**Staying in the End Zone:**

**Using Neural Networks for Lower Limb Detection in NFL Players**

**Introduction and Motivation**

The National Football League (NFL) is the most popular sports league in the U.S. and manages 32 football teams, accounting for over 1,000 players [1]. Their top priority is protecting athlete health and safety through injury detection and prevention, which is paramount for player performance and career longevity. As part of this endeavor, the NFL has partnered with Amazon Web Services (AWS) to leverage artificial intelligence and machine learning capabilities to transform player health and safety [2]. Moreover, the knowledge gained from this collaboration has the power to not only revolutionize football and other contact sports, but also other industries [2]. The major research areas of interest include reducing concussions, minimizing lower extremity injuries, providing a framework for equipment manufacturers, and measuring the impact of game rule changes [3].

For this project, we are exclusively focusing on lower extremity injuries, specifically those related to the feet. This topic is important and warrants further research as lower extremity trauma is the most common type of injury among professional athletes [4], and aside from torn ACLs, they cause the most game time lost for NFL players [5]. Therefore, it is imperative to reduce the frequency and severity of these kinds of issues, with the eventual goal of predicting the risk of injuries before they occur [2].

**Hypothesis**

Based on current literature on the use of deep neural networks for human pose estimation (HPE), we anticipate that deep neural network architectures will be able to detect and predict x and y coordinate locations of players’ toes and heels in images of football game highlights (we are using the toe and heel positions to capture the entirety of the foot). Moreover, we compare the performance of the following models using mean squared error (MSE): the pre-trained DeepPose proposed by Toshev and Szegedy (2013) [6], a modified AlexNet [7], and a modified ResNet50 [8].

