Real Time Fencing Move Classification and Detection at Touch Time during a Fencing Match

Cem Ekin Sunal, Chris G. Willcocks, Boguslaw Obara
Department of Computer Science, Durham University, Durham, UK

***Abstract*—Fencing is a fast-paced sport played with swords which are Épée, Foil, and Sabre. However, such fast-pace can cause referees to make wrong decisions. Review of slow-motion camera footage in tournaments helps referees' decision-making, but it interrupts the match and may not be available for every organisation. Motivated by the need for better decision-making, analysis and availability, we introduce the first fully-automated deep learning classification and detection system for fencing body moves at the moment a touch is made. This is an important step towards creating a fencing analysis system, with player profiling and decision tools that will benefit the fencing community. The proposed architecture combines You Only Look Once version three (YOLOv3) with a ResNet-34 classifier, trained on ImageNet settings, to obtain 83.0% test accuracy on the fencing moves. These results are exciting development in the sport, providing immediate feedback and analysis along with accessibility, hence making it a valuable tool for trainers and fencing match referees.**

***Index Terms*—Deep Learning, Fencing, Convolutional Neural Networks, Computer Vision and Supervised Learning.**

I. Introduction

Fencing is an individual sport which is played by two people who are called fencers. The aim is to reach a certain score by performing valid moves with swords. Fencing has three sword branches that are called Épée, foil and sabre; and these branches have different set of valid moves, playing speed and hitting rules. The players act fast while trying to reach their opponents. Therefore, their fastness poses a problem for referees and trainers to decide which fencer should get the score and/or track their moves to get insight.

A. Background

The usage of technology is crucial in the Fencing sport. In the early 1900s, wrong decisions and sometimes cheating took place. Even Aldo Naldi, old famous fencer, complained about the cheating in his autobiography: *"well-known fencers were often given the benefit of mistakes (so-called 'reputation touches'), and in some cases there was outright cheating"* [1].

To prevent this, new equipment was introduced to reduce the wrong scores. *"In 1933, side judges were replaced by the Laurent-Pagan electrical scoring apparatus."* [2] Currently, in modern fencing, slow motion cameras are used in international matches to help referees' decision-making. However, not all tournaments/sport centres can invest in this expensive hardware. There is also a lack of research to address these problems.

In fencing, an electronic circuit is completed on contact where two lights are used to distinguish players, Green and Red which light up when a fencer touches the opponent with a sword. Moreover, the White lamp is used only in foil to indicate "a touch has been made to invalid area". Some international tournaments provide real-time overlays to show the results and lights, where this project aims to harness this signal information to detect touches. This is preferred over vision-based touch detection, due to occlusion of the touch, where the electronic circuit is considered more accurate.

In this paper, the aim is to detect and classify the touches that are performed at touch time. As the final moves are important decision makers, the three main important move types are considered to be the Counter Attack, Lunge, and Preparation-To-Attack [3].

'Lunge' is the most common type of offensive move which is a powerful long forward jump that allows the player to reach the opponent and finish the attack. As it's a move to get a touch, it will happen in the touch time.

However, if the arm is not aimed to reach the opponent (or delayed while reaching compared to opponent or its improper), it is counted as 'preparation-to-attack'. Given that these moves frequently happen in touch time during successful/failed attacks, it was chosen to distinguish between Lunge and failed/delayed Lunges. Choosing this move as a class will help deep learning architectures to generalize 'lunge' well.

Thirdly, 'Counter Attack' was chosen. Counter attack moves are performed during the opponent's attack time. If opponent touches the player during the other's 'counter attack', and if no parry or evasive moves were performed, the opponent gets the point. This is one of the most common touch time defensive move in fencing. However, it is the only one that is performed during touch time, as it tries to reach the opponent. Moreover, in terms of posture, preparation to attack is close to counter attack. Thus, choosing both will prevent false positives.

Lastly, we introduce two new categorisations. 1) 'Not A Fencer', which distinguishes non-participants such as referees, and audience members. 2) 'Not A Valid Move', which distinguishes situations where an electronic circuit is completed but invalid contact is made, for example when a fencer is checking their kit. These both also help our method avoid false positives.

Deep Learning (DL) is the subcategory of Machine Learning, where multiple network layers learn to capture high-level features to solve difficult tasks such as in Computer Vision, Autonomous Driving, Image/Video Understanding, and Medical Imaging [4].

In world-wide applications, Computer Vision (CV) and

Image Processing (IP) techniques are used to process the visual inputs to have meaningful outputs/decisions. CV and IP's main usage areas include classification, feature extraction, and pattern recognition which are essential components for our aim.

