Batter Segmentation for Baseball Strike Zone Detection Using a Fully Convolutional Neural Network

Luis Bolanos[*]
Department of Computer Science
The University of British Columbia
luisb@mail.ubc.ca

Julian Ding[†]
Department of Computer Science
The University of British Columbia
julianzding@gmail.com

Brandon Dos Remedios[‡]
Department of Computer Science
The University of British Columbia
dosremedios.brandon@gmail.com

Abstract

In baseball, the key task of categorizing pitches into balls and strikes requires determining whether or not the ball passes through an area known as the strike zone. An electronic umpiring system known as the Automated Ball and Strike system (ABS) is being tested as a potential alternative for human home-plate umpires in Major League Baseball for performing this task. Current approaches to ABS focus on tracking the ball after the pitch, with a lack of focus on the batter metrics which determine the actual strike zone bounds. Additionally, this requires usage of additional sensing hardware like Doppler radar and high precision cameras. We propose supplementing ABS with a computer vision model to fill the hole of batter metric detection, using only input from existing live video feeds. Since the strike zone is determined by the batter's shoulders, waist, and knees, an automated system detecting these metrics from video frames must perform image segmentation; U-Net is a fully connected convolutional neural network architecture with demonstrated high performance on image segmentation tasks. We train U-Net on a dataset of 100 hand-labeled batter images and evaluate the model's performance using a 20% validation set, as well as qualitative visual analysis of its performance on new video data. Ultimately, we determine U-Net to be a good candidate for the batter segmentation task in terms of both accuracy and speed.

1 Introduction

A key mechanic in the sport of baseball is the strike zone. When a ball is thrown and enters the strike zone it is considered a “strike"; if not, it is a “ball." Distinguishing between these two outcomes during a pitch is key to the progression of the game. The spatial location of the strike zone, pictured in Figure 1, is defined by the body of the player trying to hit the ball (the batter). Specifically, the uppermost vertical boundary of the strike zone is determined by the mid point between the top of the batter's shoulders and the top of their uniform pants, and the bottom vertical boundary is determined by the batters knees, according to the rules of Major League Baseball (MLB) [Triumph Books, 2021]. During the game, the strike zone is visually determined by a human judge called the home plate umpire, who stands behind the batter and attempts to see if the ball went through the strike zone or

[*]Student #71762264
[†]Student #40434342
[‡]Student #92827583

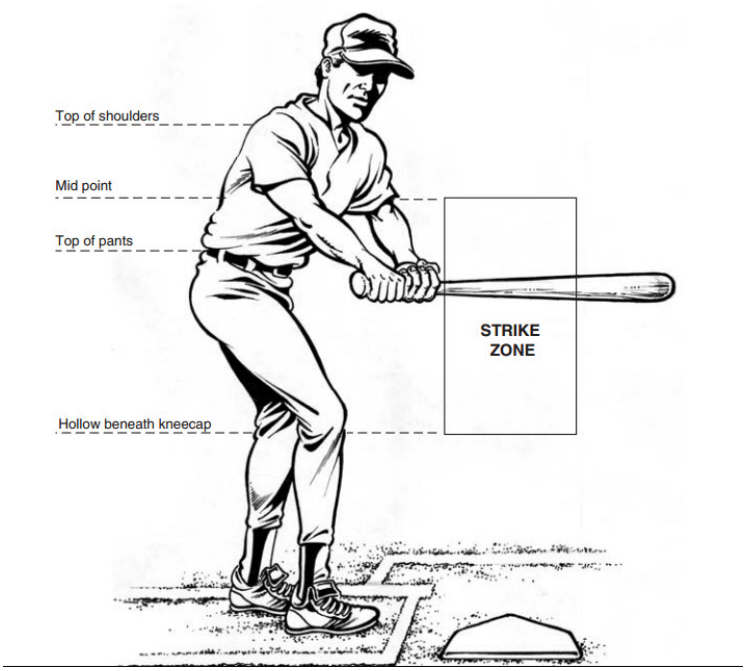

Figure 1: The strike zone in MLB (reproduced from [Gaines, 2014]).

not. However, due to the high speeds at which the ball is being thrown (rising to an average fastball pitch of over 90 miles per hour for many MLB players [Baseball Savant, 2022]), it can be difficult for the umpire to process information fast enough to correctly locate the strike zone accurately by sight alone. This is reflected in the inaccurate tendency of umpire calls, with even the best umpire of the 2021 season locating the strike zone incorrectly on average 1 in every 20 balls thrown [Singer and Schwartz, 2022].

