Multi-Action Knee Contact Force Prediction by Domain Adaptation

Iliana Loi, Evangelia I. Zacharaki, and Konstantinos Moustakas, *Senior Member, IEEE*

***Abstract*— Most recent musculoskeletal dynamics estimation methods are designed for predefined actions, such as gait, and don't generalize to various tasks. In this work, we address the problem of estimating internal biomechanical forces during more than one actions by introducing unsupervised domain adaptation into a deep learning model. More specifically, we developed a Bidirectional Long Short-Term Memory network for knee contact force prediction, enhanced with correlation alignment layers, in order to minimize the domain shift between kinematic data from different actions. Furthermore, we used the novel Neural State Machine (NSM) as a simulation platform to test and visualize our model predictions in a wide range of trajectories adapted to different 3D scene geometries in real-time. We conducted multiple experiments, including comparison with previous models, model alignment across action classes and real-to-synthetic data alignment. The results showed that the proposed deep learning architecture with domain adaptation performs better than the benchmark in terms of NRMSE and t-test. Overall, our method is capable of predicting knee contact forces for more than one action classes using a single architecture and thereby opens the path for estimating internal forces for intermediate actions, while the knowledge of the hidden state of motion may be used to support personalized rehabilitation. Moreover, our model can be easily integrated into any human motion simulation environment, which shows its potential in enabling biomechanical analysis in an automated and computationally efficient way.**

***Index Terms*— Deep learning, domain adaptation, force prediction, BiLSTM, neural state machine, musculoskeletal modeling.**

I. Introduction

COMPUTATIONAL methods that simulate human locomotion along with the internal state of the musculoskeletal system can be essential for creating personalized rehabilitation plans [1], developing clinical assessment tools [2], monitoring the movement of a patient post-operationally [3], etc. The performance of experiments simulating realistic conditions requires solving for a variety of environmental constraints accounting for interactions of the digital human with the 3D scene. Although, significant progress has been made in computer graphics research to create, with the help of machine learning (ML) or/and deep learning (DL), visually-realistic animations of digital human-scene interactions in real-time [4], such state-of-the-art simulation platforms have not been sufficiently exploited in human biomechanics research to improve performance and robustness of the simulators and to allow for a plurality of experiments under realistic conditions.

Some efforts towards this direction have recently been made to utilize simulation outcomes from musculoskeletal modeling techniques [5], [6] in order to train ML/DL surrogate models that can help improve the performance of physics-based simulators, especially in human biomechanical function estimation. In particular, many recent data-driven approaches focus on ML-based force prediction including the estimation of internal forces (i.e. knee contact forces) during a specific action, and usually this action being gait [5], [7]. It should be mentioned that there are works that produce force estimations for various movements [8], [9], by re-training their models with kinematics and/or dynamic data from different actions, but are not offering any generalization potential to more than one actions within a single framework. More importantly, the utilized kinematic data are usually obtained through measurements or computations in simplified environments with limited human-scene interactions.

To address those limitations, we exploit a novel deep auto-regressive framework, the NSM [4], as a generator of multi-action synthetic motion. NSM allows to simulate goal-driven human locomotion with periodic and non-periodic motions and precise 3D scene interactions. Our aim was to integrate into such an advanced digital human motion simulation environment, a new methodology that estimates internal biomechanical forces, i.e. knee contact forces (KCF), during different tasks.

In particular, we developed a Bidirectional Long Short-Term Memory (BiLSTM) network that minimizes the domain shift between experimental data from different movements, aiming to provide an insight into the internal body state while generalizing to more than one actions simultaneously. For that purpose, our BiLSTM model is enriched with an unsuper-

Manuscript received 11 August 2023; revised 13 December 2023; accepted 16 December 2023. Date of publication 19 December 2023; date of current version 16 January 2024. This work was supported by the EACEA (European Union's European Education and Culture Executive Agency)—Erasmus+ through the Project: EMMBIOME (Erasmus Mundus Master in Biomedical Engineering) under Grant 101082688. *(Corresponding author: Iliana Loi.)*

Iliana Loi and Konstantinos Moustakas are with the Department of Electrical and Computer Engineering, University of Patras, 26504 Patras, Greece (e-mail: loi@ceid.upatras.gr; moustakas@ece.upatras.gr).

Evangelia I. Zacharaki was with the Department of Electrical and Computer Engineering, University of Patras, 26504 Patras, Greece. She is now with the Miller School of Medicine, University of Miami, Miami, FL 33136 USA (e-mail: exz187@med.miami.edu).

Digital Object Identifier 10.1109/TNSRE.2023.3345006

vised domain adaptation layer, called CORAL (COrellation ALignment) [10], that can be used for online estimation of forces also during action transitions. Moreover, the knowledge of the hidden states of a personalized digital human model may be potentially useful for rehabilitation purposes, such as post-surgery locomotion simulation.

In order to evaluate the effectiveness of our method to produce real-time predictions, we trained the proposed BiLSTM architecture on experimental data of two representative action classes (multi-speed gait and sit-to-stand) and integrated the trained models in the NSM environment for testing. Since NSM provides a versatility of kinematic data with various scene interaction tasks, our approach is easily extendable to other action classes, beyond gait and sit-to-stand (STS), given training data availability. Overall, our main contributions are summarized as follows:

- We implemented a BiLSTM model for simulating internal biomechanical forces during two tasks by applying unsupervised domain adaptation, i.e. the CORAL method. We show that by minimizing the domain shift between experimental data from different movements, the model may generalize also in new similar actions for which it wasn't originally trained.
- By estimating knee contact forces for more than one actions within a single modeling framework, we provide a solution to the prediction of forces for intermediate actions (movement transition) that usually require more complex architectures.
- We integrate our model into the NSM environment and assess its capability to produce real-time predictions. The augmentation of NSM with joint forces' prediction allows to enhance the 3D virtual character with realistic physics-based functionalities.

II. Related Work

A. Force Estimation by Machine Learning

Recently, researchers have turned to data-driven (machine learning) approaches for estimating human biomechanics, since they are more automated, require less parameterization and manual effort, and offer real-time solutions as well. Most works develop surrogate models for force estimation or prediction that focus on estimating medial and lateral knee contact (KC) forces [5], [7], [8], [9] or muscle forces in lower extremities [5], [9], [11] using a plethora of DL techniques, such as ANNs [7], [8], RNNs, fully-connected neural networks [9], and CNNs [5], [9], or ML algorithms, such as principal component regression (i.e. a regression analysis based on PCA) [9].

In respect to application field, ML-based solutions focus mainly in estimating tibiofemoral load data during gait [5], [7], [9], [11], sit-to-stand [9] or more rarely sport movements [8]. Models were trained using raw data (marker motion data, ground reaction forces (GRFs), muscle electromyography (EMG), IMU signals) as well as derivative data from musculoskeletal analyses (e.g. KCFs). The models were validated by comparing the networks' estimations with musculoskeletal modeling calculations [5] and/or data from publicly available databases (e.g. Grand Challenge Competition [12]). Some works such as [7] examined whether GRFs affect the estimation capability of the models, and concluded that in the performed experiments, the knee loads estimated by omitting GRFs were similar to the ones produced when trained with GRFs.

All the aforementioned data-driven approaches focus on estimating joint or muscle forces during a specific movement (e.g. gait, squatting) or train different models to predict contact forces during more than one actions. Thus, they lack a mechanism that allows them to adapt the same model during the transition from one action class to another. Addressing the domain shift is the main differentiation of our current work, which envisions in the long-term to adapt predictions while performing different actions in real time.

