FenceNet: Fine-grained Footwork Recognition in Fencing

Kevin Zhu Alexander Wong John McPhee
University of Waterloo
{k79zhu, a28wong, mcphee}@uwaterloo.ca

Abstract

Current data analysis for the Canadian Olympic fencing team is primarily done manually by coaches and analysts. Due to the highly repetitive, yet dynamic and subtle movements in fencing, manual data analysis can be inefficient and inaccurate. We propose FenceNet as a novel architecture to automate the classification of fine-grained footwork techniques in fencing. FenceNet takes 2D pose data as input and classifies actions using a skeleton-based action recognition approach that incorporates temporal convolutional networks to capture temporal information. We train and evaluate FenceNet on the Fencing Footwork Dataset (FFD), which contains 10 fencers performing 6 different footwork actions for 10-11 repetitions each (652 total videos). FenceNet achieves 85.4% accuracy under 10-fold cross-validation, where each fencer is left out as the test set. This accuracy is within 1% of the current state-of-the-art method, JLJA (86.3%), which selects and fuses features engineered from skeleton data, depth videos, and inertial measurement units. BiFenceNet, a variant of FenceNet that captures the "bidirectionality" of human movement through two separate networks, achieves 87.6% accuracy, outperforming JLJA. Since neither FenceNet nor BiFenceNet requires data from wearable sensors, unlike JLJA, they could be directly applied to most fencing videos, using 2D pose data as input extracted from off-the-shelf 2D human pose estimators. In comparison to JLJA, our methods are also simpler as they do not require manual feature engineering, selection, or fusion.

1. Introduction and background

There is a current need from national-level fencing teams for the development of analytical tools to enhance performance and training. The first step is to achieve a deeper understanding of the physical, tactical, and technical demands of fencing. Once these demands are better understood, performance benchmarks can be created for different skill levels to identify gaps and more accurately evaluate athletes. This in turn contributes to athlete selection, skills progression, and training interventions. The main bottleneck is the lack of a reproducible means to collect the high quality, high resolution, objective data that is required to create these benchmarks.

Recognizing the need for automated analysis of techniques and motion in fencing, Malawski and Kwolek were among the first to apply computer vision approaches to detect and classify fine-grained actions in the sport. In [28,29], they proposed a method to classify fencing footwork by extracting 4 feature sets from visual and inertial signals, described below.

Joint dynamic (JD) features were proposed to describe the changes in motion of a fencer during the action, rather than the trajectory of motion. Skeleton data was split into windows of different sizes, for which the first 3 coefficients from the Short Time Fourier Transform computed for the velocity and acceleration of each joint along the 2 axes of 3 planes were used as features. Local trace image (LTI) features were proposed to represent the action as one image. Similar to motion energy images [5] and motion history images [12], a person's silhouette's binary images during the course of an action are superimposed, with a decay factor to better capture temporal information. To minimize noise, LTI crops out each joint, superimposes them separately, then resizes and concatenates them back together. Joint motion history context (JMHC) descriptors were proposed to capture local motion changes around joints. The absolute difference in silhouettes from depth images between two consecutive frames are described as histograms with each joint as the center. Histograms are normalized then concatenated to form a joint motion context (JMC) descriptor. The weighted sum of 3 consecutive JMC descriptors form a JMHC descriptor. Accelerometric (Acc) features from the time domain [39] were extracted from data captured by inertial measurement units (IMU).

Next, a feature selection algorithm based on feature ranking [8] was proposed to reduce the dimensionality of each feature set. The reduced feature sets are then fused with a decision-level fusion scheme [30] by training a separate support vector machine (SVM) [10] for each feature set and concatenating the outputs, which are finally fed into