Because of the ongoing issue with strike zone determination, many fans of MLB have called for the implementation of an “electronic" or “robot" umpire [Unknown Author, 2021], which would automate and standardize the strike zone determination process. Intuitively, computers can process information many orders of magnitude more quickly than human reaction times, and so may be much more accurate and reliable than human umpires if provided with reliable information. Additionally, a robot umpire could directly replace the strike-zone-determination portion of the home plate umpire’s work, allowing them to focus more on the many other judgements they must make.

The current main approach to automated umpiring focuses primarily on pitch tracking technology; however, this approach lacks focus on the batter metrics which are also key to successfully determining balls and strikes. We therefore believe that these approaches can be significantly improved with supplemental computer vision analysis leveraging live camera feeds of the pitch. The strike zone is determined based on the top of the batter’s shoulders, the top of their uniform pants, and their knees; given an algorithm for determining these locations on the batter’s body for each frame of game footage, an automatic umpire could identify the strike zone from the video. This in turn could be used in conjunction with sensor data to determine when the ball passes home plate and subsequently determine if the ball is within the strike zone, potentially culminating in a much higher overall accuracy of calling balls versus strikes.

This study investigates the feasibility and performance of using a fully convolutional neural network (FCN) to label the shoulders, belt, and knees of baseball batters in real time. FCNs are an attractive candidate for image segmentation tasks because of their ability to accept inputs of various dimensions, as well as being relatively resource-light to train and predict with. We then investigated the performance of these models on test data in the form of unseen MLB footage. Ultimately, we determine the U-Net FCN architecture to be a good candidate for a supervised image segmentation model to detect a batter’s shoulders, waist, and knees from digital footage, and suggest further steps for utilizing such a model in an Automated Ball and Strike system.

2 Related Work

The official approach for an electronic umpire system is currently in construction by MLB and is known as the Automated Ball and Strike system (ABS). A joint initiative involving the MLB-affiliate Atlantic League, sports analytic companies, and MLB has been developing ABS since 2019 [Pierce et al., 2021]. Despite some initial backlash [Reuter, 2021], ABS has thus far proven successful enough to be used within the higher stages of the 2022 AAA minor league [Papke, 2022], one league under the major league. As the name suggests, ABS is designed to automatically determine balls and strikes; however, the exact implementation is not known to the public due to ABS being a private corporate project. It is stated that the system uses an existing MLB tracking system [Pierce et al., 2021], which is likely the primary MLB pitch tracking system of Statcast [MLB, 2022]. Statcast electronically determines if a pitch has entered the strike zone by directly tracking the ball through a combination of 3D Doppler radar and computer vision systems. Thus, we infer that less work has been placed on the detection of the batter metrics which are also essential for determining the strike zone.

The application of neural networks to the problem of balls and strikes is also not unique to this particular study. Unfortunately, there is a lack of documented corporate attempts to apply neural networks in this context, but there do exist unofficial fan attempts to tackle the problem. One published example is a mixture model implementation [Cheng, 2019], which attempts to determine a ball or strike from image data. In this mixture model, a convolutional neural network (CNN) acts as an encoder converting image data to a flat 4096 feature vector, which is then fed to a decoding long short-term memory recurrent neural network (LSTM) [Hochreiter and Schmidhuber, 1997] which ultimately performs the prediction. This model achieved an F1 score of 0.786 for determining balls and strikes, but this was still lower than the author's own ability to determine them by eye (F1 score of 0.871). We conjecture this lackluster performance may be due to the model's loss of spatial information with its usage of the flattened feature vector at the end of the encoding state.