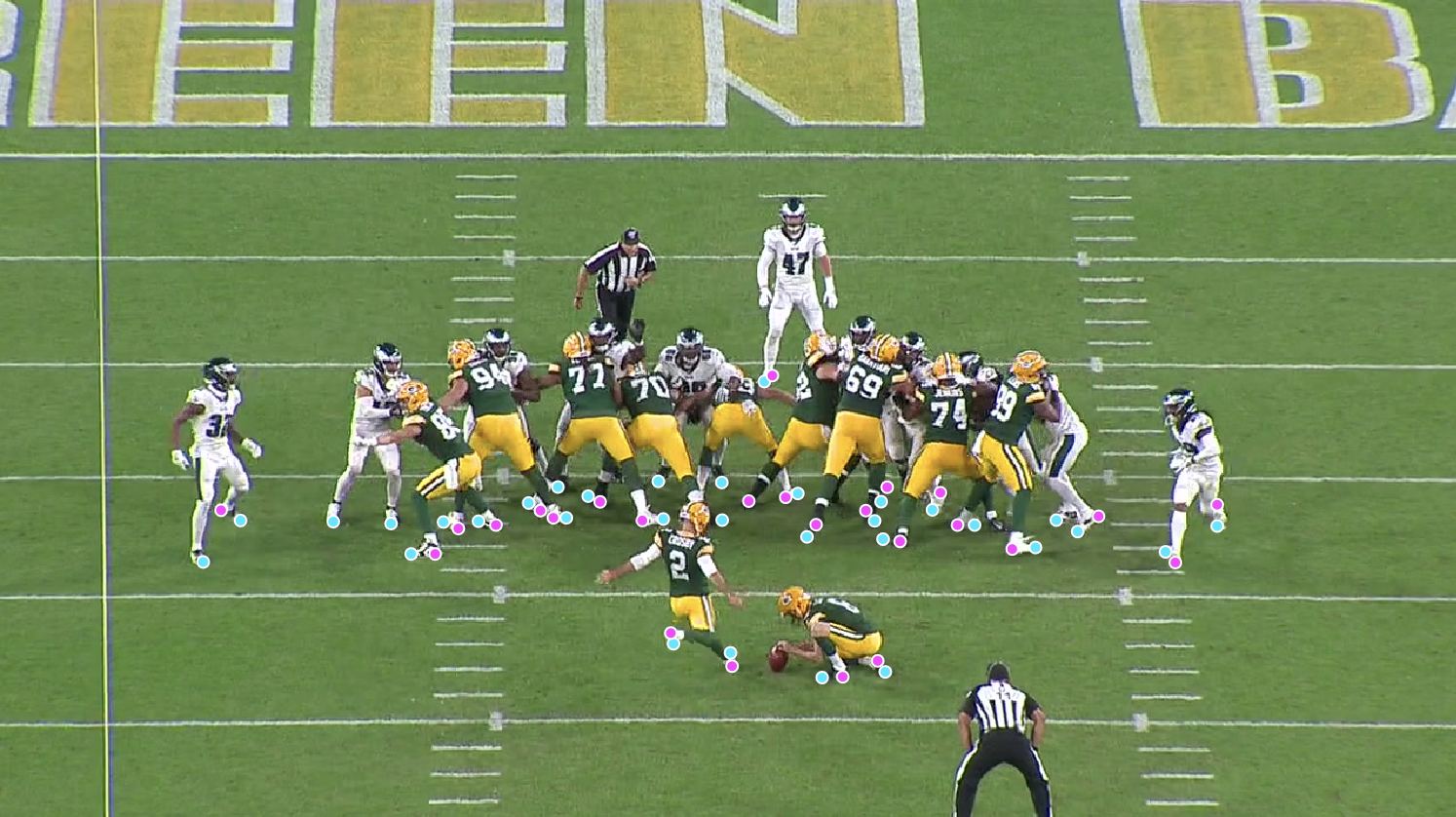
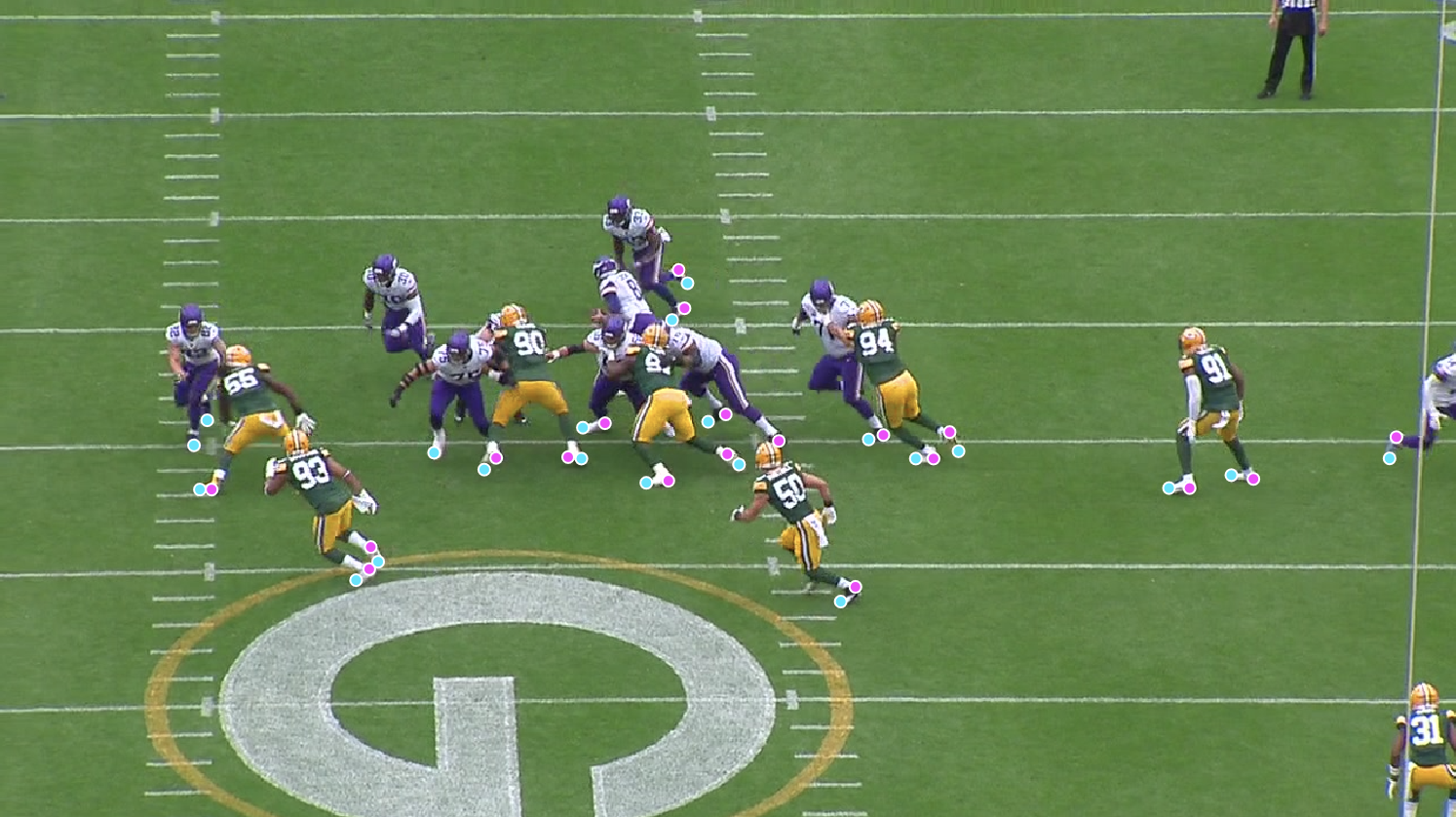
**Literature Review**