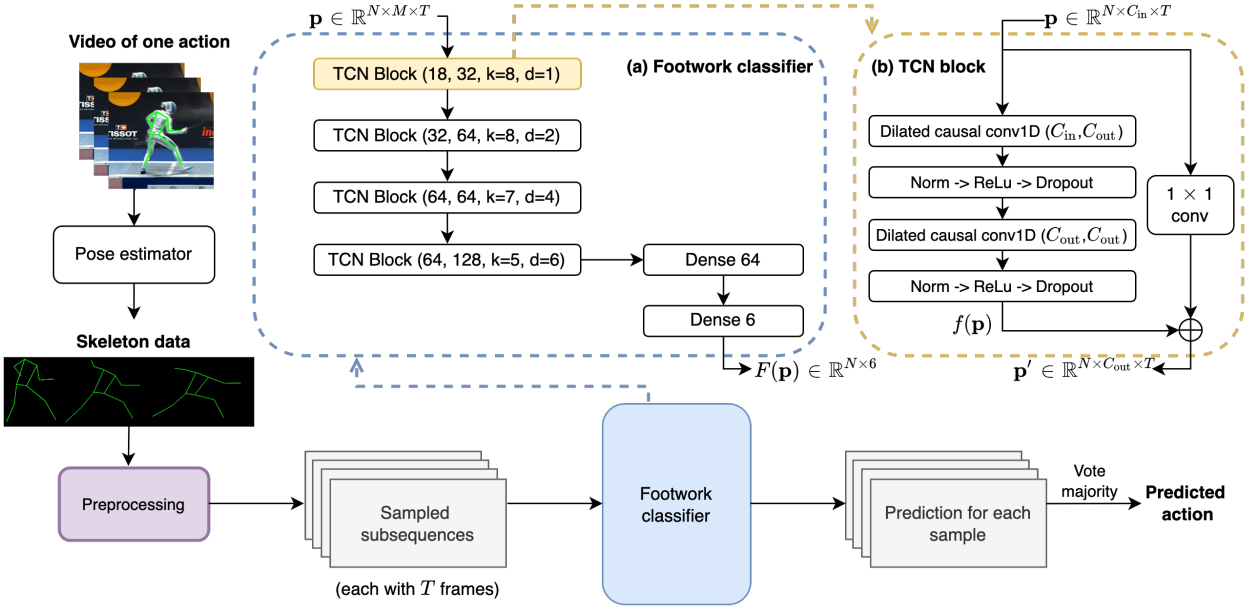

Figure 1. Network architecture of FenceNet. The footwork classifier (a) consists of stacked TCN blocks shown in (b). When training on 2D skeleton data from FFD, the pose estimator step is omitted.

a multilayer perceptron [41] for classification. We refer to this method as JLJA (JD+LTI+JMHC+Acc) moving forward.

JLJA was trained and evaluated on the Fencing Footwork Dataset (FFD) [28] that contains 6 basic fencing footwork actions – stepping forward, stepping back, and 4 types of lunges with similar motion trajectories but subtle differences in dynamics. JLJA is currently the best performing method on FFD. However, the requirement of wearable sensors and depth video limits the pool of athletes to which JLJA can be applied.

To overcome this limitation, we propose a novel architecture, FenceNet, that takes only 2D skeleton data as input for the same classification task, and achieves similar accuracy to JLJA when evaluated on FFD. We also introduce a variant, BiFenceNet, that outperforms JLJA while using the same 2D skeleton data. This way, coaches and analysts could extract information directly from videos, by training FenceNet on 2D pose data extracted from an off-the-shelf 2D pose estimator [6, 32, 51], as seen in Fig. 1. FenceNet uses a skeleton-based human action recognition approach [9, 23, 54] that incorporates temporal convolutional networks (TCN) to capture temporal information.

The concept of a TCN was first introduced by Lea *et al*. [19] for action segmentation and detection in videos. TCNs are mainly characterized by the use of two types of convolutions:

- causal convolutions to ensure no leakage of future information into the current time step.
- dilated convolutions to exponentially enlarge the receptive field.

Similar to recurrent neural networks (RNN) [16], TCN models are able to take in a sequence of variable length and produce an output of the same length as the input. In comparison to RNNs, TCNs are generally faster and require less memory for training than RNNs. Since filters are shared across layers, convolutions are done in parallel, which allows TCNs to process the input sequence as a whole. On the other hand, RNNs process the input sequentially and often require more memory to store partial results. Performance-wise, empirical evaluations from Bai *et al*. [2] showed that a TCN model was often able to achieve better results than RNN-based networks of similar size on various sequence modeling tasks.

FenceNet has the following advantages over JLJA:

- **Transferability to competition videos.** Requiring only 2D skeleton data as input allows FenceNet to be transferred and trained on competition videos, including cases where access to additional data from wearable sensors and depth videos are unavailable. This allows coaches and analysts to extract information from fencers from other competition groups, other countries, and the past.
- **Transferability to other techniques.** Actions in fencing are highly composite. For example, an attack usu-

ally consists of a long sequence of varying movements used to counteract and react to the opponent's movements. JLJA splits feature vectors into windows of 16 frames, which limits memory retention. In contrast, due to dilated convolutions, TCNs have access to substantially longer memory, allowing FenceNet to be trained to classify other techniques in fencing.

- **Simplicity and automation.** Unlike JLJA, FenceNet does not require manual feature extraction, feature selection, or feature fusion.

2. Related work

2.1. Computer vision in fencing

As described in Sec. 1, the majority of studies that involve computer vision applications in fencing were done by Malawski and Kwolek. In addition to their work on fencing footwork classification, they developed a model-based filtering algorithm for fencing footwork detection and segmentation on data acquired by a Kinect motion sensor [27]. In [26], Malawski proposed a method for blade tracking based on a single RGB camera and active markers using augmented reality.

Earlier work includes the analysis of the lunge movement from video capture data [4, 33]. Mantovani *et al*. classified weapon actions on kinematic data acquired from a motion capture system [31].

More recent work includes the fencing tracking and visualization system developed from Rhizomatiks' collaboration with Dentsu Lab Tokyo [40]. The system uses deep learning to detect sword tips without markers and real-time augmented reality synthesis to visualize the trajectory.