B. Domain Adaptation

Domain adaptation is a subcategory of transfer learning that addresses the problem of knowledge transfer between two or more related domains with different distributions, by learning domain-invariant models from data [13]. There are several domain adaptation techniques. Domain masks [14], [15] are utilized to enhance the performance of DL models by distinguishing domain-specific features from features that can be shared across domains. Another domain adaptation method is to apply a linear transformation to each domain to project the features to a common space [15], [16]. Such domain adaptation methods were used in [17] and [18] in order to minimize the domain shift between the distributions of the kinematic data obtained from different subjects and consequently enhance the inter-subject accuracy of the proposed neural network. Specifically, in [17] a regression-supervised domain adaptation framework was developed to estimate EMG-based wrist kinematics from different subjects. In [18] a supervised domain adaptation technique was utilized as the input layer of a recurrent neural network (RNN) in order to linearly transform the input features and, thus, solve the domain shift during inter-subject gesture recognition based on EMG signals.

However, domain adaptation techniques that are supervised as the aforementioned ones, cannot be applied in cases where one of the two related domains lacks labeled samples [10]. The latter case is quite common and necessitates the development of unsupervised domain adaptation methods. One approach is to learn a domain invariant subspace using both labeled source data and pseudo-labeled target data [19]. To reduce error accumulation during learning due to inaccurate pseudo-labeling, a selective pseudo-labeling strategy was proposed [19] based on unsupervised clustering analysis by exploring the structural information underlying the target domain. Other unsupervised domain adaptation techniques include domain distribution alignment [10] and matrix rank embedding to promote both feature discriminability and transferability [20]. A lot of state-of-the art works fall in this category such as [21], [22], [23], [24], and [25]. In [21] an adversarial domain adaptation network was created for Electroencephalogram (EEG) classification, which both aligns the marginal distributions of different domains and aims for decreasing the sub-domain shift. Unsupervised domain alignment was also used in [22] for deep sleep staging, while an adversarial

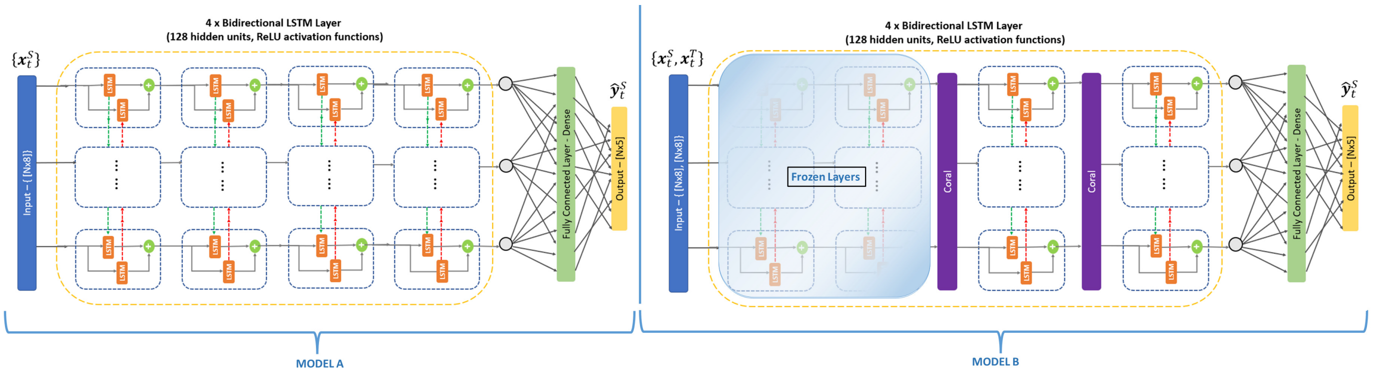

Fig. 1. A general overview of our BiLSTM architecture. Initially, we train our model with the source dataset $\mathbf{x}_t^S$ and without using the CORAL layers (Model A). Subsequently, we re-train our model (Model B) using a pair of vectors $\mathbf{x}_t^S, \mathbf{x}_t^T$ measured at the current time frame t, using the same source dataset ($\mathbf{x}_t^S$) as during initial training. During re-training, we freeze the first two pre-trained BiLSTM layers in order to use them as feature extractors and we re-train the last two as well as a fully connected layer that acts as our predictor. Each domain adaptation layer (CORAL layer) produces a D_S^* vector that consists of the transformed source features.

domain-adaptive technique was developed in [23] to detect fall events of elderly patients using sensors during different device placement and configuration scenarios. Finally, in [24] an unsupervised domain adaptation method combined with a self-guided adaptive sampling scheme was used to account for instantaneous domain shift during classifier updates. The latter enhances the EMG feature learning across subjects during gesture recognition.

In this work, we performed unsupervised domain adaptation based on correlation alignment [10] in order to minimize the domain shift between experimental data stemming from different actions. Our model may be easily integrated into any motion synthesis framework, since CORAL is unsupervised, avoiding the need for labeled data in the target domain. More details on our implementation of domain adaptation are provided in section III-A.

III. Methods

A. Unsupervised Domain Adaptation

Considering that our target domain data is unlabeled (no KCFs provided), we introduce in our modeling framework an unsupervised domain adaptation technique, in order to enable the estimation of contact forces concurrently with motion synthesis in any virtual environment, like the NSM. Let's assume that the dataset in the source domain contains n_S labeled samples and is denoted with $D_S = (\mathbf{x}_i^S, \mathbf{y}_i^S)_{i=1}^{n_S}$, where $x_i^s \in R^d$ is the feature vector of sample i (d is the number of input variables) with corresponding label $\mathbf{y}_i^S$. The dataset in the target domain, $D_T = (\mathbf{x}_j^T)_{j=1}^{n_T}$, consists of n_T unlabeled samples with same number of input variables, $\mathbf{x}_j^T \in R^d$ [13], [19].

We used CORAL, an unsupervised domain adaptation method [10], to align the distributions of the source and target domain. This is performed by computing the covariance of the source and target features and linearly transforming the source data [10], which can be mathematically formulated as:

$$C_S = cov(D_S) + H_S$$
$$C_T = cov(D_T) + H_T$$
$$A = C_S^{\frac{-1}{2}} * C_T^{\frac{1}{2}}$$
$$D_S^* = D_S * A \tag{1}$$

where $cov(D_S)$ and $cov(D_T)$ are the covariances of the domain and target features, respectively. H_S is a diagonal matrix with a small regularization parameter λ on its diagonal elements, which is set to 1. This is added to the covariance matrix of the source, to make it explicitly full rank. Matrix H_T is the same matrix as H_s but with its second dimension corresponding to the second dimension of the target matrix, D_T. By applying a linear transformation A to the original source features, the distance between the covariances of the source and target domain is minimized. We used the following (Eq. 2) residual distance as an evaluation measure of the effectiveness of the introduced CORAL layer:

$$E = \|C_{\hat{S}} - C_T\|_F^2 = \|A^T C_S A - C_T\|_F^2 \tag{2}$$

where $C_{\hat{S}}$ is the covariance of the transformed source features, D_s^* [10].

We developed CORAL as a custom layer using functional Keras API [26]. Similarly to other works [10], [15], our BiLSTM network is used as a representation learner, while the CORAL layers are introduced to align the extracted features, as illustrated in Fig.1 (Model B). We initially train our model using available motion capture data without the CORAL (Model A in Fig.1). Then, we introduce domain adaptation as the last two layers of our architecture and we re-train our model using the same (as in Model A) source data and a different dataset as the target. The domain shift can be due differences in movement pattern (action) or differences in the data generation process (real or synthetic). The different evaluation scenarios are described in section V.

