effort levels due to changing signal properties. Here, we describe a learning and classification algorithm based on hyperdimensional (HD) computing that, unlike traditional machine learning algorithms, enables computationally efficient updates to incrementally incorporate new data and adapt to changing contexts. EMG signal encoding for both training and classification is performed using the same set of simple operations on 10,000-element random hypervectors enabling updates on the fly. Through human experiments using a custom EMG acquisition system, we demonstrate 88.87% classification accuracy on 13 individual finger flexion and extension gestures. Using simple model updates, we preserve this accuracy with less than 5.48% degradation when expanding to 21 commonly used gestures or when subject to changing situational contexts. We also show that the same methods for updating models can be used to account for variations resulting from the effort level with which a gesture is performed.

Hand gestures offer a natural way for humans to control, interact with, and engage with intelligent systems and devices. Applications of hand gesture recognition range from providing touchless user interfaces for consumer electronics[1,2] to enabling natural, dexterous control of robotic arms and rehabilitative prostheses[3,4]. Though gestures can be classified through computer vision based approaches using off-body sensors, continuously worn gesture recognition systems must instead rely on biosignals recorded from the body. For example, prosthetic arm controllers typically depend on muscular[5–8] and neural[9,10] biosignals that encode performance of gestures or the intent to do so, respectively. Non-invasive recording of intact-muscle activity from surface electromyography (EMG) is compelling, but most commercially available systems employ very simple encoding or direct mapping of EMG features to actuate only a few degrees of freedom (DOFs)[7].

EMG-based pattern recognition has shown great potential, where EMG features serve as inputs to machine learning algorithms to classify more complex upper-limb gestures and movements[11–15]. However, the EMG signal can be highly variable, especially when recorded using miniaturized wearable devices in non-ideal, real-world conditions. Signal properties change with sweating, fatigue, muscle contraction effort, and electrode displacement due to changing situational contexts such as body movement and device doffing and donning **(Fig. 1)**. These variations can cause significant degradations in classification accuracy[11,16–21]. EMG-based pattern recognition systems must not only be robust to slight variations in the EMG signal, but also easy to update in an online fashion when larger variations require learning novel information[18].

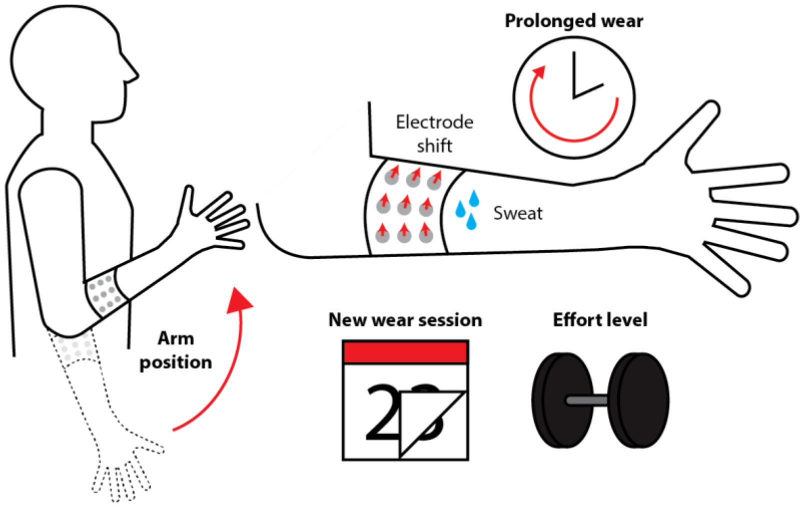


Fig. 1 Different situational contexts presented by everyday use of a surface EMG recording device worn on the forearm. Context changes include different arm positions, prolonged wear of the device in a single wear session, differences over multiple wear sessions, and the effort with which gestures are performed. These changes can introduce EMG signal variations which confound the classification of gestures.

These challenges in EMG-based pattern recognition algorithms can be addressed by the emerging hyperdimensional (HD) computing paradigm that is inherently robust against noise and supports fast learning[22]. HD computing employs random hypervectors with very high dimensionality (e.g. 10,000) to represent information, analogous to the way the human brain utilizes vast circuits of billions of neurons and synapses. Fully distributed holographic representation of data using these hypervectors makes training and classification very robust, even with minimal training data. Additionally, encoding data using simple hypervector operations is fast. The same encoding process is used for both learning and inference, enabling continuous, incremental, and online learning with efficient hardware implementations[23,24]. This is in contrast to other neuro-inspired approaches in which learning (e.g., using backpropagation) is much more computationally demanding than subsequent classification[25]. HD computing has already shown promising results in classification tasks for discriminating up to 5 classes using biosignals such as EMG[26], electroencephalography (EEG)[27,28], and electrocorticography (ECoG)[29].

In this work, we show that an algorithm based on HD computing can accurately classify hand gestures and maintains this accuracy across different situational contexts through incremental learning. The classification model can be updated with minimal new data using simple computations to adapt to new contexts and learn new gestures. To evaluate our classifier, we recorded and made available a dataset that includes 21 hand gestures performed by multiple subjects across changing situational contexts and with different effort levels. The studied situational context variations include different arm positions, different wear sessions, and prolonged wear of the device. We demonstrate one-shot learning, i.e., training on one out of five trials of each gesture, with 88.87% classification accuracy within a single situational context. Moreover, accuracy degradation from contextual variation does not exceed 5.48% when the classifier model has been updated with single gesture trials from the new context. Finally, we demonstrate how gestures performed with low, medium, and high effort levels can be incorporated into classes that cover multiple effort levels or distinguished from each other as separate classes.

Results

Dataset

No available EMG dataset for gesture recognition[30,31] covers the many situational contexts necessary to test the robustness of a classification algorithm or its ability to be incrementally updated. Therefore, we recorded a dataset of EMG signals from five able-bodied, adult male subjects who were asked to perform gestures in multiple contexts that simulated everyday use of an EMG acquisition system. We used a custom, wireless 64-channel EMG signal acquisition device[26] which enabled the subject to assume multiple arm positions, could be worn for hours at a time, and did not tether subjects to the data collection station (**Fig. 2a**). A flexible, screen-printed 16x4 array of electrodes was wrapped completely around a subject's upper forearm, capturing activity of the extrinsic flexor and extensor muscles involved in finger movements.

