



Article

# Fusion Models for Generalized Classification of Multi-Axial Human Movement: Validation in Sport Performance

Rajesh Amerineni <sup>1</sup>, Lalit Gupta <sup>1</sup> , Nathan Steadman <sup>2</sup>, Keshwyn Annauth <sup>2</sup>, Charles Burr <sup>2,3</sup>, Samuel Wilson <sup>2,4</sup>, Payam Barnaghi <sup>5,6</sup> and Ravi Vaidyanathan <sup>2,3,4,6,\*</sup> 

<sup>1</sup> Department of Electrical Engineering, Southern Illinois University, Carbondale, IL 62901, USA; rajeshamerineni@siu.edu (R.A.); lgupta@siu.edu (L.G.)

<sup>2</sup> Department of Mechanical Engineering, Imperial College London, London SW7 2AZ, UK; n.steadman17@imperial.ac.uk (N.S.); keshwyn.annauth.20@ucl.ac.uk (K.A.); charles@thecornerapp.com (C.B.); s.wilson@sergtechnologies.com (S.W.)

<sup>3</sup> Athletec Inc., Manchester M17 1QR, UK

<sup>4</sup> Serg Technologies Inc., London SW7 2LQ, UK

<sup>5</sup> Department of Brain Sciences, Imperial College London, London W12 0NN, UK; p.barnaghi@imperial.ac.uk

<sup>6</sup> Dementia Research Institute Care Research and Technology Centre (DRI-CR&T), London W12 0BZ, UK

\* Correspondence: r.vaidyanathan@imperial.ac.uk; Tel.: +44-2075-947-020

**Abstract:** We introduce a set of input models for fusing information from ensembles of wearable sensors supporting human performance and telemedicine. Veracity is demonstrated in action classification related to sport, specifically strikes in boxing and taekwondo. Four input models, formulated to be compatible with a broad range of classifiers, are introduced and two diverse classifiers, dynamic time warping (DTW) and convolutional neural networks (CNNs) are implemented in conjunction with the input models. Seven classification models fusing information at the input-level, output-level, and a combination of both are formulated. Action classification for 18 boxing punches and 24 taekwondo kicks demonstrate our fusion classifiers outperform the best DTW and CNN uni-axial classifiers. Furthermore, although DTW is ostensibly an ideal choice for human movements experiencing non-linear variations, our results demonstrate deep learning fusion classifiers outperform DTW. This is a novel finding given that CNNs are normally designed for multi-dimensional data and do not specifically compensate for non-linear variations within signal classes. The generalized formulation enables subject-specific movement classification in a feature-blind fashion with trivial computational expense for trained CNNs. A commercial boxing system, ‘Corner’, has been produced for real-world mass-market use based on this investigation providing a basis for future telemedicine translation.

**Keywords:** sports biomechanics; human performance; motion tracking; wearable sensors; IMUs; sensor fusion; DTW; CNNs; deep learning



**Citation:** Amerineni, R.; Gupta, L.; Steadman, N.; Annauth, K.; Burr, C.; Wilson, S.; Barnaghi, P.; Vaidyanathan, R. Fusion Models for Generalized Classification of Multi-Axial Human Movement: Validation in Sport Performance. *Sensors* **2021**, *21*, 8409. <https://doi.org/10.3390/s21248409>

Academic Editor: Mark Robinson

Received: 16 October 2021

Accepted: 10 December 2021

Published: 16 December 2021

**Publisher’s Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Mechatronic systems recognizing human activity are now fundamental components in biophysical analysis, with strong impact in fields such as physiotherapy, telemedicine, smart homes, rehabilitation, human-robot interface, and athletics (e.g., [1–8]). A wide range of activity-aware systems including smart phone apps (e.g., Galaxy Moves App, iPhone Moves App, iPhone Health Mate App, iPhone Fitbit App), athletic wearables (e.g., Nike Fuelband, Jawbone UP24, Fitbit Flex, Fitbit One, Fitbit Zip, Digi-Walker SW-200) and fall detection devices (e.g., Philips Lifeline, Lively Mobile, Sense4Care, Angel4) are commercially available today. Despite this range, most wearables remain limited to simple metrics such as step count, heart rate, and calories expended [9]. Though initial sales are promising, a staggering 1/3 of users abandon wearable devices [10], speaking to obvious challenges in transience and sustainability.

There is a significant need for systems that go beyond base movement metrics. Specific action classification and performance feedback on extremity movement in real-time [4,6,9,10] is in very high demand. In controlled or prepared environments, such information can be obtained with optical/camera systems, (e.g., Vicon™-Vicon, Denver, CO, USA) which have the benefit of high accuracy, but are also costly, non-portable, vulnerable to camera occlusion, and are challenging in the field. A smaller body of work has addressed these limitations with wearable vision systems (surveyed in [8]), however a robust compact system capable of tracking limb movements in the field has not been realized for large sets of classification problems. Generalizability to new or individual classes of movements without manual or bespoke feature extraction is critical for transition between types of action classification and personalized performance assessment.

Smaller systems such as inertial measurement units (IMUs) surmount issues of portability, occlusion, and price, yet demand signal fusion to provide information on movement beyond simple metrics such as step count. Recent innovation has surmounted this gap with learning algorithms fusing inertial data in specific applications. Examples include: treatment of neural dysfunction such as stroke [1] or Parkinson's Disease [11], motion recognition in smart homes [5], athletic training parameterized for specific sports [3], and artificial limb/robotic control [7]. Despite these advances, translation for widespread use demands less reliance on specific features of movement in one arena. Complex recognition problems and individual variance in movement classes must be addressed [5]. Activity recognition with no reliance on 'hand engineered' feature identification is challenging due large variability in motor movements for a given action. This necessitates broader classes of learning [12] and the capacity to fuse ensembles of individualized heterogeneous data [3,5,6,9,13]. Sensors, embedded systems, and cloud connectivity have evolved the field from a 'device' to a 'systems' perspective [9]; algorithms, hardware and IoT are fundamentally coupled, and must be treated as an integrated whole. Finally, real-world use demands algorithms be computationally efficient enough for real-time use, ideally as embedded systems on low-power edge devices (e.g., wearables) that may function through communication gaps in the field [14].

## 2. Materials and Methods

### 2.1. Scope of Work

In this investigation, we introduce a set of models to fuse information from wearable sensors to learn broad classes of human movement without dependency of any assessed features of that movement. The models are formulated generically, then validated in two sport applications. We have selected dynamic time warping (DTW) and convolution neural network (CNN) classifiers for the development of movement classification systems. The primary reason for selecting these two methods is that the multi-axial signals can be fed directly into our fusion classifiers without having to extract "hand-engineered" features. Consequently, DTW and CNN classifiers do not suffer the main drawback of classifiers that use hand-engineered features whose performance is highly dependent on the choice of the extracted features. Moreover, the selection of a set of features for a given problem is more of an art than science.

It is well known that human movements experience non-linear trial-to-trial variations which typically include expansions and/or compressions in signal segments and latency shifts in the peaks. DTW classifiers are a clear choice because they are specifically designed to handle such non-linear variations in one-dimensional signals through non-linear alignment [15–29]. Furthermore, the DTW classifiers can also serve as benchmark for comparing performance between classifiers. CNNs are not an obvious choice because they are primarily designed to classify two-dimensional and multidimensional data such as images in computer vision. Recent studies, however, have shown that CNNs can also be used to classify multivariate time series [30–33] and human-activity activity recognition problems using multi-modal sensors [34–39] through the generation of images [33,38] or by combining uni-axial signals into matrices [36,37,39]. However, the tolerance of CNNs

to non-linear signal variations and the exploitation of coupling between uni-axial signals have not been specifically addressed in the detailed manner as described in this study.

## 2.2. Multisensor Fusion Validation Application: Combat Sport

We have chosen to test and validate our fusion classifier models in combat sport given the diversity of arm and leg movements, the fact that movements are representative of those necessitating multi-axial recognition, and the capacity to collect and test large sets of meaningful data. The use of IMUs in combat sports has grown in recent years (reviewed in [6]), though existing approaches tend to be focused on metrics or specific signal features. In general, classification systems that exploit information from ensembles of multi-axial sensors are capable of improving, quite significantly, the performance over uni-axial classifiers [40–45]. However, multi-sensor classifiers tend to be more complex because of the need to incorporate fusion methods to combine “information” from the multiple sensors. The fusion methods can be divided into “input-level fusion” and “output-level” fusion. For input-level fusion, also called early fusion, the information can be input data or features extracted from the data. The information in output-level fusion, also called late fusion, is typically the decisions of the uni-axial classifiers or some measure at the outputs of the uni-axial classifiers.

The fusion models developed in this study for classifying combat sport movement are formulated generically and then validated by classifying 24 classes of kicking movements in taekwondo and 18 classes of punching movements in boxing. To our knowledge this is the first set of generalized non-feature specific models demonstrated on such a large number of classes in either activity [6].

## 2.3. Investigation Goals

Our first goal is to introduce data input models which: (a) facilitate fusion of information at the input and output levels and (b) are generalizable for use in conjunction with a broad range of diverse classifiers. The second goal is to design DTW and CNN classifiers for human movement identification using these input models. The third goal is to design experiments to classify boxing and taekwondo strikes. The final goal is to compare the DTW and CNN-based classification systems with respect to accuracy, complexity, flexibility, and the potential to obtain further improvements in performance. We offer these findings as a basis for translation of wearables for a range of human performance and healthcare applications.

## 2.4. Organisation of Paper

Section 3 describes the four movement classifier input models. Sections 4 and 5 describe the DTW and CNN models that are used in conjunction with the input models. Section 6 describes data collection and the strike movements for the validation studies in combat sports. Section 7 outlines classification results for 24-class kicking and 18-class punching movements. Section 8 briefly describes translation to a commercial product as evidence of novelty and impact while Section 9 summarizes conclusions from the investigation.

## 3. Classifier Input Models

We propose four input models, which differ in the way the multi-axial sensor signals are presented as inputs into the subsequent classification stages. The four classifier input arrangements are summarized in Figures 1–4. The models can be contrasted by noting the level of fusion incorporated in the models. In the formulations of the classification models, a movement is represented by  $I$  and it is assumed that the movement belongs to one of  $H$  movement classes,  $\omega_h$ ,  $h = 1, 2, \dots, H$ . The models are assumed to have  $G$  multi-axial sensors represented by  $S_g$ ,  $g = 1, 2, \dots, G$ , and an output of sensor  $S_g$  is represented by  $S_{gm}$ ,  $m = 1, 2, \dots, m_g$ , where,  $m_g$  is the number of multi-axial outputs.

The term “non-linear variations” will be used to encompass latency shifts (shifts in peak positions) and expansions/compressions in signal segments.

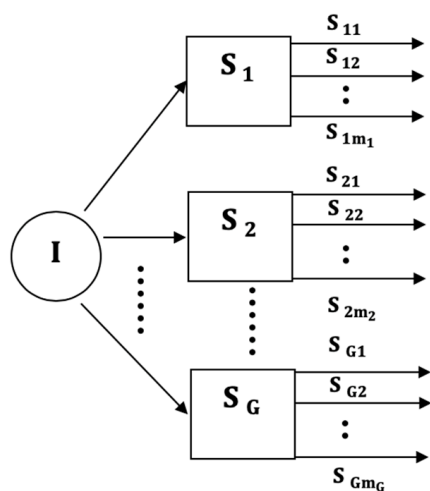


Figure 1. The Vector Input (VI) model.