B. Deep Learning Models

1) BiLSTM for Knee Contact Force Prediction: BiLSTM was introduced in [27] and it is an extension of the LSTM architecture that consists of an RNN with two parallel sequences of forward and backward feedback connections, used to remember previously parsed data and to prevent them from gradually vanishing during training. In this way, the model is exposed to both past and future information with respect to a specific time frame, which results in producing more accurate predictions than unidirectional LSTMs [27], [28].

In this work, we developed a deep BiLSTM network to predict medial and lateral knee contact forces based on human kinematics data. By the term "deep" we refer to a stacked BiLSTM architecture where the output of each BiLSTM hidden layer will be the input to the next BiLSTM hidden layer (Figure 1). As suggested by recent publications [29], [30], the stacked layered approach is considered to have better performance compared to single-layer approaches, since a plethora of hidden layers achieves higher levels of representations of time sequence data [29] and, hence its more useful for time series prediction as the problem discussed in this publication.

Our baseline BiLSTM network (Model A) consists of 4 BiLSTM layers, each containing 128 units, and a fully connected dense layer as the final layer producing 5 outputs - as many as the KCFs components we wish to predict. This architecture was designed based on experiments which showed that 2 or 3 BiLSTM layers are too few and produce high error values, while 5 or more result in almost identical results with our current 4-BiLSTM layer architecture. Fewer layers account for the use of less computational resources, hence less training time. As for the hyperparameters, each layer has a Rectified Linear Unit (ReLU) activation function. The adaptive moment estimation (Adam) [31] optimizer was utilized with a learning rate of 0.001, to ameliorate the problem of gradient descent getting stuck at local minima during training, and to accelerate the convergence of the learning process [30]. The model was trained over 30 epochs with a batch size of 32 and converged after approximately 15 epochs. Moreover, the early stopping method was used to prevent the model from over-fitting. Table I provides a comparison between different hyperparameter values for our baseline BiLSTM (Model A) in terms of Normalised Root Mean Square Error during gait action. Smaller batch size lets the model process the training dataset in smaller portions at a time, while more epochs aid in exposing the model to the same data for more iterations, thus leading to better convergence and accuracy. As for the learning rate, which determines how often the model's weights are updated, it is observed from Table I that the default value of 0.001 offers a slight boost in performance. The BiLSTM model was developed and trained in Python Keras.

The input to our final model (Model B) is a pair of vectors $\mathbf{x}_t^S, \mathbf{x}_t^T$, where $\mathbf{x}_t^S$ is the *source* input vector and $\mathbf{x}_t^T$ is the *target* input vector. Each instance of source $\mathbf{x}_t^S \in R^2$ and target $\mathbf{x}_t^T \in R^2$ input data that our model parses is a two-dimensional vector with dimensions $[t, k]$, where k is the number of joint angles measured at the current time frame t. As for the output, each instance of our source output data is a 2D vector as well, $\hat{\mathbf{y}}_t^S \in R^2$ with dimensions $[t, l]$, that consists of the knee joint forces that our model predicts. In our case, each input vector will contain 8 joint angles, as much as the virtual avatar used for visualizing NSM's results has, and each output vector will have 5 KCFs. However, every LSTM, as well as a BiLSTM model has a 3D input of dimensions $[N, n, f]$ where N is the number of samples (subjects) parsed by the model at each iteration (i.e. during training $N = batchsize$, whereas $N = 1$ during inference), n is the number of time

TABLE I
HYPERPARAMETERS COMPARISON FOR OUR BILSTM (MODEL A) MODEL IN TERMS OF NRMSE DURING GAIT MOVEMENT

BS = 32, EPS = 30	medKCFx	medKCFy	medKCFz	latKCFy	lat_KCFz
LR = 0.01	1.76	2.77	2.47	2.90	3.91
LR = 0.0001	1.80	3.42	3.57	3.25	3.26
LR = 0.001, EPS = 30	**medKCFx**	**medKCFy**	**medKCFz**	**latKCFy**	**lat_KCFz**
BS = 64	1.66	2.28	2.14	3.92	3.93
BS = 128	2.53	3.26	3.32	4.11	4.17
LR = 0.001, BS = 32	**medKCFx**	**medKCFy**	**medKCFz**	**latKCFy**	**latKCFz**
EPS = 10	1.75	2.83	2.77	3.19	3.12
EPS = 20	0.92	2.26	2.37	2.71	2.79
Final configuration	**medKCFx**	**medKCFy**	**medKCFz**	**latKCFy**	**latKCFz**
Model A	0.36	1.97	1.72	1.91	1.90

[a] Average (across all folds) NRMSE between predicted and experimental KCFs on gait. "LR" stands for learning rate, "BS" for batch size, and "EPS" for epochs. The final configuration is LR = 0.001, BS = 32, EPS = 30.

steps that the model predicts and f is the number of input features, thus, our input is reshaped accordingly. Moreover, the source features produced by the first two BiLSTM layers of our model are introduced to the CORAL layers, which produce the transformed features D_s^*, a vector of dimensions $[t, k]$.

For the BiLSTM layers of Model B, the same hyperparameters as Model A were used, i.e. 128-unit layers with ReLU activation function and Adam with 0.001 learning rate as the optimizer. Moreover, re-training was set to 20 epochs with 32 batch size. During training, the model was evaluated using the 3-fold-cross-validation scheme across all N subjects (also known as the leave-subject-out evaluation method), meaning that all instances in our dataset are split into 3 folds where the $\frac{2}{3}$ are used as training data and the remaining $\frac{1}{3}$ as test data. This process is repeated until each and every fold appears in the test data. An average prediction accuracy was then computed from the results of the three folds.

2) Artificial Neural Network Model (ANN_2): For comparison to the proposed BiLSTM architecture, we implemented a feed-forward ANN model (as an extension of previous work [7]) and trained it to predict KCFs beyond gait motion. This new ANN, which we will be referring to as ANN_2 [32], has 3 hidden layers instead of 2 and fewer neurons at each hidden layer than the original ANN [7], namely 121 instead of 400. Some network hyperparameters were also modified, i.e. the batch size was set to 256 and the learning rate to 0.0001, and the use of biases was enabled. The network was trained and tested in Python Keras [26]. During training only joint angle measurements were used and GRFs. Similarly to BiLSTM, the ANN_2 model was evaluated with 3-fold cross-validation across all subjects (n = 54 for gait and n = 19 for the STS human motion capture dataset). More details regarding the utilized datasets will be provided in Section IV.