Researchers have implemented several types of neural networks for HPE, which detects and tracks human key joints through computer vision. Here, we consider DeepPose, one of the highest performing joint detection algorithms. Many state-of-the-art pose estimation algorithms, including the widely used Fast(er) RCNN, implement Convolutional Neural Networks (CNNs). DeepPose uses a more complex version of a CNN called a Deep Convolutional Neural Network, which contains more layers. DeepPose requires the use of a top-down approach, meaning that individuals are first identified via bounding boxes, and then key points are estimated within these boxes [9]. DeepPose is a Deep Neural Network (DNN) based regression model with a prediction function where x is an image and is a vector of model parameters. The final predicted poses are given by multiplied by a normalization transformation N⁻¹. The CNN is built by layering alternating linear and non-linear layers [9]. Figure 1 shows a graphical representation of the network architecture. The input images are a fixed size of 220 x 220 with a stride of 4. The network is then formulated via a cascade of pose regressors with the first regressor estimating an initial pose and the following regressors predicting a difference of the true joint locations from the predicted joint locations.

**Figure 1. DeepPose neural network design. [6]**

**Data Source**

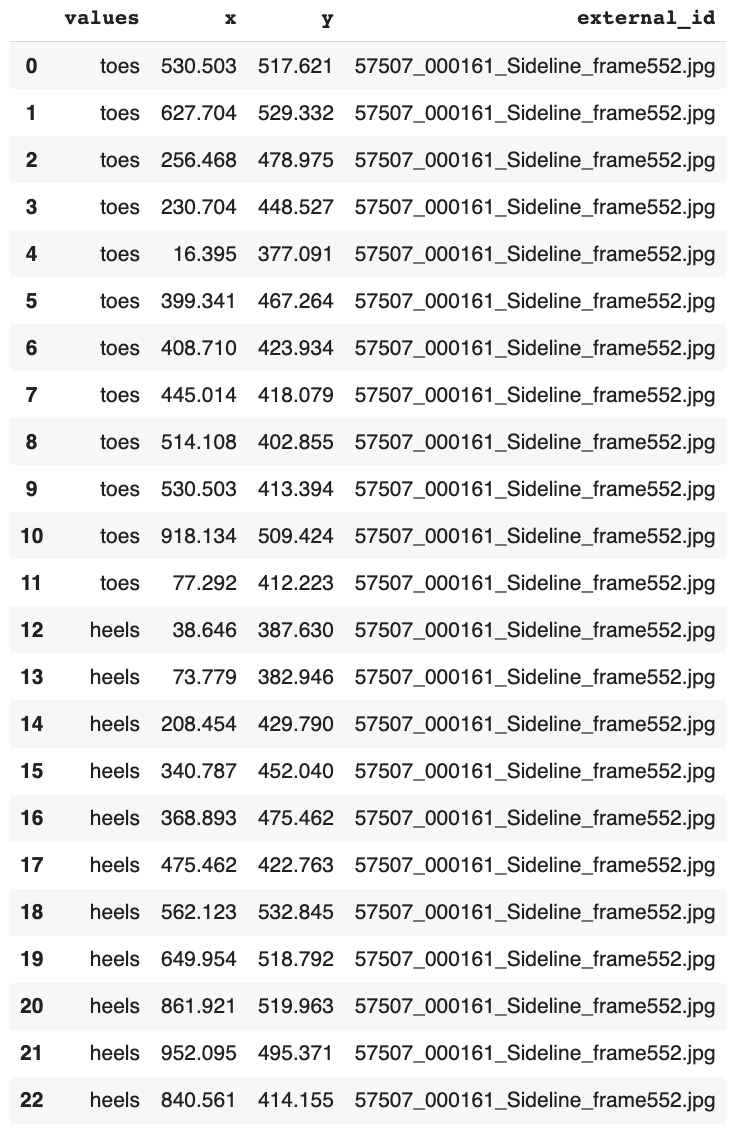
The NFL could not provide data due to concerns surrounding athlete privacy and intellectual property. However, we found open-source images of football game highlights from Kaggle [10], which served as a good alternative. Most of the photos are aerial shots of multiple players on the field in mid-motion. We only annotated players on the field, ignoring referees and any athletes on the sidelines. We used a data annotation system called Labelbox [11] to manually annotate the locations of each player’s toes and heels. We repeated this process for 1,050 images, which comprised the training and testing datasets used to build and evaluate the neural networks. Figure 2 shows examples of annotated images (cyan points represent toes and magenta points represent heels).