2.2. Skeleton-based action recognition in sports

Although end-to-end models [3, 7, 13, 21, 22, 45, 49, 52] dominate the literature in video action recognition, they are often more suited for coarse-grained classification tasks. Classification in sports are generally more fine-grained [43, 47]. Being able to classify subtechniques, such as different types of punches, is often more useful for analysis than distinguishing between a punch and a kick.

Skeleton-based methods have been a popular approach for fine-grained action recognition in sports. This method involves the use of 2D or 3D human pose as input in a human action recognition task. Representing the human skeleton as a graph with joint positions as nodes and modeling movement as the change of these graph coordinates over time allows us to capture both the spatial and temporal components of the action. Non-deep learning based approaches have been explored in sports such as wrestling [34] and Tai Chi [53].

In addition to improved performance, deep learning offers many advantages, such as automating feature engineering and feature selection. RNNs were one of the first networks used to model the temporal component of human actions. Long short-term memory (LSTM) [14] based RNNs specifically, were commonly used because traditional RNNs suffer from the vanishing/exploding gradient problem [14, 37]. Variants that incorporate graph convolutional networks (GCNs) such as GT-LSTM [20] and LSGM [15] were proposed to capture spatial information for skeleton-based action recognition.

More recently, TCN architectures have been used in place of RNN-based layers due to their ability to exhibit longer memory [2]. In table tennis, Kulkarni and Shenoy [18] used TCNs for stroke prediction and showed that their TCN model outperformed their LSTM model.

3. Fencing videos

FenceNet is trained on FFD, a publicly available fencing dataset that contains 10 intermediate to expert level fencers performing 6 types of footwork actions (lunges and steps) for 10-11 repetitions each in a practice setting, for a total of 652 videos. Czajkowski [11], known as one of the inventors of modern fencing theory, grouped the fencing lunge into 4 categories. The 6 total actions, with descriptions of the lunges by Czajkowski, are:

- *rapid lunge (R):* very fast, performed in relatively short distances.
- *incremental speed lunge (IS):* slow at beginning, accelerates during action, useful for feint attacks.
- *with waiting lunge (WW):* short pause in first stage of lunge while fencer observes reaction of opponent to counter-action.
- *jumping sliding lunge (JS):* fencer jumps forward with front leg to cover distance, back leg slides on the floor, common in complex offensive actions.
- *step forward (SF)*
- *step backward (SB)*

Figure 2. Skeleton data overlayed on depth data from FFD. Only 2D skeleton data was used for this study.

FFD contains 3D skeleton data and 640×480 16-bit depth data acquired by Kinect [1] at 30 Hz. 9 axis accelerometer, gyroscope, magnetometer, and orientation data

were captured by an x-IMU sensor at 256 Hz. For this study, we only use the x, y coordinates (see Fig. 2) of the skeleton data.

When processing the FFD files, we noticed that the skeleton data for the second repetition of *SF* for fencer 5[1] was empty and thus removed for this study.

4. FenceNet

4.1. Preprocessing phase

From Tab. 1 we see that different actions have different frame counts, with lunges being longer than steps. To allow for batch training, we sample 28 consecutive frames from each video with a random starting point from the beginning to the 20th frame. Each video is sampled at most 10 times (videos with fewer than 47 frames had less than 10 samples). We chose a window size of 28 since that is the minimum frame count for all videos.

	mean	std	min	25%	50%	75%	max
R	53.5	5.7	40	50	53	57	68
IS	65.1	8.8	49	58	64	72	98
WW	70.4	8.7	52	64	70	76	92
JS	69.9	9	51	62	70	75	98
SB	41.1	8.1	28	33	41	48	62
SF	44.2	9.8	29	37	42	50	80

Table 1. Summary of frame counts for each action in FFD.

Since *SF* and *SB* had significantly fewer frames per video than the lunge actions, sampling inevitably introduced some class imbalance to our data, as seen in Tab. 2. However, since the steps are the coarse actions, while the differences in motion among lunges are subtle, this actually helped our fine-grained action recognition performance via a form of data augmentation. This is discussed more in Sec. 5.3.

	Before sampling	After sampling
R	108 (16.7%)	1053 (18.3%)
IS	110 (16.8%)	1100 (19.1%)
WW	110 (16.8%)	1100 (19.1%)
JS	109 (16.7%)	1090 (18.9%)
SF	107 (16.4%)	761 (13.2%)
SB	108 (16.5%)	660 (11.5%)
Total	652	5764

Table 2. Video counts for each action before and after sampling.

For each sampled subsequence, we subtract the fencer's nose's position of the first frame from every joint coordinate in each frame. Then each joint coordinate in each frame is divided by the vertical distance between the head position and front ankle in the first frame. Letting $p_t^{j,c}$ be the position of joint j on the c axis during time step t, the scaled position $\tilde{p}_t^{j,c}$ is given by:

$$\tilde{p}_t^{j,c} = \frac{p_t^{j,c} - p_0^{N,c}}{p_0^{N,c} - p_0^{A,c}} \tag{1}$$

where N and A represent keypoints for the nose and front ankle, $0 < t \leq 28$, and $c \in \{x, y\}$.

