Robotica (2017) volume 35, pp. 2177–2200. © Cambridge University Press 2016
doi:10.1017/S0263574716000795

Movement prediction for a lower limb exoskeleton using a conditional restricted Boltzmann machine

Eunsuk Chong and F. C. Park*

Robotics Laboratory, Seoul National University, Seoul 08826, Korea
E-mail: bear3498@snu.ac.kr

(Accepted October 31, 2016. First published online: November 28, 2016)

SUMMARY

We propose a novel class of unsupervised learning-based algorithms that extend the conditional restricted Boltzmann machine to predict, in real-time, a lower limb exoskeleton wearer's intended movement type and future trajectory. During training, our algorithm automatically clusters unlabeled exoskeletal measurement data into movement types. Our predictor then takes as input a short time series of measurements, and outputs in real-time both the movement type and the forward trajectory time series. Physical experiments with a prototype exoskeleton demonstrate that our method more accurately and stably predicts both movement type and the forward trajectory compared to existing methods.

KEYWORDS: Lower limb exoskeleton; Unsupervised learning; Conditional restricted Boltzmann machine; Movement prediction; Trajectory generation.

1. Introduction

A lower limb exoskeleton is a wearable assistive device that delivers additional power to a human operator's legs, with applications ranging from human rehabilitation to the lifting and carrying of heavy objects.[1,2] Key to the proper operation of an exoskeleton is the ability to correctly read the operator's intended movement (e.g., walking, running, climbing stairs, squatting—recognizing the intended movement is important since the control law employed for, e.g., walking may be quite different from that used for climbing stairs), to predict the human operator's forward trajectory (e.g., as joint trajectories, or input joint torque time profiles), and to generate the correct actuator inputs corresponding to the intended trajectory. It is essential that all three of these functions be performed in real time.

Because of the real-time requirement, approaches to exoskeleton control that rely on the measurement of human muscle signals have received considerable attention. Electromyography (EMG) signals are generated before the start of a human's physical movement, and measuring EMG signals offers the possibility of predicting the intended movement before it is initiated. It has become evident, however, that the relationship between EMG signals and the generated muscle torque is highly nonlinear and complex, and that the signals are highly sensitive to both the placement of the electrodes and the state of muscle fatigue.[3–5] Despite recent progress, considerable challenges remain before EMG signals become a reliable signal for exoskeleton control. Methods based on electroencephalogram signal measurements of the human brain also suffer from many of the same drawbacks.[6]

Given the difficulties of using bioelectrical signals for predicting human movements, current practical approaches have focused on using measurements obtained directly from the exoskeleton itself. In ref. [7], measurements of joint angles, angular velocities, forces and torques obtained from sensors attached to the exoskeleton have been used to distinguish among standing, walking, and sitting. Ref. [8] predicts joint trajectories for the hip, knee, and ankle joints of a single leg based on similar measurements, whereas ref. [9] takes into account the leg dynamics to generate a reference gait trajectory corresponding to the desired movement.

* Corresponding author. E-mail: fcp@snu.ac.kr

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

The nature of the task—predicting both the intended movement type and the future trajectory from time series measurement data—is well suited to methods from machine learning, and not surprisingly many of the existing approaches employ such learning algorithms, whether by using bioelectrical signals or physical measurements, or both.[4, 10–13] Previous works for the most part adopt a supervisory learning approach, and assume a properly segmented and labeled dataset of trajectories is available for training purposes. In practice, however, it is time-consuming and difficult to measure and collect such trajectories. More generally, the problem of simultaneously predicting both the intended movement type and the precise form of the future trajectory, sometimes together with the required actuator inputs (usually joint torques), has yet to be addressed in the literature.

In this work, we propose a real-time algorithm for simultaneously predicting the movement type, and the future trajectory, for a lower limb exoskeleton like that shown in Fig. 4. Measurements of a human operator's joint trajectories (obtained from a combination of joint encoders, motion capture devices, and inertial measurement unit (IMU) sensors attached to various links of the exoskeleton and the human) are assumed available. Movement types include walking on flat ground, walking up or down an incline, climbing or going downstairs, and squatting—these are daily lower limb movements,[14] where we try to perform practical experiments. Other movements (e.g., sit-to-stand with a chair, turn), although not included in our experiments, can also be learned with our approach, provided sufficient training data is available. We do not assume that the sample trajectories are labeled with their corresponding movement types. We then develop an unsupervised stochastic learning algorithm that, after training with the unlabeled motion data, is able to predict in real time both the movement type and future trajectory based only on the recent history of state measurements. Our experimental results are demonstrated for measurements of joint angles, link orientations, and accelerations; note, however, that our algorithm is general enough to augment the inputs with other types of measurement signals, e.g., force measurements, EMG signals. In practical settings, the predicted movement type and future trajectory can be used as reference inputs to an exoskeleton tracking controller. In this paper, we do not explicitly address exoskeleton feedback control law design, and instead refer the reader to refs. [15–18] for a detailed discussion and review of exoskeleton control strategies.

While much related work can be found in the computer vision and graphics literature,[19–21] most of which are focused on tracking human body parts, pose estimation, and/or activity recognition from given image or video data, work on multistep ahead prediction and analysis remains limited; although many tracking algorithms are also able to predict the pose at the next time step, for the most part they are as yet unable to predict or generate multiple forward poses. Some approaches to trajectory generation that are closely related to multistep ahead prediction can be found in refs. [22,23], where the goal is to generate realistic human-like motions for digital characters by segmenting and analyzing human motion capture data, these methods typically require repeated k-nearest neighbor searches and large graph matrix constructions, making real-time implementation difficult.

The approach most relevant to our problem is that proposed by Taylor *et al.*,[24,25] who construct and train an artificial neural network of the conditional restricted Boltzmann machine (CRBM) type to predict human motion activities and generate trajectories. The algorithm described in ref. [25], denoted the implicit mixture of conditional restricted Boltzmann machines (imCRBM), can make use of provided motion labels (e.g., movement-type labels) for motion prediction in the supervised learning case, or it can automatically determine atomic motion activities in the unsupervised learning case.

In ref. [25], the imCRBM algorithm is successfully applied to human activity recognition and pose tracking, where it is demonstrated that the algorithm can infer transitions between activities, even when such transitions do not occur in the training data. However, our problem calls for multistep ahead trajectory prediction for different movement types, and for these class of problems, the imCRBM algorithm turns out to be highly sensitive to changes in motion activity—recall that these are automatically inferred by the algorithm, and not necessarily the same as movement types—resulting in large trajectory prediction errors for greater look-ahead intervals.

In this paper, we extend the imCRBM algorithm of ref. [25] to make it determine and predict motion activity more properly, and as a result to more reliably and accurately predict multiple step look-ahead trajectory values. Specifically, we analyze and extract patterns of atomic motion activity sequences obtained from the trained version of the unsupervised imCRBM–each pattern then contains information of a motion trajectory of which interval is set to be sufficiently larger then that of required prediction trajectory. We adopt the convolutional feature extraction technique for pattern extraction,

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

Table I. Abbreviations.

Abbreviations	Terms
CRBM	Conditional restricted Boltzmann machine[24]
imCRBM	Implicit mixture of CRBMs[25]
imCRBM-Ux	Unsupervised imCRBM with x number of motion activities
conv-imCRBM	Convolutional imCRBM
CD-$\mathcal{K}$	Contrastive divergence with $\mathcal{K}$ step iteration[33]

which aims to extract robust features from spatiotemporal data.[26,27] This technique has been adopted to related computer vision applications including human pose estimation and activity recognition,[28–32] but they have yet to be extended to unsupervised motion data segmentation for improved multistep ahead prediction.

We then encode the pattern information into the motion activity at each time step during the training phase, by clustering the patterns (which are represented in vector form). As a result, during the prediction phase, there is a natural tendency to apply the same motion activity over the interval; the original imCRBM in contrast only considers one-step ahead motion information when determining the motion activity at each time step, so that the motion activity can (and does) change frequently during multistep ahead prediction. Further details are provided later in the paper, but the net effect is that multistep ahead prediction is significantly improved even for multiple types of movements, with only marginal increases in computational overhead during the learning phase. Also, to the best of our knowledge, ours is the first work to exploit such a learning-based algorithm for exoskeleton operation and control.

The remainder of the paper is organized as follows. Section 2 formulates the problem in detail, and also reviews the main features of the CRBM and imCRBM algorithms. Section 3 describes the overall framework and details of our extended version of the imCRBM algorithm, which we call the convolutional imCRBM (conv-imCRBM) algorithm. Detailed experimental results with a lower limb exoskeleton hardware prototype are presented in Section 4. For improved clarity, we list in Table I the abbreviations used throughout this paper.

2. Problem Description and Preliminaries

We begin this section with a precise description of the problem addressed in the paper, followed by a review of the main features of the imCRBM algorithm as presented in refs. [24,25].

2.1. Problem description

Our goal is to develop a statistical learning-based algorithm that, given an input vector time series of observed lower limb exoskeleton sensor measurements (e.g., joint angles) over some time interval (e.g., 300 msec), predicts the type of motion activity and also the motion trajectory (e.g., future 800 msec vector time series) as the outputs. A sufficiently large dataset of such measurements is assumed to have been collected (using, for our purposes, the lower limb exoskeleton of Fig. 4). In the learning and training phase of the algorithm, the dataset is clustered into the user-designated number of motion activities and automatically labeled. The prediction component of the algorithm then classifies an arbitrary input measurement time series into the appropriate motion activity, and also predicts future time series values of the output trajectory over a designated time interval.

We assume that the state vector in the input and output time series are of the same dimension. Multistep ahead prediction is then achieved by developing a one-step ahead predictor, and serially cascading this one-step ahead predictor, i.e., the output to the first predictor becomes the input to the second predictor, etc. The problem can then be succinctly stated as predicting the motion activity, and also the current and future states, based on a time series sequence of observed past states.

We adopt the following notation. Let x_{h_t} denote the sequence of observed past states, where $h_t = \{t-s, t-s+1, \ldots, t-1\}$ for the given past time interval s, $x_{h_t} = [x_{t-s}; x_{t-s+1}; \ldots; x_{t-1}] \in \mathbb{R}^{Ds}$, and D is the dimension of each state. Let x_{u_t} denote the sequence of predicted states, where $u_t = \{t, t+1, \ldots, t+g-1\}$ for the prediction time interval g, and $x_{u_t} = [x_t; x_{t+1}; \ldots; x_{t+g-1}] \in \mathbb{R}^{Dg}$. Let $k \in \{1, \ldots, K\}$ be the motion activity index, where K is the user-prescribed number of motion

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

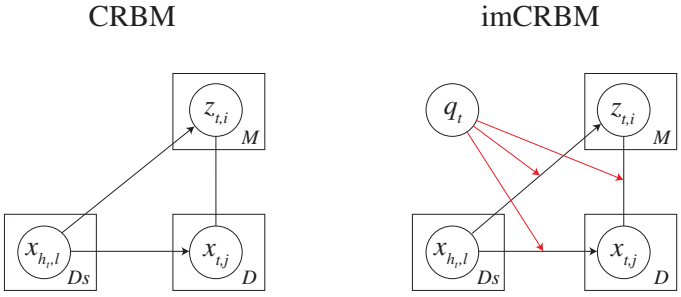


Fig. 1. Graphical models (we adopt the graphical notation of ref. [34] for compact expression). Left: CRBM. Right: implicit mixtures of CRBMs.

activities. We then seek a mapping $\Phi : \mathbb{R}^{Dh} \rightarrow \mathbb{R}^{Dg} \times \{1, \ldots, K\}$, evaluated in real-time for each time step t of the form

$$(x_{u_t}, k) = \Phi(x_{h_t}). \tag{1}$$

2.2. *Conditional Restricted Boltzmann machine and variants*

The CRBM[24] is a stochastic artificial neural network designed to capture temporal dependencies from sequential data, by having its latent and visible variables receive additional input from previous states of the visible variables (Fig. 1, left). The CRBM defines a joint probability model over a real-valued current state variable, $x_t \in \mathbb{R}^D$, and binary latent variable, $z_t \in \{0, 1\}^M$, conditioned on a variable of past states, $x_{h_t} \triangleq [x_{t-s}; x_{t-s+1}; \ldots; x_{t-1}] \in \mathbb{R}^{Ds}$, for a given past time interval s:

$$p(x_t, z_t \mid x_{h_t}) = \exp(-E_\theta(x_t, z_t \mid x_{h_t}))/Z_c(x_{h_t}), \tag{2}$$

where $Z_c = \int_{x_t, z_t} \exp(-E_\theta(x_t, z_t \mid x_{h_t}))\mathrm{d}x_t\mathrm{d}z_t$ is the partition function for the joint probability; typically, exact computation of Z_c is intractable. The joint probability is determined by the scalar function $E_\theta(x_t, z_t \mid x_{h_t})$, called the *energy function*, which is characterized by a set of model parameters, θ.[24]

To train the model given a training dataset $\{x_t^{(n)}, x_{h_t}^{(n)}\}_{n=1}^N$, the negative log-likelihood function $-\sum_{n=1}^N \ln p(x_t^{(n)} \mid x_{h_t}^{(n)})$ is minimized with respect to θ, where z_t is marginalized out. Since directly minimizing the function is difficult due to the computationally intractable partition function Z_c, the CRBM instead obtains approximations of the model parameters by using the contrastive divergence (CD) method.[33] The CD method is a learning method for approximating the exact gradient of the (negative) log-likelihood, in which data is iteratively sampled from a probability distribution ($\mathcal{K}$ steps of iterative sampling is referred to as CD-$\mathcal{K}$), and its sample mean is taken to be the expectation over the distribution. The outcome of training a CRBM is a generative model for $p(x_t \mid x_{h_t})$. We refer the reader to ref. [24] for complete details on learning and inference with CRBM.