Though the mixture model is currently unable to out-compete human umpires, the use of computer vision to assist ABS remains a tantalizing prospect. A clear benefit of computer vision methods is that they would not require any additional sensing hardware, instead using more economical and easily available media coverage feeds, thereby providing significant potential benefit for negligible hardware costs. Any algorithm seeking to determine the strike zone requires the ability to automatically find the shoulders, waist, and knees of the batter in every frame of the video. This is an image segmentation problem: the algorithm must correctly find and label the corresponding pixels in the input images. Fully convolutional neural networks (FCNs) like U-Net have shown promising results in this problem domain—U-Net was the first architecture to outperform deep convolutional neural networks on the ISBI challenge for segmentation of electron microscope scans of neurons [Ronneberger et al., 2015]. Contrasting the CNN-LSTM mixture model, an FCN implementation would also be able to fully utilize spatial information in the image data, potentially mitigating this shortcoming. Thus, we seek to apply the image segmentation capability of FCNs to the ABS problem, particularly using the U-Net architecture.

3 Methods

3.1 Dataset and Data Preparation

We constructed a dataset consisting of 100 hand-labeled images for training and testing U-Net. Each sample image is of a batter in their batting stance a few seconds before a pitch (since this is the moment that the strike zone is determined [Triumph Books, 2021]). The samples were taken from 19 different teams so that a variety of uniform colours and stadium backgrounds were included for the model to learn from. To test if data augmentation techniques such as mirroring would be sufficient to allow the model to learn to label left-handed batters, we only included images of right-handed batters in our dataset. All samples were collected from MLB broadcasts from the official MLB YouTube channel.The images were taken at manually collected time stamps of broadcast video frames during the pre-pitch moments (when the strike zone is determined during play). We then collected 1280×720 px frames at each time stamp using the `youtube-dl` package [Open Source, 2021]. For each frame, we cropped a 128×320 px sub-image centered around the batter; this allows six factor-of-two pooling reductions in the model and also removes much of the background in the image, leaving primarily only the details of the batter. Because the location of the batter was inconsistent across frames, we performed the cropping manually using an area selection GUI (Figure 2). We then manually labeled each batter's shoulder, waist, and knee using an open source pixel annotation software [Bréhéret, 2021] (Figure 3). These were the labeled images ultimately used to train U-net.

3.2 Network Architecture

We used a version of the U-Net architecture [Van Vugt and de Oliveira Neto, 2019] adapted for the PyTorch machine learning library [Paszke et al., 2019]. All of the models trained had a depth of 4 and a starting channel width of 2^4 (doubling during each max pool, and halving every up-scale). We tested two different hyper-parameter configurations for the architecture: setting the `max_pooling/upscaling` factor to the default value of 2 and testing an increased value of 4, each with both unaugmented and augmented datasets.

3.3 Image Augmentation

To improve test performance and the generalization of the model (for example, the ability to handle left-handed batters, new teams, etc.), we performed various kinds of image augmentation. Augmentation was performed at each iteration when the data loader was called to obtain a new training example. The image augmentation was done using the `transformations` module within PyTorch and consisted of the following four transformations:

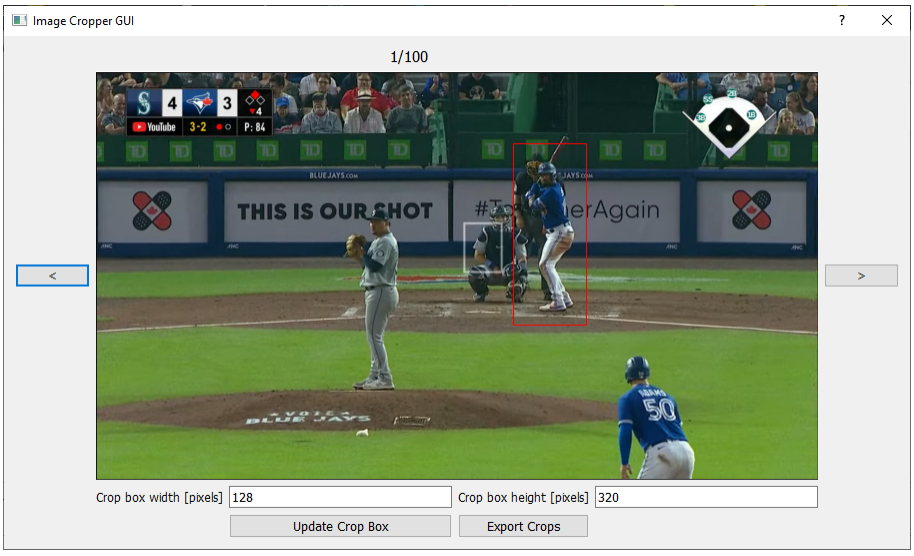

Figure 2: Example broadcast video frame within the constructed crop position selection GUI.