We take the x, y coordinates of the front wrist, front elbow, front shoulder, both hips, both knees, and both ankles as inputs to our classification model.

4.2. Footwork classifier

FenceNet consists of 6 TCN blocks followed by two dense layers (see Fig. 1a). A TCN block (Fig. 1b) contains two stacked 1D fully-convolutional layers [24], each employing causal convolutions and dilated convolutions [56]. Causal convolutions (Fig. 3b) ensure there is no leakage of information from the future to the past, meaning predictions made at time t depend only on states during and prior to t. To achieve this, for time step t and kernel size k, we convolve from $t-k$ to t (as opposed to from $t-\frac{k}{2}$ to $t+\frac{k}{2}$ in the acausal case). Let $C_{causal}(\mathbf{p}, t)$ denote causal convolution at step t for input $\mathbf{p}$:

$$C_{causal}(\mathbf{p}, t) = \sum_{i=0}^{k-1} w(i) * \mathbf{x}_{t-i} \tag{2}$$

where $*$ is the cross-correlation operator and w is the filter.

Dilated convolutions allow us to exponentially increase receptive field size in different ways, such as by increasing the number of dilated convolutional layers, increasing the dilation factor, or kernel size. In contrast, the receptive field of regular convolution is linear to depth or kernel size (Fig. 3a). Thus dilated convolutions provide better control over model size and complexity which can reduce the risk of overfitting. Furthermore, dilated convolutions could be used in place of pooling and upsampling to better retain information between layers. Dilated convolutions are implemented by skipping a fixed gap between time steps. For example, given dilation factor d and kernel size k, when combined with causal convolutions (Fig. 3c), at step t we have:

$$C_{dilated\ causal}(\mathbf{p}, t, d) = \sum_{i=0}^{k-1} w(i) * \mathbf{p}_{t-d \cdot i} \tag{3}$$

[1] File name: *2016-01-09_12-51-53.Body.mat*

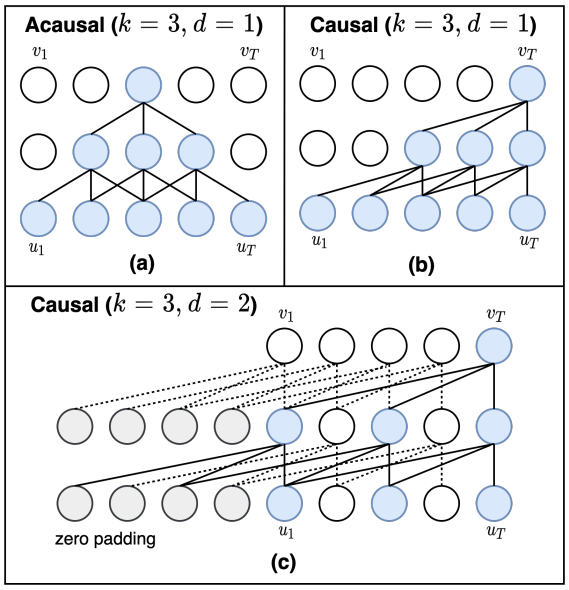

Figure 3. Given input sequence $\mathbf{u}$ and output sequence $\mathbf{v}$ (both of length $T = 5$), kernel size $k = 3$, and one hidden layer, an example of (a) normal convolutions. (b) causal convolutions. (c) dilated causal convolutions with dilation factor $d = 2$ and zero padding to ensure same length in each layer. Note: some connections in (a) and (b) are not drawn.

The convolutional layers are immediately followed by weight normalization [42], a rectified linear unit (ReLU) [35], and spatial dropout [44] to improve generalization. Lastly a residual connection is added between the input and output of each block to improve stability of the network. Given input $\mathbf{p}$ and output $f(\mathbf{p})$, this residual connection is simply:

$$\text{output} = \text{Activation}(\mathbf{p} + f(\mathbf{p})) \tag{4}$$

In the case that the input channel size could differ from the output channel size of the second convolutional layer, a 1×1 convolution is added to account for this discrepancy. Our structure is based on those used in [2, 25] but with increasing hidden size and decreasing kernel size as the number of layers increase. The increasing dilation factors are also adjusted to accommodate for the limited input length while maintaining full history coverage. To ensure each layer has the same length, zero padding is used.

From the output sequence of the last TCN block, we extract the last time-step and feed it into dense layers for prediction. Due to sampling in Sec. 4.1, for a given video, we have a predicted action for each subsequence of frames. We select the most commonly predicted action among the subsequences as our final predicted action. Details of the structure and parameters of the network can be found in Fig. 1a. These values were tuned using random search.

4.3. BiFenceNet

Causal convolutions ensure no information leakage into the future, which allows us to sequentially capture the forward motion of an action. Inspired by ELMo [38], we hope to capture "bidirectionality" of motion by using two separate networks. As seen in Fig. 4, we capture the forward motion of an action through a network of stacked TCN blocks, while feeding the reversed motion into a separate stack of TCN blocks, essentially creating a separate "anti-causal" network. The TCN blocks in the two networks are the same as in Fig. 1b. As in FenceNet, we extract the last time step from the output of each network. They are concatenated and fed into dense layers for prediction. To create BiFenceNet, we simply replace the footwork classifier seen in Fig. 1a with the bidirectional TCN-based module in Fig. 4.