In ref. [25], Taylor *et al.* extend the CRBM to the implicit mixture of CRBMs, denoted imCRBM (Fig. 1, right), by introducing a discrete variable q taking on K discrete states, such that $q_t \in \{0, 1\}^K$, $\sum_k (q_t)_k = 1$. The imCRBM allows for the modeling of different types of motion activity; each mixture mode in the imCRBM, which corresponds to a state of q_t, specializes to an atomic motion for each time step t. Similar to CRBM, the imCRBM defines a joint probability model over x_t, z_t, and $q_t \in \{0, 1\}^K$ conditioned on x_{h_t} as follows:

$$p(x_t, z_t, q_t \mid x_{h_t}) = \exp(-E_\Theta(x_t, z_t, q_t \mid x_{h_t}))/Z_{\mathrm{im}}(x_{h_t}), \tag{3}$$

where Z_{im} is the partition function for the joint probability. The corresponding energy function for the imCRBM is simply $E_\Theta = \sum_{k=1}^K (q_t)_k E_{\theta_k}$, which contains K copies of a single CRBM parameter set, θ_k for $k = 1, \ldots, K$. If the kth element of q_t is 1 (so that the remains are 0), then only θ_k governs the model, which reduces to a single CRBM (i.e., the kth CRBM). Since q_t determines the model at

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

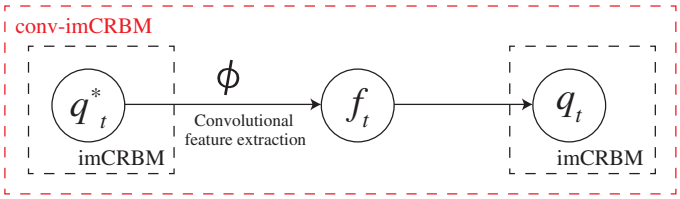


Fig. 2. Architecture of the conv-imCRBM algorithm. A pattern feature vector f_t is extracted from an unsupervised imCRBM using convolutional feature extraction, then combined with another imCRBM.

each time step t, we adopt the same terminology used in ref. [25] and refer to it as an atomic motion activation, or motion activation.

As in CRBM learning, the negative log-likelihood function $-\sum_{n=1}^{N} \ln p(x_t^{(n)} \mid x_{h_t}^{(n)})$ is minimized with respect to $\Theta = \{\theta_k\}_{k=1}^{K}$. The conditional distribution $p(x_t \mid x_{h_t})$ in the function is obtained by marginalizing Eq. (3) over q_t and z_t, which in practice is approximated statistically through the CD procedure as done for the CRBM. The only difference is that one should first sample q_t (i.e., sample a component k) from the distribution $p(q_t \mid x_t, x_{h_t})$, then continue the CD procedure for the kth CRBM model. As a result of the learning, one can obtain $p(x_t \mid x_{h_t}, q_t)$ as well as $p(q_t \mid x_t, x_{h_t})$ given observed past states.

The imCRBM provides both supervised and unsupervised learning schemes for training. Given a labeled training dataset, the supervised imCRBM treats q_t as an observed variable, in which the number of states, K, is set to be the same as the number of labels. In the unsupervised imCRBM, q_t is treated as a hidden variable that is estimated during learning. In ref. [25], the number of states for q_t is specified by the user, e.g., $K = 10$ for the walking–jogging case.

2.2.1. Multistep ahead prediction using imCRBM. We end this section by briefly describing the inference procedure used in the multistep ahead prediction or trajectory generation, using $\mathcal{K}$-step iterative (Gibbs) sampling to obtain $\{x_t, x_{t+1}, \ldots, x_{t+g-1}\}$ given x_{h_t}: (i) initialize $\tau = t$, (ii) initialize $x_\tau = x_{\tau-1}$, (iii) sample q_τ from $p(q_\tau \mid x_\tau, x_{h_\tau})$, (iv) sample x_τ from $p(x_\tau \mid x_{h_\tau}, q_\tau)$, (v) repeat steps (iii) and (iv) $\mathcal{K}$ times, (vi) set $\tau \leftarrow t + 1$ and go to step (ii) while $(\tau \leq t + g - 1)$. We omit the description of the z_t sampling procedure for marginalizing; details of the procedures are given in ref. [25].

3. Convolutional imCRBM

Although ref. [25] reports on a successful application of unsupervised imCRBM to three-dimensional human body tracking, our experience with the closely related problem of exoskeletal motion prediction—specifically, predicting the type of motion activity—suggests that imCRBM is overly sensitive, and too often results in excessively many transitions between different motion activities. According to the inference procedure for multistep ahead prediction as described in Section 2.2.1, at each time step the motion activity type q_t is predicted from $p(q_t \mid x_t, x_{h_t})$; any probabilistic inference errors therefore accumulate rapidly. We therefore seek a way of determining q_t more consistently while retaining the characteristic advantages of imCRBM, by analyzing patterns of motion activation sequences obtained from the unsupervised imCRBM algorithm.

Our strategy is to predict q_t so as to include pattern information of motions nearby in time during the learning phase, so that q_t reflects not only the current and past time steps (t and h_t, respectively), but also the nearby motion patterns at larger time steps. Although in principle this could be achieved simply by increasing the time window size for the historical time step h_t (so that the atomic motion activations learned from unsupervised imCRBM capture the information contained in the longer time steps), this is undesirable for two reasons: (i) the number of parameters to be learned during training grows by $K(D^2 + MD)$ for a single step increase in h_t, where K is the number of state activations, D is the dimension of x_t, and M is the dimension of z_t (for example, $K = 6$, $D = 32$, $M = 300$ in one of our experiments); (ii) more input data time steps are needed for inference, delaying the initial response time of the prediction.

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

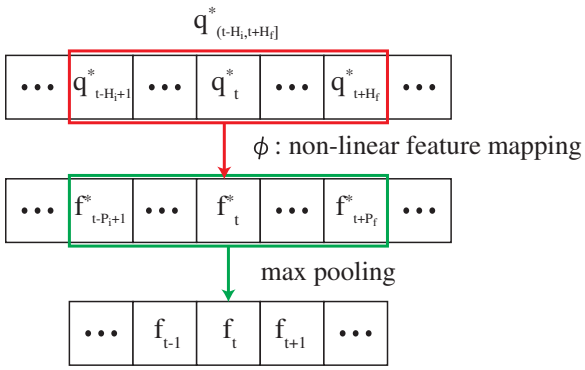


Fig. 3. Illustration of one-dimensional (time) convolutional feature extraction.

Figure 2 illustrates the architecture of our overall framework, which we refer to as conv-imCRBM. The learning procedure for our prediction algorithm proceeds in three stages as follows:

1. *Pre-training*: Train the unsupervised imCRBM for a given motion dataset and obtain motion activation sequences q_t^* as the input to Step 2.
2. *Convolutional pattern feature extraction*: Determine dominant patterns from the q_t^* sequences using the (time) convolutional feature extraction technique. The extracted pattern features are denoted by the vector f_t.
3. *Modeling with pattern features*: Combine f_t with another imCRBM as prior information, and learn the final model with appropriate motion activation, q_t, considering multistep ahead prediction.

We now describe the details of the key steps of our algorithm.

3.1. Convolutional pattern feature extraction

At each time step, we extract pattern information from the given training data using convolutional feature extraction, a widely used method of extracting features from spatiotemporal data.[26,27] Instead of directly using raw level measurement data, we make use of the motion activations obtained from an unsupervised imCRBM as an input to the feature extraction, as shown in Fig. 3. Specifically, let $q_t^* \in \{0, 1\}^Q$ denote the motion activations obtained from a trained unsupervised imCRBM. Given a time-filtering window size, H, we extract convolutional features from the motion activation sequences as follows. Let $q^*_{(t-H_i,t+H_f]} = [q^*_{t-H_i+1}, \ldots, q^*_{t+H_f}] \in \mathbb{R}^{QH}$, where $H = H_i + H_f$. We then define a feature variable, $f_t^* \in \mathbb{R}^L$, as follows:

$$f_t^* = \phi(q^*_{(t-H_i,t+H_f]}), \tag{4}$$

where $\phi : \mathbb{R}^{QH} \to \mathbb{R}^L$ is a nonlinear function that aims to extract the dominant features from $q^*_{(t-H_i,t+H_f]}$. For our application, we set $H_i = H_f = H/2$ for some even number H. ϕ is chosen to be a modified K-means clustering function suggested in ref. [35] as follows (here $\phi_l(x)$ denotes the lth component of $\phi(x)$):

$$\phi_l(x) = \max\{0, \mu(z) - z_l\}, \tag{5}$$

where $z_l = \|x - c^{(l)}\|_{\mathcal{F}}$ with $\|\cdot\|_{\mathcal{F}}$ denoting the Frobenius norm, $\mu(z)$ is the mean of the elements of z, and $c^{(l)}$ is the lth centroid of clusters. We choose this function since it performs the best among several well-known unsupervised learning methods proposed in ref. [35]; substituting for other feature extraction functions is straightforward.

The extracted features then go through a "pooling" procedure, which often follows the convolutional procedure, in order to increase the robustness to temporal shifts. Given a pooling window size P, the lth element $f_{t,l}$ of a pooled feature variable, denoted $f_t \in \mathbb{R}^L$, is defined as

$$f_{t,l} = \max\{f^*_{\tau,l}\}_{\tau=t-P_i+1}^{t+P_f}, \tag{6}$$

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

where $P_i + P_f = P$, and we choose the max function for pooling (max pooling). For our application, we set $P_i = P_f = P/2$ with P an even number. The size of P can be determined depending on the length of the prediction time interval g. For our purposes, we set P to be larger than g, so that an identical pattern is applied during the prediction time interval. The extracted pattern feature f_t is obtained via this procedure.

3.2. Modeling with pattern features

We now combine the previously obtained pattern feature, f_t, to the imCRBM model. Instead of minimizing the negative log-likelihood of $p(x_t \mid x_{h_t})$ as done in ref. [25], we instead minimize the negative log-likelihood of $p(x_t \mid x_{h_t}, f_t)$, where

$$p(x_t \mid x_{h_t}, f_t) = \sum_{q_t} \sum_{z_t} p(x_t, z_t, q_t \mid x_{h_t}, f_t) = \sum_{q_t} \sum_{z_t} p(x_t, z_t \mid x_{h_t}, q_t, f_t) p(q_t \mid f_t). \quad (7)$$

The model is designed with the assumption that q_t conditioned on f_t is independent of x_{h_t}. Note that $\sum_{z_t} p(x_t, z_t \mid x_{h_t}, q_t, f_t)$ corresponds to a CRBM, and $p(q_t \mid f_t,)$ is an as yet unknown distribution function.