Convolutional Neural Networks (CNNs) are well-known to be state-of-the-art in CV, due to their sharing of parameters and computational efficiency. CNN layers respond to different types of visual inputs, and can detect features non-linearly based on their activation. This allows efficient classification of the fencer's move based on their posture. The common CNN models are AlexNet and ResNet, which act like a prefrontal cortex for the computer, responding to objects based on their visual features.

B. Contributions

With the objective of building an accessible fencing move analysis system, this paper makes the following contributions:

1) We design and propose a modular architecture that is able to detect and classify fencer moves at touch time with 83.0% accuracy on unseen tournament footage.
2) We provide some preliminary fencing analysis using results obtained of our system, in particular we surprisingly find that both winners and losers generally have similar distributions of moves, although more analysis is needed to be definitive on this.
3) We found the combination of YOLOv3 and ResNet-34, using an auxiliary image processing function to retrieve the fencing circuit signal, to be effective in this computer vision application setting.

II. Related Work

As fencing is an action based sport, any work that focuses on players' body moves can be counted as previous attempt.

Support Vector Machines (SVM)'s usage on "lunge" move classification with lunge's velocity was proposed. However, the paper covers only one move and one person. Besides, it may be hard to use with match videos with high noise [5].

Pose estimation is also another method to classify the body postures like *Real-time Multi-Person 2D Pose Estimation* where it produces 2D joint points for human bodies' key areas and uses CNN to predict the 2D confidence maps of these key areas. Consequently, it encodes the degree of association between these key areas and creates 2D vectors per limb. Hence, combining all the information together to estimate the pose. [6] Real-time multi-person pose estimator is a closely related topic to this paper. However, this paper aims to classify fencing moves rather than recreating the body image.

Another paper which uses Artificial Neural Networks on fencing proposes a top-down solution to the addressed common problems such as human detection, pose estimation and fencing moves. The paper suggests using data mining to extract move probabilities of Egyptian fencing team, and uses high tech tools with Kinect and Arduino to correctly determine the fencer's move [7]. However, this paper focuses on more cost efficient solutions that can be easily obtained and maintained by fencing tournament officials or trainers instead. As the crucial fencing matches (e.g international tournaments finals) are recorded with professional cameras, the proposed solution can easily classify fencers' moves in real-time by using a simple 720 pixel camera and a computer.

Sports summarization and move classification is another issue. A paper on user-generated video summaries with move classification argues that exploiting players' body joints does miss the important features on proper detection and promotes usage of CNN that extracts "holistic action recognition" [8]. However, using Kendo (Japanese fencing) instead of fencing as a target sport which leaves an open research area in fencing to explore, although this paper benefited the idea of using CNN for classification.

Datasets for DL problems are as important as the implementations. However, it was found that there is no public fencing dataset available. Although there are many fencing videos, they are either not annotated for research or do not include the video overlays. Besides, despite the presence of sports datasets like ImageNet's fencing dataset, Stanford 40 action dataset or PASCAL VOC Action Images, they do not specifically contain the fencers' moves but only the general concept of fencing. Therefore, we manually annotated videos with overlays for training. **CNN** architectures that specialize in human detection and classification such as YOLOv3 and ResNet were investigated to build the system. YOLO [9], the older version of YOLOv3 [10], has a single CNN pipeline to concurrently extract object features rather than using Sliding Window approach that uses a disjoint pipeline to the same task but with longer time and less accuracy. Due to having YOLOv3 higher accuracy and lower inference time compared to other larger architectures [10], it can benefit the proposed model in fencer detection. However, YOLOv3 generally focuses on autonomous car's pedestrian detection or other custom object detectors with multi-box annotations. Therefore, the lack of fencing dataset led to find alternative models that can be trained with single object datasets such as AlexNet and similar models. AlexNet was the first deeper CNN model that won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012 [4, p11]. However, the vanishing gradient problem impacts convergence of deep networks leading to accuracy saturation and degradation [11]. Therefore, residual blocks are used to retain strong gradients with a smoother optimization landscape, as with ResNets.

Predictive analysis of fencers such as winner or future moves from the data obtained from the model was one of the model's potential features. However, a paper on the difficulties of sport prediction stated that empirical and theoretical studies show that one cannot avoid the unpredictability of the sport, although accurate prediction is the 'holy grail' of the several sectors including finance and arts [12]. Therefore, it was deemed outside the scope and removed.