C. A Deep Auto-Regressive Model - The Neural State Machine

NSM [4] is a deep auto-regressive algorithm for goal-driven prediction of a virtual character's motion and interaction with

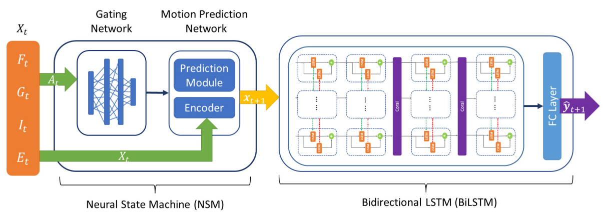

Fig. 2. Schematic diagram of our KCF prediction framework within the NSM [4]. From the output of NSM, the virtual character's predicted pose, x_{t+1} (i.e. joint angles) is given as input to our BiLSTM (Model B) model to estimate KCFs $\hat{y}_{t+1}$ at the subsequent frame $t + 1$.

scene objects in real time. As illustrated in Fig. 2, NSM consists of two machine learning networks; i) the motion prediction network consisting of a prediction and an encoder module, which synthesizes the virtual character's pose in the current and future time frames based on the user's control signals and the character's state (input X_t, at time frame t, i.e. the character's pose, trajectory data, goal position and orientation, action at the goal, the interaction scene geometry, etc.) and ii) the gating network, a fully connected network that is responsible for the transition between different action classes based on a subset of parameters of the current input, $A_t \subseteq X_t$. Overall, NSM takes as input the character's pose, trajectory, and goal data from the current frame t, and predicts those parameters for the next frame $t + 1$. Furthermore, this model enables the generation of various action classes - walk, run, sit, open, carry, climb, and idle - while supporting the automatic transition between them like sit-to-idle [4].

In this work, we integrated our BiLSTM architecture (more specifically Model B) to the NSM in order to perform real-time knee contact force prediction simultaneously with motion prediction, as illustrated in Fig. 2. More specifically, a subset of parameters from the output of NSM, i.e. the character's pose (predicted joint angles) are introduced to our BiLSTM (i.e. Model B) architecture in order to predict KCFs. We integrated our enhanced model (Model B) into NSM, rather than our baseline model (Model A), because the synthetic data produced by NSM follow a different distribution than the experimental data being available for model training. Another issue was that NSM does not provide force predictions and so we could not train a network solely on synthetic data from NSM. Consequently, before incorporating our model into NSM, we re-trained our network using real (human experimental) data as source and synthetic (virtual data from NSM) as target, minimizing the domain shift. The estimated KCFs are displayed in real-time in the NSM environment for visual assessment.

IV. Datasets

A. Gait of Variable Speed

The experimental gait dataset was obtained from previous work [7] and it contains gait data of 54 healthy subjects both young (mean age 22 years) and older (mean age 69.6 years) who were asked to perform gait trials of variable speed, i.e. speed increasing from 3 to 7 km/h, with an increment of 1 km/h and monitored with a 10-camera VICON system. The overall dataset consists of 4874 gait trials. The trajectory of 42 markers was recorded, which were placed on specific anatomical landmarks of the subjects' body, in order to capture the motion of the lumbar, hip, knee, ankle, and pelvis.

Since this experimental dataset contains physics-based simulations of body forces, it can be used as ground truth for training our BiLSTM model. However, the trained model cannot be directly applied in a virtual environment such as the one of NSM due to potential differences in the input data distributions. In order to mitigate the domain shift through domain adaptation, the target data space had to be determined. For this purpose, we created a synthetic gait dataset in the virtual environment by monitoring the movement of the virtual avatar in the NSM, while performing gait trials of various speeds in real-time. More specifically, the joint angles of the 3D character were recorded using C++ scripts, and the data were stored in Excel files. During this process, we provided user signals (goals) that would lead the 3D character only to the gait state avoiding transitions to other states.

As for the pre-processing, the KCFs were calculated through musculoskeletal modeling. Specifically, the markers' spatial trajectories and GRFs were used to extract joint kinematics (angles) and KCFs through musculoskeletal modeling processes carried out in OpenSim [33]. Through this process, joint angles for lumbar extension/bending/rotation, hip flexion/adduction/rotation, pelvis tilt/list/rotation, knee angle, patella-knee-angle, ankle angle, and subtalar angle are computed, as well as six components of medial and lateral KCFs. Subsequently, these data are being transformed to dataframe format using Python Data Analysis Library (Pandas) [34].

In other words, our input data are stored in a 2D vector of dimensions $[M_g \cdot W_g, k_g]$, where M_g is the number of trials for all subjects, W_g, is the duration of each input gait instance and $k_g = 13$ is the number of joint angles. Likewise, our output is a 2D vector of dimensions $[M_g \cdot W_g, l_g]$, where $l_g = 5$ is the number of KCFs. In particular, as we mentioned earlier, we computed three medial (in the x, y and z axes) and three lateral (in the x, y and z axes) knee contact force components, but we consider only two lateral forces, since the lateral force in the x-axis, is practically negligible, thus, resulting in 5 KCFs in total. All time series are synchronized by resampling in order to have the same duration which is essential for temporal pattern comparison in time series forecasting problems.

Before introducing the data to our network we omit variables whose standard deviation is less than 10^{-6}, as well as the patella-knee-angle, subtalar angle, lumbar extension, lumbar bending and lumbar rotation, since there is no correspondence for these two joints with the skeletal avatar of NSM. These variables are not included in the contact force prediction problem for the skeletal virtual character in the NSM application, i.e. the size of input data is reduced to $k_g = 8$. All input and output data are normalized in the range $[0, 1]$. More specifically, for the normalization of output data, scaling parameters were stored as part of the modeling framework and used to scale the predicted values back to their original range. Furthermore, we performed outlier detection and trimming by rejecting values that were higher than the 99% quantile of the data distribution or lower than the 1% quantile of the data distribution. The same procedure was followed to pre-process the corresponding synthetic gait dataset.

B. Sit-To-Stand

To evaluate whether our method generalizes to actions other than gait, we tested our model using a STS human motion capture dataset, which is part of the publicly available KIT Whole-Body Human Motion Database[1] [35], [36]. The KIT database, contains motion capture, auxiliary (e.g. external and internal forces), and anthropometric data as well as video recordings from 53 different subjects (16 female and 37 male) aged from 15-55 years, while performing a wide range of actions including environment and human-object interactions. Moreover, this database was recently enriched by data obtained from 2 subjects while performing 12 actions of bimanual daily household activities (cooking chores like peeling fruits and vegetables, mixing ingredients, pouring, etc., cleaning chores such as sweeping and more) [37].

The STS dataset extracted from KIT contains 266 trials of both left and right leg joint angle measurements from 19 subjects, which were recorded using an optical Vicon MX motion capture system with 56 markers covering specific anatomical landmarks of the whole body. Similarly to the gait motion case, this STS dataset will be used as the source dataset in our experiments, thus, knee joint forces were added through OpenSim Analyses as described in Section IV-A. Furthermore, a STS synthetic dataset was also created using the NSM framework in order to record the virtual avatar's joint angles while sitting on chairs with various geometries. This dataset was used as the target dataset in one of our experiments.

The experimental STS dataset was pre-processed following the pipeline described in [38], hence the same parameters, i.e. joint angles during pelvis tilt/list/rotation, hip flexion/adduction/rotation, lumbar extension/bending/rotation, knee angle, patella-knee-angle, ankle angle and subtalar angle, as well as internal knee forces, were extracted by motion marker's raw positions and GRFs via OpenSim's IK and JRA, respectively. During this movement, the joint angles of lumbar extension, lumbar bending, lumbar rotation and subtalar angle have very small (close to 0) values, thus, were omitted. In addition to the subtalar angle, the patella-knee angle was also omitted from the dataset, since the skeleton of NSM's 3D character does not have these two joints.

The input data $\mathbf{x}$ of the STS dataset are stored in a 2D dataframe of dimensions $[M_s \cdot W_s, k_s]$, where M_s is the total number of trials for all 19 subjects, W_s, is the duration of each trial and $k_s = 8$ is the number of joint angles after omitting the aforementioned parameters. The output, $\mathbf{y}$, is also a 2D vector of dimensions $[M_s \cdot W_s, l_s]$, where $l_s = 6$ are the number of the KCFs. In contrast to the gait movement, here, the medial force in the y-axis is negligible and thus is omitted, reducing the number of the KCFs to $l_s = 5$. Synchronization by resampling was subsequently performed, similarly to the gait dataset. Both input and output data were normalized by subtracting the mean and dividing by the standard deviation. Outlier detection and trimming was also performed in this case. Moreover, we followed the above procedure to pre-process the synthetic STS dataset.