To specify the relationship between f_t and q_t, we define the joint probability $p(f_t, q_t)$ as follows. Let $f_t \in \mathbb{R}^L$ follow a multivariate Gaussian mixture distribution with K modes, characterized by the mixture weights $w = \{w_1, \ldots, w_K\}$, the means $\mu\{\mu_1, \ldots, \mu_K\}$, and covariance matrices $\Sigma = \{\Sigma_1, \ldots, \Sigma_K\}$. The mixture distribution is then given by

$$p(f_t \mid \mu, \Sigma, w) = \sum_k w_k \mathcal{N}(f_t; \mu_k, \Sigma_k), \quad (8)$$

where $\mathcal{N}$ denotes the normal distribution. Further, define $q_t \in \{0, 1\}^K$, $\sum_k (q_t)_k = 1$ to indicate which Gaussian mode the pattern f_t is derived from (e.g., $(q_t)_k = 1$ if f_t is from $\mathcal{N}(\mu_k, \Sigma_k)$; otherwise, $(q_t)_k = 0$). The joint probability for q_t and f_t is then given by

$$p(f_t, q_t \mid \mu, \Sigma, w) = \prod_k [w_k \mathcal{N}(f_t; \mu_k, \Sigma_k)]^{q_{t,k}}. \quad (9)$$

To learn the mixture distribution parameters $\{\mu, \Sigma, w\}$ from a set of extracted features $\{f_t^{(n)}\}_{n=1}^{N_a}$, we maximize the log-likelihood of Eq. (9), which is given by

$$\mathcal{L}(\mu, \Sigma, w) = \sum_{n,k} q_{t,k}^{(n)} [\ln w_k + \ln \mathcal{N}(f_t^{(n)}; u_k, \Sigma_k)]. \quad (10)$$

To make the problem more tractable, we take the expectation over Eq. (10) with respect to q_t:

$$\mathcal{L}_q(\mu, \Sigma, w) = \mathbb{E}_q[\mathcal{L}(\mu, \Sigma, w)]. \quad (11)$$

Equation (11) can be maximized via, e.g., the EM algorithm, see ref. [34].

To further enhance computational tractability of the problem, we set $w_k = 1/K$ and $\Sigma_k = I$ (the identity matrix) for all $k = 1, \ldots, K$. The log-likelihood then becomes $\mathcal{L}(\mu) = -\frac{1}{2} \sum_{n,k} q_{t,k}^{(n)} \| f_t^{(n)} - \mu_k \|^2 + \text{constant}$. Maximizing $\mathcal{L}(\mu)$ with respect to μ is then equivalent to performing K-means clustering on $\{f_t^{(n)}\}_{n=1}^{N_a}$, with μ_k given by each resulting cluster mean.[34] We then obtain $p(q_t \mid f_t)$ of Eq. (7) via Bayes rule: $p(q_t \mid f_t) = p(f_t, q_t) / \sum_{q_t} p(f_t, q_t)$.

3.3. Learning and inference

Similar to ref. [25], once $p(q_t \mid f_t)$ is determined, our model can be trained by minimizing the negative log-likelihood (7); the only difference is that we use $p(q_t \mid f_t)$ instead of $p(q_t \mid x_t, x_{ht})$ for the initial sampling procedure in the CD. The learning procedure is described in more detail in Algorithm 1.

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

Algorithm 1 Learning conv-imCRBM (combining f_t to the imCRBM)

Given: Extracted patterns $\{f_t^{(n)}\}_{n=1}^{N_a}$, number of motion activations K, number of iterations J
Output: Model parameter set $\Theta = \{\theta\}_{k=1}^{K}$ for an imCRBM
1: Initialize Θ
2: **for** each $i \in [1, J]$ **do**
3: **for all** $n \in [1, N_a]$ **do**
4: Sample $q_t^{(n)} \sim p(q_t \mid f_t^{(n)})$
5: **end for**
6: Obtain $\Delta\Theta_i$ via CD from the imCRBM and $\{q_t^{(n)}\}_{n=1}^{N_a}$
7: $\Theta \leftarrow \Theta + \Delta\Theta_i$
8: **end for**

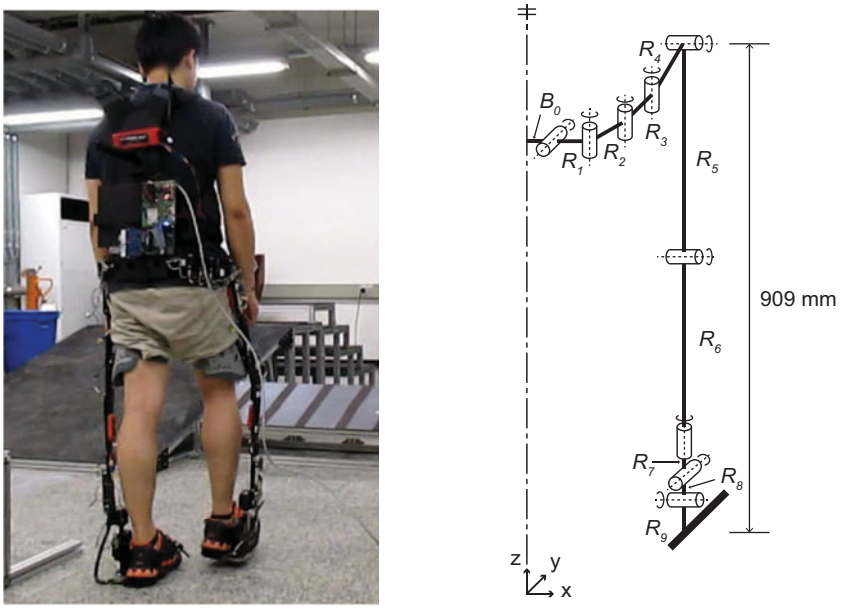


Fig. 4. Left: lower limb exoskeleton prototype hardware (developed by Hanyang University C&R Lab). Right: a schematic of the exoskeleton model. The exoskeleton consists of 19 links and 18 revolute joints; a main body link B_0, joints attached to the left lower limb L_1 to L_9, and joints attached to the right lower limb R_1 to R_9. Only the right side of the model is shown in the schematic, since it is symmetric with respect to the zy plane shown in the figure (e.g., R_1 to R_9 correspond to L_1 to L_9, respectively).

In practice, we find that for most cases, simply using $\hat{q}_t = \text{argmax}_{q_t}\, p(f_t, q_t)$ instead of samples obtained from the distribution $p(q_t \mid f_t)$ in Line 4 of Algorithm 1 works well enough. As a result of the learning, we obtain a sampling method through which we can sample q_t from $p(q_t \mid x_t, x_{h_t})$, and sample x_t from $p(x_t \mid x_{h_t}, q_t)$, so that by following the procedure described in Section 2.2.1 we can perform prediction for both the type of motion activity and also for the multistep ahead trajectory.

4. Experimental Results

4.1. Lower limb exoskeleton hardware prototype

The lower limb exoskeleton hardware prototype used in our experiments is shown in Fig. 4, left. The hardware consists of 19 links and 18 rotational joints, corresponding to the pelvis, thigh, shank links, and the hip, knee, and ankle joints of the human body (Fig. 4, right). The main body link B_0, links of the left lower limb L_0 to L_4, and links of the right lower limb R_0 to R_4 correspond to the pelvis. Five revolute joints are attached to the left (right) sides of the pelvis links, corresponding to the human left (right) hip joint. The thigh link L_5 (R_5) and the shank link L_6 (R_6) are connected by a left (right) knee joint. Three ankle joints for the left (right) lower limb connect the shank link and the foot link

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

Table II. Sensors installed on the lower limb exoskeleton.

Sensor type (model)	Installation place (equipped number on both limb)
Rotary encoder (RBM 20)	hip (2), knee (2), ankle (4)
Rotary encoder (MA3 Abs.)	hip (8), ankle (2)
AHRS (3DM-GX3-25)	torso (1)
IMU (EBIMU-9DoF)	foot (2)
Load cell (CBFSB)	foot (8)
Force sensing resistor (F-Scan)	foot (2)

Table III. Subject information.

Subject	Age	Gender	Height (cm)	Weight (kg)
S1 (Kim**)	29	male	174.5	78
S2 (Lee**)	27	male	170	80
S3 (Lim**)	27	male	173.8	70.3
S4 (Moon**)	27	male	176.5	70

L_9 (R_9), including intermediate links (gaps between the ankle joints) L_7 (R_7) and L_8 (R_8). The torso is connected to the pelvis (link B_0), where a CPU device is mounted for measurement processing and communication with a recording computer.

The exoskeleton hardware is equipped with custom harnesses on the torso, both thighs, and both feet to attach the exoskeleton to the human operator. The available measurements consist of joint angles (recorded from joint encoders attached to all the joints), link orientations and accelerations (recorded from IMUs attached to the links B_0, L_9, and R_9) with respect to a fixed coordinate frame on ground plane, and ground reaction forces (recorded from load cells attached to each foot plate). More detailed information on the sensors installed on the exoskeleton is given in Table II. The exoskeleton hardware weighs 6.5 kg including all sensors and electronics, and is 909 mm in height. All actuators have been removed since only measurements are recorded, so that the movement of the human-exoskeleton system is fully driven by the human operator, i.e., the lower limb exoskeleton is unactuated through all of our experiments. All measurements are recorded at a constant frame rate of 62.5 Hz. We denote the measured state at time step τ by $x_\tau \in \mathbb{R}^D$, where D corresponds to the number of measurements.

4.2. Lower limb Eexoskeleton motion dataset

The data used in our experiments consist of 18 joint angle measurements, 8 orientation angle measurements for the links of the body and two feet, and 6 acceleration measurements for the foot links, resulting in a 32-dimensional measurement. Four healthy male subjects, S1 to S4 as described in Table III, perform two types of motion data sets—D1 and D2, described in Table IV and V, respectively—in an indoor environment as shown in Fig. 5 for D1, and Fig. 15(a) for D2. Each motion in D1 consists of a different number of frames, with each subject repeating each motion sequence three times: the first two repetitions are used for training, whereas the third is used as an out-of-sample test. The composite motions of D2 are each performed twice, with one set use for training and the other for out-of-sample testing. All training data dimensions are normalized to have zero mean and unit variance following the same preprocessing steps of ref. [24]. The test data dimensions are normalized using the sample mean and variance obtained from the training data.

4.3. Measuring prediction accuracy

We use a left-invariant distance metric on the Special Euclidean group $SE(3)$ of rigid body motions[36] to measure the accuracy of the predicted motions with respect to the actual motions. Given two frames $X_1 = \left[\begin{smallmatrix}\theta_1 & b_1\\ 0 & 1\end{smallmatrix}\right]$ and $X_2 = \left[\begin{smallmatrix}\theta_2 & b_2\\ 0 & 1\end{smallmatrix}\right]$ in $SE(3)$, where $\theta_1, \theta_2 \in SO(3)$ and $b_1, b_2 \in \mathbb{R}^3$, the distance metric

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

Table IV. Experimental data, D1.

Dataset, D1								
Description								
Consists of six separate motion classes:								
(walking) on flat ground, up an incline, down an incline, upstairs, downstairs, and squatting.								
Each motion class is performed by each subject, S1,S2,S3, and S4.								
Training set (TR): trial-1,2								
Test set (TST): trial-3								
	Number of Frames (repeated cycles)							
	S1		S2		S3		S4	
	TR	TST	TR	TST	TR	TST	TR	TST
(Walking)								
on flat ground	1160 (9–10)	450 (4)	1300 (10–12)	600 (5–6)	1250 (10)	500 (5–6)	1050 (8–10)	450 (4)
Up an incline	665 (4–6)	345 (2–3)	700 (4–6)	350 (2–3)	650 (4–6)	375 (2–3)	575 (4–6)	300 (2–3)
Down an incline	625 (4–6)	325 (2–3)	775 (6–8)	325 (3–4)	750 (4–6)	400 (2–3)	600 (2–4)	300 (1–2)
Upstairs	600 (4–6)	230 (2–3)	675 (4–6)	400 (2–3)	775 (4–6)	399 (2–3)	650 (4–6)	375 (2–3)
Downstairs	525 (4–6)	210 (2–3)	700 (4–6)	300 (2–3)	750 (4–6)	450 (2–3)	675 (4–6)	375 (2–3)
Squatting	775 (6)	400 (3)	900 (6)	400 (3)	575 (6)	275 (3)	800 (6)	375 (3)
Total frames	4350	1960	5050	2375	4750	2399	4350	2175

Table V. Experimental Data, D2.