III. Method

The proposed method is split into three easy-to-solve subproblems, discussed in the three subsections accordingly: (1)

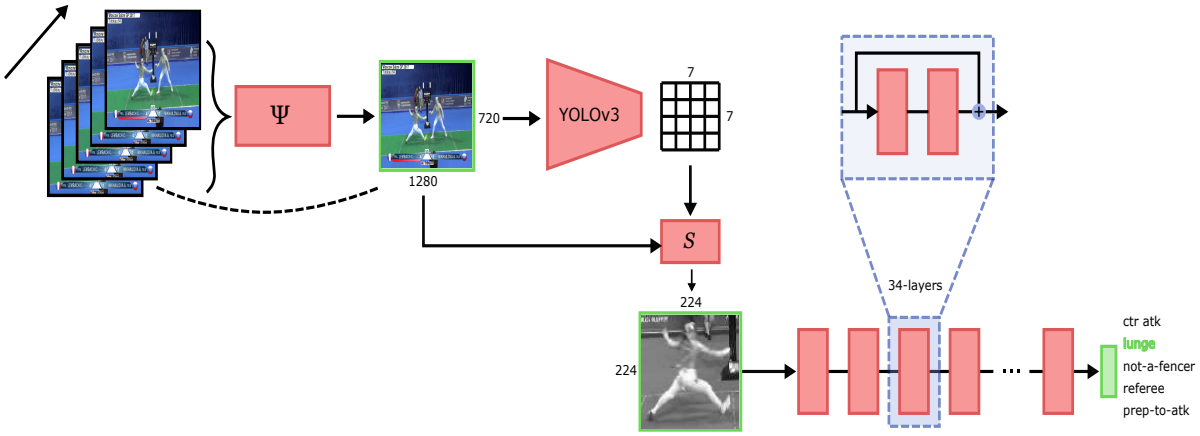

Fig. 1. The proposed architecture initially extracts contact frames using the colour function Ψ, then detects and crops players before the final classification using a residual architecture.

Touch-time detection, given live video. (2) Person detection, given touch-time-only images. (3) Classification, given cropped images of people at touch-time.

A. Touch-time detection

The objective is to classify in real-time live fencing videos. When touch is made an electronic circuit is completed; ideally we would have direct access to this supervisory signal, however for simplicity we extract it directly from the interface overlaid on the input fencing video stream. Therefore, we use a simple function $\Psi(\mathbf{v}) \rightarrow \{0, 1\}$ that takes an input a frame $\mathbf{v}$ from the set of video frames $\mathbf{v} \sim \mathcal{V}$ and, using simple colour image processing, extracts the notification of contact directly from the video overlay feed. This means we have a new dataset $\mathcal{T} \subset \mathcal{V}$ representing only the image frames where touch (contact) is made (Figure 2A).

B. Person detection

People are easy objects to detect using off-the-shelf object detectors, as there exists a surplus of publicly available imaging data of people (the data distribution p_{data}) and excellent available high-performing 'person detectors'. In particular, we use YOLOv3, which estimates the probability of a person $p(\text{person} \mid \mathbf{x})$ where $\mathbf{x} \sim p_{\text{data}}$, with predictions:

$$\begin{aligned} b_x &= \sigma(t_x) + c_x \\ b_y &= \sigma(t_y) + c_y \\ b_w &= p_w e^{t_w} \\ b_y &= p_h e^{t_h} \end{aligned} \tag{1}$$

where the YOLOv3 network $Y(\mathbf{x})$ takes $\mathbf{x} \sim \mathcal{T}$ as input, and outputs a feature map consisting of cells, offset by (c_x, c_y) which estimate k box coordinates t_x, t_y, t_w, t_h of prior width p_w, p_h. These are then fed into our sampling function $\mathcal{S}(Y(\mathbf{x}), \mathbf{x})$ which, using the $Y(\mathbf{x})$ outputs, crops the original image $\mathbf{x}$ when confident people are detected. This gives n colour 224^2 cropped images of each person labeled $\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n$ accordingly, where labels are determined simply by the box x-centers. We choose two most likely boxes to be fencers' boxes based on their horizontal alignment. This works, as it is considered an illegal move when players cross x-axis positions in fencing. From our sampling function $\mathcal{S}(Y(\mathbf{x}), \mathbf{x})$, we now have another distribution of player-labelled $(\mathbf{y}_i)$ cropped images $\mathcal{C}$ at touch-time (Figure 2B).