[1]https://motion-database.humanoids.kit.edu/list/motions/?page=1&advanced_mdt_search_term=sit+AND+stand

V. Results

We conducted three sets of experiments that are summarized in Table II). First, we evaluate our baseline model without the CORAL layers (Model A), by training it with either the gait or the STS experimental dataset (columns *Exp1_Gait* and *Exp1_STS*. The results are compared with methods from previous works [7], [32] utilizing the same datasets and cross-validations setting.

In all experiments the loss function used to train both BiLSTM and ANN models was the *Mean Square Error (MSE)* between biomechanical simulations (serving as ground truth) and model predictions. Moreover, to assess the obtained results we used the *Normalised Root Mean Squared Error (NRMSE)* [7] which, being scale-invariant, allows to compare errors across action classes and components of forces (medial, lateral).

Subsequently, we present the results of our baseline model when enriched with the unsupervised domain adaptation technique, i.e. CORAL layers [10] (column *Exp2* in Table II). In this experiment, the first layers of the pre-trained (using the experimental gait dataset) Model A provide a representation learner mechanism, while CORAL is implemented within the last two layers of our architecture (Model B) to align source and target features. We then re-trained our model using the gait experimental dataset as the source and the STS experimental dataset as the target dataset. Finally, we tested our enhanced model (Model B) on the STS (target) data in order to assess whether a model originally trained on gait data can be used for KCF estimation during a different motion type (action) using a simple domain adaptation method, such as CORAL.

Finally, in order to assess whether our model can be integrated into a virtual environment for real-time motion synthesis, such as the NSM, we conducted a similar experiment as before, but using the synthetic gait and STS datasets (described in Section IV) as target domain (columns *Exp3_Gait* and *Exp3_STS* in Table II). More specifically, we trained Model B (i) using the gait experimental dataset as the source and the gait synthetic dataset as the target and (ii) using the STS experimental dataset as the source and the STS synthetic as the target dataset, respectively. Similarly to [32], we visualize the medial and lateral KCFs on the virtual character, so as to provide real-time force estimation feedback. In this experiment, we address a completely different problem than in Experiment 2 since we try to calibrate two datasets that have different distributions, yet they share the same action class, whereas in Experiment 2 we try to align data of different distributions and classes.

A. Experiment 1: Evaluation of Proposed BiLSTM

To assess our framework, we compared the performance of our Model A (without using any domain adaptation) with previous machine learning models in a knee contact force estimation scenario In Table III the average across folds

TABLE II
SUMMARY OF EXPERIMENTS PERFORMED IN THIS STUDY

	Exp1_Gait	Exp1_STS	Exp2	Exp3_Gait	Exp3_STS
woDA	✓	✓	-	-	-
wDA	-	-	✓	✓	✓
real_Gait	✓	-	✓(source)	✓(source)	-
real_STS	-	✓	✓(target)	-	✓(source)
synth_Gait	-	-	-	✓(target)	-
synth_STS	-	-	-	-	✓(target)
Results	Table III	Table IV, Table V	-	-	

[b] Two experiments were performed without domain adaptation (woDA) and three experiments with domain adaptation (wDA). The utilized datasets included experimental (real_Gait, real_STS) or virtual (synth_Gait, synth_STS) gait or STS measurements. Source and target domain are indicated for wDA experiments.

TABLE III
COMPARISON OF OUR BILSTM (MODEL A) MODEL AGAINST PREVIOUS MODELS IN TERMS OF NRMSE DURING GAIT AND SIT-TO-STAND

$NRMSE_{Gait}$	medKCFx	medKCFy	medKCFz	latKCFy	latKCFz
(Model A)	0.36	1.97	1.72	1.91	1.90
SVR [7]	1.73	5.85	3.80	4.66	4.65
ANN [7]	1.81	5.39	3.85	4.59	4.59
$NRMSE_{STS}$	medKCFx	medKCFy	medKCFz	latKCFy	latKCFz
(Model A)	0.11	0.71	0.00	0.10	0.19
ANN_2	0.75	3.49	5.86	3.02	3.48

[c] Average (across all folds) NRMSE between predicted and experimental KCFs and on the same gait and STS dataset, respectively.

NRMSE of ANN and SVR models [7] are reported. All compared models were trained with the same dataset, i.e. without taking into consideration motion-dependent variables (e.g. GRFs), and were also evaluated using the same leave-subject-out validation scheme.

Subsequently, we conducted the same experiment by training our BiLSTM model (Model A) with the experimental STS dataset and compared our results against the ANN_2 model [7], [32], which was also trained and tested using the same STS dataset. The results are also reported in Table III (rows 6 and 7).

We observe that in both cases (gait and sit-to-stand) Model A performs better than previous models presented in [7] and [32]. This positive result most probably is attributed to the ability of RNNs to generate predictions by exploiting temporal dependencies, something that is lacking in fully connected feed-forward networks like ANN. RNNs have a recurrent architecture that acts like a memory mechanism. Thus, unlike feed-forward networks, RNNs have the ability to process time sequences and to provide both estimations (of the same time frame t) or predictions (of the next time frame $t+1$). Especially a BiLSTM can manipulate both past and future time dependencies due to its architecture (it contains both a forward and a backward sequence) and is usually preferred over conventional models and unidirectional LSTMs.

B. Experiment 2: Model Alignment Across Action Classes

The average NRMSE for all folds are reported in Table IV and illustrate how well our model performs with and without the use of the CORAL layers in the STS dataset. Lower

TABLE IV
MODEL COMPARISON WITH AND WITHOUT DOMAIN ADAPTATION IN STS DATASET

RMSE	medKCFx	medKCFy	medKCFz	latKCFy	latKCFz
Model A (woDA)	0.17∗	0.18∗	0.03	0.88∗	0.15∗
Model B (wDA)	0.15∗	0.07∗	0.09	0.72∗	0.07∗
NRMSE	medKCFx	medKCFy	medKCFz	latKCFy	latKCFz
Model A (woDA)	0.78	0.25	< 0.01	0.74	0.39
Model B (wDA)	0.57	0.10	< 0.01	0.60	0.019
R^2	medKCFx	medKCFy	medKCFz	latKCFy	latKCFz
Model A (woDA)	0.90	0.86	0.92	0.80	0.84
Model B (wDA)	0.93	0.91	0.94	0.85	0.89

[d] Comparison of proposed Model B (w/ domain adaptation) against the baseline Model A (w/o domain adaptation). Average (across all folds) NRMSE between predicted and experimental KCFs and on the same STS dataset. Moreover,The values marked with an asterisk (*), are multiplied by 10^{-3}.

NRMSE values indicate that the model estimates better new (unseen) data. As mentioned above, we apply the CORAL layers to our pre-trained model and re-train the last layers using the gait experimental dataset as the source and the STS experimental dataset as the target. Then, we test our model using the STS dataset. In order to evaluate the effectiveness of our architecture's domain adaptation (i.e. CORAL) layer, we computed the residual distance of source and target features (Eq.2) before and after domain adaptation. The domain shift before applying linear transformation A (see Eq.1) to the source features is $E_{before} = 73739.36$, while after linearly transforming the data (using A), it reduces down to $E_{after} = 0.62$, indicating that our implementation works as expected.