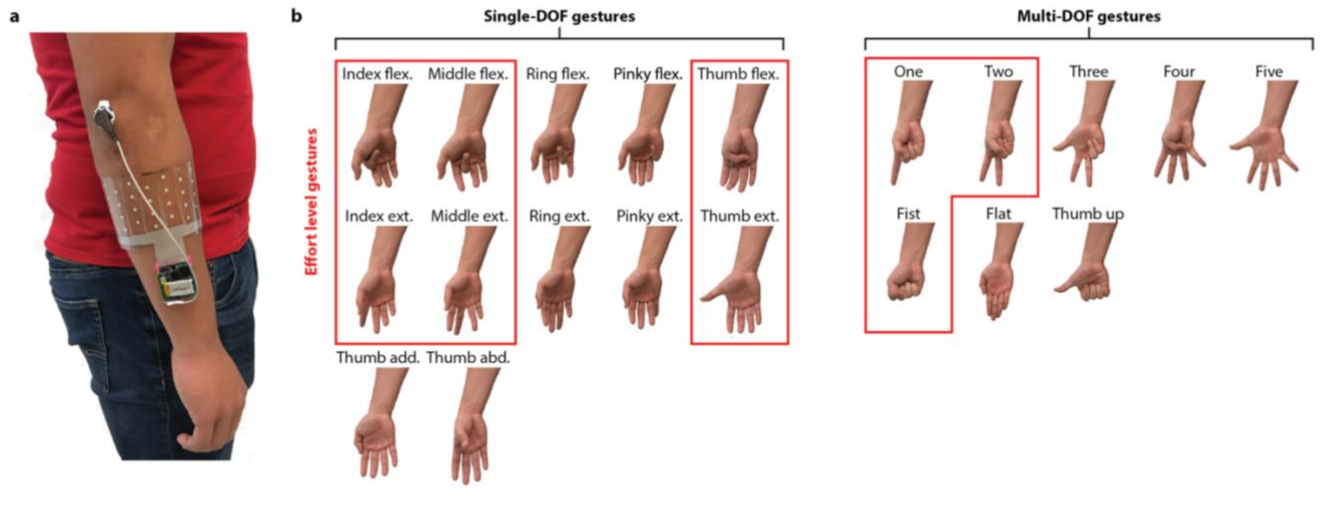


Fig. 2 Wireless EMG recording system and hand gesture classes used in the study. **a** A custom-built biosignal acquisition device donned on the forearm of a subject enables long-term recording and wireless streaming of EMG data in a small form factor. A custom-designed, flexible, screen-printed 16x4 array of electrodes conforms to the forearm to provide high-density, large-area EMG recordings without individual wires. A single Ag/AgCl electrode is attached to the elbow to provide a reference voltage. **b** The single degree-of-freedom (DOF) gesture subset includes individual finger flexions (flex.) and extensions (ext.). The multi-DOF gesture subset includes common, isometric hand postures involving multiple fingers. A third subset of gestures containing both single- and multi-DOF gestures were used for the effort level experiments (boxed in red).

Subjects were asked to perform a total of 21 different gestures consisting of a rest/relax position and two subsets of finger movements and positions: (1) single-degree-of-freedom (DOF) flexions and extensions of individual fingers, and (2) multi-DOF hand postures involving multiple fingers (**Fig. 2b**). An additional subset of gestures was selected for measuring accuracy across different effort levels and consists of both single- and multi-DOF gestures.

Each subject performed a total of 8 recording sessions, with each session being dedicated to a different subset of gestures or a different context simulating variation from everyday use (**Table 1**). Sessions 1 and 2 were designed to test a baseline accuracy for the single- and multi-DOF gesture subsets, respectively, in a baseline relaxed context. Session 3 then introduced an arm position context variation to the single-DOF gestures. The subject performed each gesture with their elbow rested on an armrest in an arm-wrestling position. For sessions 4, 5, and 6, the subjects performed effort level gestures at low, medium, and high effort levels, respectively. Effort levels were assigned as percentages of the maximum voluntary contraction for that gesture, as measured by signal energy (**Fig. 3**). Sessions 7 and 8 were recorded one day after session 1, after doffing and re-donning the device in approximately the same location on the arm. Session 7 thus introduced a new wear session context variation relative to session 1. Subjects were given a two-hour break between sessions 7 and 8, during which they wore the EMG acquisition device while going about their daily activities. This introduced a prolonged wear contextual variation between sessions 7 and 8.

Table 1 EMG recording sessions for various gesture subsets and situational contexts.

Session	**Gestures**	**Context**
1	Single-DOF	Baseline, relaxed arm position
2	Multi-DOF	Baseline, relaxed arm position
3	Single-DOF	New arm position (arm wrestle)
4	Effort level	Low effort, relaxed arm position
5	Effort level	Medium effort, relaxed arm position
6	Effort level	High effort, relaxed arm position
7	Single-DOF	New wear session, relaxed arm position
8	Single-DOF	Prolonged wear, relaxed arm position

During each recording session, the subject performed 5 trials of each of the gestures. Each trial lasted 8 seconds (**Fig. 3**), and the middle 4 seconds consisting of a steady-state hold of the gesture were used for training and inference.

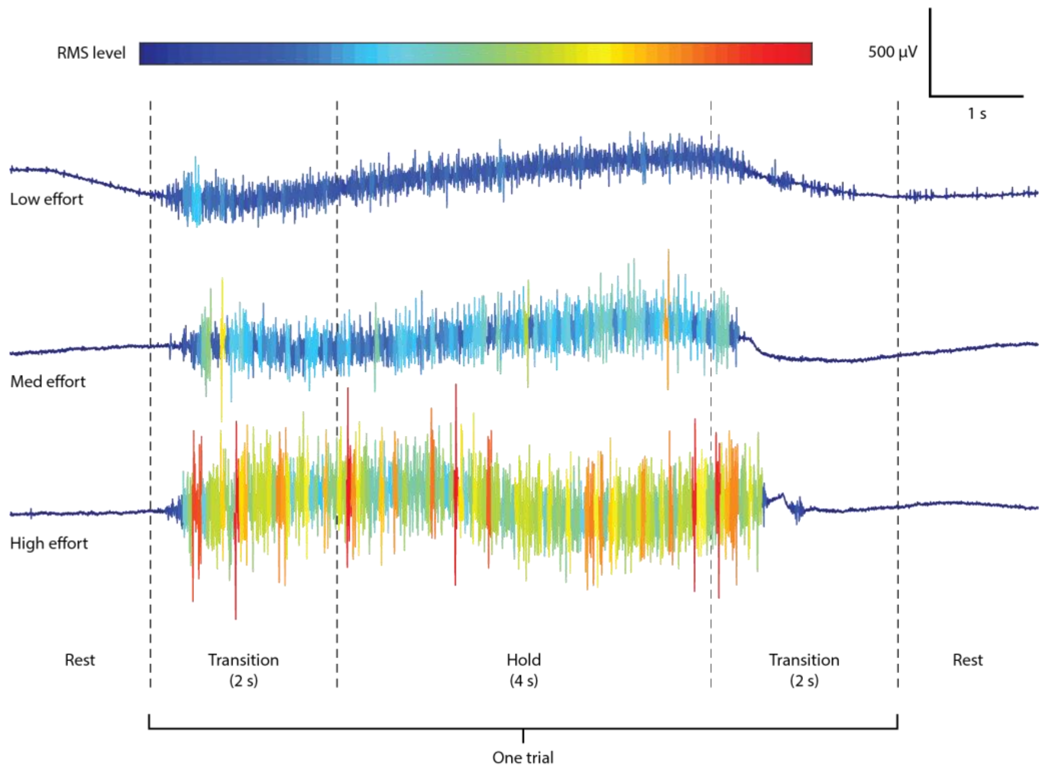


Fig. 3 Representative EMG signals from one electrode channel recorded during a low, medium, and high effort level trial of the same gesture. The vertical dotted lines divide a single 8 s gesture trial into 2 s transition periods and a 4 s hold period based on the instructions given to the subject. The color of the waveform indicates the effort level, as measured by windowed signal power (RMS calculated over 200 ms windows with 150 ms overlap).

HD computing architecture

Training and classification using HD computing consists of encoding data as hypervectors and performing comparisons between these hypervectors. A fixed symbol table, or item memory (IM), is built from an initial set of hypervectors taken randomly from a hyperdimensional space, in this case with 10,000 dimensions. Each hypervector consists of an equal number of randomly placed +1's and -1's. A fundamental property is that, with an extremely high probability, these hypervectors will all be orthogonal to each other. The hypervectors in the IM can be combined to form new composite hypervectors using well-defined vector space operations, including element-wise multiplication (*), element-wise addition (+), scalar multiplication (x), and permutation (ρ). These resulting composite hypervectors can be used to robustly represent an event or class of interest[22].