C. Fencing Move Classification

The fencing move classification is now a simple task of estimating the probability of a fencing move, given the cropped image and player identification label $\mathbf{y}$: $p(\text{move} \mid \mathbf{x}, \mathbf{y})$ where $(\mathbf{x}, \mathbf{y}) \sim \mathcal{C}$. Moves can be any one of {counter lunge, invalid, referee, prep} represented as a one-hot-encoding and trained in a supervised way, where the classifier estimates moves $m = M(\mathbf{x})$ set to categorical probabilities $\hat{p}$ by the softmax:

$$\hat{p}_i = \frac{e^{m_i}}{\sum_{j=1}^{K} e^{m_j}} \tag{2}$$

and the classifier M objective is standard cross entropy with the ground-truth labelled target moves p:

$$\mathcal{L}_{\text{M}}(\hat{p}, p) = -\sum_i p_i \log(\hat{p}_i) \tag{3}$$

Given that, at inference, we know which $\mathbf{x} \sim \mathcal{C}$ corresponds to which player $\mathbf{y}$, we know which player performed which move (Figure 2C). In the end, the results are output (Figure 2D).

D. Data Gathering and Annotating

The main dataset used has images of fencers performing the chosen moves. The fact that there is no appropriate fencing move dataset other than fencing videos pushed us to form our own move dataset from the raw videos. Most international tournaments make their match videos available to public, so the videos from [13] are used. To form the dataset, videos were processed to have square cropped regions of the fencers, which needed to be manually annotated. Images were also resized to RGB $3 \times 224 \times 224$ pixels to work with existing pre-trained models.

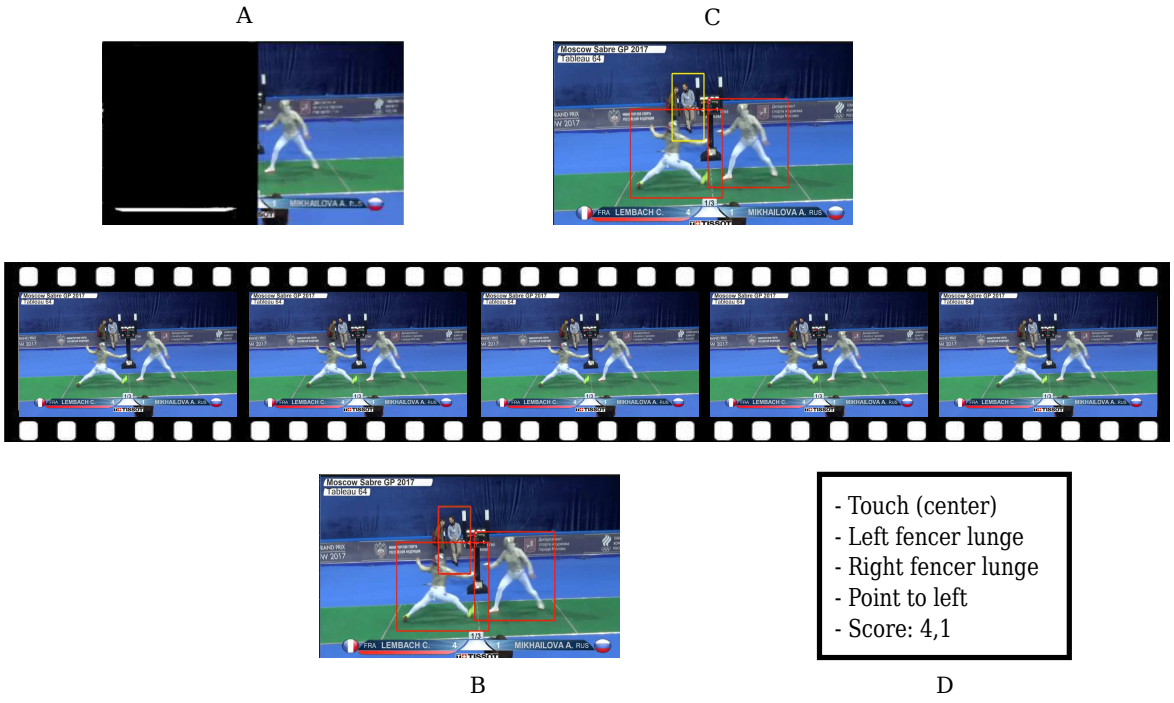

Fig. 2. Move classification of a single round. Initially the notification of contact is extracted from colour information in the video overlay feed. YOLOv3 estimates object proposals, and our residual architecture estimates the move and keeps track of the final scores. Further details in Section III.

IV. Implementation

The following parts detail the implementation of our touch time fencer detection and classification.