Dataset, D2								
Description								
A composite motion consisting of a sequence of the following:								
walking on flat ground, squatting, walking on flat ground, walking up an incline, and walking downstairs.								
Each motion sequence is performed by each subject, S1,S2,S3, and S4.								
Training set (TR): trial-1								
Test set (TST): trial-2								
Number of frames for each motion is counted by manually segmenting the composite motions, only for describing this dataset.								
	Number of Frames (repeated cycles)							
	S1		S2		S3		S4	
	TR	TST	TR	TST	TR	TST	TR	TST
Walking on flat ground	330 (2–3)	400 (2–3)	400 (2–3)	400 (2–3)	280 (2–3)	360 (2–3)	400 (2–3)	360 (2–3)
Squatting	290 (3)	390 (3)	260 (3)	400 (3)	340 (3)	280 (3)	360 (3)	380 (3)
Walking on flat ground	230 (2–3)	210 (2–3)	340 (3–4)	280 (3–4)	300 (2–3)	400 (2–3)	340 (2–3)	260 (2–3)
Walking up an incline	250 (2–3)	300 (2–3)	380 (2–3)	340 (2–3)	280 (2–3)	260 (2–3)	260 (2–3)	200 (2–3)
Walking downstairs	260 (2–3)	260 (2–3)	340 (2–3)	260 (2–3)	300 (2–3)	300 (2–3)	300 (2–3)	320 (2–3)
Total frames	1360	1560	1720	1680	1500	1600	1660	1520

$d_{SE3}(X_1, X_2)$ is defined as

$$d_{SE3}(X_1, X_2) = \sqrt{a_d \| \log(\theta_1 \theta_2^T) \|^2 + c_d \| b_1 - \theta_1 \theta_2^T b_2 \|^2}, \tag{12}$$

where $\log(\cdot)$ is the matrix logarithm, a_d, c_d are user-defined scale factors, and $\| \cdot \|$ denotes the Euclidean norm.

To measure the performance of our prediction algorithm for future time steps, we use the following error measure. The hardware prototype used in our experiments has a main body link B_0 (the pelvis), joints attached to the left lower limb L_1 to L_9, and joints attached to the right lower limb R_1 to R_9. Coordinate frames are attached to all 19 of these links. Let $T_{X(t)Y(\tau)} \in SE(3)$ denote

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

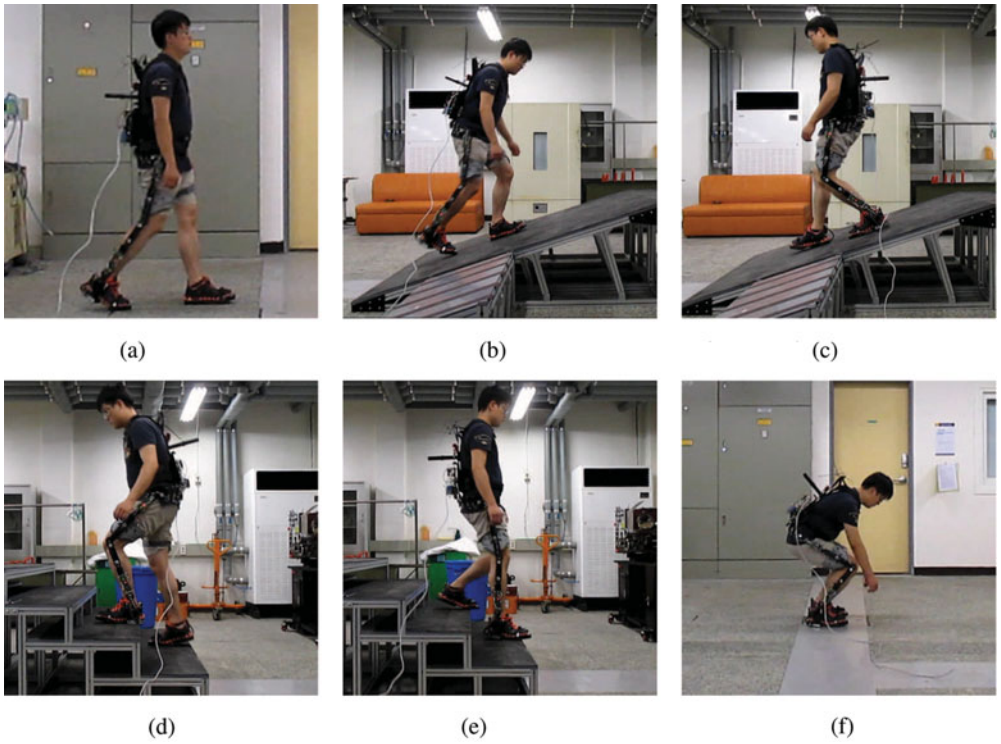


Fig. 5. Six separate motions are performed by each subject in an indoor environment shown in the figure, for datasets D1 (TR) and D1 (TST) described in Table IV: (a) walking on flat ground, (b) walking up an incline, (c) walking down an incline, (d) walking upstairs, (e) walking downstairs, (f) squatting.

the displacement of frame Y at time τ as seen from frame X at time t. Define the collection $U = [T_{B_0L_1}, T_{L_1L_2} \ldots, T_{L_8L_9}, T_{B_0R_1}, T_{R_1R_2}, \ldots, T_{R_8R_9}]$, where $U_k(t)$ indicates the kth $SE(3)$ element of U at time t (e.g., $U_2(t) = T_{L_1(t)L_2(t)}$). Let $T^{(t)} \in SE(3)$ indicate the predicted displacement for T at time t. The motion prediction error for the τth step from t, $\mathcal{E}_t(\tau)$, is then defined as follows:

$$\mathcal{E}_t(\tau) = d_{SE(3)}(T_{B_0(t+\tau-1)B_0(t+\tau)}, T^{(t)}_{B_0(t+\tau-1)B_0(t+\tau)}) + \sum_k d_{SE(3)}(U_k(t+\tau), U_k^{(t)}(t+\tau)). \tag{13}$$

The motion prediction error at time t for future g steps is then denoted $\mathcal{E}_{t,g}$ and defined as follows:

$$\mathcal{E}_{t,g} = \sum_{\tau=1}^{g} \mathcal{E}_t(\tau). \tag{14}$$

4.4. Experiments with multiple classes of motion (D1)

We first report on results of data segmentation, motion activity recognition, and trajectory prediction experiments performed on the D1 data set.

4.4.1. Learning configurations. We first train the unsupervised imCRBM as a pre-training stage, then train the conv-imCRBM by minimizing the negative log-likelihood of Eq. (7). In both stages, we use the CD method with five iterative sampling steps (CD-5, see Section 2.2), with 4000 learning iterations (epochs) for each stage, and 20 past state time steps for x_{h_t}. Three noise-corrupted versions of each data set are created and concatenated, with scaled and shifted versions of small noise added; adding small noise to the inputs with a fixed output target while training a neural network is known to be equivalent to regularizing the objective function, and an effective means of avoiding overfitting.[37]

We set the number of hidden variables M to 300 for $z_t \in \{0, 1\}^M$, $Q = 20$ for $q_t^* \in \{0, 1\}^Q$, and $K = 6$ for $q_t \in \{0, 1\}^K$. Values for the remaining optimization parameters, e.g., the learning rate, are

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

Table VI. Mean and standard deviation (std.) of R^2 values over all the measurement dimensions (averaged over all the subjects) evaluated after (i) pre-training stage (imCRBM-U20), (ii) full conv-imCRBM training, for D1.

	D1 (TR)		D1 (TST)	
Training stage	Mean	std.	Mean	std.
Pre-training	0.7761	0.2073	0.6875	0.2443
Conv-imCRBM	0.8803	0.1646	0.7620	0.2106

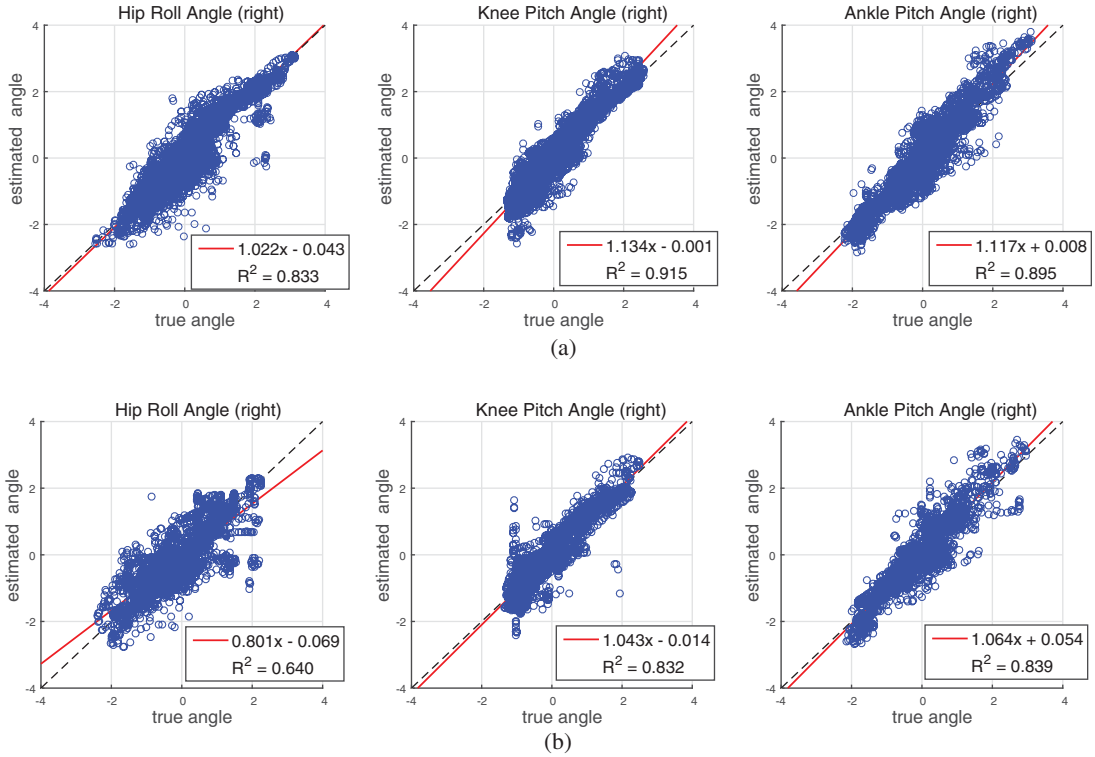


Fig. 6. Model performance after pre-training, (a) D1 (TR) and (b) D1 (TST), subject S1. Real data points x_t (x-axis) with respect to $\mathbb{E}[x_t|x_{h_t}]$ (y-axis) for three joint angles (hip roll, knee pitch, ankle pitch of the right lower limb) of the D1 dataset are shown. Linear regression results with R^2 value are also shown. The angle values shown are normalized to have zero mean and unit variance as explained in Section 4.2.

adopted from ref. [25]. We train and test our algorithm with respect to each subject applying the same learning configurations as stated. All numerical experiments are performed using MATLAB R2013a running on a PC with an Intel Core i7-4770 CPU at 3.40 GHz. The entire learning procedure consumed 14.65 h (average per subject) in our experimental procedure.

4.4.2. Training performance. We analyze and compare model performance after the pre-training stage with full conv-imCRBM. The real data points x_t with respect to the estimated data points $\mathbb{E}[x_t|x_{h_t}]$, given past observations x_{h_t}, are illustrated in Figs. 6 and 7 for three joints (subject S1). We evaluate $\mathbb{E}[x_t|x_{h_t}]$ by estimating the mean value over $p(x_t|x_{h_t}, q_t = \hat{q}_t)$, where $\hat{q}_t$ is set to $\text{argmax}_{q_t}\ p(f_t, q_t)$ instead of marginalizing q_t as described in Section 3.3. For each figure, the linear regression results—all the points should lie on a 45-degree line if the model perfectly estimates the real data—are specified with R^2 values reflecting the regression performance. Higher R^2 values (up to 1) imply a greater explanatory power of the trained model for the given dataset. The mean and standard deviation of R^2 over all the measurement dimensions, averaged over the subjects, are given in Table VI. Both results show that the conv-imCRBM convincingly improves training performance from the pre-trained model for the training set D1 (TR) and the test set D1 (TST).