Fig. 4a describes the process of encoding EMG data into hypervectors used for training and inference. Data is first preprocessed to extract the features to be used as inputs to the HD algorithm. Across a time window, a single feature is calculated for each of the 64 electrode channels. These 64 features are encoded *spatially* to form a single spatial hypervector representing that feature window. Multiple spatial hypervectors from consecutive feature windows are then encoded *temporally* into a single spatiotemporal hypervector, which is the output of the encoding process. These spatiotemporal hypervectors are calculated for all feature sequences to be used as training hypervectors or inference hypervectors in classification.

The spatial and temporal encoding steps involve various hypervector algebraic operations to encode feature values, the electrode channels they are recorded from, and sequences of features across all electrode channels[26]. Electrode channels are represented by an immutable IM composed of approximately orthogonal hypervectors E_i representing each electrode i. For each feature window t, electrode hypervectors, E_i, are modulated by the features from their respective electrode channels, f_i^t, and summed together. The spatial hypervector is then formed as $S_t = \sigma(\Sigma(E_i \cdot f_i^t))$, where σ is a bipolar thresholder that turns a positive element to +1 and a negative element to -1 (**Fig. 4b**).

The resultant spatial hypervector is nearest the electrode hypervectors weighted with the largest feature values. The extracted features should then represent the level of activity local to each electrode channel, such that the spatial hypervector represents the overall pattern of muscle activity across the forearm. We examined six widely-used EMG features and found that root-mean-square (RMS) and mean absolute value (MAV), both indicative of overall signal power and thus muscle activity, led to the highest accuracies with this encoding scheme. We chose MAV, calculated using non-overlapping 50ms feature windows, as our input feature since it is more hardware efficient.

A sequence of n spatial hypervectors is then combined to encode relevant *temporal* information, i.e. the order in which they appear. A hypervector's position in the sequence, $k \in [0, n-1]$ with $k=0$ being the newest hypervector, is encoded using permutation by rotating the hypervector over k positions. The n permuted hypervectors are bound together through element-wise multiplication to form the spatiotemporal hypervector, G (**Fig. 4c**). A new spatiotemporal hypervector can be calculated after each new feature window to encode the previous n windows. We found that sequences of $n=5$ non-overlapping feature windows resulted in the best classification accuracy while introducing a latency of 250 ms, acceptable for real-time control of prosthetics[32,33].

The same encoding process is used to produce spatiotemporal hypervectors for both training and inference (**Fig. 4d**). Training hypervectors G_t calculated using data from a single gesture class are accumulated to form a prototype hypervector representing that class. Prototype hypervectors are thresholded with σ and stored in an associative memory (AM). This accumulation and thresholding operation constitutes the entire training process, enabling fast learning and updating. For classification, inference hypervectors G_i are compared to each entry of a fully trained AM. The inferred gesture is selected by finding the closest prototype hypervector in the AM using cosine similarity as the distance metric.

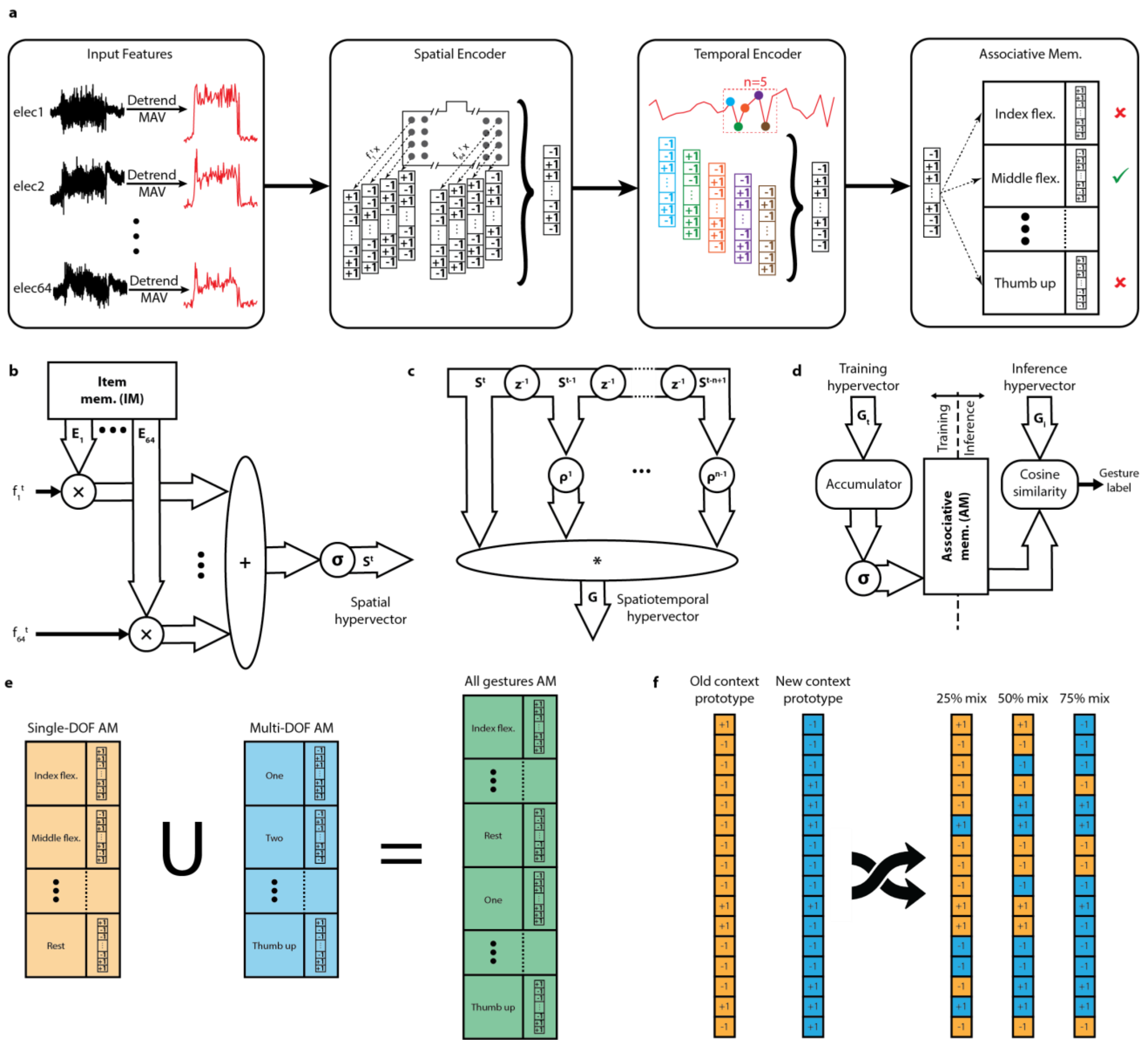


Fig. 4 Hyperdimensional computing algorithm for learning and classification. **a** High-level flow diagram for encoding 64 electrode channels of EMG data into hypervectors. From left to right: mean absolute value (MAV) features are extracted from a segment of each electrode channel and encoded spatially into a spatial hypervector. Multiple spatial hypervectors are then encoded temporally into a spatiotemporal hypervector for comparison with prototype entries of an associative memory (AM). **b** A spatial hypervector S^t is formed as a weighted sum of item memory (IM) hypervectors that represent each electrode channel. The weights for each electrode channel hypervector are the calculated features. After summation, the spatial hypervector is bipolarized to +1's and -1's. **c** Spatial hypervectors from consecutive feature windows are bound together through permute and multiply operations, where a k-element permutation ρ^k is a k-element rotation of the hypervector. Element wise multiplication of the rotated hypervectors forms a spatiotemporal hypervector, G. **d** Spatiotemporal hypervectors can be used for training (G_t, left) or inference (G_i, right). All training hypervectors from the same gesture class are bundled together through element-wise accumulation, and the resulting prototype hypervector is stored in the associative memory (AM). For inference, a new spatiotemporal hypervector is compared with the entries of the AM using cosine similarity, with the closest entry being output as the inferred gesture class. **e** Updating the AM with new gesture classes involves appending new prototype hypervectors as new memory entries without modifying existing entries. **f** Updating the AM with new contexts for an existing class involves forming an updated prototype hypervector for that class. The updated hypervector randomly takes elements from the initial context prototype hypervector and the new context prototype hypervector. The proportion of bits taken from each hypervector determines the relative weight of each context in the updated prototype.