1) Touch Time Detection: In fencing, after touch has been made, the game stops and referee declares the score of that round. Now with modern technology, we can understand it just by looking at red and green lights and slow motion camera footages. Therefore, to build our system, overlay information was used for touch detection such that we don't require any auxiliary hardware setup. Given that red and green lights are clearly shown at bottom of screen, Hue/Saturation/Value (HSV) boundary filters were used. Their good performance on invariant objects [14] like score overlays and intuitiveness on people's RGB color experience [15] made an excellent solution. As piste colors are white and shadows have dark colors, clear saturated colors like red/green can easily be distinguished. Therefore, this enabled the algorithm to detect bright values (in 230-256 "Value" range) and saturated non-white areas (in 100-256 "Saturation" range). Applied filters converted the frame into binary images to detect light changes easily. Moreover, the region of interest (ROI) was limited to overlay area to avoid interference. The final classification using this approach did not have any false-positives in testing.

Figure 2A, Figure 3 and Figure 4 show the before and after version filter application in ROI to detect touches. White lights detection system was not developed, because white light resembles touches that were made to invalid area rather than invalid touch itself. Therefore, they do not affect classification of moves.

Example of touch time: both sides have touched each other.

Fig. 3. RGB version of the ROI. Original image is from [13] FE F S Individual Moscow RUS Grand Prix 2017 T32 03 red WOZNIAK USA vs BAKASTOVA UKR.

Fig. 4. HSV binary filter version of ROI. Original image: [13] FE F S Individual Moscow RUS Grand Prix 2017 T32 03 red WOZNIAK USA vs BAKASTOVA UKR.

2) Extraction of fencers from the frame into separate images: In the second part, the extraction of fencers into separate images from the captured screen was implemented. Although standard image processing techniques, Histogram of Oriented Gradients (HOG) methods were tried to simplify the process, YOLOv3 was found to be the best performing method for the cause. One of our sub-aims was to make the algorithm work in real time. As YOLOv3 was found to be efficient in the application of self-driving cars [16], we found that extrapolating the pedestrian detection problem for fencer detection would simultaneously solve both problems.

As YOLOv3 divides the image into grids, create bounding boxes and predicts their objectness score with logistic regression [10], it can easily detect multiple objects such as fencers in one large frame. Therefore, images obtained from touch

detector can be fed to YOLOv3, retrieving object proposals. YOLOv3 was implemented from [17].

Proposals which have "person" label and are within a 120-220 grey intensity range were separated to get prospective fencer proposals. The grey filter is an effective thresholding heuristic as Fencers' standard grey protective clothing makes them easy to detect. The final filtered images were resized to 224x224 pixel and saved for annotation in Greyscale (Figure 2B).

3) Data Augmentation: Several data augmentation methods were applied to improve generalisation, and we found that both horizontal flipping and random cropping to be effective. In particular, horizontal flips are suitable because left and right fencers can do the same move.

4) Hardware & Software Tools: For accessibility, a consumer grade setup was used. A computer with `OpenCV`, `PyTorch` and `Pandas` libraries and a mid-range GPU (with CUDA support) and either a camera that sends the frames to computer or pre-recorded videos are sufficient to run the program. The current development setup is Intel 7700HQ CPU, 16GB RAM, GeForce GTX 1050 4GB RAM Graphics Card.

V. Results

In this section, we present the training/testing results for several residual architectural variations, which are known to have state-of-the-art performance in detection and classification [18], [19]. The model was also evaluated for both greyscale and RGB inputs, and we also tried a reduced Canny edge detector representation to evaluate whether the model was overfitting to background information.

A. Training hyperparameters

The model was trained for 50 epochs, where each training per dataset/model was repeated 10 times giving means and standard deviations. The training dataset and the testing dataset were sampled from different distribution (different tournaments in different venues) where we have equal amounts of data for training and testing.

For RGB color space, we trained ResNet-18 and ResNet-34. Each model was trained on augmented over 10000 fencing move images, $\eta = 0.01$ learning rate, and we used the Adam optimization algorithm [20]. We used a batch size of 32 images per iteration with 5 workers.

The Figures 5,6,7,8 and 9's Y-axis of right hand size is the validation accuracy, Y-axis of left hand size is the training loss. X-axis is the epochs.

The final accuracy and confusion matrix is shown in Table 10. We observed that our model can identify Counter-Attack and Lunge properly, but does not perform so well on Preparation-to-Attack. It is also satisfying to see that the model well-identifies referees and distinguishes 'Not Valid Moves' to eliminate False Positives, which would hinder usability of the solution.