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

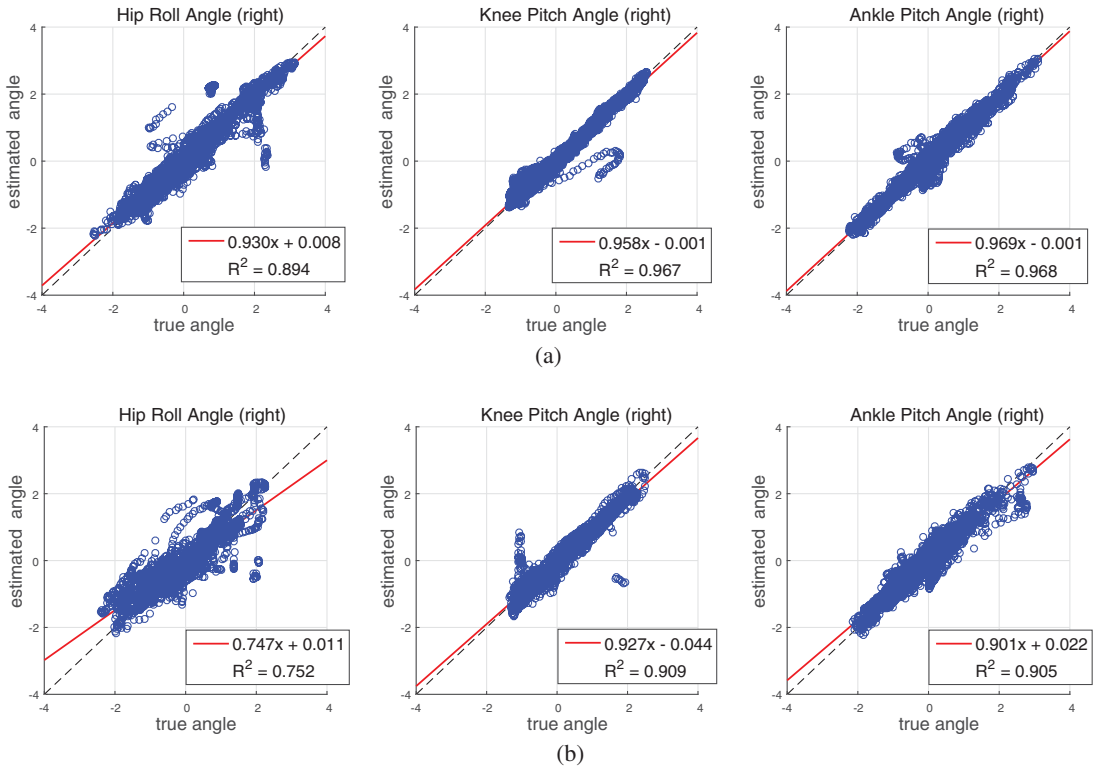


Fig. 7. Model performance after full conv-imCRBM, (a) D1 (TR) and (b) D1 (TST), subject S1. Real data points x_t (x-axis) with respect to $\mathbb{E}[x_t|x_{h_t}]$ (y-axis) for three joint angles (hip roll, knee pitch, ankle pitch of the right lower limb) of the D1 dataset are shown. Linear regression results with R^2 value are also shown. The angle values shown are normalized to have zero mean and unit variance as explained in Section 4.2.

Error histograms for D1 (TR) and D1 (TST) are also illustrated in Fig. 8 (for subject S1), where we define the estimation error to be $\|x_t - \mathbb{E}[x_t|x_{h_t}]\|^2$. The right-skewed bell shape of the error histograms implies that our model has been properly trained. Errors with a value less than 20 after the pre-training stage constitute 95.43% of D1 (TR) and 85.57% for D1 (TST)), whereas training performance is improved after full conv-imCRBM training, i.e., most errors have a value less than 20 (99.45% for D1 (TR), 97.37% for D1 (TST)). Whether this level of performance (error) is sufficient for our application requires further evaluation, since the above performance measures consider neither the exoskeleton kinematics nor multistep ahead trajectories. More practical aspects of our model are evaluated in the following experiments, where we apply and evaluate appropriate performance measures for each case.

4.4.3. Segmentation and recognition. We obtain motion activity segments from D1 using $p(q_t \mid f_t)$ without label information. First, appropriate sizes for the convolutional windows, H and P need to be determined, in order to obtain an identical motion activation during the prediction time interval. For our experiments, we set $H = 20$ and $P = 150$ for 50 frames ahead prediction. Then with the extracted patterns f_t from D1, we evaluate $p(q_t \mid f_t)$ so that each motion activation to be applied at each time frame is determined for training. In determining the motion activation, in addition to using $\hat{q}_t = \text{argmax}_{q_t}\, p(f_t, q_t)$ from Section 3.3, we further set a threshold condition such that $p(\hat{q}_t \mid f_t) > \eta$ is satisfied (we set $\eta = 0.5$) in our experiments; this is to prevent unreliable motion activations (i.e., motion activations with lower probability) from being applied during training. As a result, we obtain motion activity segments for the training data, where the results are shown in Fig. 9. From these experimental settings, we can verify that the motion activation segments correspond well to real motion labels of the D1 data. We attach labels to the segmentation results shown in Figs. 9 and 10.

We next evaluate the accuracy and speed of our algorithm in classifying the D1 data set based on segmented motion activity indices. The segmented indices for D1 (TST) are compared with the

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

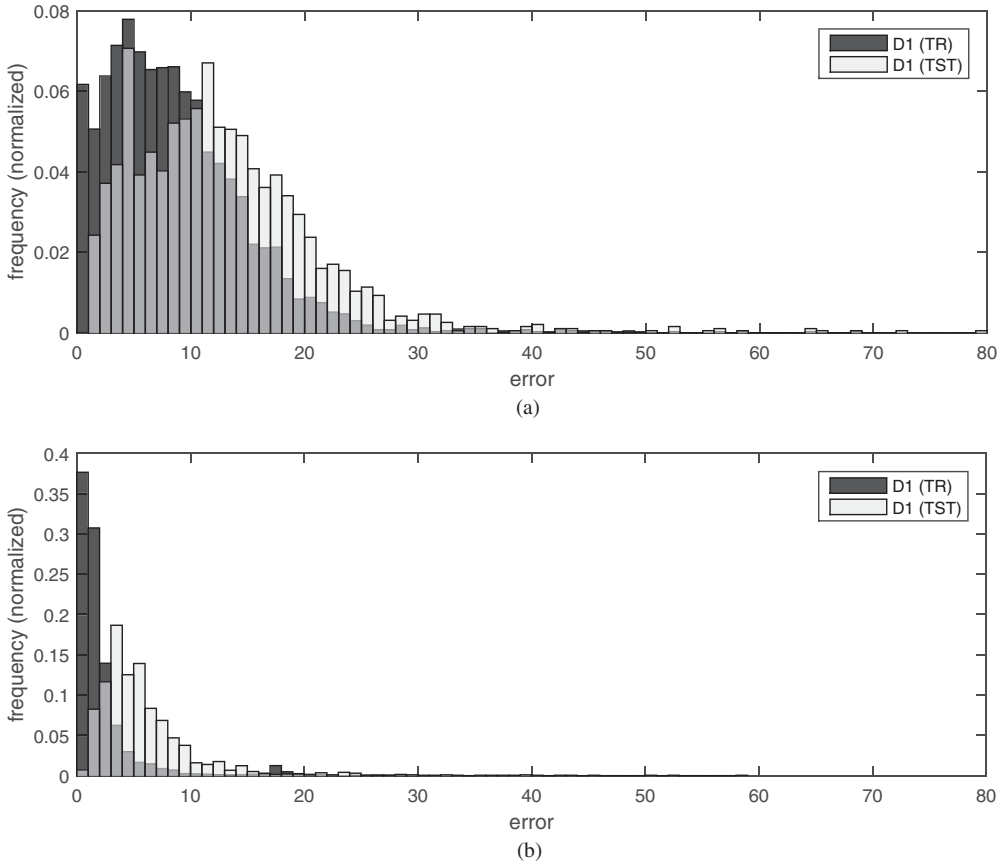


Fig. 8. Error histograms for D1, (a) after pre-training and (b) after full conv-imCRBM training, with a bin size of 1 (subject S1). Error in the x-axis is defined as $\|x_t - \mathbb{E}[x_t|x_{h_t}]\|^2$. Dark bars denote D1 (TR), whereas light bars denote D1 (TST). The gray regions denote overlapping areas of the two histograms. Frequency in the y-axis is normalized (sum of frequencies for each histogram is normalized to one) to compare the two histograms, since the sample sizes for each dataset are different.

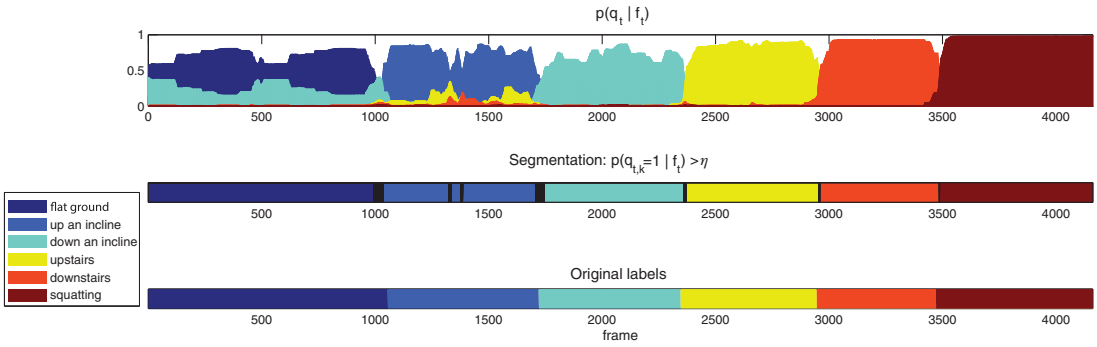


Fig. 9. Segmentation result for D1 (TR) for subject S1, based on $p(q_t \mid f_t)$ without label information, compared to original data labels. Black regions in the middle figure indicate that no state has a probability exceeding $\eta = 0.5$. The color codes are manually matched after the segmentation.

motion activation q_t inferred for the prediction (i.e., q_t obtained from the procedure given in Section 2.2.1). The original label information is only used for matching the segmented motion indices to the real labels. The confusion matrix in Fig. 10 shows the motion classification results for subject S1, for 1960 frames, performed in 0.091 s. Note that the frame length corresponds to 31.36 s (classifying time/classified motion time = 0.29%).

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

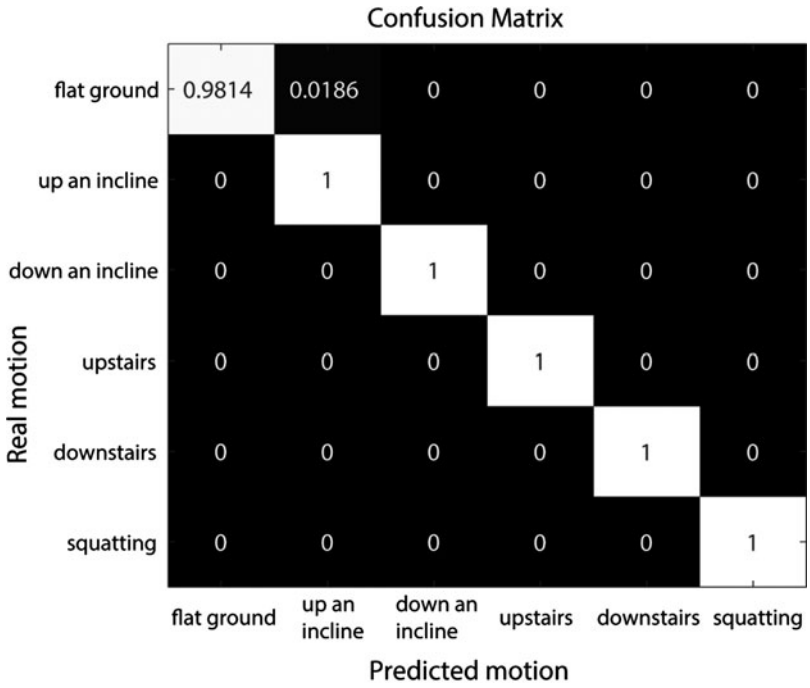


Fig. 10. Recognition results (confusion matrix) for D1 (TST) using segmented motion indices from D1 (TR), for subject S1. The total estimation time is 0.091 s (averaged over 100 trials) for 1960 frames. The sum of the values in each row is normalized to one. Note that above labels are attached after the classification.

4.4.4. Prediction. We evaluate the accuracy for multistep forward prediction using the measure $\mathcal{E}_{t,g}$ defined in Eq. (14). In all of our prediction experiments, we set the scale parameters of the $SE(3)$ metric to $a_d = 0.5$ and $c_d = 0.5$ in (12). We also evaluate the accuracy measure for the case of no displacement with respect to the true motion (i.e., $U_k^{(t)}(t+\tau) = U_k^{(t)}(t)$, for $\tau \in \{1, \ldots, g\}$) at each time t, as a *baseline* measurement. Table VII shows the 50 frames ahead prediction results ($g = 50$) of the baseline, the unsupervised imCRBM (with $q_t \in \{0, 1\}^{20}$), and the conv-imCRBM (the result values are averaged over the subjects). We use the same total number of iterations during learning for the unsupervised imCRBM and the conv-imCRBM. The results show that our algorithm performance far exceeds the baseline, whereas the basic imCRBM fails to show any meaningful results for the learning configurations.