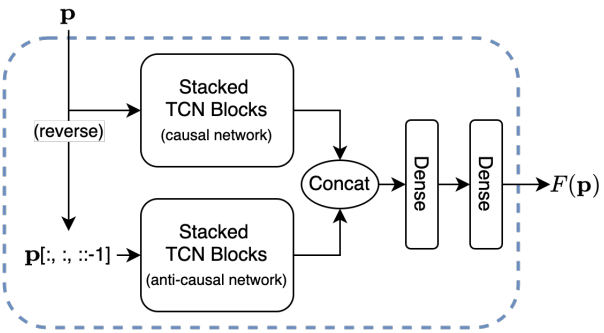

Figure 4. A bidirectional TCN-based module that replaces the footwork classifier (Fig. 1a) in BiFenceNet. The TCN blocks remain the same as in Fig. 1b.

5. Experimental results

5.1. Evaluation

Our model is trained and evaluated on FFD using 10-fold cross-validation, where in each fold, data from one fencer is taken out as the test set. This scenario provides a better representation of how the model generalizes to new fencers than randomly splitting the training and testing data. Malawski and Kwolek referred to this scenario as the person-independent (PI) case when evaluating JLJA. Since Malawski and Kwolek did not provide the train-test split for their random 5-fold cross-validation scenario, we omit the comparison for that case. The JLJA result displayed in this section is the top performing result from the various combinations of features and parameters used in [28, 29].

Under the PI case, FenceNet achieved a classification accuracy of 85.4%, within 1% of JLJA (86.3%), after training for 103 epochs for all 10 folds. BiFenceNet achieved a classification accuracy of 87.6%, outperforming JLJA after training for 94 epochs for all 10 folds, with 4 layers for each of the two stacked TCN blocks. In addition to

JLJA, Malawski and Kwolek also evaluated methods such as SkeletonNet [17], C3D [48], EigenJoints [55], HON4D [36], LOP/FTP [50], and MHI [5]. Comparisons of the methods can be found in Tab. 3 (note that results from all non-FenceNet methods were computed by Malawski and Kwolek in [29]). As mentioned in Sec. 3, we removed 1 of the 653 files from FFD for this study as we were unable to process the data in that file. However, since the removed file was an *SF* action, and we were able to separate this class well in both FenceNet and BiFenceNet (Tab. 4 and Tab. 5), we believe this one missing observation will not alter our results and that our results are still comparable to the other methods shown in Tab. 3.

Method	Accuracy %
JLJA [29] *	**86.3**
EigenJoints [55] *	29.9
MHI [5] *	61.3
SkeletonNet [17] *	64.4
C3D [48] *	67.6
HON4D [36] *	75.9
LOP/FTP [50] *	76.1
FenceNet (ours)	**85.4**
BiFenceNet (ours)	**87.6**

Table 3. Classification accuracy for the PI case (* results taken directly from Sec. 5 of [29]).

From the confusion matrices (Tabs. 4, 5, 6) we observe that, compared to JLJA, our methods are also better at distinguishing between coarse actions – steps from lunges. This could be useful in future action segmentation tasks for fencing matches.

For all three methods (Tabs. 4, 5, 6), we find *IS* to be the worse performing class, often mixed with *WW*. This is likely due to different fencers subjectively interpreting the two classes differently, as the two actions share almost identical motion trajectories but differ in speeds at different time points. Having some prior knowledge of the fencer can improve results. For example, during the random split case, where we randomly take out 20% of the repetitions for each action for each fencer as the test set, FenceNet achieves a significantly higher accuracy of 96.5% on the test set. JLJA's ability to better identify *IS* is likely due to its JMHC descriptor directly incorporating the change in depth image between consecutive frames to better capture the "incremental" change in speed of the movement, whereas in FenceNet, only skeleton data is used as input. This can cause FenceNet to misclassify "faster" *IS* repetitions as *R* and "slower" ones as *WW*, since different athletes perform techniques at different overall speeds. Future variants of FenceNet can explore incorporating the change in skeleton data as input. Furthermore, distinguishing *IS* and *WW* in a binary classification scenario could be a focus in future work. A hierarchical approach where *IS* and *WW* are treated as one class during initial classification and separated in the second stage as a binary case may improve overall accuracy.

	R	IS	WW	JS	SF	SB
R	**88.9**	7.4	1.9	1.9	-	-
IS	15.5	**51.8**	17.3	15.5	-	-
WW	0.9	15.5	**82.7**	0.9	-	-
JS	-	10.1	-	**89.9**	-	-
SF	-	-	-	-	**100**	-
SB	-	-	-	-	-	**100**

Table 4. Confusion matrix for FenceNet for the PI case (prediction accuracy **85.4%**).