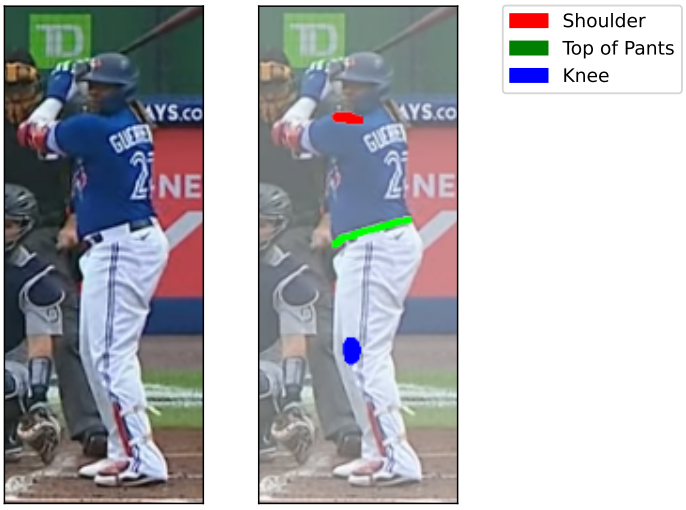

Figure 3: Example cropped sample image (left subplot) and the corresponding manual pixel label (right subplot) of batter shoulder, top of uniform pants, and knee.

1. a colour jitter that varies the hue by ± 0.15, the brightness by ± 0.05, the contrast by ± 0.10, and the saturation by ± 0.10
2. a Gaussian blur with a kernel size of 5 and a random sigma between $[0.001, 2.0]$
3. a random horizontal flip with probability of 0.25
4. an affine transformation which varies the rotation between ± 5.0 degrees, translation between $\pm 0.05\%$ on both the x and y axes, scale between $[0.95, 1.05]$, and a shear between ± 2.0 degrees on the x axis.

On top of this affine transformation, each label/target image also received an additional smaller translation between $\pm 0.01\%$ of the image size.

3.4 Training

We trained 10 models for each of the hyper-parameter combinations tested for 200 epochs on an NVIDIA RTX 6000 (40 models total). Each model had a random 80-20 train-test split. For training, we used the Adam optimizer [Kingma and Ba, 2014] with `AMSGrad` set to true, and the Dice loss function [Sudre et al., 2017] as our objective function. The Dice loss is akin to the intersection over union metric and improves over cross entropy loss when there is a large class imbalance (which in our case there is; most of the image is background).

Throughout every epoch, we kept track of the average loss on the training data, and at the end of every epoch calculated the average loss for the testing data. Both values were saved to an array for later analysis. Additionally, for unseen video inference, we retrained the models on all of the 100 training images for 200 epochs.

4 Discussion of Results

As we expected, the training data reached a very low loss when the data was not augmented, while the test loss leveled off rather quickly. Interestingly, increasing the max pooling factor to 4 decreased the variance between models (Figure 4). When the data was augmented, the training loss plateau was reached slower (at about epoch 100 instead of epoch 50) and the ultimate training loss was higher. The test scores however, improved significantly compared to when they were not augmented ($p = 6.99 \times 10^{-6}$ for max pool factor of 2, $p = 4.24 \times 10^{-11}$ for factor of 4, $n = 10$; see Figure 5). There was no significant improvement between max pool factors of 2 and 4 on the test dataset.