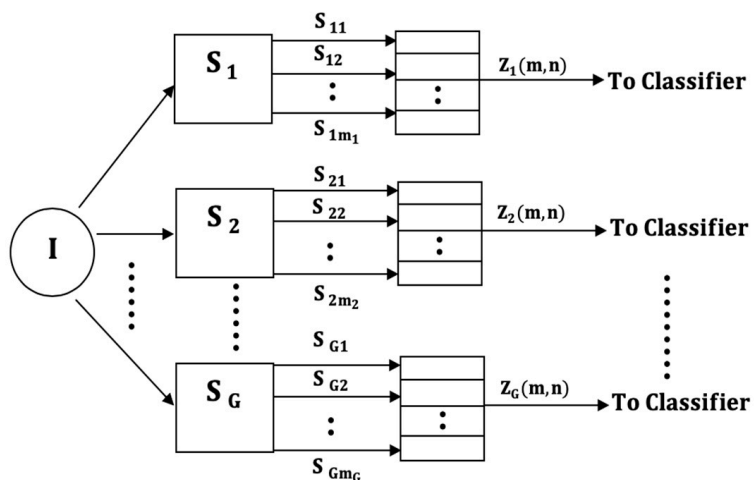


Figure 2. The Local Matrix Input (LMI) model.

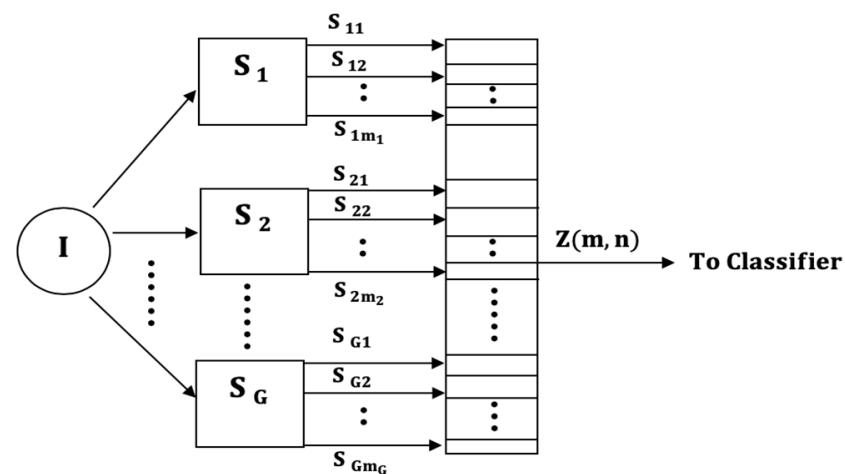
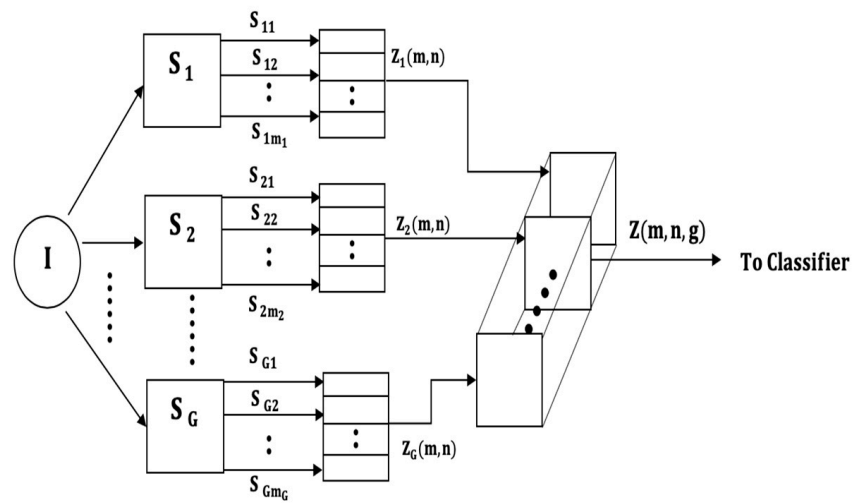


Figure 3. The Global Matrix Input (GMI) model.



**Figure 4.** The Global Cuboid Input (GCI) model.

### 3.1. Vector Input (VI) Model

The VI model, shown in Figure 1, is straightforward because it does not involve any form of input-level fusion. The figure simply shows the labeling of the sensors and the sensor outputs. This model is suitable for systems that classify each uniaxial signal  $s_{gm}$  independently. The number of independent classifiers in such a system, therefore, is  $M_G = \sum_{g=1}^G m_g$ . Systems using this input model need fusion at the output level to determine the class of the movement signal. Of the four input models, the VI model is the most versatile because the sensors can be heterogeneous and can have a different number of axes. Furthermore, the sensor outputs can have different durations and do not have to be synchronized with respect to non-linear variations. However, the resulting classifiers are the most complex because they require a classifier for each multi-axial signal and output-level fusion to combine the information from the  $M_G$  classifier outputs in order to determine the input class.

### 3.2. Local Matrix Input (LMI) Model

The LMI model is designed for systems that classify the uniaxial outputs of each sensor separately by fusing the signals of each sensor into a matrix as shown in Figure 2. That is, the outputs of each multi-axial sensor  $S_g$  are fused into a local intra-sensor matrix

$$Z_g(m, n), m = 1, 2, \dots, m_g; n = 1, 2, \dots, n_g \quad (1)$$

where,  $n_g$  is the duration of the outputs of sensor  $S_g$  (assumed equal in each sensor). The number of matrices is equal to the number of sensors  $G$ . The intra-sensor matrix to classify the signals of sensor  $S_g$  can be written as

$$Z_g(m, n) = \nabla_{m=1}^{m_g} s_{gm}, g = 1, 2, \dots, G \quad (2)$$

where, the fusion operation is represented by  $\nabla$ . Each matrix can be classified independently, and some form of output-level fusion can be applied to determine the class of the movement signal. The resulting classification system, therefore, is a hybrid system which includes both input and output-level fusion. This LMI input model is more restrictive than the previous model because the multi-axial sensor outputs must have the same durations within each sensor (not across all sensors) in order to fuse them into a matrix. Moreover, the multi-axial sensor outputs are assumed to experience synchronized non-linear variations within each sensor. The advantage of the LMI model is that the number of classifiers is reduced to  $G$  when compared with the  $M_G$  classifiers needed in the previous VI model.

### 3.3. Global Matrix Input (GMI) Input Model

The third model, involving only input-level fusion in the classifiers, is designed to classify the uniaxial sensor signals of all sensors by fusing the signals into a global inter-sensor matrix shown in Figure 3. The inter-sensor matrix is formed by fusing all multi-axial outputs into a matrix  $Z(m, n)$ ,  $m = 1, 2, \dots, M_G$ ;  $n = 1, 2, \dots, N$ , where,  $N$  is the duration of each sensor output (assumed equal). That is, each row of  $Z(m, n)$  is an output of a multi-axial sensor. The global input matrix is, therefore, given by

$$Z(m, n) = \nabla_{g=1}^G \nabla_{m=1}^{m_g} s_{gm} \quad (3)$$

This fusion operation is equivalent to fusing the LMI matrices into a matrix, therefore, the global matrix can also be written as

$$Z(m, n) = \nabla_{g=1}^G Z_g(m, n) \quad (4)$$

Unlike the two previous models, classifiers using this input model do not require output-level fusion because only a single classifier is needed to classify the global matrix. However, it is important to note that the resulting classifier is more restrictive than the two previous models because the following assumptions are made:

1. The multi-axial outputs have the same durations within and across all sensors in order to fuse them into a global matrix.
2. The multi-axial sensor outputs experience synchronized non-linear variations within and across all sensors.

### 3.4. Global Cuboid Input (GMI) Input Model

If the number of uniaxial outputs and the output durations of all  $G$  sensors are assumed equal, the LMI matrices can also be fused in a cuboid which can be represented by

$$Z(m, n, g) = \Delta_{g=1}^G Z_g(m, n), \quad m = 1, 2, \dots, M; \quad n = 1, 2, \dots, N; \quad g = 1, 2, \dots, G$$

where  $\Delta$  is the cuboid fusion operation,  $M$  is the number of uniaxial outputs of each sensor and  $N$  is the duration of each uniaxial signal. This input model, shown in Figure 4, is the most restrictive because it requires an additional condition to be met, viz., the sensors must have an equal number of uniaxial outputs.

In summary, there is a trade-off between classifier complexity and flexibility using the four input models. A suitable way to overcome the equal duration restriction is to duration normalize the signals to have a common length through linear expansion or compression. For example, the common length can be chosen to be the average duration of all sensor outputs for the GMI model and the average of outputs of each sensor for the LMI model. However, there is no simple way to overcome the non-linearity restriction and the performance of the GMI and LMI models can be expected to drop if this assumption is violated. The GCI model cannot be implemented if the number of outputs across all sensors is not equal. Another issue to take into account is the similarity of the sensors. If the sensors are heterogeneous, the heterogeneous signal amplitudes have to be normalized, for example, using min-max normalization. Even if the sensors are homogeneous, normalization within each sensor is also required to account for the varying ranges of the uni-axial signal amplitudes.

The sections that follow present the formulations of three DTW based models and the four CNN based models to classify multi-axial multiple-sensor movement signals using the four input models. It can be shown that for the DTW implementations, the GCI model is equivalent to GMI model, therefore, the GCI model is not implemented. A DTW classifier is explicitly split into two operations: discrepancy computation and a decision rule. The discrepancy computation operation determines the dissimilarity score between the aligned test and reference signals of movements and the decision rule uses these discrepancy scores to assign the test movement into one of the movement classes.



#### 4. Dynamic Time Warping (DTW) Classifier

DTW has been applied in numerous applications to measure the dissimilarity between pairs of sequences that experience non-linear variations in the segments of the sequences. A sample of applications employing DTW include speech recognition [19,23], shape recognition [15,16,25], clustering [17,24], gene expression [26], financial time series matching [29], and classifying human actions in sports [27]. In order to facilitate the understanding of the formulations of the three DTW-based fusion models, a brief description of one, two, and three-dimensional DTW algorithms follows next.

##### 4.1. Dynamic Time Warping (DTW) Algorithms

Given a pair of signals  $X$  and  $Y$  and a local cost function  $w(k)$  to reflect the discrepancy between the elements of  $X$  and  $Y$ , the goal of DTW is to determine an alignment function  $W = \{w(1), w(2), \dots, w(K)\}$ , such that the overall normalized cost

$$A_{XY} = (1/K) \sum_{k=0}^{K-1} d[w(k)] \quad (5)$$

is minimized subject to a set of end-point, monotonicity, and continuity constraints.  $A_{XY}$  is a measure of the discrepancy between signals  $X$  and  $Y$  after optimal alignment. The most often used cost functions include the Euclidean, Manhattan, and Euclidean-squared distance metrics. Dynamic programming is used to solve the optimization problem.

The steps to align one, two, and three-dimensional signals are quite similar except for the computation of the local cost function. For example, if the Euclidean distance is used, the cost function for the one-dimensional (vector) DTW algorithm is

$$d[w(k)] = \|X(i(k)) - Y(j(k))\|. \quad (6)$$

For the two-dimensional (matrix) case, the cost function is given by

$$d[w(k)] = \|X(:,i(k)) - Y(:,j(k))\| \quad (7)$$

where, the notation  $Z(:,t)$  is used to denote column  $t$  of a matrix  $Z$ . Note that the number of rows in the two matrices must be equal but the number of columns can be different. Similarly, the cost function for the three-dimensional (cuboid) extension is given by

$$d[w(k)] = \|X(:,i(k),:) - Y(:,j(k),:)\| \quad (8)$$

where, the notation  $Z(:,t,:)$  is used to denote a depth-frame  $t$  of a cuboid  $Z$ . For this case, the number of rows and depth of the two cuboids must be equal but the number of columns can be different. Also note that cuboid alignment can also be implemented as matrix alignment by fusing the height-width frames into an augmented matrix because the resulting column-to-column cost function is equal to the frame-to-frame cost function. However, the matrix alignment cannot be implemented as cubic alignment if the number of rows in the frames are unequal. In this study which involves the classification of sensor signals arranged as vectors, matrices, and cuboids, the corresponding DTW classifiers will be referred to as V-DTW, M-DTW, and C-DTW, respectively.