Computationally efficient training and updating

The training process of building an AM allows fast implementation of two types of incremental learning: updating the model (1) with new classes and (2) for new contexts. Adding new classes to a trained classification model is a challenging task in many state-of-the-art classification algorithms, such as neural networks. Doing so often requires training a new model from scratch (requiring access to the original training data) or modifying the model architecture (i.e. identifying new support vectors in a support vector machine). The HD computing model, however, is inherently flexible and simple to update by appending the prototype hypervector for a new class to the AM without changing the existing prototypes. AMs containing different classes can be merged together, even if they were trained on separate occasions (**Fig. 4e**).

Incremental learning of new contexts in HD computing is performed by merging the gesture prototype hypervector from an initial model with a prototype hypervector trained in a new context. An "updated" prototype hypervector is formed, which takes its elements from both the initial and new prototype hypervectors (**Fig. 4f**). Different proportions of new and old elements can be taken to weight the contribution of each context. As more contexts are encountered, this proportion can be adapted to tune the decay rate in order to avoid catastrophic forgetting of initial contexts.

For both cases, the computational resources required for training and updating an HD model online are an accumulator for building the prototype hypervectors and a random bit-position generator for merging multiple prototype hypervectors. This could be implemented in hardware as very low overhead to a system already capable of performing inference with a pre-trained HD classifier.

Baseline Accuracy and Appending New Classes

Due to the limited amount of data for both training and inference, we implemented cross-validation to measure classification accuracy. Because the HD computing model has an inherent robustness to mismatched elements[22], the AM can provide an acceptable level of accuracy for inference after bundling only a small number of spatiotemporal training hypervectors. Therefore, we implemented two different cross-validation schemes with different amounts of training and testing data. For each experiment, the relevant dataset was divided into 5 folds, with each stratified fold consisting of a single trial of each gesture. For the first cross-validation scheme, we implemented the standard leave-one-out cross validation (LOOCV) using 4 folds for training and 1 fold for inference. For the second scheme, we implemented a leave-four-out or *reverse* cross validation (RCV) using a single fold for training and the remaining 4 folds for inference. This way we were able to validate one-shot learning with our model using a small, single-trial training dataset.

We performed both LOOCV and RCV to determine classification accuracy within a single situational context for the single- and multi-DOF gesture subsets separately and for all 21 gestures together. **Fig. 5a** shows LOOCV classification accuracies using our HD computing model as well as three other machine learning models widely used for gesture recognition: support vector machine (SVM), linear discriminant analysis (LDA), and random forest (RF). The HD classifier achieves 92.47% accuracy for single-DOF, 91.71% for multi-DOF, and 89.85% for all gestures, which are comparable or better than the three other methods. The updated model for classifying all 21 gestures was created by joining the AMs previously trained separately for the single- and multi-DOF gesture subsets (**Fig. 4e**). This involved no additional computation, whereas other methods required training a new model from scratch. Furthermore, when measuring accuracy with RCV (**Fig. 5b**), the HD classifier experienced a small amount of degradation compared to LOOCV accuracy, demonstrating its amenability to one-shot learning. Accuracy for single-DOF gestures was 88.87%, with the 3.87% degradation resulting from the smaller training set. Thus, for the remainder of the results, we report accuracies obtained using RCV which were slightly lower than those obtained by LOOCV.

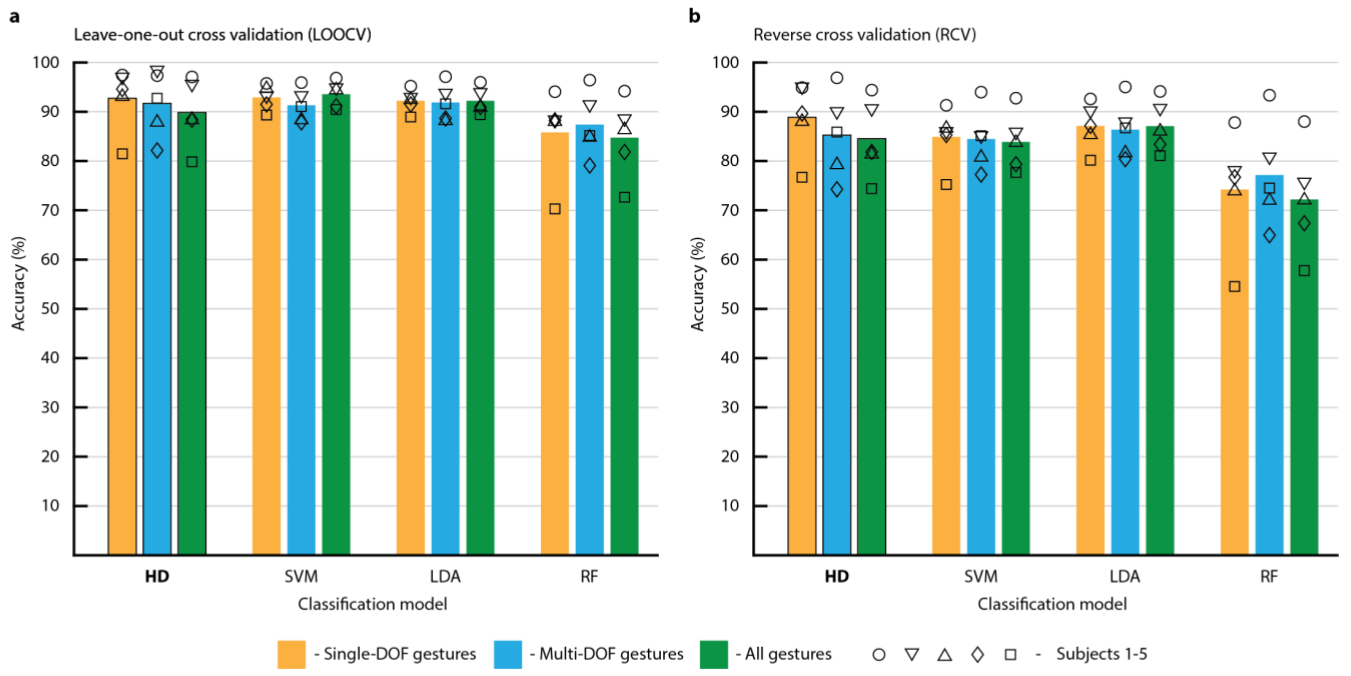


Fig. 5 Classification accuracies on different gesture subsets within a single situational context. Gesture subsets included single-DOF gestures only, multi-DOF gestures only, and all gestures combined, with training and testing splits determined using **a.** leave-one-out cross validation (LOOCV), and **b.** reverse cross validation (RCV). Accuracies were compared for hyperdimensional computing (HD, bold), and three other traditional learning models: support vector machine (SVM), linear discriminant analysis (LDA), and random forest algorithm (RF). Bars represent the mean accuracy across five subjects, with overlaid data points for each individual subject.