We then tested the models trained on all of the training data on new videos of batters which had not been used in the labeled dataset. As expected, augmented models were able to handle left-handed batters much better than the unaugmented models due to the augmentation pipeline including a random horizontal flip (Figure 6a, Supplemental Video 1). Interestingly, while the max pooling factors did not show significant differences on quantitative test performance, increasing the max pooling factor did improve results on new video inference. As shown in Figure 6b and Supplemental Videos 2a and 2b, the model with a max pooling and upscaling factor of 4 was more stable over new poses, whereas the factor of 2 model would often place knee or waist artifacts where they should not be (highlighted by the red circles). We believe that, for the same number of layers/max pooling steps, the increased shrinking of the image resulting from a larger max pooling factor resulted in a larger "field of view" for each convolution, allowing the model to use more global information to accurately identify the body parts. Thus, increasing the depth of the model by a factor of 2 may also achieve the same effect, but the downside of that is a significantly larger model that is much slower to train and perform inference on.

Overall, the model is only 2.5 megabytes in size, and we were able to perform over 50 iterations of training (inference, loss calculation, and backpropagation) per second on the RTX 6000. Thus, we are confident the trained model will be able to perform inference in much faster than real time on 60 frames-per-second video feeds[4], and likely on significantly less powerful hardware.

5 Conclusions and Next Steps

In this study, we have quantitatively and qualitatively assessed the performance of the U-Net FCN architecture on the image segmentation task of labeling the shoulder, waist, and knees of MLB batters before a pitch. We conclude that using image augmentation during training, especially colour jitter and mirroring, helps generalize the trained model to different uniform colours and left versus right-handed batters, respectively. Furthermore, increasing the max pooling and upscaling factor from 2 to 4 resulted in reduced variance in the loss across different test examples, as well as better inference on unseen video data as evidenced by fewer misplaced image segmentation artifacts, likely due to increased network field-of-view. Further testing is warranted to obtain more quantitative performance metrics such as receiver operating characteristics and F1 scores. Due to the relatively low computational cost of performing inference with the U-Net architecture, we were also able to perform inference on 60 frames-per-second videos, in faster than real time—a key desideratum for incorporating the model into ABS.

Given that the greatest shortcoming of this preliminary study was the small training set, we believe this already promising approach can be vastly improved with increased labeled data. Additionally, determining the strike zone from these labeled images is relatively straightforward: we set the top boundary using the average position of the shoulder and waist, the bottom boundary using the average position of the knee, and set the left and right boundaries using the position of the home plate. Used in conjunction with the existing ABS sensors, U-Net can thereby form the basis for a system for determining whether or not the baseball is within the strike zone when it passes over the home plate simply by checking its position within the corresponding video frame. Cropping the input video to focus on the batter, as was done manually in this study, can also be automated using existing bounding box computer vision models.

[4]Unfortunately, we lost access to the computing hardware before we were able to perform runtime tests on the trained model.