	R	IS	WW	JS	SF	SB
R	**95.4**	0.9	-	3.7	-	-
IS	14.5	**59.1**	13.6	12.7	-	-
WW	1.8	14.5	**83.6**	-	-	-
JS	-	3.7	7.3	**89.0**	-	-
SF	-	-	0.9	-	**99.1**	-
SB	-	-	-	-	-	**100**

Table 5. Confusion matrix for BiFenceNet for the PI case (prediction accuracy **87.6%**).

	R	IS	WW	JS	SF	SB
R	**85.3**	12.0	1.8	0.9	-	-
IS	11.0	**71.6**	5.6	11.8	-	-
WW	4.6	18.2	**77.3**	-	-	-
JS	-	13.6	-	**86.4**	-	-
SF	-	-	-	-	**100**	-
SB	-	-	-	-	2.7	**97.3**

Table 6. Confusion matrix for JLJA for the PI case (prediction accuracy **86.3%**), taken from the top performing version in [29].

5.2. Causality

To investigate the effect of causality, we examine cases where the input sequence is reversed (anti-causal) and shuffled (acausal). FenceNet outperforming these cases (rows 3-4 of Tab. 7) provides evidence that the forward trajectory of motion contains useful information when distinguishing actions.

To investigate the effect of the additional network in BiFenceNet that aims to capture the reverse direction of movement, we replace the anti-causal network with another causal network. BiFenceNet outperforming this case (Tab. 7 row 5) provides evidence that the reversed motion trajectory contains additional information for distinguishing actions, and that the better performance from BiFenceNet is

	Parameters (10^6)	Prediction accuracy(%)	Class accuracies (%)					
			R	IS	WW	JS	SF	SB
FenceNet	2.6	**85.4**	89	52	83	90	100	100
BiFenceNet	5.4	**87.6**	95	59	84	89	99	100
FenceNet (reversed)	2.6	84.4	86	61	78	83	99	100
FenceNet (shuffled)	2.6	84.4	87	50	84	87	100	99
FenceNet (forward $\times 2$)	7.0	86.2	95	50	86	86	100	100
FenceNet (wide)	5.9	85.4	90	56	80	88	100	100
FenceNet (regular conv1D)	4.8	83.3	92	41	83	84	100	100
FenceNet (zero padding)	2.6	76.5	82	28	74	77	100	100
FenceNet (full body)	2.7	83.1	89	52	72	87	100	100
FenceNet (lower body)	2.6	82.4	77	60	72	90	99	97
LSTM	2.7	81.9	92	34	78	89	100	100
Bi-LSTM	5.5	83.1	93	39	80	88	100	100

Table 7. Experimental results. Rows 3-7 correspond to results in Sec. 5.2. Row 8 corresponds to Sec. 5.3. Rows 9-10 correspond to Sec. 5.4. Rows 11-12 correspond to Sec. 5.5. All methods were evaluated under the PI case.

not simply due to an ensemble effect. Different parameters and structures for the second causal network were tested and the best result was recorded. In Tab. 7 row 6, we compare BiFenceNet to a wider version of FenceNet by increasing channel size to show that the improved performance is not simply due to an increase in model size. In Tab. 7 row 7, we compare BiFenceNet to a version of FenceNet with only dilated 1D convolution layers and no causality to show that training the forward and reverse directions of the input sequence separately is different from using regular 1D convolutions. Instead of taking the last time step of the output from the last block, we flatten the last 1D convolution layer before feeding into the dense layers.

5.3. Sampling versus zero padding

During the preprocessing phase (Sec. 4.1), to ensure videos of the same length for batch training, an alternative to sampling would be to pad zeros to the end of shorter videos, as done in [18]. However, we chose to sample subsequences as this process simultaneously augments our training data.

Segmenting a sequence of movements into actions, even by manual cropping, is prone to error, and could lead to inconsistencies in defining the start and end of an action. Sampling subsequences of frames essentially augments the training data, and allows the model to better deal with this problem. From Tab. 7 row 8, we see that FenceNet outperforms the zero padding case significantly.

5.4. Keypoint selection

Despite the lunge being characterized as a lower body movement, we included the front wrist, elbow, and shoulder joint into our input. This is because Czajkowski described *IS* as often being associated with feint attacks and *WW* with counter-actions. We hypothesize that information extracted from the front arm could capture some of the aforementioned associations, and improve our lunge prediction. We compare this to using keypoints from the whole body (by including the nose and back arm), as well as only the lower body (both hips, both knees, both ankles). From rows 9-10 of Tab. 7, we see that both these cases perform worse than our original method, aligning with our hypothesis.

5.5. Model comparison

Lastly, Tab. 7 row 11 shows FenceNet outperforming an LSTM of similar size and Tab. 7 row 12 shows BiFenceNet outperforming a bidirectional LSTM of similar size. The training times for the LSTM and bidirectional LSTM were both more than 3 times that of FenceNet and BiFenceNet, respectively. These results align with observations from [2, 18], which state that TCNs often outperform LSTMs in sequential tasks.