**Figure 2. Examples of annotated images using Labelbox.**

**Methods**

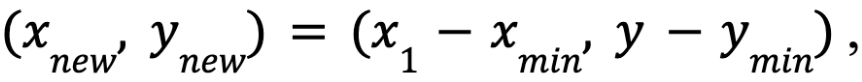
*Data Preprocessing*

After manually identifying and annotating a total of 16,198 toes and 16,710 heels, the data were read into Google Colab in JSON format. Unlabeled images were dropped (either no player on the field or severely blurred photo quality) and this took the total number of images from 1,050 to 1,038. From the external data file, we extracted the following information: classification (toe or heel), x-coordinate, y-coordinate, and corresponding image ID. These details were then saved as a pandas dataframe for subsequent analyses (Figure 3).

**Figure 3. Example of extracted player information for one image.**

Next, we used YOLOv5 [12], which is an object detection model, to identify bounding boxes encompassing individual athletes. YOLOv5 detects objects by using a neural network to create a grid of cells, each responsible for object detection within itself [13]. For the scope of this project, YOLOv5 was preferred over RCNN and Fast(er) RCNN due to its higher accuracy and improved efficiency. Using this pre-trained model, consisting of a single CNN, potential bounding boxes were outputted along with their probabilities for capturing a single player.

We created a function to select all images with four or more annotated points - each player can have a maximum of four annotated points with unique x and y coordinates. We were not concerned with images that had less than four annotated points due to the maximum key point restriction. Furthermore, for any images with more than four key points, we randomly sampled four annotations to satisfy the maximum key point condition.

Each bounding box was cropped with an added padding of 10 and saved as its own image. Furthermore, the coordinates of the annotated points were simultaneously converted to the respective bounding box coordinates via the following equation:

where are the coordinates of an annotated point on the original image, and and are the minimum x and y coordinates of the bounding box for the respective cropped image. The origin in both cases is defined with respect to the top left corner of the photo and bounding box. See the figure below for a sample cropped photo along with its annotated points.

**Figure 4. Example of cropped bounding box with converted annotations.**

The final data preprocessing step involved reshaping the images to a fixed size of 220 x 220 pixels for the DeepPose and modified AlexNet models, and 224 x 224 pixels for the modified ResNet50 model, once again converting the annotated points to the respective coordinates of the new bounding box. The data was then split into train and test sets of 70% and 30%, respectively.

*Model Building: Pre-trained DeepPose Model*

We first implemented the pre-trained DeepPose model, which consisted of eight outputs and MSE as the loss metric. The architecture of this model is as follows: a sequence of (1) convolutional layer, (2) local response normalization layer, (3) pooling layer repeated twice followed by three convolutional layers, one pooling layer and two fully connected layers. It should be noted, in this model, local response normalization was preferred over batch normalization to encourage local contrast enhancement, since we were more concerned with lateral inhibition than internal covariate shift [14]. We ran the model with a learning rate of 0.0005, as was done in the original paper, and tested the following number of epochs: 20, 50, and 100. We observed that increasing the number of epochs did not meaningfully reduce the loss. Therefore, we set the final number of epochs to 20. Note: we just used the first stage of the DeepPose algorithm proposed by Toshev and Szegedy (2013).