To better visualize the prediction errors, we illustrate the predicted trajectories (red circled lines) for three joints at certain time frames (red vertical lines) compared with the true trajectories (blue solid lines) as shown in Fig. 11. Each estimation for predicting ahead 50 frames (0.8 s) is performed in 0.035 s on average (predicting time/predicted motion time = 4.375%).

4.5. Experiments with composite motions (D2)

We perform the same set of motion recognition and prediction experiments on the D2 composite motion dataset, which include transitions between motion types, extra motions (e.g., turning, standing), and contain fewer number of cycles for each motion type. None of the motions are labeled.

4.5.1. Learning configurations. We downsample the D2 motion data, keeping only every fourth frame (from 62.5 to 15.625 Hz). The model is trained using CD-10 with 2000 learning iterations (epochs) for the pre-training stage, 6000 epochs for the conv-imCRBM, and using the five past state time-steps for x_{h_t}. Five noise-corrupted versions of each data set are created and concatenated, with scaled and shifted versions of small noise added. We set $Q = 50$ for $q_t^* \in \{0, 1\}^Q$, and the remaining parameters to the same values listed in Section 4.4. We train and test our algorithm with respect to each subject applying the same learning configurations as stated. The learning consumed a total of 2.42 h (average per subject) in our experimental setup.

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

Table VII. Motion prediction error, $\mathcal{E}_{t,g}$, for D1 ($g = 50$).

		Motion prediction error: mean (std.) over frames (averaged over all the subjects)					
		Test data: D1 (TST)					
Model	Training data	Flat ground	Up an incline	Down an incline	Upstairs	Downstairs	Squatting
None (baseline)	·	13384.08 (2831.63)	13440.13 (5174.84)	13043.31 (3299.52)	15134.73 (6245.44)	14810.52 (4191.34)	21783.71 (8308.66)
imCRBM-U20	D1 (TR)	10822.56 (4643.64)	11106.34 (4667.06)	14784.10 (5519.47)	12396.86 (5552.05)	12604.30 (5455.09)	21392.99 (12335.48)
conv-imCRBM	D1 (TR)	**4961.23 (2511.59)**	**3993.24 (1256.48)**	**6468.61 (3003.00)**	**6101.01 (4005.83)**	**6476.04 (2879.80)**	**7432.12 (4842.75)**

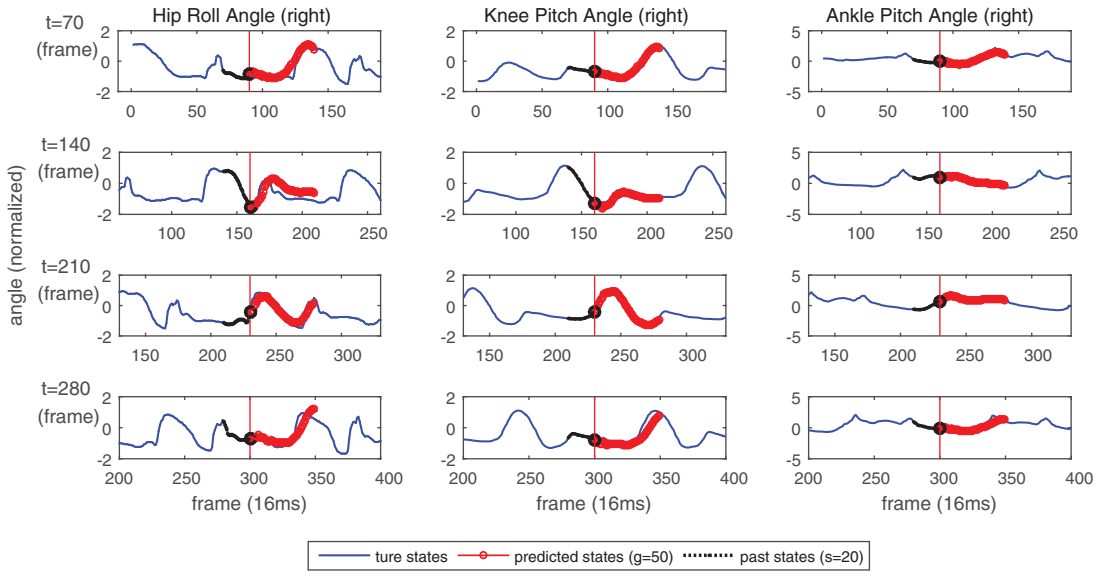


Fig. 11. 50 frames ahead prediction (walking on flat ground) for three joint angles (hip roll, knee pitch, ankle pitch of the right lower limb) of the D1 dataset are shown (subject S1). Predicted motions (red circled lines) are estimated using observed past states (black dotted lines). Trajectories of true motions are shown as blue solid lines. All angle values shown are normalized to have zero mean and unit variance as explained in Section 4.2.

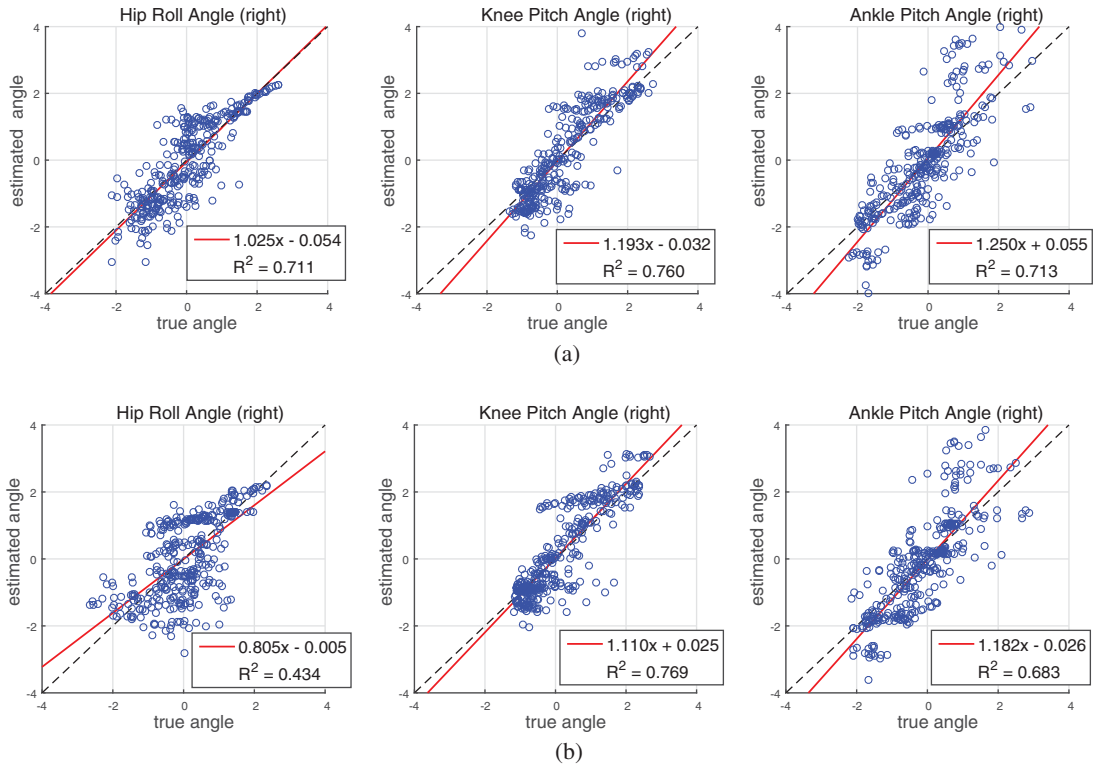


Fig. 12. Model performance after pre-training, (a) D2 (TR) and (b) D2 (TST), subject S1. Real data points x_t (x-axis) with respect to $\mathbb{E}[x_t|x_{h_t}]$ (y-axis) for three joint angles (hip roll, knee pitch, ankle pitch of the right lower limb) of the D2 dataset are shown. Linear regression results with R^2 value are also shown. The angle values shown are normalized to have zero mean and unit variance as explained in Section 4.2.

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

Table VIII. Mean and standard deviation (std.) of R^2 values over all the measurement dimensions (averaged over all the subjects) evaluated after (i) pre-training stage (imCRBM-U50), (ii) full conv-imCRBM training, for D2.

	D2 (TR)		D2 (TST)	
Training stage	Mean	std.	Mean	std.
Pre-training	0.5924	0.2089	0.4161	0.2379
Conv-imCRBM	0.9205	0.0615	0.6241	0.1946

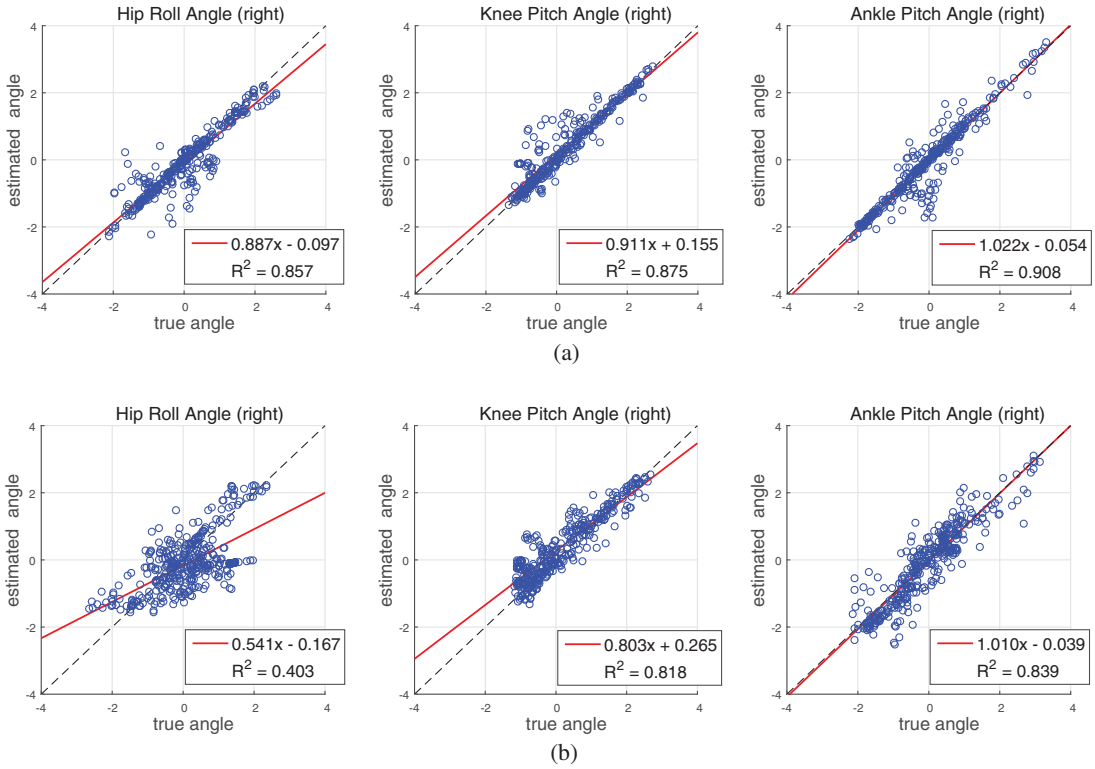


Fig. 13. Model performance after full conv-imCRBM training, (a) D2 (TR) and (b) D2 (TST), subject S1. Real data points x_t (x-axis) with respect to $\mathbb{E}[x_t|x_{h_t}]$ (y-axis) for three joint angles (hip roll, knee pitch, ankle pitch of the right lower limb) of the D2 dataset are shown. Linear regression results with R^2 value are also shown. The angle values shown are normalized to have zero mean and unit variance as explained in Section 4.2.

4.5.2. Training performance. We now analyze and compare model performance after the pre-training stage with full conv-imCRBM. The real data points x_t with respect to the estimated data points $\mathbb{E}[x_t|x_{h_t}]$, given past observations x_{h_t}, are illustrated in Figs. 12 and 13 for three joints (subject S1). We evaluate $\mathbb{E}[x_t|x_{h_t}]$ as specified in Section 4.4.

Linear regression results are shown in each figure with corresponding R^2 values. The mean and standard deviation of R^2 over all the measurement dimensions, averaged over the subjects, are given in Table VIII. The results show that conv-imCRBM clearly improves training performance over the pre-trained model, both for the training set D2 (TR) and the test set D2 (TST). Error histograms for D2 (TR) and D2 (TST) are also illustrated in Fig. 14 (subject S1). For 91.34% of the D1 (TR) data set, and 92.73% of the D1 (TST) data set, the error values after full conv-imCRBM are less than 20. (The percentage of error values less than 20 after the pre-training stage is 57.61% for D1 (TR), and 44.94% for D1 (TST)).