New Situational Contexts and Updating Classes

To measure classification accuracy of the HD model across different situational contexts, we performed three experiments in which a model was trained using data from an initial context and then used to classify gestures in a new dataset from a different context. **Fig. 6** shows the classification accuracy for the three tested contextual variations: arm position, new wear session, and prolonged wear. Although training and inference were performed on separate datasets, we still used a single trial of each gesture to train the model as in RCV. For each of the contextual variations, we saw accuracy degradations of 26.17% (arm position), 17.34% (new day/wear session), and 9.95% (prolonged wear), suggesting that incremental updates were necessary.

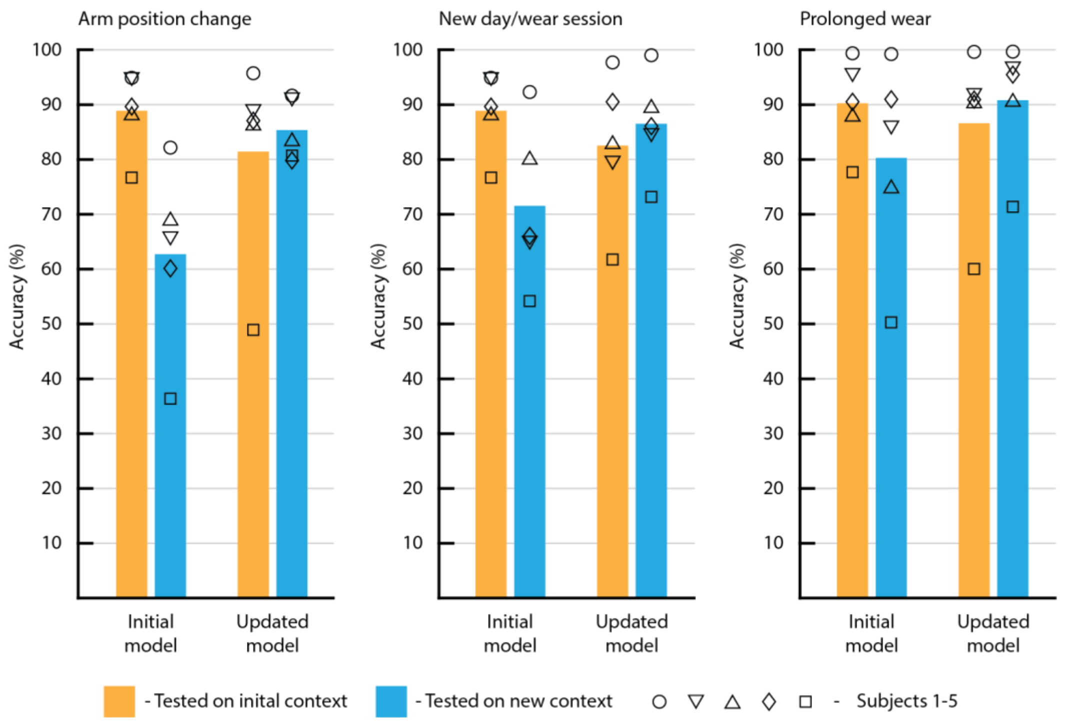
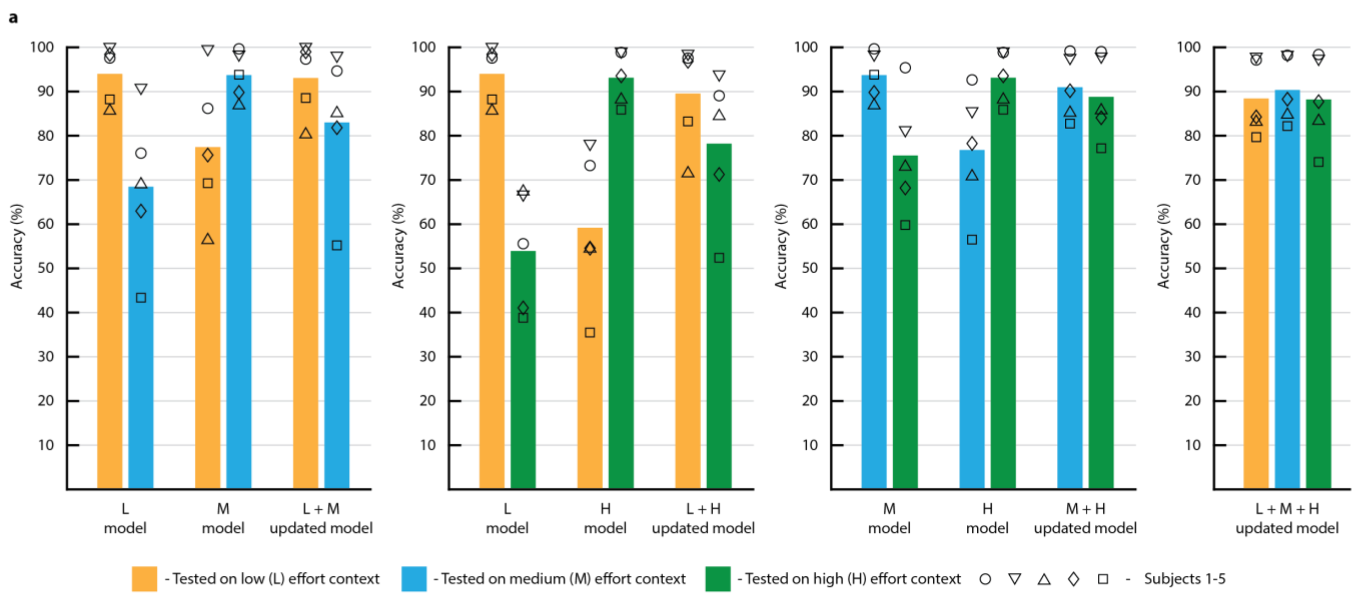


Fig. 6 Classification accuracy across different situational contexts, before and after incremental model updates. For three different context changes (arm position, new wear session, and prolonged wear), an initial model was trained on the initial context and tested on both the initial context and the new context (left pair of bars). The model was then updated using a single trial of each gesture in the new context, and again tested on both contexts (right pair of bars). Bars represent the mean accuracy across five subjects, with overlaid data points for each individual subject.

Prototype hypervectors from the new contexts were also trained using only a single trial of each gesture before being merged with the initial model. We wanted both the initial and new contexts to have equal contributions to the updated model, so the merged hypervector randomly took 50% of its elements from the initial prototype hypervector and the other 50% from the new prototype hypervector (**Fig. 4f**). After updating initial models for the new arm position, wear session, and prolonged wear contexts, the classification accuracy on gestures in the new context was improved significantly (22.66%, 14.96%, and 10.51%, respectively) at the expense of small degradation (7.46%, 6.37%, and 3.65%, respectively) in classifying the initial context gestures.

Multi Levels of Gesture Effort

A subject can exert different levels of effort while performing a gesture, resulting in different EMG signal properties[34]. This effort level variation can be incorporated into the HD computing model in two different ways, depending on the application. If discrimination between different gestures is the only goal, the classifier should output the same gesture class regardless of the subject's effort level. Multiple levels of effort can be considered as different contexts for the same gesture, and a single gesture prototype hypervector can be incrementally updated to include information from these different effort contexts. If, on the other hand, different effort levels are relevant to the application (e.g. controlling different levels of force for gripping using a prosthetic hand), different effort levels for the same gesture must be treated as separate output classes.