In order to design a DTW classifier for a given problem, a reference template for each pattern is typically estimated from the signals in their respective training sets. The sample mean vector is used often because it best represents the signals in the training set in the sense of minimizing the sum of squared distances from itself to the signals in the training set. However, this does not necessarily imply that the sample mean is the best template choice for a particular problem. Modified averaging procedures which take non-linear variations into account have been proposed to generate templates that can be used in DTW algorithms [18]. In fact, other measures of central tendency (C-T) such as the median, Winsorized mean, trimmed mean, and tri-mean can also be used. What is important to note is that a better C-T estimate does not necessarily result in a better template for classification problems. Therefore, attempting to predict which C-T estimate will yield the best template

for a given problem is not easy and the selection of a template is usually determined through trial-and-error.

#### 4.2. DTW Implementation of VI Model (DTW-1)

The DTW based classification model which uses the VI model is illustrated in Figure 5. The model, referred to as DTW-1, consists of one independent V-DTW classifier for each multi-axial sensor output. Therefore, the number of V-DTW classifiers is  $M_G$ . The discrepancy scores of the  $M_G$  classifiers are fused through averaging in order to determine the class of the impact signal.

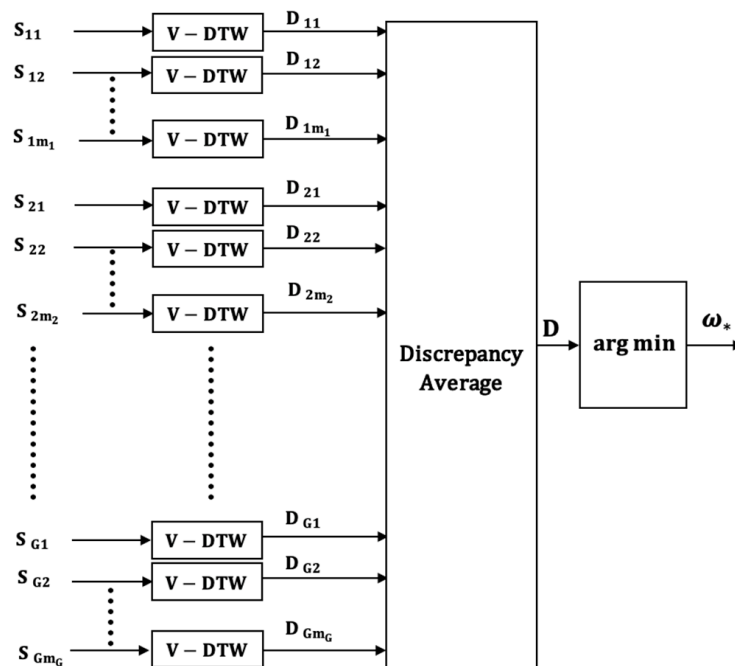


Figure 5. The DTW-1 classification model.

*Discrepancy computation:* The output of the V-DTW operator for the uni-axial signal  $s_{gm}$  is the discrepancy score vector  $D_{gm} = (d_{gm}^{\omega_1}, d_{gm}^{\omega_2}, \dots, d_{gm}^{\omega_H})$ , where,  $d_{gm}^{\omega_h}$  is the discrepancy between a test sequence  $s_{gm}^T$  and the reference sequence  $s_{gm}^h$ .

*Output Fusion Rule:* The discrepancy scores of the  $M_G$  V-DTW operators are averaged and the resulting averaged discrepancy fusion vector is given by

$$D = (D^{\omega_1}, D^{\omega_2}, \dots, D^{\omega_H})$$

where:

$$D^{\omega_h} = \left( \frac{1}{M_G} \right) \left[ \sum_{m=1}^{m_1} d_{1m}^{\omega_h} + \sum_{m=1}^{m_2} d_{2m}^{\omega_h} + \dots + \sum_{m=1}^{m_G} d_{Gm}^{\omega_h} \right]. \quad (9)$$

*Decision Rule:* The test movement  $I_T$  is assigned to the class that yields the least discrepancy using the following rule:

$$\omega_* = \arg \min [ D^{\omega_h} ], \quad h = 1, 2, \dots, H. \quad (10)$$

#### 4.3. DTW Implementation of the LMI Model (DTW-2)

The use of M-DWT in conjunction with the LMI model is illustrated in Figure 6. In this hybrid input and output-level fusion approach, each intra-sensor matrix is classified independently using M-DWT and the class of the movement is determined by averaging the discrepancy scores of each classifier.



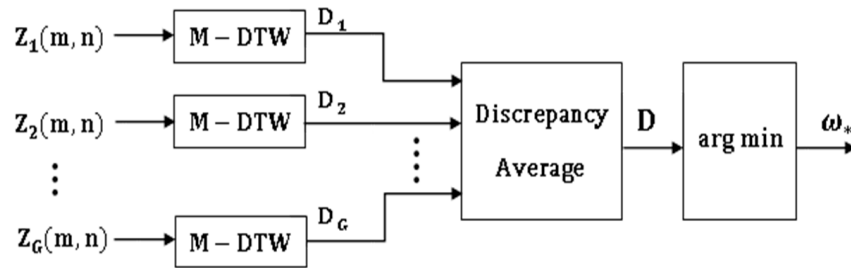


Figure 6. The DTW-2 classification model.

*Local Discrepancy Computation:* The system has one M-DTW classifier for the outputs of each multi-axial sensor. For the M-DTW classifier for sensor  $S_g$ , let  $Z_{g,T}(m, n)$  and  $Z_{g,h}(m, n)$  be the local input matrix of a test movement and a reference movement of class  $h$ , respectively, and let the output of the M-DTW operator be the discrepancy vector  $D_g = (d_g^{\omega_1}, d_g^{\omega_2}, \dots, d_g^{\omega_H})$ . The element  $d_g^{\omega_h}$  is the discrepancy score between  $Z_{g,T}(m, n)$  and  $Z_{g,h}(m, n)$ .

*Output Fusion Rule:* For this case, the outputs (discrepancy scores) of the  $G$  DTW operators are fused using an averaging operation. The averaged discrepancy fusion vector is given by

$$D = (D^{\omega_1}, D^{\omega_2}, \dots, D^{\omega_H}) \quad (11)$$

where,

$$D^{\omega_h} = \left( \frac{1}{G} \right) \sum_{g=1}^G d_g^{\omega_h} \quad (12)$$

*Decision Rule:* The test movement  $I_T$  is assigned to the class  $\omega_*$  using the rule in Equation (10).

#### 4.4. DTW Implementation of the GMI Model (DTW-3)

The DTW classifier that uses the GMI model is illustrated in Figure 7. In this input-level fusion approach, the system has one M-DTW classifier to classify the global inter-sensor matrix. The discrepancy scores between a test movement and reference movements are computed and the test movement is assigned to the class which yields the smallest score.

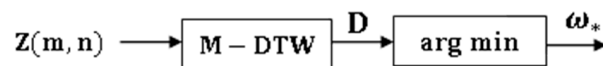


Figure 7. The DTW-3 classification model.

*Discrepancy Computation:* If the global input matrices of a test movement  $I_T$  and reference template of movement  $I_h$  are represented by  $Z_T(m, n)$  and  $Z_h(m, n)$ , respectively, the output of the M-DTW operator is the discrepancy vector  $D = (D^{\omega_1}, D^{\omega_2}, \dots, D^{\omega_H})$  in which element  $D^{\omega_h}$  is the discrepancy score between  $Z_T(m, n)$  and  $Z_h(m, n)$ .

*Output Fusion Rule:* none required.

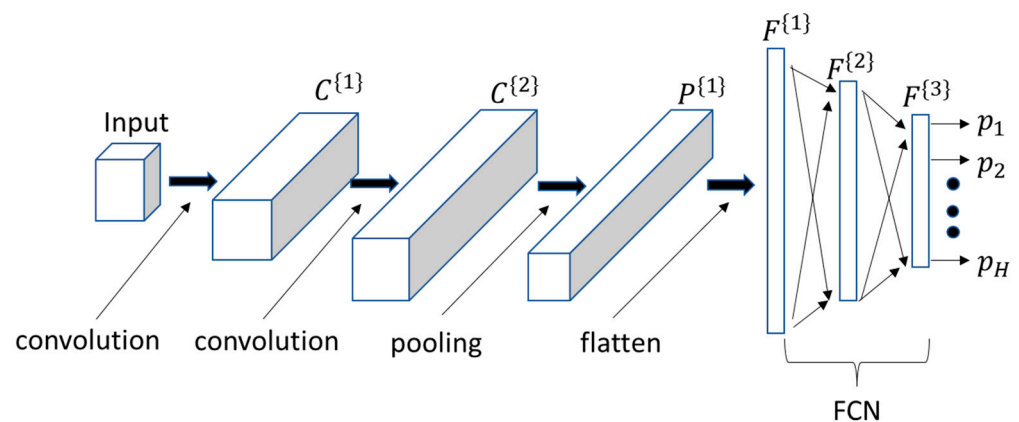
*Decision Rule:* The test movement  $I_T$  is assigned to the class  $\omega_*$  the using the rule in Equation (10).

## 5. Convolution Neural Network (CNN) Classifiers

CNNs are a class of deep learning networks that is capable of performing well in computer vision problems such as large-scale object classification and detection in images [46–51]. One of the most striking features of CNNs when compared with other traditional classifiers, including fully connected neural networks (FCNs), is that a minimal amount of preprocessing is required to generate the input to the network. For example, an image can be processed directly without having to convert it into a vector. Converting

images to vectors results in a very long input vector which can lead to the curse of dimensionality in traditional classifiers and a large network for FCNs which in turn results in a large number of network parameters and overfitting problems. Though seldom discussed, converting an image into a vector leads to a poor representation of the input image because it loses the relationship between a pixel and its vertical and diagonal neighbors which is important for local feature detection. The most often used methods to overcome the dimensionality-related problems is through feature extraction. However, as noted in the introduction, selecting a set of features for a given problem is more an art than science and features are typically selected through trial-and-error. CNNs overcome these problems by applying feature extracting filters directly to the image and most importantly, learning the filter weights through training rather than using prior knowledge to hand-engineer the weights. Moreover, the overfitting problem is reduced through parameter sharing in which the same filter is used to determine each element in the feature map.

A typical CNN has an input layer, an output layer, and hidden layers consisting of convolution, pooling, and fully connected layers. The network architecture is defined by the number and arrangement of the convolution and pooling layers. Figure 8 is an illustration of a CNN with two convolution layers  $C^{\{1\}}$  and  $C^{\{2\}}$  followed by a pooling layer  $P^{\{1\}}$  and a FCN with layers  $F^{\{1\}}$ ,  $F^{\{2\}}$ , and  $F^{\{3\}}$  (output layer). The input to the first fully connected layer is the flattened (concatenated) output from the pooling layer. In general, the dimension of a convolution layer depends on the number of convolution filters, the filter stride, and the type of convolution (valid or same). A pooling layer dimension depends on the size and stride of the pooling filters. For classification problems, the output layer is typically a softmax layer with one output for each pattern class. The network is trained using the gradient descent backpropagation algorithm.