As illustrated in Table IV, our model produces more accurate predictions when using the CORAL-based domain adaptation technique, compared to not using it. Both Pearson's Correlation Coefficient and NRMSE values in Table IV show that predictions with CORAL (Model B) tend to be more correlated (higher R^2) and closer (smaller NRMSE) to ground-truth (experimental) measurements for all KCF components.

To explore whether the predictions obtained w/ or w/o domain adaptation, i.e. $\hat{\mathbf{y}_{wDA}}$ produced by Model A and $\hat{\mathbf{y}_{woDA}}$ by Model B are significantly different, we calculated a paired samples t-test at significance level $\alpha = 0.05$. The null hypothesis was that the mean difference between the two sets of paired results is zero, hence we test if there is a statistically significant difference between the means of our models' predictions. As "pair", we refer to KCF predictions during STS or gait movement at the same time frame t. Moreover, we conducted the same t-test between the predictions of the proposed Model B ($\hat{\mathbf{y}_{wDA}}$) and the ground truth, i.e. the experimentally measured KCFs during STS or gait ($\mathbf{y}$). To sum it up, we calculated 4 paired samples t-tests: i) Model A vs Model B during STS movement to test if the improvement in the KCF prediction through our approach is statistically significant (the smaller p-values the better the new approach), ii) Model B vs Ground-truth STS data to understand whether the residual errors are significant (in this case the larger p-values the better our method), and same statistical tests for the gait data. The p-values were computed automatically, using built-in Python functions. What is important is that if the p-value is less than the level of significance α, then the null hypothesis is rejected indicating that the means of the compared results are statistically different.

The obtained p-values of paired samples t-tests are illustrated in Table V. All p-values in the first row are less than α (i.e. null hypothesis is rejected), indicating that the predicted KCF components during STS w/ or w/o domain adaptation are significantly different and thus highlighting the contribution of the proposed architecture using CORAL (Model B). The latter results were expected, since in this experiment both Model A and B are trained based on the gait experimental dataset (source). Therefore, a model solely trained on gait (Model A) would produce predictions that differ from the ones of a model with domain adaptation (Model B), which has the advantage of incorporating knowledge of the target, STS, dataset. Furthermore, the second row shows that the KCF predictions of Model B do not significantly differ from the STS ground-truth values (real human KCF data), thereby further supporting the validity of our approach. As for the KCF predictions in the gait dataset, we observe that the proposed model reliably reproduces ground truth (Table V, fourth row), and its predictions are not significantly different from the ones w/o domain adaptation (Table V, third row). To further support our findings the p-values between the predictions of Model B and the ones of ANN_2 during STS movement are less than $alpha$, meaning that there is a statistically significant difference between the means of the predictions of the two models, indicating our proposed model (Model B) performing better. Finally, in the last 3 rows of Table V, we provide the p-values for our 1^{st} Experiment. Hence, we test whether the predictions of Model A are significantly different from the ones of i) ANN and ii) SVR architectures that were proposed in [7] during gait movement, as well as the ones of iii) ANN_2 [7], [32] during STS action. As observed, in all cases the p-values are greater than $\alpha = 0.05$, meaning that, although the predictions of Model A are better (smaller NRMSE) than the ones from previous frameworks, the difference is not statistically significant.

C. Experiment 3: Real-to-Synthetic Model Alignment

Our enhanced with domain adaptation model (Model B in Fig.1) is integrated into the real-time auto-regressive motion prediction/synthesis framework to test the capability of our model to produce real-time predictions and to exploit the synthesized motion trajectories in order to forecast the contact forces required to achieve a desired goal. To do so, we re-trained our model using either the experimental gait dataset as source and the synthetic gait dataset as target, or the experimental STS dataset as source and the synthetic STS dataset as target as illustrated in the 3rd column of II. Since NSM does not generate joint forces, we cannot train a network based entirely on synthetic data from NSM, and, thus, unsupervised domain adaptation (i.e. integration of CORAL layers) is necessary in order to minimize the domain shift between the distributions of real and virtual data that are of the same action class. Although we are not able to quantitatively assess the predictions in the virtual environment due to lack of "ground truth", we use the evidence obtained from the gait-to-STS

TABLE V
SIGNIFICANCE OF DIFFERENCES IN PREDICTION (P-VALUE) DURING GAIT AND STS

Comparison with Ground-Truth					
P-value	latKCFy	latKCFz	medKCFx	medKCFy	medKCFz
Model A vs B (STS)	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01
Model B vs G-T (STS)	0.48	0.47	0.09	0.76	0.21
Model A vs B (Gait)	0.88	0.10	0.23	0.22	0.96
Model B vs G-T (Gait)	0.11	0.49	0.56	0.12	0.35
Comparison with Previous Methods					
P-value	latKCFy	latKCFz	medKCFx	medKCFy	medKCFz
Model B vs ANN_2 (STS)	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01
Model A vs ANN (Gait)	0.61	0.55	0.33	0.13	0.19
Model A vs ANN_2 (STS)	0.44	0.16	0.49	0.30	0.48
Model A vs SVR (Gait)	0.96	0.13	0.48	0.32	0.65

[e] P-values of the paired samples t-test for two sets of outcomes: (i) comparing models w/ or w/o domain adaptation (Model A vs Model B) and (ii) assessing the proposed model against the experimentally measured KCFs (Model B vs Ground-Truth) as well as iii) comparing our baseline (Model A) and proposed model (Model B) with previous frameworks. Acronym "G-T" stands for "Ground-Truth".

experiments (Section V-B) and deduce that our implementation using CORAL improves predictions in the case of domain shift.

In order to qualitatively assess the performance of our framework in a human motion simulation environment, we incorporated a visualization mechanism in NSM. Particularly, we visualized the medial and lateral KCFs on the digital human model while the character moves and interacts with virtual scene objects in real time. As shown in Fig. 3, we embedded a sphere on the left knee of the human model, using a cyan-to-red color scale to indicate the magnitude of the force (cyan/red corresponding to minimum/maximum values of the predicted KCF force, respectively). By observing Fig. 3, during the idle state the sphere is colorized cyan, since KCFs have the least possible value, whereas the sphere becomes red when the walking speed of the 3D character is increased. All of the above confirm our expectations and indicate that our framework is applicable in real-time scenarios. Moreover, we added two lines (green and red lines as illustrated in Fig. 3) on the knee contact points (force action points), which represent both the direction and the magnitude of each one of the forces, medial or lateral (i.e. the length of the lines change according to the KCFs magnitude). It is worth mentioning that in Fig. 3, the arrows depicted on the floor of the application, indicate the trajectory of the movement of the character, and they are part of the NSM virtual environment. Moreover, our model runs at approximately 60 frames-per-second (fps), pointing out the capability of our framework to produce real-time predictions.