The shapes of the histograms, which are highly concentrated near zero for the training set D2 (TR) compared to the shapes for D2 (TST) in Fig. 14(a) and (b), suggest that the model may be overfitted. Although adjusting the parameters for the learning configurations, e.g., the number of

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

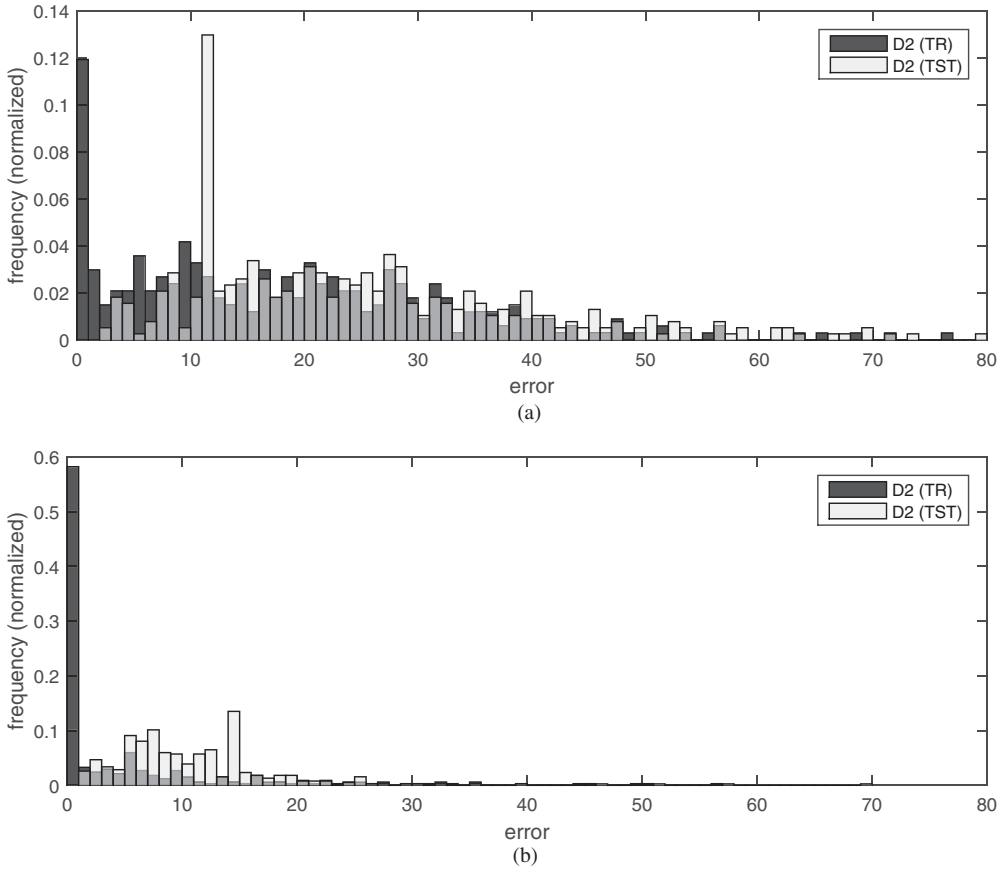


Fig. 14. Error histograms for D2, (a) after pre-training and (b) after full conv-imCRBM training, with bin size of 1 (subject S1). Error in the x-axis is defined as $\|x_t - \mathbb{E}[x_t|x_{h_t}]\|^2$. Dark bars denote D2 (TR), light bars denote D2 (TST). Gray regions denote the overlap area of the two histograms. Frequency in the y-axis is normalized (sum of frequencies for one histogram to be one) to compare the two histograms, since the sample sizes of each dataset are different.

hidden variables, may lead to a more effective general model, we do not do so, and for the purposes of this paper interpret the results as errors for the test set D2 (TST).

4.5.3. Segmentation and recognition. We segment the composite motions of D2 (TR) in the same way as described in Section 4.4, and use the resulting primitives (i.e., motion activations) to recognize the motion types for D2 (TST). Experimental results are shown in Fig. 15. We show the corresponding motion snapshots and two joint trajectories for comparison (subject S1), since we do not have exact label information for D2. In Fig. 15(c), the composite motion is segmented into the following motion activities: walking on flat ground (segment number 3), squatting (6), walking up an incline (2), walking downstairs (4), and standing (1). Segment 5 is an empty segment, where we allow empty clusters to exist while performing clustering on the patterns, f_t. The movement type classification is performed in 0.0153 s for all frames shown in the figure. Figure 16 illustrates the motions generated by each motion activity learned from our model that are used for multistep ahead prediction.

4.5.4. Prediction. We now evaluate the accuracy of multistep forward prediction using the measure defined in Eq. (14), $\mathcal{E}_{t,g}$. Prediction performance is assessed on the two test datasets D1 (TST) and D2 (TST). Results of the 20 frames ahead ($g = 20$) prediction on D1 (TST) are shown in Table IX, comparing the baseline, the unsupervised imCRBM (with $q_t \in \{0, 1\}^{50}$), and the conv-imCRBM. We use the same total number of iterations during learning for the unsupervised imCRBM and the conv-imCRBM. Performance of the CRBM is also given for comparison, trained by each corresponding class of motion to the test motions. The frames in D1 are also down-sampled by keeping every fourth frame only. All the result values are averaged over the subjects. Our experimental results show that performance of the conv-imCRBM exceeds both the baseline and the imCRBM. Although the CRBM

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

Table IX. Motion prediction error, $\mathcal{E}_{t,g}$, for D1, trained by D2 ($g = 20$).

		Motion prediction error: mean (std.) over frames (averaged over all the subjects)			
		Test data: D1 (TST)			
Model	Training data	Flat ground, D1 (TST)	Squatting, D1 (TST)	Up an incline, D1 (TST)	Downstairs, D1 (TST)
None (baseline)	·	5567.12 (1146.31)	9985.41 (2333.04)	6409.21 (1212.85)	6683.59 (1492.32)
CRBM	Flat ground, D1 (TR)	3006.74 (599.24)	–	–	–
	Squatting, D1 (TR)	–	10080.56 (3378.18)	–	–
	Up an incline, D1 (TR)	–	–	2268.45 (866.50)	–
	Downstairs, D1 (TR)	–	–	–	3600.69 (1228.71)
imCRBM-U50	D2 (TR)	7973.35 (2377.85)	11162.52 (3309.95)	7813.87 (1648.39)	8515.41 (2439.02)
conv-imCRBM	D2 (TR)	**3803.01 (1243.61)**	**5797.08 (2865.59)**	**4018.86 (1573.49)**	**4737.39 (1470.51)**

Table X. Motion prediction error, $\mathcal{E}_{t,g}$, for D2, trained by D2 ($g = 20$).

Model	Training data	Motion prediction error for D2 (TST): mean (std.) over frames (averaged over all the subjects)
None (baseline)	·	6796.68 (3022.91)
imCRBM-U50	D2 (TR)	8439.51 (3372.54)
conv-imCRBM	D2 (TR)	**4361.26 (2665.17)**

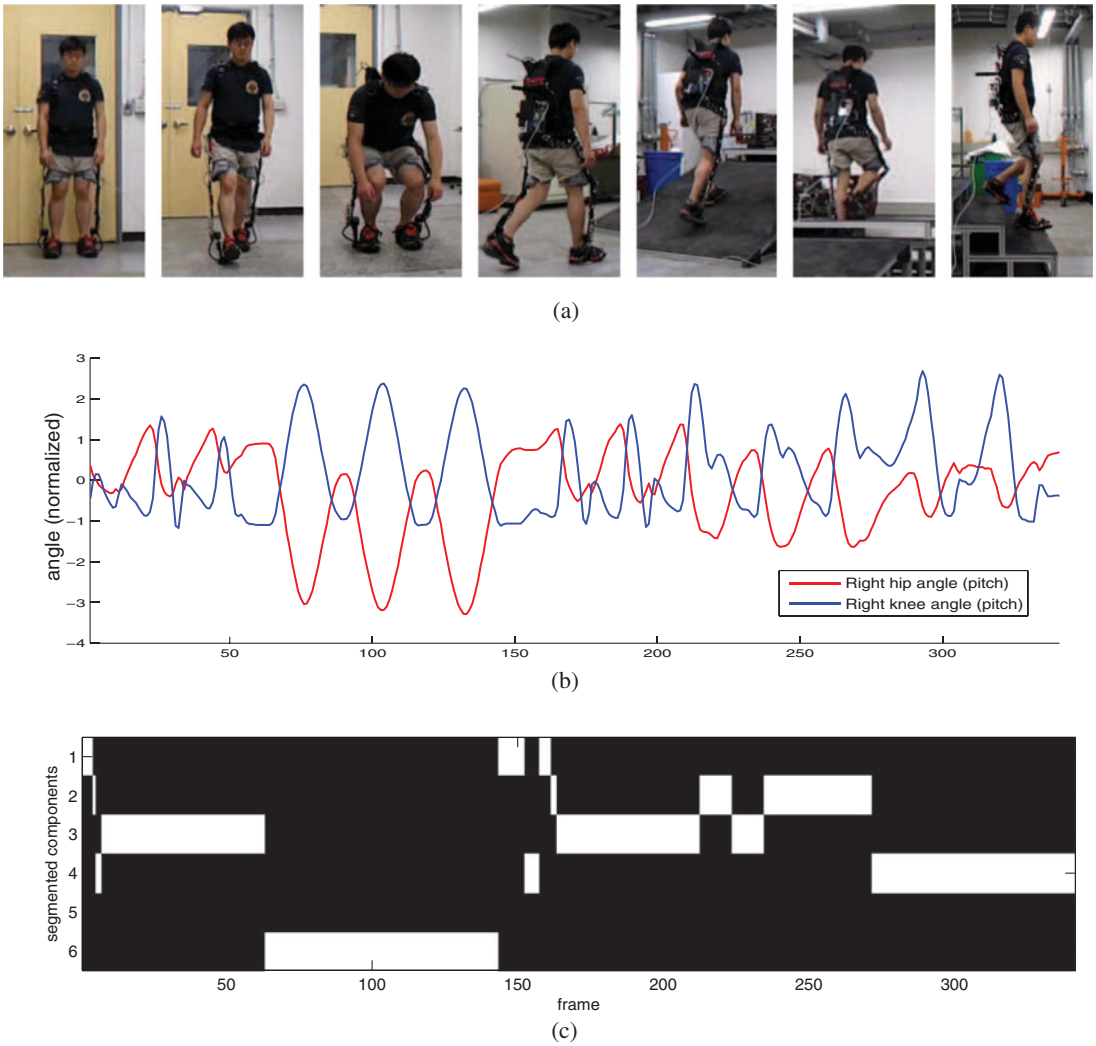


Fig. 15. Segmentation and recognition results for D2 (subject S1). (a) Composite motion experiment snapshots for D2 (TST). (b) Right hip and right knee pitch angle trajectories for D2 (TST). (c) Classification results for D2 (TST) using segmented components obtained from D2 (TR).

performs better for some motions, which to some extent is expected—recall that the CRBM is trained and tested using the same class of motion which the subject repeated under the same settings, and also that the CRBM does not include motion activity prediction given the true motion class label—the error values of the conv-imCRBM are still comparable with the CRBM. The prediction errors for D2 (TST) are given in Table X, showing that our approach outperforms both the baseline and the imCRBM. We also illustrate the predicted trajectories (red circled lines) for three joints at certain time frames (red vertical lines) compared with the true trajectories (blue solid lines) as shown in Fig. 17.