**Figure 8.** Block diagram of a CNN with two convolution layers and a pooling layer.

The two notable operations performed in CNNs are convolution and pooling. Each convolution layer contains a set of filters which have spatial dimensions much smaller than those of the image, however, the depth (number of channels) is usually the same as the input. A bias is added to the filtered outputs which are then passed through a non-linear activation such as the *ReLU* function to yield the feature maps. The feature maps are stacked into cuboids to form the output of the convolution layer in which the number of channels is equal to the number of filters. If the convolution layer is followed by a pooling layer, the spatial dimension is reduced by subsampling blocks in each feature map in the convolution layer output. Max pooling, which replaces a block with the maximum value, is the most often used pooling operation. Pooling serves two purposes: it progressively reduces the spatial dimension thus decreasing the overfitting problem through the reduction in the number of parameters and selects the most robust features.

The actual operation that is performed in the convolution layer is correlation and not convolution. The term “convolution”, therefore, is incorrectly used. However, if the input or the filter is folded (1-d case) or rotated (2-d and 3-d cases), the correlation and

convolution operations are equivalent. Therefore, it is assumed that one of the inputs has been pre-folded or pre-rotated prior to the actual correlation operation performed in the convolution layer.

The following sections describe four implementations of CNNs that use the vector, matrix, and cuboid input models. The models can be distinguished by the convolution operations in the first stage and the output-level fusion operation. In order to do so, the input and output of the first convolution layer are assumed to be generalized cuboids with dimensions  $(H^{[0]} \times W^{[0]} \times D^{[0]})$  and  $(H^{[1]} \times W^{[1]} \times D^{[1]})$ , respectively. Using this notation, a  $d$ -dimensional vector and  $(m \times n)$  matrix are represented as  $(1 \times d \times 1)$  and  $(m \times n \times 1)$  generalized cuboids, respectively. If a pooling layer follows, the output of the pooling layer is assumed to have dimensions  $(H^{[1,p]} \times W^{[1,p]} \times D^{[1,p]})$ . The filters in the first convolution layer have dimensions represented by  $(f_h^{[1]} \times f_w^{[1]} \times D^{[0]})$  and the pooling filter by  $(f_h^{[1,p]} \times f_w^{[1,p]} \times 1)$ . The dimensions are related as follows:

$$H^{[1]} = [1 + (H^{[0]} - f_h^{[1]} + 2p) / s_c] \quad (13)$$

$$W^{[1]} = [1 + (W^{[0]} - f_w^{[1]} + 2p) / s_c] \quad (14)$$

$$D^{[1]} = K^{[1]} \quad (15)$$

$$H^{[1,p]} = [1 + (H^{[1]} - f_h^{[1,p]}) / s_p] \quad (16)$$

$$W^{[1,p]} = [1 + (W^{[1]} - f_w^{[1,p]}) / s_p] \quad (17)$$

$$D^{[1,p]} = D^{[1]} \quad (18)$$

where,  $p$ ,  $s_c$ ,  $s_p$ , and  $K^{[1]}$  represent the zero-padding amount, convolution stride, pooling stride, and the number of filters in the first stage, respectively. Zero-padding is employed in "same convolution" to keep the input and output dimensions equal. If  $p = 0$ , the output of the "valid convolution" operation has smaller dimensions than those of the input.

Just as in the development of the DTW classifiers in the previous section, the CNN classifiers are explicitly split into two operations: computation of the posterior class probabilities and a decision rule. The posterior class probabilities are computed by the CNN and the decision rule uses these probabilities to assign the test movement into one of the movement classes. Although a pooling layer may or may not follow a convolution layer, it will be assumed that a convolution layer is followed by a pooling layer for consistency in the formulations. It will also be assumed that the convolutions are "same." The output dimensions can be easily adjusted if the convolutions are "valid."

### 5.1. CNN Implementation of the VI Model (CNN-1)

The CNN-1 classification model which uses the VI model is illustrated in Figure 9. The CNN-1 model is characterized by vector convolutions in the first layer to extract local intra-axial features and output-level fusion for combining the  $M_g$  classifier outputs.

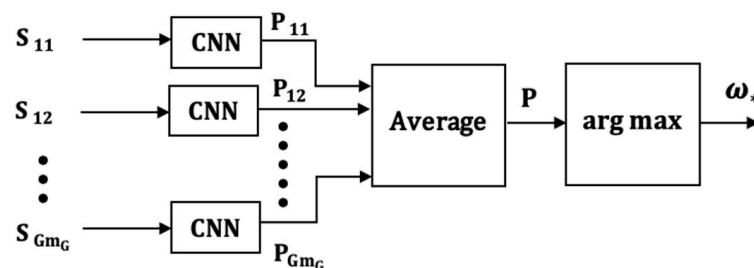


Figure 9. The CNN-1 classification model.

Because the uni-axial classifiers are identical, the CNN classifier for one uniaxial signal  $s_{gm}$  is first described and the method for combining the outputs of the  $M_g$  classifiers is described next.

*Posterior Probability Computation:* In the first convolution layer, the input vector  $s_{gm}$  with generalized cuboid dimensions  $(1 \times n_{gm} \times 1)$  is convolved with  $K^{[1]}$  filters, each with dimensions  $(1 \times f_w^{[1]} \times 1)$ . Because the convolution is assumed “same”, the output  $\hat{s}_{gm}^{[1,k]}$  of the  $k$ th filter will have the same dimensions as the input  $s_{gm}$ . A bias  $b_{gm}^{[1,k]}$  is added to the filtered output and passed through the nonlinear *ReLU* activation function so that the activation of filter  $k$  in the first layer is given by

$$\hat{s}_{gm}^{[1,k]}(n) = \text{ReLU}[\hat{s}_{gm}^{[1,k]}(n) + b_{gm}^{[1,k]}] \quad (19)$$

where,  $\text{ReLU}[\delta] = \text{Max}[0, \delta]$ . The output of the first convolution layer is the  $K^{[1]}$  activations combined into  $(1 \times n_{gm} \times K^{[1]})$  unit height cuboid represented by  $S_{gm}^{[1]}$ . If pooling follows and the stride and size of the pooling filter are  $r$  and  $(1 \times \gamma \times 1)$ , respectively, the output  $S_{gm}^{[1,p]}$  of the pooling layer will have dimension  $(1 \times [(n_{gm} - \gamma)/r] + 1) \times K^{[1]}$ .

In the second convolution layer, if each filter has dimension  $(1 \times f_w^{[2]} \times K^{[1]})$ , the output  $\hat{s}_{gm}^{[2,k]}$  of the  $k$ th filter will have dimension  $(1 \times [(n_{gm} - \gamma)/r] + 1) \times 1$ . Note that although the two functions convolved are unit height cuboids, the output is a vector. After adding a bias and passing each filtered output through the *ReLU* activation function, the  $K^{[2]}$  activations are combined into a unit height cuboid. The width of the unit height cuboid is adjusted according to the stride if a pooling layer is added. If necessary, the convolution and pooling operations can be repeated. A flattening operation is employed to combine the rows of the last cuboid into a vector which forms the input to a fully connected feed forward neural network with  $N_{gm}$  layers. Typically, the sigmoidal or tanh functions are used as activations in the intermediate hidden layers and the softmax activation is used in the output layer of the fully connected network (FCN). Cross-entropy is employed for the loss-function. Because of the softmax activation function, the outputs can be regarded as estimates of posterior probabilities given by

$$p_{gm}(h) = \frac{e^{q_h}}{\sum_{h=1}^H e^{q_h}}, \quad h = 1, 2, \dots, H \quad (20)$$

where,  $q_h$  is the weighted sum of the inputs into a neuron  $h$  in the output layer.

The output of the CNN classifier for signal  $s_{gm}$  is represented by the vector

$$P_{gm} = (p_{gm}(1), p_{gm}(2), \dots, p_{gm}(H)); \quad g = 1, 2, \dots, G, \quad m = 1, 2, \dots, m_g \quad (21)$$

*Decision Rule:* The  $H$  probabilities of the  $M_G$  CNN classifiers are averaged into a probability fusion vector represented by

$$P = (P^{\omega_1}, P^{\omega_2}, \dots, P^{\omega_H}) \quad (22)$$

where,

$$P^{\omega_h} = (1/M_G) \left[ \sum_{m=1}^{m_1} p_{1m}(h) + \sum_{m=1}^{m_2} p_{2m}(h) + \dots + \sum_{m=1}^{m_G} p_{Gm}(h) \right]. \quad (23)$$

Using the maximum response rule, the CNN assigns the input movement to the class associated with the output that yields the largest value. That is, a test movement is assigned to class  $\omega_h$  if

$$P^{\omega_h} > P^{\omega_j}, \quad \text{for all } j \neq h \quad (24)$$

Equivalently, the test movement is assigned to the class given by

$$\omega_* = \arg \max [P^{\omega_h}], \quad h = 1, 2, \dots, H. \quad (25)$$

The CNN-1 model shares similarities with the Channel-Based Late Fusion models (CB-LF) described in [35,39] in the sense that there is one CNN per axis. The main difference is that the CB-LF model has one FCN and the input to the FCN is the concatenation of the features from the last convolution layer of each axis. The late fusion, therefore, is a form of inter-channel feature fusion. The CNN-1 model has one FCN for each axis and the late fusion is a form of decision fusion that occurs at the outputs of the CNNs.

### 5.2. CNN Implementation of the LAI Model (CNN-2)

The CNN-2 classification model which uses the LMI model is illustrated in Figure 10. It is characterized by one CNN classifier for each sensor, matrix convolutions in the input layer to extract local intra-sensor features, and output-level fusion for combining the outputs of the  $G$  CNN classifiers.

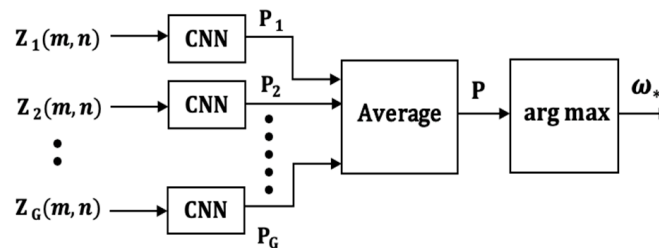


Figure 10. The CNN-2 classification model.

*Posterior Probability Computation:* In the first layer, the sensor matrix  $Z_g(m, n)$  with generalized dimensions  $(m_g \times n_g \times 1)$  is convolved with with  $K^{[1]}$  filters, each with dimensions  $(f_h^{[1]} \times f_w^{[1]} \times 1)$ . The output  $\hat{Z}_g^{[1,k]}(m, n)$  of the  $k$ th filter is a matrix with the same dimensions as the input. A bias  $b_g^{[1,k]}$  is added to the filtered output and passed through the nonlinear *ReLU* activation function. The activation of the filter, therefore, is given by

$$\tilde{Z}_g^{[1,k]}(m, n) = \text{ReLU}[\hat{Z}_g^{[1,k]}(m, n) + b_g^{[1,k]}]. \quad (26)$$

The  $K^{[1]}$  filtered outputs are combined into a  $(m_g \times n_g \times K^{[1]})$  cuboid  $Z_g^{[1]}(m, n, k)$ . If pooling follows and the stride and size of the pooling filter are  $r$  and  $(\gamma \times \gamma \times 1)$ , respectively, the output is the cuboid

$$Z_g^{[1,p]}(m, n, k), \quad m = 1, 2, \dots, m_g^{[1,p]}, \quad n = 1, 2, \dots, n_g^{[1,p]}, \quad k = 1, 2, \dots, K^{[1]}$$

where,  $m_g^{[1,p]} = (((m_g - \gamma)/r) + 1)$ , and  $n_g^{[1,p]} = (((n_g - \gamma)/r) + 1)$ .