6. Discussion

A limitation of FenceNet is its dependency on the quality of 2D pose input. Without a marker-based motion capture system, pose data extracted from off-the-shelf 2D pose estimators could be noisy or inaccurate, limiting the performance of FenceNet. To address this issue, future work can focus on the preprocessing step to obtain more robust inputs. This includes data augmentation, smoothing, pose normalization, or incorporating methods such as VIPE [46] to directly extract pose features from 2D input. The next step involves collecting high quality labeled fencing competition video data to test these methods to obtain more robust results. Competition video data also allows us to further explore different computer vision tasks in fencing, such as action segmentation and retrieval of more complex techniques.

Preliminary data collection and tagging on competition videos are currently being conducted with help from video analysts from the Canadian Olympic fencing team. We found actions from FFD not directly transferable to a competition setting. The footwork classes are highly imbalanced. For example, in the Grand Prix Turin 2020 Women's Foil Final[2], we tagged all 45 lunges (2266 total frames) executed by French fencer Ysarora Thibus according to the description in Sec. 3 and noticed that the frequency of the *JS* lunge dominates the other actions, as seen in Tab. 8.

Action	Count
Rapid lunge (R)	3 (6.7%)
Incremental speed (IS)	4 (8.9%)
With waiting (WW)	3 (6.7%)
Jumping sliding (JS)	35 (77.8%)

Table 8. Class frequency for each lunge performed by Ysarora Thibus in the Grand Prix Turin 2020 Women's Foil Final. The lunges are tagged according to the descriptions in Sec. 3.

Although there are multiple scenarios where Thibus adjusts the speed of her lunge as in *IS* or *WW*, they are almost always accompanied by a jumping, sliding motion in the lower body, as described by *JS*. She very rarely keeps her back foot still during the entirety of a lunge, which is the case for all non-*JS* lunges in FFD. For footwork recognition in a competition setting, we recommend creating more classes or modifying the definitions of the current ones, as techniques performed during competition seem to be much more complex and dynamic.

7. Conclusion

The current state-of-the-art fencing footwork recognition algorithm, JLJA, fuses information from skeleton data, depth videos, and IMU sensors to classify footwork techniques in fencing. However, the requirement of depth videos and wearable sensors imposes technological and physical difficulties on users, limiting the pool of athletes that they are able to analyze. To address this shortcoming, we introduce FenceNet, a fencing footwork recognition model that relies only on 2D pose data. FenceNet is a lightweight network that incorporates skeleton-based human action recognition with TCN architectures to capture both spatial and temporal components of human motions. Experimental results indicate that FenceNet's classification accuracy came within 1% of JLJA, while BiFenceNet was able to outperform JLJA, despite both only using 2D skeleton data as input. We hope FenceNet and future variants are able to contribute to automating the analysis in sports.

[2] https://www.youtube.com/watch?v=H-v6DfxnjF8

Acknowledgements. We acknowledge financial support from the Canada Research Chairs Program, the Canadian Sports Institute Ontario, and a Mitacs grant. We also acknowledge Fencing Canada for their help with data collection, tagging, and fencing expertise.

References

[1] Kinect Windows app development. Kinect for windows. 3

[2] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv:1803.01271*, 2018. 2, 3, 5, 7

[3] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *Proceedings of the International Conference on Machine Learning*, volume 139, 2021. 3

[4] Tadeusz Bober, Alicja Rutkowska-Kucharska, Sebastian Jaroszczuk, Maciej Barabasz, and Wojciech Woźnica. Kinematic characterisation of the lunge and the fleche in epee fencing: Two case studies. *Polish Journal of Sport and Tourism*, 23, 2016. 3

[5] Aaron F. Bobick and James W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23, 2001. 1, 6

[6] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43, 2021. 2

[7] João Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3

[8] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers and Electrical Engineering*, 40, 2014. 1

[9] Vasileios Choutas, Philippe Weinzaepfel, Jérôme Revaud, and Cordelia Schmid. Potion: Pose motion representation for action recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018. 2

[10] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20, 1995. 1

[11] Zbigniew Czajkowski. *Understanding Fencing. The Unity of Theory and Practice*. SKA Swordplay Books, 2005. 3

[12] James W. Davis and Aaron F. Bobick. Representation and recognition of human movement using temporal templates. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997. 1

[13] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 3

[14] Sepp Hochreiter and J J Urgen Schmidhuber. Long short term meomory (lstm). *MEMORY Neural Computation*, 9, 1997. 3

[15] Junqin Huang, Zhenhuan Huang, Xiang Xiang, Xuan Gong, and Baochang Zhang. Long-short graph memory network for skeleton-based action recognition. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 2020. 3

[16] Michael I Jordan. Serial order: A parallel distributed processing approach. *ICS Report*, 8604, 1986. 2

[17] Qiuhong Ke, Senjian An, Mohammed Bennamoun, Ferdous Sohel, and Farid Boussaid. Skeletonnet: Mining deep part features for 3-d action recognition. *IEEE Signal Processing Letters*, 24, 2017. 6