Table I shows the final reported test accuracies on the unseen tournament venues. Results indicate the model overfits

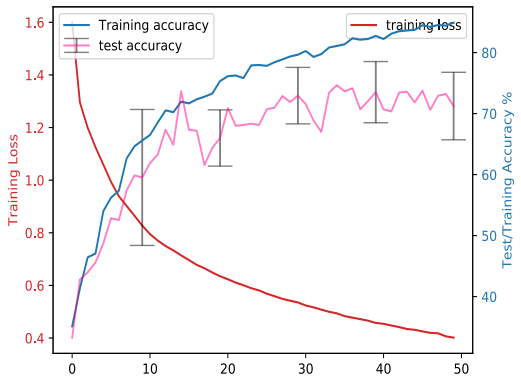

Fig. 5. Mean Train/Test accuracy of ResNet-18 on RGB dataset over 10 runs, with error bars as standard deviation. The x-axis shows the training Epochs.

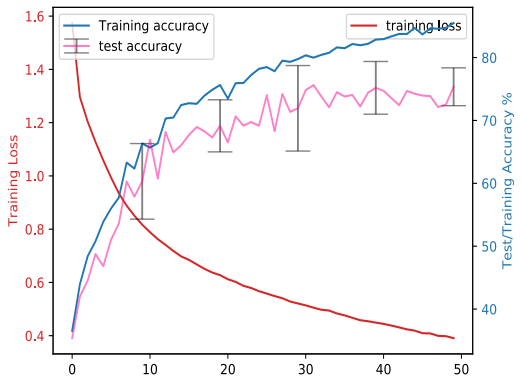

Fig. 6. Mean Train/Test accuracy of ResNet-34 on RGB dataset.

to some background colour information, where training with ResNet-34 only on Greyscale inputs was found to give the best generalisation performance. Given that there is no linear mapping between HSV and RGB, we experimented in HSV although found the results to be worse.

TABLE I
TEST ACCURACY ON UNSEEN TOURNAMENT VENUES. WE SHOW THE MEANS AND STANDARD DEVIATIONS BEST TEST ACCURACY FOR THE MODEL TRAINED TEN TIMES, AND THE SINGLE BEST TEST ACCURACY.

Model/Color	Mean/Std test acc	Best test acc
ResNet-18 RGB	$79.7\% \pm 1.8\%$	84%
ResNet-34 RGB	$81.6\% \pm 2.4\%$	86%
ResNet-34 HSV	$77.2\% \pm 3.0\%$	81%
ResNet-34 Gray	$\mathbf{83.0\% \pm 1.3\%}$	**86%**
ResNet-34 Edge	$77.8\% \pm 1.5\%$	81%

Figure 10 shows the Confusion Matrix for the best generalising ResNet-34 Greyscale model that was obtained during the training, with the 86% test accuracy. Despite three main classes introduced earlier, five classes are shown to prevent false positives for YOLOv3 model's human detections. "NotValidMove" represents fencers who do not perform any moves despite they have lighted up the score board.

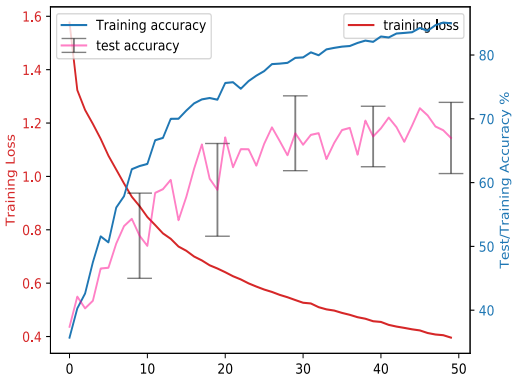

Fig. 7. Mean Train/Test accuracy of ResNet-34 on HSV dataset.

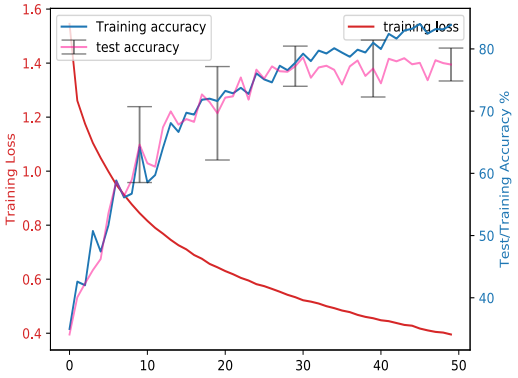

Fig. 8. Mean Train/Test accuracy of ResNet-34 on Grayscale dataset.