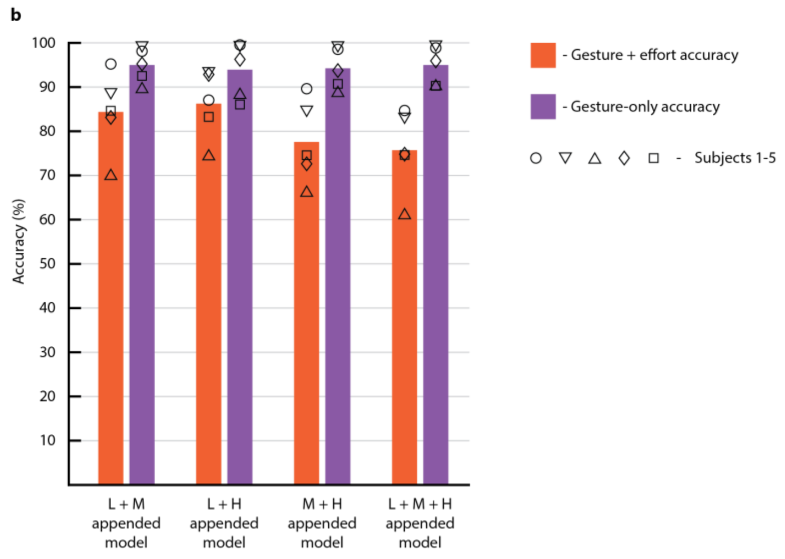


Fig. 7 Classification accuracy with different gesture effort levels. **a** Classification accuracy measured before and after incremental model updates when treating effort levels as different contexts of the same gesture class. Accuracies across effort contexts before and after updating were calculated for each pair of effort levels: low (L) and medium (M), low and high (H), and medium and high. Accuracy for each effort level was also calculated using a model updated with all three effort level contexts. **b** Classification accuracy measured when treating different effort levels of the same gesture as different classes. Accuracy was calculated as the success rate of matching both gesture type and effort level (red) as well as gesture type only (purple). For both **a** and **b**, bars represent the mean accuracy across five subjects, with overlaid data points for each individual subject.

We first treated different effort levels as different contexts of the same gesture class. We performed the same set of experiments as we did for other types of contextual variations on each pair of the three effort levels (low, medium, high). An initial model was trained with gestures from one effort level context. It was then cross validated within the same context and also used to classify gestures from the other level within the pair, without model updates (**Fig. 7a**, first and second pairs). When training and testing within the same effort level context, classification accuracy remained better than 93.11%. However, across different effort level contexts, classification accuracy dropped by between 16.57% and 39.17%, with the worst performance when the difference between effort levels was highest, i.e. low and high effort. After updating the model to include both effort contexts, the classification accuracy was recovered to above 78.21% (**Fig. 7a**, third pairs). The poorest recovered accuracies resulted from training an initial model on medium or high effort level gestures, and then updating with low effort gestures. While prototype hypervectors for medium and high effort level gestures were more similar to each other, prototype hypervectors for low effort level gestures were more distant due to smaller variance in the calculated feature values. For a model trained with all three contexts, accuracy was at least 88.19% for all three effort contexts (**Fig. 7a**, right). In this case, the final updated hypervector is weighted to be more similar to medium and high effort level prototype hypervectors enabling higher accuracies in those contexts.

When treating different effort levels of a single gesture as different classes, we trained a new AM entry for each gesture and effort level, increasing the total number of classes. We calculated classification accuracy in two different ways (**Fig. 7b**). For the first method (**Fig. 7b**, red bars), an accurate classification required matching both the gesture type and its effort level to the label. For the second, (**Fig. 7b**, purple bars), an accurate classification required only matching the gesture type. Notably, if we disregard the effort level classification output from this model, we achieve a better *gesture-only* classification accuracy than in the case where we treated different effort levels as different contexts.

Discussion

In summary, we have demonstrated that a classifier based on HD computing can be used to recognize hand gestures with a high degree of accuracy while training on limited data. Minimizing training data is crucial in subject-specific applications where data collection can be very time consuming. The algorithm is comparable to or better than many traditional machine learning methods in terms of classification accuracy, and it has an additional, key advantage of being very easy to update. For a typical machine learning model, the computation required for training or updating is more complex than and very different from the computation required for inference[18,35]. This results in a large resource overhead to the relatively lightweight calculations needed to produce an inferred class. In contrast, the HD classification model leverages the same encoder to generate hypervectors for classification, training, and updating, enabling efficient implementations that can satisfy the tight resource constraints of wearable systems.

We have shown two methods in their respective use cases for updating the AM of the HD computing model: (1) appending new prototype hypervectors to the AM to add new classes, and (2) merging new prototype hypervectors into existing prototypes to cover more situational contexts with a single AM entry. However, as demonstrated with gesture effort levels, a third method could involve adding new prototype hypervectors to the AM while preserving the number of classes, allowing multiple prototype entries to represent the same class. When bundling different effort level or situational context prototype hypervectors through bitwise merging, the resultant updated prototype hypervector is an approximation of the mean of the original hypervectors. This approximation becomes worse as the similarity between the original hypervectors decreases, which can affect the classification accuracy when using the updated prototype hypervector. The decision of whether to store different contexts or effort levels of the same gesture as separate prototype hypervectors in the AM could be made adaptively by calculating the similarity between the separate prototypes. This decision would also involve weighing the accuracy benefits against the memory and computation costs of having more AM entries.

The ability to incrementally update a model is key for everyday use of a prosthetics controller to account for confounding factors from varying situational contexts[11]. We demonstrated this feature of the HD model by first collecting a new, more comprehensive EMG dataset using a custom, wireless device resembling an actual wearable medical or commercial system. Across multiple recording sessions, we intentionally introduced realistic variations and non-idealities to the data, enabling more extensive experiments than existing EMG datasets recorded in a single context. We then tested the HD model across multiple contextual variations including arm position, wear session, prolonged wear, and effort level. Our results show that while new contexts degraded classification accuracy using an initial model, performance loss was quickly recovered through incremental updates.

Methods

Screen-printed electrode array fabrication

The high-density electrode array (**Fig. 2a**) was printed onto a flexible PET substrate (NovaCentrix DE-SP1) using screen-printing silver ink (NovaCentrix FG57b) in a commercial process (NovaCentrix). It consists of 64 circular electrodes with 4.3 mm diameter, uniformly distributed in a 16 x 4 grid covering up to a 29.3 cm x 8.2 cm area on the forearm. A dielectric encapsulation layer covers the conductive traces while exposing the electrode pads through circular openings.

An adapter PCB with a 64-position Flat Flexible Connector (FFC, XF2W-6415-1AE, Omron Electronics) on one side and two DF12 connectors (Hirose Electric) on the other side was fabricated to interface the flexible array traces with the EMG signal acquisition system.

EMG signal acquisition device fabrication

A custom wireless neural recording module[26] capable of recording up to 128 channels of electrical biosignal data and wirelessly streaming up to 96 channels back to a base station (**Fig. 2a**) is attached to the flexible electrode array. It records and digitizes the EMG signals from the electrodes using a custom neuromodulation IC (NMIC, Cortera Neurotechnologies, Inc.) with DC-coupled front-ends and a 1 kHz sampling rate[36]. Data is aggregated on an FPGA SoC (SmartFusion2 M2S060T, Microsemi) and then streamed out over a 2.4 GHz radio (nRF51822, Nordic Semiconductor). These components enable a small form factor and low power operation (~6 hours of streaming using a standard 250 mAh 4.1 V Li-ion battery), while eliminating the need for bulky individual cables connecting each electrode to the neural front-end, making the device comfortably wearable for extended periods with enhanced signal quality. Moreover, digitizing the signal next to its source increases the signal to noise ratio (SNR).