In the next convolution stage, the cuboid is convolved with a cuboid filters with dimensions  $(f_h^{[2]} \times f_w^{[2]} \times K^{[1]})$ . Each filtered output  $\hat{Z}_g^{[2,k]}(m, n)$  resulting from the cuboid convolution is a matrix. The series of convolutions and pooling operations terminate into an FCN with a softmax output layer.

If  $p_g(h)$  is the output of neuron  $h$  in the output layer, then, the output of classifier for matrix  $Z_g(m, n)$  can be represented by the vector

$$P_g = (p_g(1), p_g(2), \dots, p_g(H)); \quad g = 1, 2, \dots, G. \quad (27)$$

*Decision Rule:* The outputs of the  $G$  classifiers can be averaged and represented by the vector

$$P = (P^{\omega_1}, P^{\omega_2}, \dots, P^{\omega_H}) \quad (28)$$

where,

$$P^{\omega_h} = \left(\frac{1}{G}\right) \sum_{g=1}^G p_g(h). \quad (29)$$

A test movement is then assigned to class  $\omega_h$  using the rule in Equation (25).

The CNN-2 model is somewhat similar to the Sensor-Based Late Fusion models (SB-LF) described in [34,39] in the sense that there is one CNN per sensor. The main difference is that the SB-LF model has one FCN and the input to the FCN is the late fusion of the features from the last convolution layer of each sensor. The CNN-2 model has one FCN for each sensor and the late fusion is a form of decision fusion that occurs at the outputs of the CNNs.

### 5.3. CNN Implementation of the GAI Model (CNN-3)

The CNN-3 classification model using the GMI model, shown in Figure 11, is characterized by matrix convolutions in the first layer to extract local intra-sensor features and no output-level fusion. A small number of inter-sensor features are also extracted from the bordering uni-axial outputs from adjacent sensors in the input matrix. The input is the global matrix  $Z(m, n)$ .

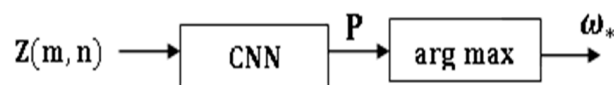


Figure 11. The CNN-3 classification model.

*Posterior Probability Computation:* In the first layer, the matrix  $Z(m, n)$  with dimension  $(M_g \times N \times 1)$  is convolved with with  $K^{[1]}$  filters, each with dimensions  $(f_h^{[1]} \times f_w^{[1]} \times 1)$ . The output of the  $k$ th filter yields a matrix  $\hat{Z}^{[1,k]}(m, n)$ . A bias  $b^{[1,k]}$  is added to the filtered output and passed through the nonlinear *ReLU* activation function. The activation of the filter, therefore, is given by

$$\tilde{Z}^{[1,k]}(m, n) = \text{ReLU} [\hat{Z}^{[1,k]}(m, n) + b^{[1,k]}] \quad (30)$$

The  $K^{[1]}$  filtered outputs are combined into a  $(M_g \times N \times K^{[1]})$  cuboid  $Z^{[1]}(m, n, k)$  which is pooled to give the cuboid  $Z^{[1,p]}(m, n, k)$ . The cuboid pooling operation is not described because it is similar to the one used in the previous model. The pooled cuboid is filtered by  $K^{[2]}$  cuboid filters and the output of the  $k$ th filter is a matrix represented by

$$\hat{Z}^{[2,k]}(m, n), m = 1, 2, \dots, M^{[1]}, n = 1, 2, \dots, N^{[1]} \quad (31)$$

where,  $M^{[1]}$  and  $N^{[1]}$  are the height and width of the pooled output  $Z^{[1,p]}(m, n, k)$ . The series of convolutions and pooling operations terminate into a FCN with a softmax output layer which gives an estimate of the  $H$  movement probabilities. The softmax output is represented by the vector

$$P = (P^{\omega_1}, P^{\omega_2}, \dots, P^{\omega_H}). \quad (32)$$

*Decision Rule:* a test movement is assigned to class  $\omega_h$  using the rule in Equation (25).

The CNN-3 model is similar to the Early Fusion (EF) model described in [37,39]. The difference is mainly in the selection of the dimensions of the filters in the convolution layers.

### 5.4. CNN Implementation of the CI Model (CNN-4)

The CNN-4 classification model, shown in Figure 12, is implemented using the cuboid representation which is obtained by fusing the LMI local matrices into a cuboid. Cuboid convolutions in the first layer extract coupled intra-sensor and inter-sensor features throughout the input.



Figure 12. The CNN-4 classification model.

*Posterior Probability Computation:* The cuboid input  $Z(m, n, g)$  is convolved with cuboid filters  $(f_h^{[1]} \times f_w^{[1]} \times G)$  and the output of the  $k$ th filter, is represented by



$$Z^{[1,k]}(m,n), m = 1,2,\dots,M; n = 1,2,\dots,N. \quad (33)$$

Note that convolving two cuboids with the same depth results in a matrix. The  $K^{[1]}$  filtered outputs are combined into a  $(M \times N \times K_1)$  cuboid after the biases are added and passed through the *ReLU* activation function. The height and width of the cuboid is adjusted if a pooling layer follows the convolution layer. Subsequent convolutions are also cuboid convolutions which result in matrices which are then combined into cuboids. An FCN with softmax outputs is implemented after the last pooling layer. The softmax output is represented by the vector

$$P = (P^{\omega_1}, P^{\omega_2}, \dots, P^{\omega_H}). \quad (34)$$

*Decision Rule:* a test movement is assigned to class  $\omega_h$  using the rule in Equation (25).

The CNN-4 model is unique because, to the best of our knowledge, there are no similar models which combine the uniaxial signals of each sensor into matrices, combine the matrices into a cuboid, and extract a combination of intra-sensor and inter-sensor features.

## 6. Experimental Data Collection

Motion capture for both taekwondo and boxing was conducted using custom Inertial Measurements Units (IMUs), developed in previous motion tracking research [7] as a basis for a commercial product. The IMU consists of a 3-axis accelerometer and 3-axis gyroscope; the ranges of the two sensor modules was set at  $\pm 16$  g and  $\pm 2000$  dps respectively to capture the full range of motion in both sports. Sampling frequency was constant for both sports, at 100 Hz. Data was streamed in real time from the IMU to the control computer via Bluetooth 4.0 communication. The IMU module was placed on the striking limb and held by Velcro straps. A pouch was sewn on the inside of the strap to keep IMU positioning consistent throughout the data collection process. Positioning and axis orientation of the IMU for boxing and taekwondo are outlined for sample movements in Figures 13 and 14, respectively. Note that these axes are relative and rotate along with the limb.



Figure 13. Data collection example: left jab punch (pad) (sensors under glove on wrist).

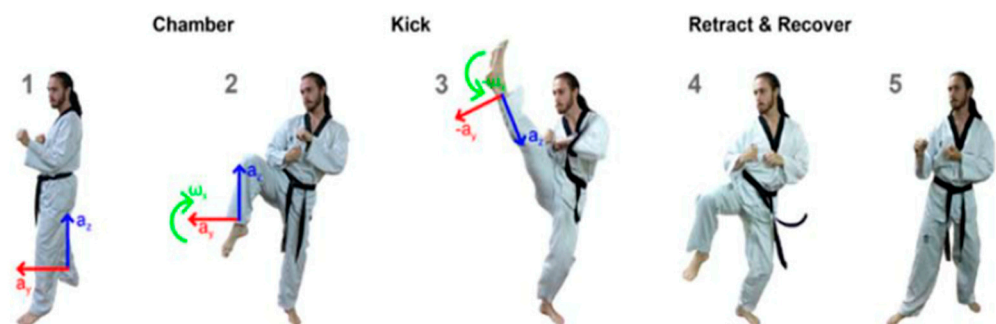


Figure 14. Data collection example: front right kick (sensors strapped to each ankle).

Experiments were designed to demonstrate the application and evaluation of the three DTW and four CNN classification models developed in this study. Motion capture data was collected from 15 martial artists of varying experience. 18 classes of boxing punches

and 24 classes taekwondo kicks (6 kicks for each leg for shadow and bag strikes) were collected as consistent with the entire range of movements for each sport. The classification models were given no a priori information on sensor placement or left/right limb to make the systems robust to using either sensor on either limb without polarization. Moreover, the models were not presented any a priori movement features. To our knowledge, no previous system has classified this wide range of movement and no existing classification model has demonstrated generalizability to both sports [6].

Boxing punches were acquired by placing the IMU on the wrists of each martial artist for 6 different punch classes during shadow boxing, punching a heavy bag, and with a trainer holding pads (2880 strikes, 18 classes). For taekwondo, an IMU was placed on the ankle of each martial artist executing kicking motions (2880 strikes, 24 classes). The signals were segmented using a signal-energy based algorithm [52] to locate the start and end-points of the strikes. The boxing punch classes, and taekwondo kick classes are listed in Tables 1 and 2, respectively. Note the right- and left-hand classes will be switched for left-handed (Southpaw) boxers. Table 3 shows examples of superimposed ensembles of boxing and taekwondo strikes. For clarity, Figures 15 and 16 show enlarged versions of the ensembles of one boxing and one taekwondo strike extracted from Table 3, respectively. From the blur in each figure, it is clear that signal peaks and valleys within each class are not aligned. Such nonlinear variations are typical in other boxing and taekwondo strike classes (and other human movements).

Given that the number of sensors  $G$  is 2, the number of axes  $m_g$  in each sensor is 3, and the total number of axes  $M_g$  is 6, the 4 input models are characterized by the following:

- VI: 6 vectors of dimension  $(3 \times n_{gm})$ , where  $n_{gm}$  is the dimension of uni-axial signal  $s_{gm}$ .
- LMI: 2 matrices of dimension  $(3 \times n_g)$ , where  $n_g$  is the normalized duration of the uni-axial signals of sensor  $S_g$ .
- GMI: A  $(6 \times N)$  matrix, where  $N$  is the normalized duration of all  $M_g = 6$  uni-axial signals.
- GCI: A  $(3 \times N \times 2)$  cuboid, where  $N$  is the normalized duration of all  $M_g = 6$  uni-axial signals.