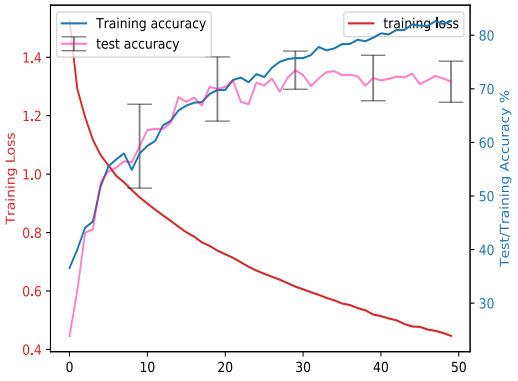

Fig. 9. Mean Train/Test accuracy of ResNet-34 on Canny Edge dataset.

B. Fencing Data Analysis

The data obtained from the fencing matches is presented in the following tables. The "Not-a-fencer" class data is omitted, as it does not have statistical importance. Over 300 videos are analysed in creation of the following Table II, where we found the move distribution is roughly equal in fencing matches. The reason that loser column is nearly half of the winner column is because loser side is eliminated from the subsequent matches.

From Table III, we can see that most of the touches have been made in the centre of the piste. The reason is thought to be that simultaneous attacks are performed more than normal attacks that pushed the fencers to the end of the piste.

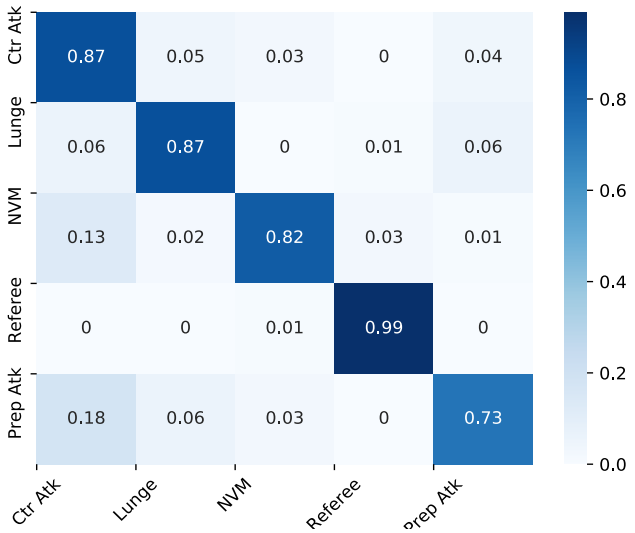

Fig. 10. Confusion Matrix for the best performing model, ResNet-34 Gray, on the Test Dataset. 'NVM' is the 'Not A Valid Move' class.

TABLE II
FENCING MOVE DISTRIBUTION FOR WINNER AND LOSER

Winner			Loser		
Ctr-atk	Lunge	Prep-to-atk	Ctr-atk	Lunge	Prep-to-atk
1237	2592	236	832	1729	184

TABLE III
LOCATIONS OF THE PISTE THAT TOUCH HAS BEEN MADE

Left	Center	Right
1078	6311	710

From Table IV, we found that Lunge is dominating other touches. As the touches at centre locations are also common due to simultaneous attacks, the data in Table 2 supports Table 3 too. Thus, by using ratios we can understand the proportion and the probability of fencer moves.

TABLE IV
FREQUENCY OF FENCING MOVES IN THE TOUCH TIME

Ctr-Atk	Lunge	Prep-to-Atk
5464	10935	1219

It can be seen that some numbers are not adding up properly when comparing with other tables due to null data problem when YOLOv3 could not detect the fencers properly. However, their correctness can be shown that the proportion of the moves in Table 3 31.0%, 62.1%, 6.9% respectively whereas in Table 1 they are 30.4%, 63.8%, 5.8% and 30.3%, 63.0%, 6.7%.

C. Model Performance

On mid-range hardware, the touch time component takes less than 50ms for its decision. The YOLOv3 fencer detection component takes ∼0.2s, and the final ResNet-34 classification takes ∼0.2s. In total, this whole process takes ∼0.5s to complete, however this is calculated asynchronously to the video footage. Therefore, decisions are available in reasonable time to when analysis is needed.

VI. Discussion

In this section, the strengths and limitation are explained.

A. Strengths

The strengths of our approach are in its simplicity and modular reliance on existing publicly available and pretrained Deep Learning networks, such as YOLOv3 and Residual architectures. This makes it easy to extend and improve the individual components accordingly. For example, in the future, should the video overlay touch-time graphical interface used in tournament footage change, this component can be simply updated and replaced without requiring a full end-to-end retraining of our network.

B. Challenges & Limitations

The main challenge of the project has been the lack of an appropriate high-quality public annotated fencing dataset. There was a need to create a new specific dataset, where dataset creation took considerable amount of time. Besides, during annotation, some images may be wrongly annotated due to human performance, as sometimes the context may change the move's nature. Moreover, knowing that the trained models are never perfect, the statistical analysis that are created from the application may have slight deviations from reality.