VI. DISCUSSION AND FUTURE WORK

In this work, we developed a joint contact force prediction framework based on deep learning and unsupervised domain adaptation techniques. DL approaches such as the one proposed in this work, are surrogate models for conventional musculoskeletal and computational techniques. These

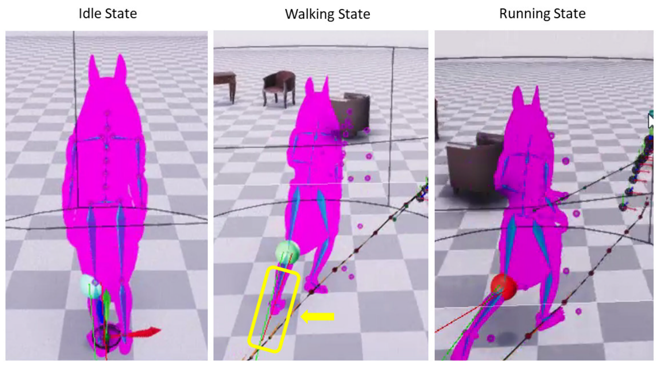

Fig. 3. Visualization of the magnitude of medial and lateral KCFs during idle, walking, and walking with increased speed states (the term "running" is misused here). Green and red lines (indicated with a yellow box) represent the direction of the corresponding force, medial or lateral.

surrogate models have been trained based on widely used musculoskeletal tools like OpenSim and, thus, follow their limitations. Therefore, the benefit of machine learning models is not higher accuracy but reproducibility, robustness, and efficiency. As for domain adaptation, it has been widely used for giving the ability to machine and deep neural networks to learn shared features from multiple data sources. Our network architecture is inspired by works like [10], [13], and [15], which use the first layers of their models as representation learner mechanisms and then feed the extracted features to domain adaptation (CORAL) layers whose output feeds the last layer or layers of their neural networks providing classification or regression predictions.

Specifically, we conducted three sets of experiments in order to evaluate different aspects of our proposed model. In the first experiment, we trained and tested our baseline model (Model A in Fig. 1) using human capture data from two different action classes, gait, and STS, and evaluated our model's predictions using the NRMSE measure. Our results as reported in Table III indicated that our framework performs well in both activity cases and it showed better results compared to previous similar methods. In this experiment, our goal was to assess our BiLSTM model (Model A) in order to utilize it as a "base" in our enhanced with domain adaptation Model B. Nevertheless, more validation is required, especially for the STS activity pattern for which the training dataset was relatively small (266 trials vs ∼4900 trials contained in the gait dataset).

The main aim of this work is to provide a model that can simulate the internal biomechanical forces during more than one tasks. To do so, we enriched our model with an unsupervised domain adaptation technique, the CORAL method, [10] in order to minimize the domain shift between data from different action classes. As a result, our model could eventually be trained offline to recognize a specific class (e.g. gait as source) and then could be applied to fit other classes (e.g. STS, stair ascending/descending, etc. as target) by adapting features during test time. Especially, the developed Model B (depicted in Fig. 1) whose frozen layers are trained using gait data, is able to produce valid predictions for any action when retrained as described in subsection III-B.1. Overall, we benchmark our method using the example of gait and sit-to-stand due to the availability of such data, while this use is indicative and non-restrictive.

Our results in Table IV indicate that our model (Model B in Fig.1) produces more accurate predictions on the target dataset when enriched with the domain adaptation layers, i.e. the CORAL layers than without them (Model A in Fig. 1). It is worth mentioning that both Models A and B were trained on the source (gait) dataset and tested on the target (STS) dataset, thus, the comparison was performed between a model that was enhanced with a transfer learning technique (Model B) and a model (Model A) that was trained with a completely different dataset (gait) than the one that was applied to (STS). However, if we compare Model B's results (Table IV) to the predictions of Model A when trained with the STS dataset (Table III rows 6 and 7), Model A performs better as expected, since Model A is cleanly trained on STS dataset, whereas Model B is fit to the same dataset via a transfer learning method. Overall, using this simple, yet effective unsupervised domain adaptation approach (CORAL), we create a network that can predict KCFs for more than one action class simultaneously (e.g. gait and STS) and since the predictions of two actions can be guaranteed, then our framework offers the ability to produce predictions for intermediate actions that usually require more complex and computationally expensive architectures.

Our third experiment was to integrate our model into a real-time motion synthesis framework, the NSM, in order to enhance it with joint force prediction along with motion prediction and synthesis, and test it in a real-time force estimation scenario. Moreover, since this motion synthesis framework can produce motion trajectory predictions, our model is given the ability to perform long-term forecasting of KCFs required to perform goal-driven actions. Our model runs at approximately 60fps and the knee contact forces of a virtual avatar were visualized at the same rate, while the character was moving and interacting in the virtual scene. It is worth pointing out that since NSM cannot generate joint forces we had no way to quantitatively assess the performance of our model. Furthermore, our model is capable of being easily integrated into any motion synthesis framework, since the implemented domain adaptation is unsupervised and thus applicable when target output data are not available.

We acknowledge that while our framework presents advantages over biomechanical modeling approaches in respect to computational speed, robustness and potential for extrapolation to out-of-sample distributions, it lacks precision in cases with limited samples, such as for patients with disease (e.g. osteoarthritis). Moreover, our modeling framework is flexible to address different clinical scenarios if enough data are available for proper estimation of the domain shift, but we should also note that adaptation is limited to specific impairments, e.g. in lower extremities due to total knee replacement, etc. Overall, we mainly envision the usability of our framework for the creation of models of a healthy state, that can later be adapted to the motions of impaired individuals through healthy-to-impaired state adaptation. We plan to investigate the latter application scenario in our future work.

Furthermore, as future work, we intend to further exploit the benefits of deep domain adaptation, such as in deep CORAL [39], where CORAL is embedded into a deep neural network, not as a layer, but as a differentiable loss function that acts as a nonlinear transformation that aligns the correlations of the source and target domains. Our model could also be applied to other joints of the lower or upper body without altering its architecture. In the case of hip or ankle contact forces estimation, we could re-train our models (both Model A and B) using the same gait dataset [7], i.e. the same joint angles as input, and compute the corresponding hip and ankle contact forces as output by repeating only the JRA process during musculoskeletal modeling in OpenSim for creation of the training data (used as ground truth). Finally, we would also like to try to incorporate physics-based constraints into our framework to generate more accurate results.

VII. Conclusion

In this work, we developed and trained a BiLSTM for predicting knee contact joint forces without using motion-dependent variables (GRFs, EMGs, etc.) but relying only on kinematic data. By integrating a simple, yet effective unsupervised domain adaptation technique, i.e. CORAL, our model is rendered able to predict KCFs for more than one action classes simultaneously and even open the path to estimating internal loads for intermediate actions, a task that usually requires rather complex solutions. Our results indicated that our enhanced model with CORAL (Model B) performs better than without using it in terms of NRMSE and t-test statistical analysis. Moreover, we integrated our framework into a deep auto-regressive algorithm for goal-driven motion synthesis, NSM, to i) test the real-time capability of our network and the ease to be incorporated in any motion synthesis system and ii) augment NSM with a model that simulates internal biomechanical forces. Since there is no robust method to evaluate the real-time predictions of our network in NSM execution, we provide a visualization mechanism to qualitatively assess the validity of our model predictions. Our work offers a simple solution that opens the path for ergonomically-adjusted motion estimation and physiology-aware simulation, which can be used as a tool for rehabilitation planning, as well as the prediction of human motion in rich environments with realistic scenes and character-scene interactions.

Acknowledgment

The authors would like to thank Prof. Sabine Verschueren and Prof. Ilse Jonkers from KUL and all the study participants, for their permission to utilize the acquired gait motion capture data for research purposes, and also would like to thank Dr. Georgios Giarmatzis for providing the musculoskeletal modeling results used for ML model training.

References

[1] M. Sartori, D. G. Llyod, and D. Farina, "Neural data-driven musculoskeletal modeling for personalized neurorehabilitation technologies," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 5, pp. 879–893, May 2016.