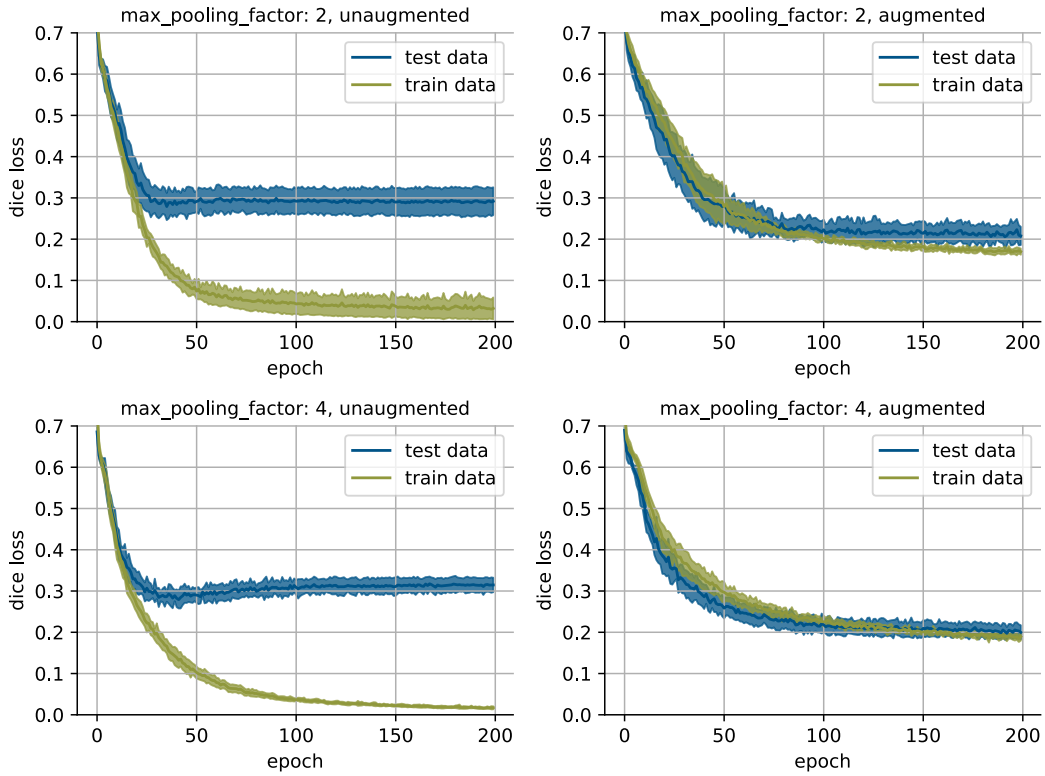

Figure 4: Train and test error throughout the training of the models. Area shows mean ± 1 standard deviation, $n = 10$.

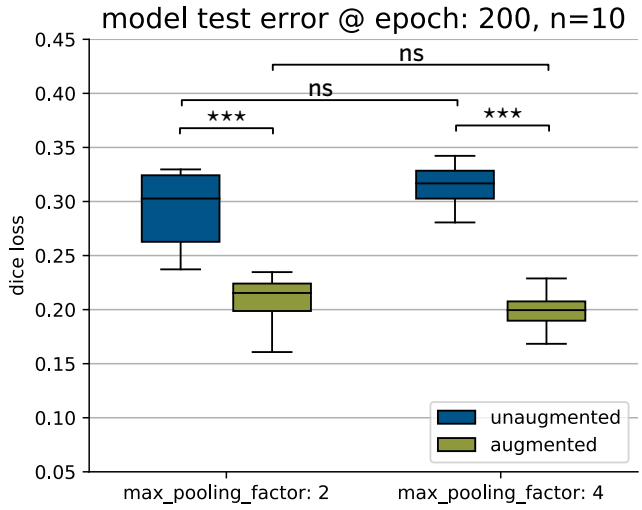

Figure 5: Test errors at the 200$^{\text{th}}$ epoch ($n = 10$). Independent t-test showed significant improvements between unaugmented and augmented data (max pooling 2: $p = 6.99 \times 10^{-06}$, max pooling 4: $p = 4.24 \times 10^{-11}$). Comparisons between max pooling factors of 2 and 4 did not show any significant differences.

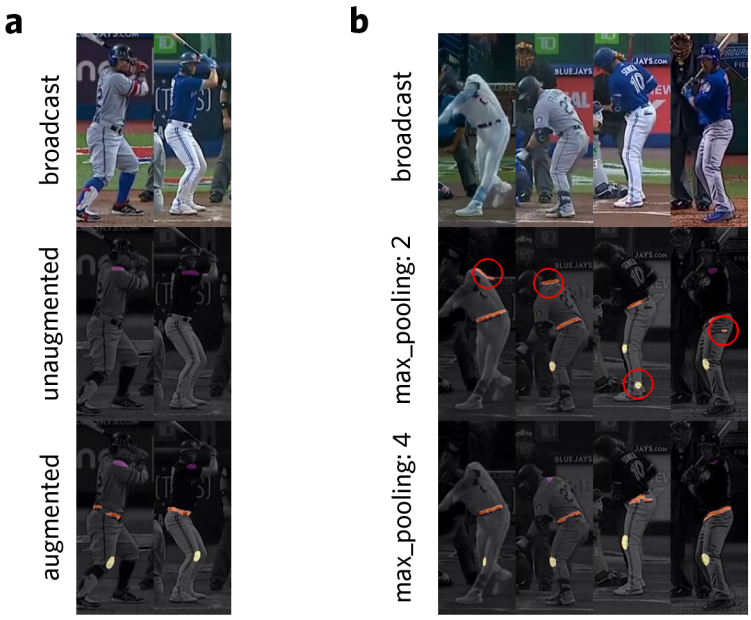

Figure 6: **a.** Model performance on unseen left-handed batters. The model trained on augmented data was able to handle the segmentation, see supplemental_video_01.mp4 for full-length video. **b.** Model performance on unseen batters is more stable when the max_pooling/upscaling factor is doubled. See supplemental_video_02{ab}.mp4 (included in the submission ZIP file) for full-length videos.