**Table 1.** Boxing Strikes: 18 classes.

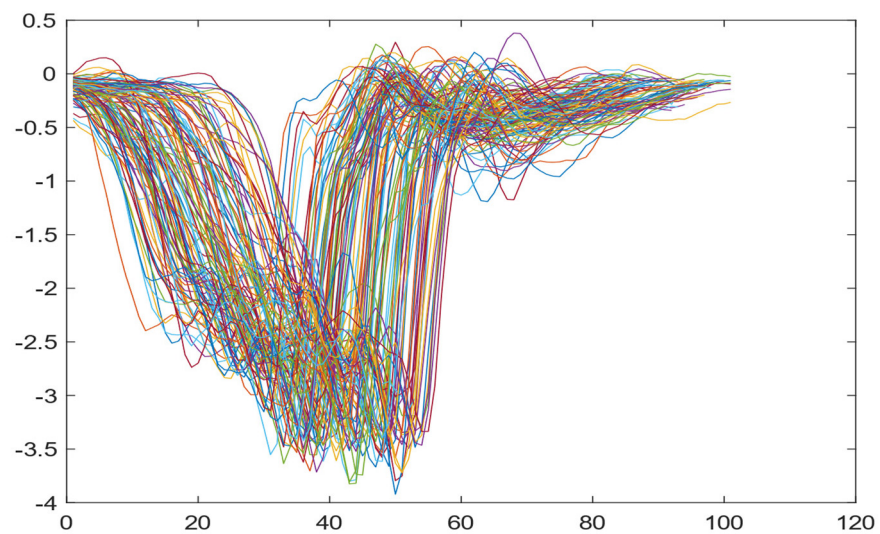
Hand	Shadow Boxing	Heavy Bag	Pads Strike
Right	Cross	Cross	Cross
	Hook	Hook	Hook
	Upper cut	Upper cut	Upper cut
Left	Jab	Jab	Jab
	Hook	Hook	Hook
	Upper Cut	Upper Cut	Upper Cut

**Table 2.** Taekwondo Kicks: 24 classes (12 shadow, 12 bag).

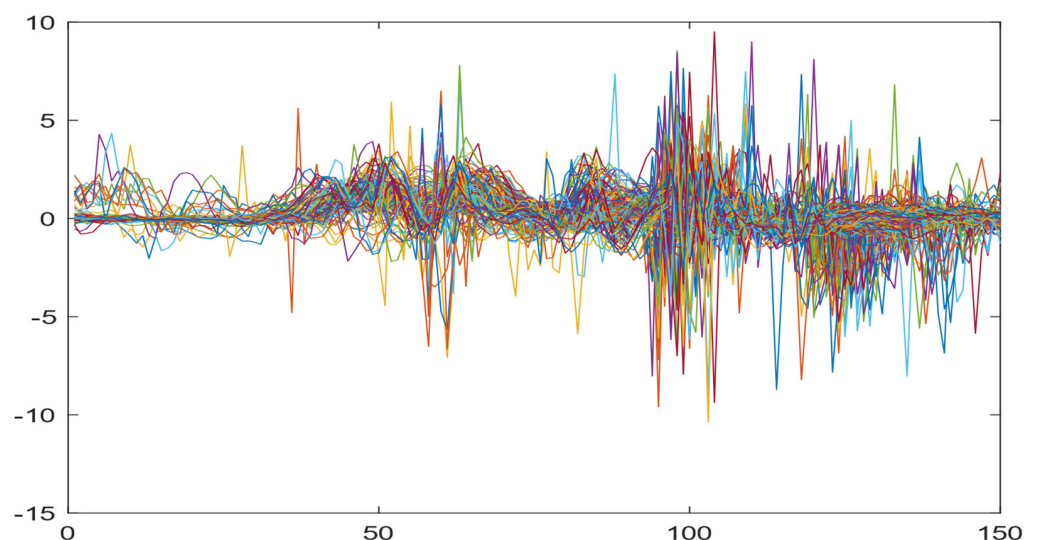
Kick Type (Shadow)		Kick Type (Heavy Bag)	
Right Leg	Left Leg	Right Leg	Left Leg
Turn Kick	Turn Kick	Turn Kick	Turn Kick
Axe Kick	Axe Kick	Axe Kick	Axe Kick
Front Kick	Front Kick	Front Kick	Front Kick
Back Kick	Back Kick	Back Kick	Back Kick
Side Kick	Side Kick	Side Kick	Side Kick
Reverse Hook Kick	Reverse Hook Kick	Reverse Hook Kick	Reverse Hook Kick

Table 3. Boxing and taekwondo strikes superimposed.

Boxing Strikes			
Accelerometer	x-axis	y-axis	z-axis
Right Hand Shadow Hook			
Right Hand Bag Hook			
Gyroscope	x-axis	y-axis	z-axis
Right Hand Shadow Hook			
Right Hand Bag Hook			
Taekwondo Strikes			
Accelerometer	x-axis	y-axis	z-axis
Right Axe Contact Kick			
Right Back Contact Kick			
Gyroscope	x-axis	y-axis	z-axis
Right Axe Contact Kick			
Right Back Contact Kick			



**Figure 15.** Superimposed strikes of the boxing Right Hand Shadow Hook ensemble acquired from the x-axis of the accelerometer.



**Figure 16.** Superimposed strikes of the taekwondo Right Axe Contact Kick ensemble acquired from the x-axis of the accelerometer.

## 7. System Training and Convergence

Each data set was divided randomly into a training set and a test set containing approximately 80% and 20% of the strikes, respectively. The average classification accuracy for the test set was determined. The random partitioning into training and test sets was repeated 100 times, with classification accuracies across repetitions averaged to obtain a final estimate of the classification accuracy.

For the DTW classifiers, the reference templates for the strike classes were determined by averaging the signals in their respective training sets. The CNN classifiers were initialized with a different set of random weights for each random partitioning of the data sets. Consequently, the final classification accuracy was obtained by averaging the results of 100 different CNNs. In order to keep the comparisons fair, the number of convolutions, pooling, and FC layers were fixed for all experiments. Moreover, the ordering of the layers was fixed. Given that the dimensions of the data were relatively small (2 sensors, 3 axes/sensor), a deep network with a large number of convolution and pooling layers was not needed. The CNN, therefore, consisted of a convolution layer, convolution layer, pool-

ing layer, and 2 FC layers in which the first FC layer used sigmoidal activation functions and the last FC layer used softmax activation functions. The “same” operation was used in the convolution layer and max pooling was used in the pooling layer. The number of filters were 32 and 32 in the first and second convolution layers, respectively. The filter dimensions in the first and second convolution layers were as follows:  $(1 \times 3 \times 1)$  and  $(1 \times 3 \times 32)$  for CNN-1,  $(3 \times 3 \times 1)$  and  $(3 \times 3 \times 32)$  for CNN-2,  $(3 \times 3 \times 1)$  and  $(3 \times 3 \times 32)$  for CNN-3, and  $(3 \times 3 \times 2)$  and  $(3 \times 3 \times 32)$  for CNN-4, respectively. The networks were implemented using the Keras library [53–55].

Training times were benchmarked for each input model and classifier. Time efficiency profiling was conducted by using the MATLAB 2021a internal profiler for DTW models and the Python 3.8.12 c Profile function for CNN models. All evaluations were conducted on a system using Windows 10 Home Edition with an Intel Core i7-6700 k 4GHz Quad Core CPU, GeForce GTX 980 Ti GPU and 32GB RAM.

## 8. Results and Analysis

Figure 17 shows the total time in seconds for model parameterization. The CNN training time increase is expected given the repeated layering design as opposed to the single pass in the DTW. CNN1 and CNN2 are also setup for multiple neural networks in training, hence their increase in training times. CNN3 and CNN4, however, show parameter convergence in comparable time to DTW in the single network training. It is also interesting to note that boxing data actually took longer to train or had negligible differences to taekwondo in CNN implementations, despite being an 18-class problem versus 24-class. Boxing punches very more between the dominant and non-dominant side, but boxing must also distinguish between pad (human held and bag strike classification which could account for comparable or longer time to converge. We also note that differences between an uppercut and hook punch can include highly nonlinear variations in movement waveform.

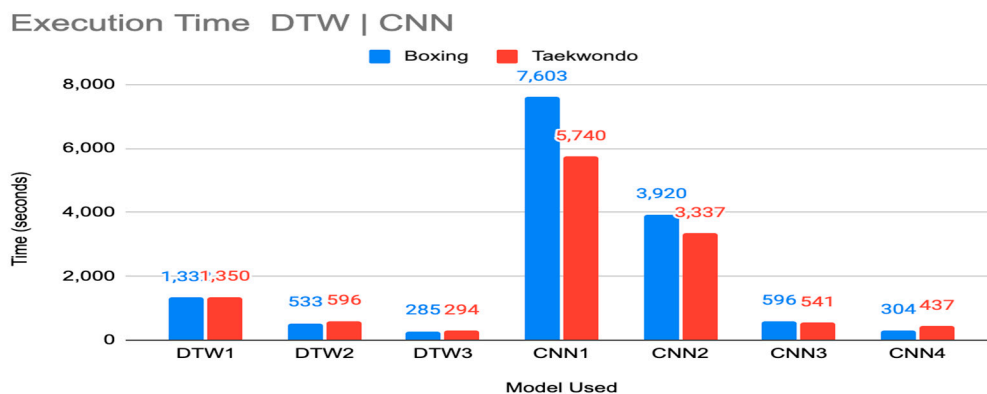


Figure 17. Training parameterization time for DTW and CNN models.

While the training times for CNN1 and CNN2 were significant, it is also important to note that this has no impact on online use for real-time classification. Training, in particular for such large data sets will be done offline with identification in real-time occurring with trained models. Deep neural networks have well-established properties for computationally compact representations of nonlinear models enabling use online, even in time critical applications with limited computational resources (e.g., [56]). Complete identification of strikes occurs in negligible (msc) timeframes for all CNN models. DTW models are not as computationally lean for online use as they require a comparison of each data point of an incoming motion to each data point in the movement class. However, the time to execute this in real-time is still suitable for most online human-motion tracking applications. We envision training for individualized movements to be completed on cloud servers with online use parameters updated to edge devices to execute in firmware, as implemented in our commercial systems.



Outputs of the classification experiments after training are summarized in Tables 4 and 5. Table 4 shows the results of classifying each uniaxial signal independently for both data sets. For each classifier (table row), the result of the best axis channel is shown in boldface. For example, for the uni-axial boxing DTW classifiers, the best result of 65.1% was obtained from the x-axis channel of the accelerometer. An accuracy of 65.1% may not be strong, however, in comparison, an accuracy of only 5.6% can be expected through the random classification for an 18-class problem. The classification accuracies of the seven fusion classifiers are shown in Table 5 for both data sets. The best result for each data set is shown in boldface. Note that for each classifier type and data set, the worst fusion result in Table 5 is much better than the best uni-axial result in Table 4. This clearly demonstrates the merits in fusing information from multiple sensors and axes. Also note that in Tables 4 and 5, the accuracies of the CNN classifiers are much higher than those of the DTW classifiers.

**Table 4.** Uniaxial classification accuracies (%).

Sport	Classifier	Accelerometer Axes			Gyroscope Axes		
		X	Y	Z	X	Y	Z
Boxing	DTW	<b>65.1</b>	53.1	59.4	58.1	57.9	59.4
	CNN	75.8	71.2	70.8	70.4	67.6	<b>77.1</b>
Taekwondo	DTW	<b>52.3</b>	46	36.4	41.2	49.7	36.4
	CNN	<b>81.7</b>	75.1	70.3	75.8	81.1	77.1

**Table 5.** Fusion classification accuracies (%).

Classifier	Boxing (18 Class)	Taekwondo (24 Class)
DTW-1	77.42	64.00
DTW-2	80.43	69.32
DTW-3	79.59	68.69
CNN-1	87.21	86.89
CNN-2	89.70	88.02
CNN-3	<b>92.08</b>	88.14
CNN-4	91.26	<b>88.70</b>

The fact that the CNN classifiers performed better than the DTW classifiers is quite unexpected for the following reasons: (a) unlike DTW classifiers, CNNs do not readily appear to be a good choice for classifying signals that are not naturally in a 2-d or multidimensional array formats, and (b) unlike the design of DTW classifiers, the design of CNN classifier do not typically focus on addressing the non-linear variations problem. From the results, it is interesting to note the following:

- (a) The CNN classifiers performed well in spite of the fact that the uni-axial strike signals typically experience non-linear variations as seen in Table 3 and in Figures 15 and 16. The reason for this performance can be explained by noting that features are detected locally and not globally. Consequently, the local features tend to be invariant to latency shifts. Moreover, the local features are unaffected in the segments that do not experience non-linear variations. The trial-to-trial variations of the signals within each training set can be regarded as a natural form of “data augmentation” which is a technique commonly used to artificially increase the diversity in the training set without having to collect additional data. The CNN classifiers are capable of learning the typical variations in the signals by presenting the network with representative signals during training.
- (b) Using the same input data, the CNN implementations using the four input models extracted different types of local features for classification. The CNN classifiers, therefore, offer many choices of local features which can be selected depending on the type of coupling assumed or desired between the intra and inter-sensor outputs. For example, if the uni-axial outputs of all sensors are assumed independent, CNNs



using the VI model can be selected. CNNs using the LMI model can be selected if the channels in a sensor carry complementary information for determining the output class. If complementary information is shared across all sensors, CNNs using the GCI model will be an effective choice. The manner in which the inputs are fused can take other factors into account, for example, the geographical locations (co-located or dispersed) of the sensors. The sensor outputs can also be fused in other ways. For example, the x-axis channels of all sensor can be combined into a matrix. The y-axis and z-axis channels can be combined in a similar manner. The intra and inter-sensor coupling assumptions can, therefore, be used to choose a particular classification model for a given problem.