[18] Kaustubh Milind Kulkarni and Sucheth Shenoy. Table tennis stroke recognition using two-dimensional human pose estimation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2021. 3, 7

[19] Colin Lea, Michael D. Flynn, René Vidal, Austin Reiter, and Gregory D. Hager. Temporal convolutional networks for action segmentation and detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[20] Hongsheng Li, Guangming Zhu, Liang Zhang, Juan Song, and Peiyi Shen. Graph-temporal lstm networks for skeleton-based action recognition. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 12306 LNCS, 2020. 3

[21] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 3

[22] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. *arXiv:2106.13230*, 2021. 3

[23] Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, and Wanli Ouyang. Disentangling and unifying graph convolutions for skeleton-based action recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020. 2

[24] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015. 4

[25] Xianglong Luo, Wenjuan Gan, Lixin Wang, Yonghong Chen, and Enlin Ma. A deep learning prediction model for structural deformation based on temporal convolutional networks. *Computational Intelligence and Neuroscience*, 2021. 5

[26] Filip Malawski. Real-time first person perspective tracking and feedback system for weapon practice support in fencing. In *Frontiers in Artificial Intelligence and Applications*, volume 310, 2018. 3

[27] Filip Malawski and Bogdan Kwolek. Real-time action detection and analysis in fencing footwork. In *International Conference on Telecommunications and Signal Processing*, 2017. 3

[28] Filip Malawski and Bogdan Kwolek. Recognition of action dynamics in fencing using multimodal cues. *Image and Vision Computing*, 75, 05 2018. 1, 2, 5

[29] Filip Malawski and Bogdan Kwolek. Improving multimodal action representation with joint motion history context. *Journal of Visual Communication and Image Representation*, 61, 04 2019. 1, 5, 6

[30] Utthara Gosa Mangai, Suranjana Samanta, Sukhendu Das, and Pinaki Roy Chowdhury. A survey of decision fusion and feature fusion strategies for pattern classification. *IETE Technical Review (Institution of Electronics and Telecommunication Engineers, India)*, 27, 2010. 1

[31] G. Mantovani, A. Ravaschio, P. Piaggi Pa, and A. Landi. Fine classification of complex motion pattern in fencing. In *Procedia Engineering*, volume 2, 2010. 3

[32] William McNally, Kanav Vats, Alexander Wong, and John McPhee. Rethinking keypoint representations: Modeling keypoints and poses as objects for multi-person human pose estimation. *arXiv:2111.08557*, 2021. 2

[33] Kevin C. Moore, Frances M.E. Chow, and John Y.H. Chow. Novel lunge biomechanics in modern sabre fencing. In *Procedia Engineering*, volume 112, 2015. 3

[34] Ali Mottaghi, Mohsen Soryani, and Hamid Seifi. Action recognition in freestyle wrestling using silhouette-skeleton features. *Engineering Science and Technology, an International Journal*, 23, 2020. 3

[35] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the International Conference on Machine Learning*, 2010. 5

[36] Omar Oreifej and Zicheng Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013. 6

[37] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, 2013. 3

[38] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, June 2018. 5

[39] Juha Pärkkä, Miikka Ermes, Panu Korpipää, Jani Mäntyjärvi, Johannes Peltola, and Ilkka Korhonen. Activity classification using realistic data from wearable sensors. *IEEE Transactions on Information Technology in Biomedicine*, 10, 2006. 1

[40] Rhizomatiks. Fencing tracking and visualization system. 3

[41] D.E Rumelhart, G.E Hinton, and R.J Williams. Learning internal representations by error propagation. *Explorations in the Micro-Structure of Cognition Vol. 1 : Foundations*, 1986. 2

[42] Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*, 2016. 5

[43] Dian Shao, Yue Zhao, Bo Dai, and Dahua Lin. Finegym: A hierarchical video dataset for fine-grained action understand-

ing. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020. 3

[44] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15, 2014. 5

[45] Swathikiran Sudhakaran, Sergio Escalera, and Oswald Lanz. Gate-shift networks for video action recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020. 3

[46] Jennifer J. Sun, Jiaping Zhao, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, and Ting Liu. View-invariant probabilistic embedding for human pose. In *Proceedings of the European Conference on Computer Vision*, 2020. 7

[47] Shan Sun, Feng Wang, Qi Liang, and Liang He. Taichi: A fine-grained action recognition dataset. In *Proceedings of the ACM International Conference on Multimedia Retrieval*, 2017. 3

[48] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015. 6

[49] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann Lecun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018. 3

[50] Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012. 6

[51] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43, 2021. 2

[52] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9912 LNCS, 2016. 3

[53] Leiyang Xu, Qiang Wang, Lin Yuan, and Xiang Ma. Using trajectory features for tai chi action recognition. In *Proceedings of the International Instrumentation and Measurement Technology Conference*, 2020. 3

[54] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018. 2

[55] Xiaodong Yang and Yingli Tian. Effective 3d action recognition using eigenjoints. *Journal of Visual Communication and Image Representation*, 25, 2014. 6

[56] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *Proceedings of the International Conference on Learning Representations*, 2016. 4