In conclusion, we believe the U-Net FCN architecture to be a promising candidate for a supervised image segmentation model that can supplement the current Automated Ball and Strike system being tested for MLB.

Supplemental Video Links

The following are links to the supplemental videos. Please download and play back locally.

1. Supplemental Video 1: https://osf.io/myxv8/download
2. Supplemental Video 2a: https://osf.io/yfvua/download
3. Supplemental Video 2b: https://osf.io/52xtr/download

Acknowledgements

This work was supported by resources made available through the Dynamic Brain Circuits cluster and the NeuroImaging and NeuroComputation Centre at the UBC Djavad Mowafaghian Centre for Brain Health (RRID SCR_019086). Some of the hardware within the centre was provided by NVIDIA through the NVIDIA Academic Hardware Grant Program.

References

Triumph Books, *2021 Official Rules of Major League Baseball*, ser. Official Rules Series. Triumph Books, 2021.

Baseball Savant, "Statcast Pitch Arsenals leaderboard," Apr. 2022, Accessed on: Apr. 22, 2022. [Online]. Available: https://baseballsavant.mlb.com/leaderboard/pitch-arsenals.

E. Singer and E. Schwartz, "Umpires," Apr. 2022, Accessed on: Apr. 22, 2022. [Online]. Available: https://umpscorecards.com/umpires/.

C. Gaines, "What an MLB strike zone really looks like and why players are always so mad about it," *Business Insider*, Sep. 2014, Accessed on: Apr. 22, 2022. [Online]. Available: https://www.businessinsider.com/mlb-strike-zone-2014-9.

Unknown Author, "Major league baseball considers dramatic change with 'robot umpires'," *CBS News*, Nov. 2021, Accessed on: Apr. 22, 2022. [Online]. Available: https://www.cbsnews.com/news/robot-umpires-mlb-consider/.

P. Pierce, M. Whitrock, and S. Cheshir, "Developing MLB's Automated Ball/Strike System (ABS)," *MLB Technology Blog*, May. 2021, Accessed on: Apr. 22, 2022. [Online]. Available: https://technology.mlblogs.com/developing-mlbs-automated-ball-strike-system-abs-d4f499deff31.

J. Reuter, "Why robot UMPS aren't the best solution to MLB's umpiring problem...yet," *Bleacher Report*, Oct. 2021, Accessed on: Apr. 22, 2022. [Online]. Available: https://bleacherreport.com/articles/2949584-why-robot-umps-arent-the-best-solution-to-mlbs-umpiring-problemyet.

G. Papke, "Automatic strike zone coming to AAA in '22," *MLB News*, Jan. 2022, Accessed on: Apr. 22, 2022. [Online]. Available: https://www.mlb.com/news/triple-a-to-have-automated-ball-and-strike-system.

MLB, "Statcast," 2022, Accessed on: Apr. 22, 2022. [Online]. Available: https://www.mlb.com/glossary/statcast.

R. Cheng, "Building a Robot Umpire with Deep Learning HD Video Analysis (Part Three)," *The Hardball Times*, May 2019, Accessed on: Apr. 22, 2022. [Online]. Available: https://tht.fangraphs.com/building-a-robot-umpire-with-deep-learning-hd-video-analysis-part-three/.

S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

Open Source, "Youtube-dl," https://github.com/ytdl-org/youtube-dl, Dec. 2021.

A. Bréhéret, "PixelAnnotationTool," https://github.com/abreheret/PixelAnnotationTool, Jun. 2021.

J. Van Vugt and N. C. de Oliveira Neto, https://github.com/jvanvugt/pytorch-unet, Jul. 2019.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. Jorge Cardoso, "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations," in *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, 2017, pp. 240–248.