- (c) It is unlikely that the performance of DTW classifiers can be improved by increasing the size of the training set because the template, which is the training set average, will change only marginally after a certain point and this marginal change will have little effect on the performance. Contrarily, CNN classifiers have the potential to improve performance by extracting more complex features by increasing the network depth and training data. Furthermore, by increasing the network depth and training data, CNNs are capable of accurately classifying a larger number of classes, whereas, the performance of traditional classifiers such as DTW classifiers will tend to drop as the number of classes increase.
- (d) It is interesting to compare the performances of the one, two, and three-dimensional classifiers resulting from the VI, LMI & GMI, and GCI input models, respectively. By comparing the results for the DTW classifiers in Table 5, it is first noted that the classification accuracies vary marginally for all DTW classifiers across both sets of data. The best results for the boxing and taekwondo data were obtained by the 2-dimensional DTW-2 classifiers. The classification accuracies also varied marginally across the CNN classifiers for both data sets. The best results were obtained by the 2-dimensional CNN-3 and CNN-4 classifiers for the boxing and taekwondo data, respectively.

It is also worth noting that CNNs offer particularly intriguing potential for widespread use in commercial wearables due to low computational expense of online use. A cloud-based training system working in conjunction with an embedded wearable would enable real-time training feedback coupled with updates and adaptation as movements change with time.

Finally, it should be stressed the results presented here are designed to demonstrate the capacity of the input modelling and classification approaches in the most challenging of circumstances. Tables 4 and 5 show output for the maximum number of classes with **zero** knowledge of movement and the broadest class of athletes. While the accuracies by themselves are enough for commercial use, further improvements are easily possible in practice. The eighteen-class boxing data, for example, yielded fusion classification accuracies of 95% + for CNN-3 with a subset (1/3) of the boxers who were not beginners. Furthermore, it is unlikely that boxers or martial artists even with simple training will mix striking a heavy bag, shadow boxing, or pad striking in the same round. Such measures will virtually eliminate misclassification such that the only errors are erratic strikes from the user that do not fit any strike model.

## 9. Translation for Mass Market Athletic Training

The research executed in this investigation has led to the design, fabrication, and commercial translation of a complete IoT sensor system for smart boxing. Our design reflects the evolution of wearables from a 'device' to a 'systems' perspective [9], and consists of original sensors, embedded code, apps for use with a smart phone for data collection, and cloud computing for data storage and visualization. The system, shown in Figure 18, has been released as a commercial product by Corner Wearables based in Manchester, UK. It was first trialed with the boxing team at Imperial College London and subsequently expanded into a full product for sale worldwide. The integrated system consists of a

small sensor in the boxer's hand wraps that fits under boxing gloves. All code for punch identification is embedded onboard with a microcontroller to detect and classify movement history, which is sent via Bluetooth to a smart device for display and storage through an app on a smart phone. The embedded code performs all pattern classification hence transmission is only necessary for statistics saving the need to send raw data over Bluetooth. The first-generation commercial system tracks 6 classes of punches (dominant hand–cross, hook, uppercut, non-dominant hand–jab, hook, uppercut). Subsequent releases will classify the full 18-class problem outlined in Section 6 using the full deep learning architecture outlined in this investigation. Corner, featured in IEEE Spectrum [57], is the first ever smart boxing tracker which does not need polarized (left-right specified) sensors. It was recently assessed in a boxing study as a part of this special issue of Sensors [58] as having the capacity to track both beginners and experienced boxers, though beginner punches are less consistent due to immaturity of technique. Thousands of devices are currently in use, providing an intriguing database for analysis in future work. The commercial system has also been used in live boxing matches, including the World Series of Boxing, to provide real-time statistics to spectators, judges, and trainers to evaluate match performance.



**Figure 18.** Corner Boxing System: Sensor with dimensions (top), sensor, phone app and wrist attachment (middle), and sample group training session with data from several users compiled and displayed simultaneously (bottom).

## 10. Conclusions

The goal of this investigation was to develop models to classify human movement by fusing information from ensembles of wearable multi-axial inertial sensors. The specific contributions resulting from the investigation include: (a) the introduction of four multi-sensor multi-axial input models that can be used in conjunction with diverse classifiers, (b) demonstrating the use of the input models to develop three DTW and four CNN fusion-based classifier models that do not require a set of predetermined hand-engineered features, (c) testing the validity of the classifier on boxing and taekwondo sport data, (d) demonstrating the merits of multi-axial fusion by showing that the worst fusion classifiers outperform the best uniaxial classifiers, (e) demonstrating that high classification accuracies can be obtained with the CNN fusion classifiers on signals that experience large non-linear variations and on signals belonging to a large number of classes, (f) demonstrating the surprising result that the CNN fusion classifiers outperform the DTW classifiers, (g) explaining the ability of the CNN classifiers to extract local features which depend on the type of coupling assumed or desired between the intra and inter-sensor outputs, and (h) noting that CNN classifiers have the potential to improve performance and handle a larger number of classes through both training and network scaling.

To our knowledge this is the first set of models demonstrated on this large a problem class in either activity [7] and the first generalized non-feature specific classification over multiple movement ranges. Also noteworthy is that due to the generalized formulations, the classifiers can be easily adapted to classify multi-dimensional signals of multiple sensors in various other applications.

Future work involves refining the system for exact learning of individual users for performance assessment, analyzing time series data from training of large groups of athletes, and implementation for live performance streaming in professional fights to enhance spectator experience support of fight scoring. As a completely feature-blind generic classification strategy, translation is also underway in other sports (e.g., tennis) as well as in wearables for telemedicine in neural motor dysfunction conditions such as stroke and Parkinson's Disease [59,60]. We believe these results provide a foundation for a new set of human movement classification paradigms based on fusion and deep learning.

**Author Contributions:** Conceptualization, L.G., C.B., P.B. and R.V.; sensor design, S.W., C.B. and R.V.; methodology, R.A., L.G., N.S. and R.V.; software, R.A. and N.S.; data collection, K.A.; writing—original draft preparation, L.G., P.B. and R.V.; writing—review and refinement, L.G., P.B., K.A. and R.V.; funding acquisition, R.V. and L.G.; commercial translation, C.B. and R.V. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the EPSRC (EP/R511547/1), the EPSRC CDT in Neurotechnology, the Department of Mechanical Engineering and UK DRI CR&T at Imperial College London (ICL) and Athletec Inc.

**Institutional Review Board Statement:** The study was approved by the Imperial College Research Ethics Committee (ICREC) under study reference 15IC3068. Human subject permission does not explicitly give permission for data release.

**Informed Consent Statement:** Informed consent was obtained from all subjects involved in the study.

**Data Availability Statement:** Please contact corresponding authors for data availability. Ethics permission does not explicitly give permission for public release of human performance data at the time of publication.

**Acknowledgments:** We gratefully acknowledge Master Luke Jamie Robinson and all the athletes at Titan Academy who participated in data collection, testing, and refinement and the Imperial College London Boxing Club. We also acknowledge the continued generous support of Athletec, Serg Technologies and the Department of Mechanical Engineering at Imperial College London.

**Conflicts of Interest:** C.B. and R.V. are co-founders of the company Corner, which has released the commercial boxing system described in Section 9.