[2] A. Esrafilian et al., "An EMG-assisted muscle-force driven finite element analysis pipeline to investigate joint- and tissue-level mechanical responses in functional activities: Towards a rapid assessment toolbox," *IEEE Trans. Biomed. Eng.*, vol. 69, no. 9, pp. 2860–2871, Sep. 2022.

[3] C. Curreli, F. Di Puccio, G. Davico, L. Modenese, and M. Viceconti, "Using musculoskeletal models to estimate in vivo total knee replacement kinematics and loads: Effect of differences between models," *Frontiers Bioeng. Biotechnol.*, vol. 9, Jul. 2021, Art. no. 703508.

[4] S. Starke, H. Zhang, T. Komura, and J. Saito, "Neural state machine for character-scene interactions," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 1–14, Dec. 2019.

[5] L. Rane, Z. Ding, A. H. McGregor, and A. M. J. Bull, "Deep learning for musculoskeletal force prediction," *Ann. Biomed. Eng.*, vol. 47, no. 3, pp. 778–789, Mar. 2019.

[6] G. Giarmatzis, E. I. Zacharaki, and K. Moustakas, "Neural network based prediction of knee contact forces for different gait speeds," in *Proc. IEEE Int. Conf. Bioinf. Biomed. (BIBM)*, Dec. 2020, pp. 2590–2595, doi: 10.1109/BIBM49941.2020.9313190.

[7] G. Giarmatzis, E. I. Zacharaki, and K. Moustakas, "Real-time prediction of joint forces by motion capture and machine learning," *Sensors*, vol. 20, no. 23, p. 6933, Dec. 2020.

[8] B. J. Stetter, S. Ringhof, F. C. Krafft, S. Sell, and T. Stein, "Estimation of knee joint forces in sport movements using wearable sensors and machine learning," *Sensors*, vol. 19, no. 17, p. 3690, Aug. 2019.

[9] W. S. Burton, C. A. Myers, and P. J. Rullkoetter, "Machine learning for rapid estimation of lower extremity muscle and joint loading during activities of daily living," *J. Biomech.*, vol. 123, Jun. 2021, Art. no. 110439.

[10] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," in *Proc. 30th AAAI Conf. Artif. Intell. (AAAI)*, 2016, pp. 2058–2065.

[11] J. Zhang et al., "Physics-informed deep learning for musculoskeletal modelling: Predicting muscle forces and joint kinematics from surface EMG," 2022, *arXiv:2207.01435*.

[12] B. J. Fregly et al., "Grand challenge competition to predict in vivo knee loads," *J. Orthopaedic Res.*, vol. 30, no. 4, pp. 503–513, Apr. 2012.

[13] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," 2015, *arXiv:1502.02791*.

[14] D. Hal, "Frustratingly easy domain adaptation," in *Proc. ACL*, 2007, pp. 256–263.

[15] N. Peng and M. Dredze, "Multi-task domain adaptation for sequence tagging," in *Proc. 2nd Workshop Represent. Learn. (NLP)*, 2017, pp. 91–100.

[16] R. Vega and R. Greiner, "Domain-shift adaptation via linear transformations," 2022, *arXiv:2201.05282*.

[17] T. Bao, S. A. R. Zaidi, S. Xie, P. Yang, and Z.-Q. Zhang, "Inter-subject domain adaptation for CNN-based wrist kinematics estimation using sEMG," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 29, pp. 1068–1078, 2021.

[18] I. Ketykó, F. Kovács, and K. Z. Varga, "Domain adaptation for sEMG-based gesture recognition with recurrent neural networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–7.

[19] Q. Wang and T. P. Breckon, "Unsupervised domain adaptation via structured prediction based selective pseudo-labeling," 2019, *arXiv:1911.07982*.

[20] T. Xiao, C. Fan, P. Liu, and H. Liu, "Simultaneously improve transferability and discriminability for adversarial domain adaptation," *Entropy*, vol. 24, no. 1, p. 44, Dec. 2021.

[21] X. Hong et al., "Dynamic joint domain adaptation network for motor imagery classification," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 29, pp. 556–565, 2021.

[22] J. Fan et al., "Unsupervised domain adaptation by statistics alignment for deep sleep staging networks," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 30, pp. 205–216, 2022.

[23] K.-C. Liu et al., "Domain-adaptive fall detection using deep adversarial training," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 29, pp. 1243–1251, 2021.

[24] X. Zhang, X. Zhang, L. Wu, C. Li, X. Chen, and X. Chen, "Domain adaptation with self-guided adaptive sampling strategy: Feature alignment for cross-user myoelectric pattern recognition," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 30, pp. 1374–1383, 2022.

[25] Y. Zhou et al., "Cross-task cognitive workload recognition based on EEG and domain adaptation," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 30, pp. 50–60, 2022.

[26] F. Chollet, “Keras,” GitHub, 2015. [Online]. Available: https://github.com/fchollet/keras

[27] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[28] I. K. Ihianle, A. O. Nwajana, S. H. Ebenuwa, R. I. Otuka, K. Owa, and M. O. Orisatoki, “A deep learning approach for human activities recognition from multimodal sensing devices,” *IEEE Access*, vol. 8, pp. 179028–179038, 2020.

[29] Z. Cui, R. Ke, Z. Pu, and Y. Wang, “Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction,” 2018, *arXiv:1801.02143*.

[30] M. Ullah, H. Ullah, S. D. Khan, and F. A. Cheikh, “Stacked LSTM network for human activity recognition using smartphone data,” in *Proc. 8th Eur. Workshop Vis. Inf. Process. (EUVIP)*, Oct. 2019, pp. 175–180.

[31] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, *arXiv:1412.6980*.

[32] I. Loi, E. I. Zacharaki, and K. Moustakas, “Integrating ANN-based real-time joint force prediction with deep auto-regressive goal-driven motion synthesis [abstract],” in *27th Congr. Eur. Soc. Biomech.*, Porto, Portugal, Jun. 2022.

[33] A. Seth et al., “OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement,” *PLoS Comput. Biol.*, vol. 14, no. 7, Jul. 2018, Art. no. e1006223.

[34] W. McKinney, “Data structures for statistical computing in Python,” in *Proc. 9th Python Sci. Conf.*, 2010, pp. 56–61.

[35] C. Mandery, Ö. Terlemez, M. Do, N. Vahrenkamp, and T. Asfour, “The KIT whole-body human motion database,” in *Proc. Int. Conf. Adv. Robot. (ICAR)*, Jul. 2015, pp. 329–336.

[36] C. Mandery, Ö. Terlemez, M. Do, N. Vahrenkamp, and T. Asfour, “Unifying representations and large-scale whole-body motion databases for studying human motion,” *IEEE Trans. Robot.*, vol. 32, no. 4, pp. 796–809, Aug. 2016.

[37] F. Krebs, A. Meixner, I. Patzer, and T. Asfour, “The kit bimanual manipulation dataset,” in *Proc. IEEE-RAS 20th Int. Conf. Humanoid Robots (Humanoids)*, Jul. 2021, pp. 499–506.

[38] I. Loi, D. Stanev, and K. Moustakas, “Total knee replacement: Subject-specific modeling, finite element analysis, and evaluation of dynamic activities,” *Frontiers Bioeng. Biotechnol.*, vol. 9, Apr. 2021, Art. no. 648356.

[39] B. Sun and K. Saenko, “Deep CORAL: Correlation alignment for deep domain adaptation,” 2016, *arXiv:1607.01719*.