Graphical user interface

A custom graphical user interface (GUI) developed in Python language was run on the base station for receiving and logging the streamed data, configuring the EMG signal acquisition system, and providing subjects with gesture information, timing, and effort level feedback. The GUI enabled experimenters to select gesture subsets, update metadata for the save files, and update gesture timing prior to each recording session. A bar graph for effort level feedback showed the gesture effort level between 0% and 100% during recording sessions. Effort level was calculated (over 200 ms windows with 150 ms overlap) as the mean signal energy across all electrode channels. For each gesture and subject, experimenters could tune normalization parameters to display different signal energy levels on the 0-100% scale.

Array application

Before wrapping the array around the subject's forearm, a small drop of highly conductive hydrogel (SignaGel Electrode Gel, Parker Laboratories, Inc.) was applied to each electrode to improve the skin-electrode interface impedance. The ends of the array were tightly taped together, holding it in place on the subject's forearm. A single commercial Ag/AgCl electrode (H124SG, Covidien Kendall) was attached to the subject's elbow on the same arm as a reference potential for the voltage measurements (**Fig. 2a**).

Dataset collection

All experiments were performed in strict compliance with the guidelines of IRB and were approved by the Committee for Protection of Human Subjects at University of California, Berkeley (Protocol title: Flex EMG Study. Protocol number: 2017-10-10425). A total of 5 healthy, able-bodied, male, adult volunteers were recruited for participation in the study, and informed, signed consent was obtained from each individual.

Each subject continuously wore the device through Sessions 1-3, with 5 minutes of rest between the sessions. Session 1 consisted of only the single-DOF gestures (**Fig. 2b**, left) performed with the arm relaxed and at the subject's side, as shown in **Fig. 2a**. Session 2 consisted of the multi-DOF isometric, isotonic hand postures (**Fig. 2b**, right) performed in the same arm position. Session 3 consisted of the single-DOF gestures performed with the arm in an arm-wrestling position and the elbow resting on a flat surface (**Fig. 1**). Sessions 4-6 were performed after doffing the array, 2 hours of rest, and re-donning the array. Although small misalignments were unavoidable, a small indicator was marked on the subject's arm for realigning the array between different wear sessions. Since Sessions 4-6 were relatively cumbersome, we only chose a subset of gestures (index finger flexion and extension, middle finger flexion and extension, thumb flexion and extension, One, Two, and Fist) to be included in the dataset. For each effort level gesture, we started with a calibration phase during which the subject was asked to perform the gesture with the maximum effort level, also known as maximum voluntary contraction (MVC). This value was normalized to map to 100% in the GUI feedback bar graph. The subject was asked to target three different effort levels (Session 4 with low effort at 25%, Session 5 with medium effort at 50%, and Session 6 with high effort at 75%) for each gesture.

Sessions 7 and 8 were performed one day after Sessions 1-6, after the device had been doffed and re-donned on the next day. Sessions 7 and 8 consisted of the single-DOF gestures performed with the arm relaxed at the subject's side. Session 7 was recorded after the device was freshly donned (in a similar timeframe to Session 1). Session 8 was recorded 2 hours after Session 7, with the subject going about their daily activities while wearing the array in between.

During each recording session, the subject performed 5 trials of each of the gestures. Each trial lasted 8 seconds (**Fig. 3**), with 3 seconds of rest before the next trial. The subject was told to begin the gesture within a 2-second transition window which would contain the transient, non-stationary part of the EMG signal for that gesture. After the 2-second transition window, the subject was asked to hold the gesture for 4 seconds, constituting the steady-state part of the EMG signal. Finally, the subject was directed to return to the rest position within another 2-second transition window. These directions ensured that the steady-state portion of the gesture could easily be labeled as part of the middle 4 second segment. Data were automatically labeled with the gesture class and saved as .mat files for processing in MATLAB (MathWorks, Inc.).

Data segmentation

For this study, only data from the 4-second steady-state hold period were used for classification. During training and testing, we generated a spatiotemporal vector for every 250 ms segment of data, sliding by 50 ms MAV feature windows (200 ms overlap). Thus, within a single 4-second gesture trial, 76 different vectors were encoded either for accumulating in the AM or for making 76 different inferences. Classification accuracy was calculated as the percentage of inference results that matched the labeled gesture (and effort level, when applicable), without any post processing or voting.

Accuracy metrics

Cross-validation is a procedure used to evaluate machine learning models on a limited data sample. We implemented cross-validation to measure classification accuracy when training and testing with data from the same situational context. We divided each experimental dataset into 5 stratified folds containing a single trial of each gesture. For LOOCV, the model was trained on 4 of the 5 folds with 1 fold left out for inference. For RCV, the model was trained on 1 fold with the remaining 4 folds left out for inference. For both methods, classification accuracy was measured as the mean accuracy of the 5 possible different data partitions.

When training on an initial context and testing on a different context without updates, cross-validation was not necessary. However, to maintain the same amount of training data used, we used a similar method of dividing the training data into 5 folds, and training with either 1 or 4 of those folds. 5 different initial models could be trained, and cross-context classification accuracy was measured as the mean accuracy of all the models tested on the entire testing dataset.

When updating a model for new contexts, we partitioned both the initial and new context datasets into 5 folds. For LOOCV (RCV), we chose 4 (1) folds from the initial dataset to train the initial model, 4 (1) folds from the new dataset to update the model, and then tested the updated model on the remaining 1 (4) folds from each dataset. Classification accuracies on the old and new context dataset were measured as the mean accuracy of the 25 possible different data partitions.

Algorithm implementation

The HD computing algorithm for learning and classification was implemented in MATLAB (MathWorks, Inc.). Scripts were run on a single node (24 cores, 64GB RAM) of a research computing cluster. We leveraged MATLAB's Parallel Computing Toolbox to expedite running multiple experiments in parallel.

SVM, LDA, and RF algorithms were also implemented in MATLAB. For SVM, we used a linear kernel and cost parameter C = 500. For LDA, we implemented the standard MATLAB function included in the Statistics and Machine Learning toolbox, with maximum regularization (gamma = 1) for the covariance matrix estimation. For the RF algorithm, we implemented the standard MATLAB function included in the Statistics and Machine Learning toolbox with 40 decision trees and 2 predictors selected at random for each decision split.

Code availability

Source code is available from Github at https://github.com/flexemg/flexemg_v2.

Data availability

The dataset generated for this study is available at https://github.com/flexemg/flexemg_v2.