## References

1. Burridge, J.H.; Lee, A.C.W.; Turk, R.; Stokes, M.; Whittall, J.; Vaidyanathan, R.; Clatworthy, P.; Hughes, A.-M.; Meagher, C.; Franco, E.; et al. Telehealth, Wearable Sensors, and the Internet: Will They Improve Stroke Outcomes Through Increased Intensity of Therapy, Motivation, and Adherence to Rehabilitation Programs? *J. Neurol. Phys. Ther.* **2017**, *41* (Suppl. 3), S32–S38. [CrossRef]
2. Woodward, R.B.; Shefelbine, S.J.; Vaidyanathan, R. Pervasive Monitoring of Motion and Muscle Activation: Inertial and Mechanomyography Fusion. *IEEE/ASME Trans. Mechatron.* **2017**, *22*, 2022–2033. [CrossRef]
3. Ahmadi, A.; Mitchell, E.; Richter, C.; Destelle, F.; Gowing, M.; O'Connor, N.E.; Moran, K. Toward Automatic Activity Classification and Movement Assessment during a Sports Training Session. *IEEE Internet Things J.* **2015**, *2*, 23–32. [CrossRef]
4. Camomilla, V.; Bergamini, E.; Fantozzi, S.; Vannozzi, G. Trends Supporting the In-Field Use of Wearable Inertial Sensors for Sport Performance Evaluation: A Systematic Review. *Sensors* **2018**, *18*, 873. [CrossRef] [PubMed]
5. Bianchi, V.; Bassoli, M.; Lombardo, G.; Fornacciari, P.; Mordonini, M.; De Munari, I. IoT Wearable Sensor and Deep Learning: An Integrated Approach for Personalized Human Activity Recognition in a Smart Home Environment. *IEEE Internet Things J.* **2019**, *6*, 8553–8562. [CrossRef]
6. Worsey, M.T.; Espinosa, H.G.; Shepherd, J.B.; Thiel, D.V. Inertial Sensors for Performance Analysis in Combat Sports: A Systematic Review. *Sports* **2019**, *7*, 28. [CrossRef]
7. Wilson, S.; Eberle, H.; Hayashi, Y.; Madgwick, S.O.H.; McGregor, A.; Jing, X.; Vaidyanathan, R. Formulation of a new gradient descent MARG orientation algorithm: Case study on robot teleoperation. *Mech. Syst. Signal Process.* **2019**, *130*, 183–200. [CrossRef]
8. Rodgers, M.M.; Alon, G.; Pai, M.M.; Conroy, R.S. Wearable technologies for active living and rehabilitation: Current research challenges and future opportunities. *J. Rehabil. Assist. Technol. Eng.* **2019**, *6*, 2055668319839607. [CrossRef] [PubMed]
9. Seshadri, D.R.; Li, R.T.; Voos, J.E.; Rowbottom, J.R.; Alfes, C.M.; Zorman, C.A.; Drummond, C.K. Wearable sensors for monitoring the internal and external workload of the athlete. *NPJ Digit. Med.* **2019**, *2*, 71. [CrossRef]
10. Bindi, T. A Third of Wearable Devices Abandoned by Consumers: Gartner. ZDnet. Available online: <https://www.zdnet.com/article/a-third-of-wearable-devices-abandoned-by-consumers-gartner/> (accessed on 7 December 2021).
11. Huo, W.; Angeles, P.; Tai, Y.F.; Pavese, N.; Wilson, S.; Hu, M.T.; Vaidyanathan, R. A Heterogeneous Sensing Suite for Multisymptom Quantification of Parkinson's Disease. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2020**, *28*, 1397–1406. [CrossRef]
12. Ordóñez, F.J.; Roggen, D. Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. *Sensors* **2016**, *16*, 115. [CrossRef]
13. Lapinski, M.; Brum Medeiros, C.; Moxley Scarborough, D.; Berkson, E.; Gill, T.J.; Kepple, T.; Paradiso, J.A. A Wide-Range, Wireless Wearable Inertial Motion Sensing System for Capturing Fast Athletic Biomechanics in Overhead Pitching. *Sensors* **2019**, *19*, 3637. [CrossRef] [PubMed]
14. Ravi, D.; Wong, C.; Lo, B.; Yang, G.-Z. A Deep Learning Approach to on-Node Sensor Data Analytics for Mobile or Wearable Devices. *IEEE J. Biomed. Health Inform.* **2017**, *21*, 56–64. [CrossRef] [PubMed]
15. Gupta, L.; Srinath, M.D. Invariant planar shape recognition using dynamic alignment. *Pattern Recognit.* **1988**, *21*, 235–239. [CrossRef]
16. Gupta, L.; Malakapalli, K. Robust partial shape classification using invariant breakpoints and dynamic alignment. *Pattern Recognit.* **1990**, *23*, 1103–1111. [CrossRef]
17. Gupta, L.; Tammana, R. A discrepancy measure for improved clustering. *Pattern Recognit.* **1995**, *28*, 1627–1634. [CrossRef]
18. Gupta, L.; Molfese, D.L.; Tammana, R.; Simos, P.G. Nonlinear alignment and averaging for estimating the evoked potential. *IEEE Trans. Biomed. Eng.* **1996**, *43*, 348–356. [CrossRef] [PubMed]
19. De Wachter, M.; Matton, M.; Demuynck, K.; Wambach, P.; Cools, R.; Van Compernelle, D. Template-based continuous speech recognition. *IEEE Trans. Audio Speech Lang. Process.* **2007**, *15*, 1377–1390. [CrossRef]
20. Ten Holt, G.A.; Reinders, M.J.; Hendriks, E. Multi-dimensional dynamic time warping for gesture recognition. In Proceedings of the Thirteenth Annual Conference of the Advanced School for Computing and Imaging, Montreal, QC, Canada, 9–14 September 2007.
21. de Mello, R.F.; Gondra, I. Multi-Dimensional Dynamic Time Warping for Image Texture Similarity. In Proceedings of the 19th Brazilian Symposium on Artificial Intelligence: Advances in Artificial Intelligence, Salvador, Brazil, 26–30 October 2008; Springer: Berlin/Heidelberg, Germany, 2008.
22. Wöllmer, M.; Al-Hames, M.; Eyben, F.; Schuller, B.; Rigoll, G. A multidimensional dynamic time warping algorithm for efficient multimodal fusion of asynchronous data streams. *Neurocomputing* **2009**, *73*, 366–380. [CrossRef]
23. Muda, L.; Begam, M.; Elamvazuthi, I. Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW) techniques. *arXiv* **2010**, arXiv:1003.4083.
24. Petitjean, F.; Ketterlin, A.; Gançarski, P. A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognit.* **2011**, *44*, 678–693. [CrossRef]
25. Raheja, J.; Minhas, M.; Prashanth, D.; Shah, T.; Chaudhary, A. Robust gesture recognition using Kinect: A comparison between DTW and HMM. *Optik* **2015**, *126*, 1098–1104. [CrossRef]
26. Radović, M.; Ghalwash, M.; Filipović, N.; Obradović, Z. Minimum redundancy maximum relevance feature selection approach for temporal gene expression data. *BMC Bioinform.* **2017**, *18*, 9. [CrossRef]
27. Hachaj, T.; Piekarczyk, M.; Ogiela, M. Human actions analysis: Templates generation, matching and visualization applied to motion capture of highly-skilled karate athletes. *Sensors* **2017**, *17*, 2590. [CrossRef]



28. Shokoohi-Yekta, M.; Hu, B.; Jin, H.; Wang, J.; Keogh, E. Generalizing DTW to the multi-dimensional case requires an adaptive approach. *Data Min. Knowl. Discov.* **2017**, *31*, 1–31. [[CrossRef](#)]
29. Kim, S.H.; Lee, H.S.; Ko, H.J.; Jeong, S.H.; Byun, W.H.; Oh, K.J. Pattern Matching Trading System Based on the Dynamic Time Warping Algorithm. *Sustainability* **2018**, *10*, 4641. [[CrossRef](#)]
30. Le Guennec, A.; Malinowski, S.; Tavenard, R. Data Augmentation for Time Series Classification using Convolutional Neural Networks. In Proceedings of the ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data, Riva del Garda, Italy, 19–23 September 2016.
31. Zhao, B.; Lu, H.; Chen, S.; Liu, J.; Wu, D. Convolutional neural networks for time series classification. *J. Syst. Eng. Electron.* **2017**, *28*, 162–169. [[CrossRef](#)]
32. Hatami, N.; Gavet, Y.; Debayle, J. Classification of Time-Series Images Using Deep Convolutional Neural Networks. In Proceedings of the Tenth International Conference on Machine Vision (ICMV 2017), Vienna, Austria, 13–15 November 2017; International Society for Optics and Photonics: Bellingham, WA, USA, 2017.
33. Sezer, O.B.; Ozbayoglu, A.M. Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. *Appl. Soft Comput.* **2018**, *70*, 525–538. [[CrossRef](#)]
34. Martinez, H.P.; Bengio, Y.; Yannakakis, G.N. Learning deep physiological models of affect. *IEEE Comput. Intell. Mag.* **2013**, *8*, 20–33. [[CrossRef](#)]
35. Zeng, M.; Nguyen, L.T.; Yu, B.; Mengshoel, O.J.; Zhu, J.; Wu, P.; Zhang, J. Convolutional Neural Networks for Human Activity Recognition Using Mobile Sensors. In Proceedings of the 6th International Conference on Mobile Computing, Applications and Services, Austin, TX, USA, 6–9 November 2014; IEEE: New York, NY, USA, 2014.
36. Yang, J.; Nguyen, M.N.; San, P.P.; Li, X.; Krishnaswamy, S. Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015; IJCAI: Palo Alto, CA, USA.
37. Neverova, N.; Wolf, C.; Lacey, G.; Fridman, L.; Chandra, D. Learning human identity from motion patterns. *IEEE Access* **2016**, *4*, 1810–1820. [[CrossRef](#)]
38. Dehzangi, O.; Taherisadr, M.; Changalvala, R. IMU-based gait recognition using convolutional neural networks and multi-sensor fusion. *Sensors* **2017**, *17*, 2735. [[CrossRef](#)]
39. Münzner, S.; Schmidt, P.; Reiss, A.; Hanselmann, M.; Steifelhagen, R. CNN-based sensor fusion techniques for multimodal human activity recognition. In Proceedings of the 2017 ACM International Symposium on Wearable Computers, Maui, HI, USA, 11–15 September 2017; ACM: New York, NY, USA, 2017.
40. Kook, H.; Gupta, L.; Molfese, D.; Fadem, K. C Multi-stimuli multi-channel data and decision fusion strategies for dyslexia prediction using neonatal ERPs. *Pattern Recognit.* **2005**, *38*, 2174–2184. [[CrossRef](#)]
41. Gupta, L.; Chung, B.; Srinath, M.D.; Molfese, D.L.; Kook, H. Multichannel fusion models for the parametric classification of differential brain activity. *IEEE Trans. Biomed. Eng.* **2005**, *52*, 1869–1881. [[CrossRef](#)]
42. Polikar, R. *Ensemble Machine Learning*; Springer: Boston, MA, USA, 2012; pp. 1–34.
43. Kota, S.; Gupta, L.; Molfese, D.; Vaidyanathan, R. Diversity-Based Selection of Polychotomous Components for Multi-Sensor Fusion Classifiers. *J. Eng. Med.* **2013**, *227*, 655–662.
44. Kuncheva, L.I. *Combining Pattern Classifiers: Methods and Algorithms*; John Wiley & Sons: New York, NY, USA, 2014.
45. Amerineni, R.; Gupta, R.; Gupta, L. Multimodal Object Classification Models Inspired by Multisensory Integration in the Brain. *Brain Sci.* **2019**, *9*, 3. [[CrossRef](#)] [[PubMed](#)]
46. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [[CrossRef](#)]
47. Sermanet, P.; Eigen, D.; Zhang, X.; Mathieu, M.; Fergus, R.; LeCun, J. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv* **2013**, arXiv:1312.6229.
48. Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; Li, F.-F. Large-scale video classification with convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; IEEE: New York, NY, USA, 2014.
49. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
50. Zeiler, M.D.; Fergus, R. Visualizing and Understanding Convolutional Networks. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Boston, MA, USA, 2014.
51. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Wang, X.; Lui, T.; Wang, L.; Wang, G. Recent advances in convolutional neural networks. *Pattern Recognit.* **2018**, *77*, 354–377. [[CrossRef](#)]
52. Vaidyanathan, R.; Chung, B.; Gupta, L.; Kook, H.; Kota, S.; West, J.D. Tongue-Movement Communication and Control Concept for Hands-Free Human–Machine Interfaces. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2007**, *37*, 533–546. [[CrossRef](#)]
53. Chollet, F. Keras Deep Learning Library Tensorflow. Available online: <https://keras.io/> (accessed on 26 November 2021).
54. Sarkar, D.; Bali, R.; Ghosh, T. *Hands-On Transfer Learning with Python: Implement Advanced Deep Learning and Neural Network Models Using TensorFlow and Keras*; Packt Publishing Ltd.: Birmingham, UK, 2018.
55. Shanmugamani, R. *Deep Learning for Computer Vision: Expert Techniques to Train Advanced Neural Networks Using TensorFlow and Keras*; Packt Publishing Ltd.: Birmingham, UK, 2018.

- 
56. Vaidyanathan, R.; Chen, C.; Jeong, C.D.; Williams, C.; Ritzmann, R.E.; Quinn, R.D. A Reflexive Control Architecture based on a Neural Model of the Cockroach Escape Response. *J. Syst. Control Eng.* **2012**, *226*, 699–718. [[CrossRef](#)]
  57. Lightman, K. Next-Gen Sensors Make Golf Clubs, Tennis Rackets, and Baseball Bats Smarter Than Ever. *IEEE Spectrum*. 25 February 2016. Available online: <https://spectrum.ieee.org/consumer-electronics/gadgets/nextgen-sensors-make-golf-clubs-tennis-rackets-and-baseball-bats-smarter-than-ever> (accessed on 26 November 2021).
  58. Omcirk, D.; Vetrovsky, T.; Padecky, J.; Vanbelle, S.; Malecek, J.; Tufano, J.J. Punch Trackers: Correct Recognition Depends on Punch Type and Training Experience. *Sensors* **2021**, *21*, 2968. [[CrossRef](#)] [[PubMed](#)]
  59. Formstone, L.; Hou, W.; Wilson, S.; McGregor, A.; Bentley, P.; Vaidyanathan, R. Quantification of Motor Function Post-stroke using Novel Combination of Wearable Inertial and Mechanomyographic Sensors. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2021**, *29*, 1158–1167. [[CrossRef](#)] [[PubMed](#)]
  60. Madgwick, S.O.H.; Wilson, S.; Turk, R.; Burridge, J.; Kapatos, C.; Vaidyanathan, R. An Extended Complementary Filter for Full-Body MARG Orientation Estimation. *IEEE/ASME Trans. Mechatron.* **2020**, *25*, 2054–2064. [[CrossRef](#)]