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

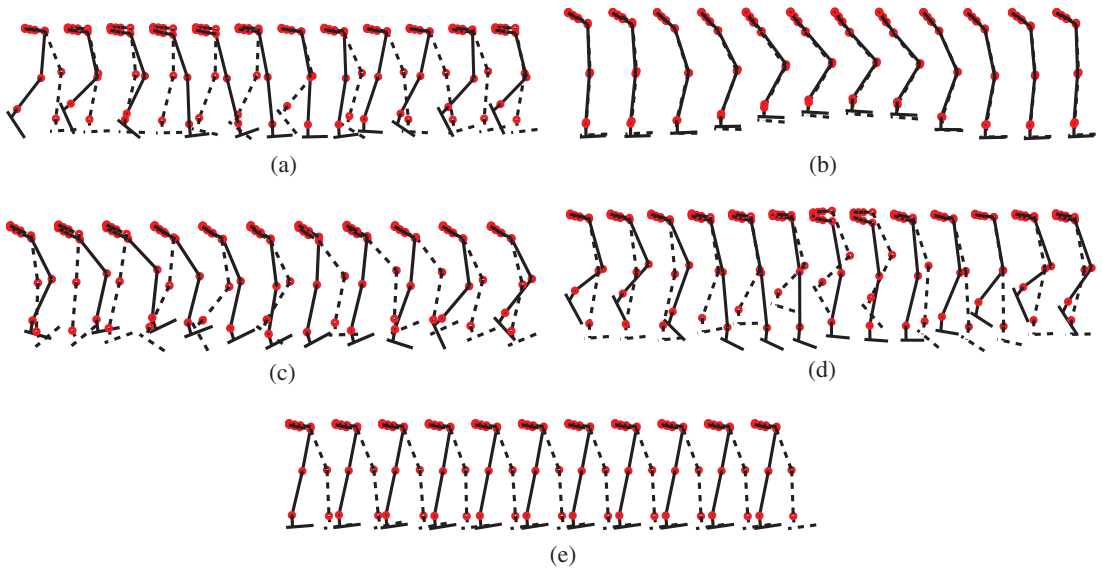


Fig. 16. Illustration of motions generated by each motion activity (which corresponds to each element in q_t) learned from conv-imCRBM. Revolute joints of the exoskeleton are indicated by red circles in the figure; each ankle consists of three overlapping joints. Position of the body link (pelvis) is represented to be fixed, since we do not have transitional information on it. (a) Walking on flat ground (segment 3). (b) Squatting (segment 6). (c) Walking up an incline (segment 2). (d) Walking downstairs (segment 4). (e) Standing (segment 1).

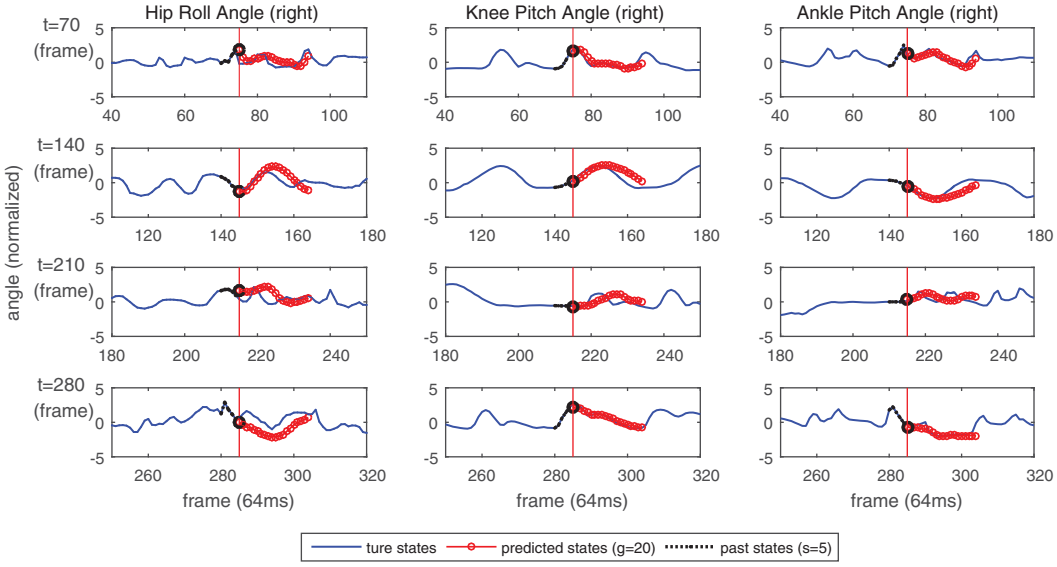


Fig. 17. 20 frames ahead prediction (composite motion) for three joint angles (hip roll, knee pitch, ankle pitch of the right lower limb) of the D2 dataset are shown (subject S1). Predicted motions (red circled lines) are estimated using observed past states (black dotted lines). Trajectories of true motions are shown as blue solid lines. All angle values shown are normalized to have zero mean and unit variance as explained in Section 4.2.

5. Conclusion

In this paper, we have proposed a neural-network-based machine-learning algorithm, called the conv-imCRBM, to predict a lower limb exoskeleton wearer's movement type and future trajectory as inputs to an exoskeleton controller. We extend the imCRBM to automatically segment unlabeled motion data into appropriate motion activities, which in turn allows for more accurate multistep ahead trajectory prediction with multiple movement types. Experimental results with a lower limb exoskeleton performing various motions (walking on flat ground, walking up and down an incline, climbing

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

and going downstairs, squatting, standing) clearly demonstrate the advantages of our approach to existing methods.

While proper movement segmentation, e.g., by clustering, is important in analyzing and modeling movement data, especially when the data consists of different types of motion, there is no standard answer for how to perform such a segmentation, as it depends on each task. Our approach provides a reasonable and effective solution for the case of multistep ahead prediction, and serves as a useful guideline for similar applications beyond exoskeleton control.

Acknowledgment

This research was supported in part by the Dual-Use Technology Program of MOTIE/DAPA/CMTC under grant 13-DU-MC-16, the BK21+ program in mechanical engineering at SNU, the SNU Biomimetic Robotics Research Center (BMRR), and SNU-IAMD.

References

1. A. M. Dollar and H. Herr, "Lower extremity exoskeletons and active orthoses: Challenges and state-of-the-art," *IEEE Trans. Robot.* **24**(1), 144–158, (2008).
2. A. B. Zoss, H. Kazerooni and A. Chu, "Biomechanical design of the Berkeley lower extremity exoskeleton (BLEEX)," *IEEE/ASME Trans. Mechatronics* **11**(2), 128–138, (2006).
3. C. Fleischer, C. Reinicke and G. Hommel, "Predicting the Intended Motion with EMG Signals for an Exoskeleton Orthosis Controller," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2005)*, IEEE (2005) pp. 2029–2034.
4. K. Kiguchi, T. Tanaka and T. Fukuda, "Neuro-fuzzy control of a robotic exoskeleton with EMG signals," *IEEE Trans. Fuzzy Syst.* **12**(4), 481–490 (2004).
5. T. Lenzi, S. M. M. D. Rossi, N. Vitiello and M. C. Carrozza, "Intention-based EMG control for powered exoskeletons," *IEEE Trans. Biomed. Eng.* **59**(8), 2180–2190 (2012).
6. G. Pfurtscheller, C. Guger, G. Mller, G. Krausz and C. Neuper, "Brain oscillations control hand orthosis in a tetraplegic," *Neurosci. Lett.* **292**(3), 211–214 (2000).
7. H. A. Varol, F. Sup and M. Goldfarb, "Multiclass real-time intent recognition of a powered lower limb prosthesis," *IEEE Trans.Biomed. Eng.* **57**(3), 542–551 (2010).
8. L. Wang, E. H. Van Asseldonk and H. Van Der Kooij, "Model Predictive Control-Based Gait Pattern Generation for Wearable Exoskeletons," *Proceedings of the IEEE International Conference on Rehabilitation Robotics (ICORR)*, IEEE (2011) pp. 1–6.
9. J. F. Veneman, R. Kruidhof, E. E. Hekman, R. Ekkelenkamp, E. H. Van Asseldonk and H. Van Der Kooij, "Design and evaluation of the LOPES exoskeleton robot for interactive gait rehabilitation," *IEEE Trans. Neural Syst. Rehabil. Eng.* **15**(3), 379–386 (2007).
10. S. Kwon and J. Kim, "Real-time upper limb motion estimation from surface electromyography and joint angular velocities using an artificial neural network for humanmachine cooperation," *IEEE Trans. Inform. Technol. Biomed.* **15**(4), 522–530 (2011).
11. Y. M. Aung and A. Al-Jumaily, "sEMG based ANN for shoulder angle prediction," *Procedia Eng.* **41**, 1009–1015 (2012).
12. Y. H. Yin, Y. J. Fan and L. D. Xu, "EMG and EPP-integrated humanmachine interface between the paralyzed and rehabilitation exoskeleton," *IEEE Trans. Inform. Technol. Biomed.* **16**(4), 542–549 (2012).
13. E. Aertbelin and J. De Schutter, "Learning a Predictive Model of Human Gait for the Control of a Lower-limb Exoskeleton," *Proceedings of the 5th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics*, IEEE (2014) pp. 520–525.
14. H. He, K. Kiguchi and E. Horikawa, "A study on lower-limb muscle activities during daily lower-limb motions," *Int. J. Bioelectromagnetism* **9**(2), 79–84 (2007).
15. R. Jimenez-Fabian and O. Verlinden, "Review of control algorithms for robotic ankle systems in lower-limb orthoses, prostheses and exoskeletons," *Med. Eng. Phys.* **34**(4), 397–408 (2012).
16. K. Anam and A. A. Al-Jumaily, "Active exoskeleton control systems: State of the art," *Procedia Eng.* **41**, 988–994 (2012).
17. T. Yan, M. Cempini, C. M. Oddo and N. Vitiello, "Review of assistive strategies in powered lower-limb orthoses and exoskeletons," *Robot. Auton. Syst.* **64**, 120–136 (2015).
18. Y. Han, S. Zhu, Z. Zhou, Y. Shi and D. Hao, "Research on a multimodal actuator-oriented power-assisted knee exoskeleton," *Robotica*, 1–17 (2016), doi: 10.1017/S0263574716000576.
19. J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake and R. Moore, "Real-time human pose recognition in parts from single depth images," *Commun. ACM* **56**(1), 116–124 (2013).
20. T. B. Moeslund, A. Hilton and V. Krger, "A survey of advances in vision-based human motion capture and analysis," *Comput. Vis. Image Underst.* **104**(2), 90–126 (2006).
21. R. Poppe, "A survey on vision-based human action recognition," *Image Vis. Comput.* **28**(6), 976–990 (2010).

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press

22. F. Zhou, F. De la Torre and J. K. Hodgins, "Hierarchical aligned cluster analysis for temporal clustering of human motion," *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(3), 582–596 (2013).
23. A. Vgele, B. Krger and R. Klein, "Efficient Unsupervised Temporal Segmentation of Human Motion," In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (pp. 167–176). Eurographics Association.
24. G. W. Taylor, G. E. Hinton and S. T. Roweis, "Modeling Human Motion using Binary Latent Variables," In *Advances in Neural Information Processing Systems* (2006) pp. 1345–1352.
25. G. W. Taylor, L. Sigal, L. Fleet and G. E. Hinton, "Dynamical Binary Latent Variable Models for 3d Human Pose Tracking," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE (2010) pp. 631–638.
26. Y. LeCun, F. J. Huang and L. Bottou, "Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting," *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004*, IEEE (2004) pp. II-97-104.
27. H. Lee, P. Pham, Y. Largman and A. Y. Ng, "Unsupervised Feature Learning for Audio Classification Using Convolutional Deep Belief Networks," *Advances in Neural Information Processing Systems* (2009) pp. 1096–1104.
28. J. W. Davis, "Hierarchical Motion History Images for Recognizing Human Motion," *Proceedings of the IEEE Workshop on Detection and Recognition of Events in Video*, IEEE (2001) pp. 39–46.
29. Q. V. Le, W. Y. Zou, S. Y. Yeung and A. Y. Ng, "Learning Hierarchical Invariant Spatio-Temporal Features for Action Recognition with Independent Subspace Analysis," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE (2011) pp. 3361–3368.
30. S. Ji, W. Xu, M. Yang and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(1), 221–231 (2013).
31. J. Tompson, M. Stein, Y. Lecun and K. Perlin, "Real-time continuous pose recovery of human hands using convolutional networks," *ACM Trans. Graph. (TOG)* **33**(5), 169 (2014).
32. A. Jain, J. Tompson, M. Andriluka, G. W. Taylor and C. Bregler, "Learning Human Pose Estimation Features with Convolutional Networks" *Proceedings of International Conference on Learning Representations 2014 (ICLR 2014)*, Banff, Canada (2014) pp. 1–14.
33. M. A. Carreira-Perpinan and G. E. Hinton, "On Contrastive Divergence Learning," *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, NP: Society for Artificial Intelligence and Statistics (2005) pp. 33–40.
34. C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer, Berlin, 2006).
35. A. Coates, A. Y. Ng and H. Lee, "An Analysis of Single-Layer Networks in Unsupervised Feature Learning," *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Fort Lauderdale, FL, USA, **15** (2011).
36. F. C. Park, "Distance metrics on the rigid-body motions with applications to mechanism design," *J. Mech. Des.* **117**(1), 48–54 (1995).
37. C. M. Bishop, "Training with noise is equivalent to Tikhonov regularization," *Neural Comput.* **7**(1), 108–116 (1995).

https://doi.org/10.1017/S0263574716000795 Published online by Cambridge University Press