References

1. Haque, F., Nancel, M. & Vogel, D. Myopoint: Pointing and Clicking Using Forearm Mounted Electromyography and Inertial Motion Sensors. in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15* 3653–3656 (ACM Press, 2015).
2. Zhang, Z. Microsoft kinect sensor and its effect. *IEEE Multimedia* **19,** 4–10 (2012).
3. Clement, R.G.E., K.E. Bugler, C. W. O. D. Bionic prosthetic hands: A review of present technology and future aspirations. *Neurol. Neurocir. y Psiquiatr.* **44,** 128–132 (2011).
4. Cognolato, M. *et al.* Multifunction control and evaluation of a 3D printed hand prosthesis with the Myo armband by hand amputees. (2018).
5. Weir, R. F. *et al.* Implantable myoelectric sensors (IMESs) for intramuscular electromyogram recording. *IEEE Trans. Biomed. Eng.* **56,** 159–171 (2009).
6. Pasquina, P. F. *et al.* First-in-man demonstration of a fully implanted myoelectric sensors system to control an advanced electromechanical prosthetic hand. *J. Neurosci. Methods* **244,** 85–93 (2015).
7. Farina, D. *et al.* The extraction of neural information from the surface EMG for the control of upper-limb prostheses: Emerging avenues and challenges. *IEEE Trans. Neural Syst. Rehabil. Eng.* **22,** 797–809 (2014).
8. Wang, N., Lao, K. & Zhang, X. Design and Myoelectric Control of an Anthropomorphic Prosthetic Hand. *J. Bionic Eng.* **14,** 47–59 (2017).
9. Velliste, M., Perel, S., Spalding, M. C., Whitford, A. S. & Schwartz, A. B. Cortical control of a prosthetic arm for self-feeding. *Nature* **453,** 1098–1101 (2008).
10. Dhillon, G. S. & Horch, K. W. Direct neural sensory feedback and control of a prosthetic arm. *IEEE Trans. Neural Syst. Rehabil. Eng.* **13,** 468–472 (2005).
11. Gu, Y., Yang, D., Huang, Q., Yang, W. & Liu, H. Robust EMG pattern recognition in the presence of confounding factors: features, classifiers and adaptive learning. *Expert Syst. Appl.* **96,** 208–217 (2018).
12. Benatti, S., Milosevic, B., Farella, E., Gruppioni, E. & Benini, L. A Prosthetic Hand Body Area Controller Based on Efficient Pattern Recognition Control Strategies. *Sensors* **17,** 869 (2017).
13. Geng, W. *et al.* Gesture recognition by instantaneous surface EMG images. *Sci. Rep.* **6,** 36571 (2016).
14. Zhang, H. *et al.* An adaptation strategy of using LDA classifier for EMG pattern recognition. *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS* **2013,** 4267–4270 (2013).
15. Oskoei, M. A. & Hu, H. Support vector machine-based classification scheme for myoelectric control applied to upper limb. *IEEE Trans. Biomed. Eng.* **55,** 1956–1965 (2008).
16. Young, A. J., Hargrove, L. J. & Kuiken, T. A. The effects of electrode size and orientation on the sensitivity of myoelectric pattern recognition systems to electrode shift. *IEEE Trans. Biomed. Eng.* **58,** 2537–2544 (2011).
17. Hargrove, L., Englehart, K. & Hudgins, B. A training strategy to reduce classification degradation due to electrode displacements in pattern recognition based myoelectric control. *Biomed. Signal Process. Control* **3,** 175–180 (2008).
18. Castellini, C., Bongers, R. M., Nowak, M. & van der Sluis, C. K. Upper-limb prosthetic

myocontrol: Two recommendations. *Front. Neurosci.* **9,** 1–4 (2016).

19. Jiang, N., Dosen, S., Muller, K. R. & Farina, D. Myoelectric control of artificial limbsis there a need to change focus? [In the Spotlight]. *IEEE Signal Process. Mag.* **29,** 148–152 (2012).

20. Tkach, D., Huang, H. & Kuiken, T. A. Study of stability of time-domain features for electromyographic pattern recognition. *J. Neuroeng. Rehabil.* **7,** 1–13 (2010).

21. Milosevic, B., Farella, E. & Benaui, S. Exploring Arm Posture and Temporal Variability in Myoelectric Hand Gesture Recognition. *Proc. IEEE RAS EMBS Int. Conf. Biomed. Robot. Biomechatronics* **2018–Augus,** 1032–1037 (2018).

22. Kanerva, P. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognit. Comput.* **1,** 139–159 (2009).

23. Rahimi, A. *et al.* High-Dimensional Computing as a Nanoscalable Paradigm. *IEEE Trans. Circuits Syst. I Regul. Pap.* **64,** 2508–2521 (2017).

24. Wu, T. F. *et al.* Hyperdimensional computing exploiting carbon nanotube FETs, resistive RAM, and their monolithic 3d integration. *IEEE J. Solid-State Circuits* **53,** 3183–3196 (2018).

25. Lecun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521,** 436–444 (2015).

26. Moin, A. *et al.* An EMG Gesture Recognition System with Flexible High-Density Sensors and Brain-Inspired High-Dimensional Classifier. *2018 IEEE Int. Symp. Circuits Syst.* 1–5 (2018).

27. Rahimi, A., Tchouprina, A., Kanerva, P., Millán, J. D. R. & Rabaey, J. M. Hyperdimensional Computing for Blind and One-Shot Classification of EEG Error-Related Potentials. *Mob. Networks Appl.* 1–12 (2017).

28. Rahimi, A., Kanerva, P., Benini, L. & Rabaey, J. M. Efficient Biosignal Processing Using Hyperdimensional Computing: Network Templates for Combined Learning and Classification of ExG Signals. *Proc. IEEE* (2018).

29. Burrello, A., Schindler, K., Benini, L. & Rahimi, A. One-shot Learning for iEEG Seizure Detection Using End-to-end Binary Operations: Local Binary Patterns with Hyperdimensional Computing. (2018).

30. Atzori, M. *et al.* Electromyography data for non-invasive naturally-controlled robotic hand prostheses. *Sci. Data* **1,** 1–13 (2014).

31. Du, Y., Jin, W., Wei, W., Hu, Y. & Geng, W. Surface EMG-based inter-session gesture recognition enhanced by deep domain adaptation. *Sensors (Switzerland)* **17,** 6–9 (2017).

32. Smith, L. H., Hargrove, L. J., Lock, B. A. & Kuiken, T. A. Determining the optimal window length for pattern recognition-based myoelectric control: Balancing the competing effects of classification error and controller delay. *IEEE Trans. Neural Syst. Rehabil. Eng.* **19,** 186–192 (2011).

33. Englehart, K. & Hudgins, B. A Robust, Real-Time Control Scheme for Multifunction Myoelectric Control. *IEEE Trans. Biomed. Eng.* **50,** 848–854 (2003).

34. Khushaba, R. N., Al-Timemy, A., Kodagoda, S. & Nazarpour, K. Combined influence of forearm orientation and muscular contraction on EMG pattern recognition. *Expert Syst. Appl.* **61,** 154–161 (2016).

35. Liu, J. Adaptive myoelectric pattern recognition toward improved multifunctional prosthesis control. *Med. Eng. Phys.* **37,** 424–430 (2015).

36. Johnson, B. C. *et al.* An implantable 700μW 64-channel neuromodulation IC for simultaneous recording and stimulation with rapid artifact recovery. *IEEE Symp. VLSI Circuits, Dig. Tech. Pap.* C48–C49 (2017).

Acknowledgements

The authors would like to thank Prof. Elad Alon, Prof. Rikky Muller, and Cortera Neurotechnologies Inc. for contributing to the EMG signal acquisition device fabrication, Novacentrix for screen-printed electrode array fabrication, and Ken Lutz and Haylie Wu for help with data collection. This work was supported in part by the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.

This research used the Savio computational cluster resource provided by the Berkeley Research Computing program at the University of California, Berkeley (supported by the UC Berkeley Chancellor, Vice Chancellor for Research, and Chief Information Officer). Support was also received from sponsors of Berkeley Wireless Research Center, NSF Graduate Research Fellowship under Grant No. 1106400, ETH Zurich Postdoctoral Fellowship program, and the Marie Curie Actions for People COFUND Program.