Additionally, although pre-trained state-of-the-art object detections like YOLOv3 was used in fencer detection, some fencers were not always detected. For example, in some match videos, the fencers moved out of the frame during touch time. As YOLOv3 cannot detect something that is not in the frame, this problem caused null data in the CSV files that are used for our statistical analysis.

C. Availability

The model and our dataset are made publicly available at: https://github.com/CodLiver/RT-Fencing, released under the MIT licence.

VII. Conclusions

In conclusion, we found that a modular architecture combining YOLOv3 and ResNet-34 gives an excellent estimate of fencer body moves at touch time. Despite limitations with our manually annotated dataset, we were surprised that the model performs over 83.0% accuracy with an equal distribution of move frequencies regardless of who was winning. This indicates that the approach will be a useful fencing match analysis tool for anyone interested in the sport, which will find benefit especially by trainers and referees. In the future, we would like to extend its capabilities and deploy it in a production-grade software package for real-time use, where we hope that the use of Deep Learning technology in fencing will continue to increase and benefit the fencing community.

References

[1] A. Nadi, L. C. Lobo, and W. M. Gaugler, *The living sword*. Laureate Press, 1995.

[2] M. Alaux, *Modern fencing: Foil, epee, sabre from initiation to competition*, 1st ed. Macmillan Pub Co, 1975.

[3] T. Lee, *Beginner Fencers' Guide Version 2018*. Metro Tacoma Fencing Club, 2018.

[4] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, M. Hasan, B. C. V. Esesn, A. A. S. Awwal, and V. K. Asari, "The history began from alexnet: A comprehensive survey on deep learning approaches," *Computing Research Repository (CoRR)*, vol. abs/1803.01164, 2018.

[5] F. Malawski and B. Kwolek, "Real-time action detection and analysis in fencing footwork," in *International Conference on Telecommunications and Signal Processing*, 2017, pp. 520–523.

[6] Z. Cao, G. H. Martinez, T. Simon, S. Wei, and Y. A. Sheikh, "Openpose: Realtime multi-person 2D pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2019.

[7] A. M. Abdel-Mawgoud, "The appliance of neural networks in fencing sport," Ph.D. dissertation, Department of Information Technology, Cairo University, 06 2015.

[8] A. T. de Pablos, Y. Nakashima, T. Sato, N. Yokoya, M. Linna, and E. Rahtu, "Summarization of user-generated sports video by using deep action recognition features," *IEEE Transactions on Multimedia*, vol. 20, no. 8, pp. 2000–2011, 2018.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[10] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *Computing Research Repository (CoRR)*, vol. abs/1804.02767, 2018.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[12] R. Y. S. Aoki, R. M. Assunção, and P. O. S. V. de Melo, "Luck is hard to beat: The difficulty of sports prediction," *Computing Research Repository (CoRR)*, vol. abs/1706.02447, 2017.

[13] "Fencing vision," youtube playlists was accessed on 10/10/2018. [Online]. Available: https://www.youtube.com/channel/UCA40s4GODjkaJ9JeM0fBcdg/

[14] S. Arivazhagan, R. N. Shebiah, S. S. Nidhyanandhan, and L. Ganesan, "Fruit recognition using color and texture features," *Journal of Emerging Trends in Computing and Information Sciences*, pp. 620–624, 2010.

[15] S. Kolkur, D. Kalbande, P. Shimpi, C. Bapat, and J. Jatakia, "Human skin detection using rgb, hsv and ycbcr color models," in *International Conference on Communication and Signal Processing*, 2016, pp. 324–332.

[16] Z. Cui, L. Heng, Y. C. Yeo, A. Geiger, M. Pollefeys, and T. Sattler, "Real-time dense mapping for self-driving vehicles using fisheye cameras," *Computing Research Repository (CoRR)*, vol. abs/1809.06132, 2018.

[17] Y.-S. Yun, "pytorch-0.4-yolov3," https://github.com/andy-yun/pytorch-0.4-yolov3, 2018.

[18] S. Akcay, M. E. Kundegorski, C. G. Willcocks, and T. P. Breckon, "Using deep convolutional neural network architectures for object classification and detection within X-ray baggage security imagery," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 9, pp. 2203–2215, 2018.

[19] S. Bianco, R. Cadene, L. Celona, and P. Napoletano, "Benchmark analysis of representative deep neural network architectures," *IEEE Access*, vol. 6, pp. 64 270–64 277, 2018.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 2014.