*Model Building: Modified AlexNet Model*

Next, we wished to experiment with a modified AlexNet model by adding layers to increase complexity. We were curious to see if repeating the sequence of (1) convolutional layer, (2) local response normalization layer, (3) pooling layer three times instead of twice, with the last pooling layer removed, would improve model performance. Furthermore, we changed the activation function from linear to sigmoid to create a (0, 1) bound on the values and prevent the early neurons from interfering with intermediate layers through the addition of non-linearity [15]. We explored the effect of changing the learning rate on the performance of the model. This parameter scales the magnitude of the weight updates to minimize the network’s loss function [16]. The optimal learning rate typically lies around 0.001, and so this is the value we used for this model. After increasing the number of epochs to 100, which led to overfitting, we settled on 20 as the optimal value for this parameter.

*Model Building: Modified ResNet50 Model*

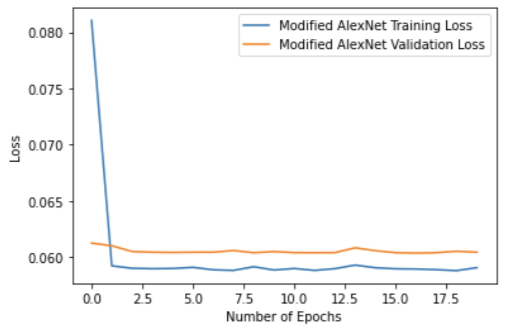
To further evaluate our models, we wanted to compare the performance of DeepPose and modified AlexNet to another pre-trained model. We selected ResNet50, which is often used for image classification, object localisation, and object detection [17]. ResNet50 is a variation of the traditional ResNet and gets its name from being a Deep Convolutional Neural Network with 50 layers [17]. We performed some hyperparameter tuning, which included changing the learning rate to 0.001 (perhaps this value was a bit too high because the loss began to diverge) and finding the best freezing layer. Since we had a relatively small dataset compared to the number of model parameters, we froze the first few layers and trained the remaining layers to let the model learn more about the dataset [18].

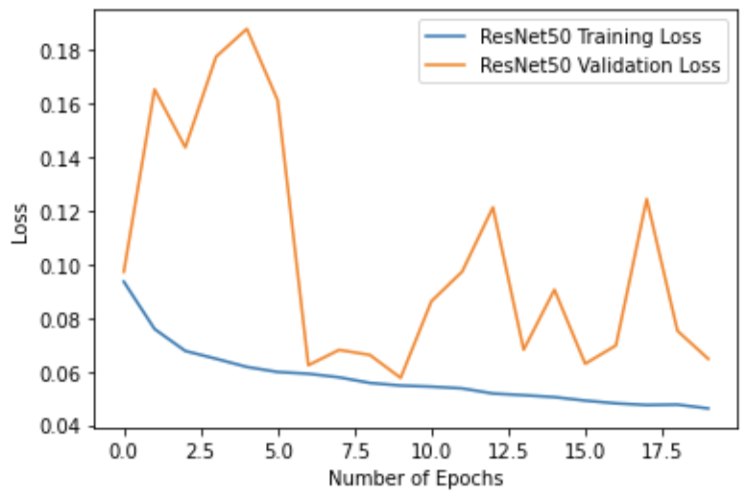
**Analysis and Interpretation**

After running our three models and observing the loss associated with each, the modified AlexNet was the best with the lowest validation loss (0.0604 with 20 epochs). When comparing between the different epoch numbers (20, 50, and 100), we noticed that at 100 epochs, the model was overfitting with respect to the validation loss. Furthermore, because there was not a significant difference between 20 and 50 epochs, we proceeded to use 20 epochs. Our DeepPose model had a slightly higher validation loss of 0.0606 with 20 epochs. We observed no real reduction in loss when increasing the number of epochs to 50 and 100, respectively. Our modified ResNet50 model had the highest validation loss of 0.0647 with 20 epochs. For this model, we were unable to experiment past 20 epochs due to a lack of RAM and found that epochs lower than 20 led to an even higher loss.

We expected that deep neural network architectures would be able to detect x and y coordinates of players’ heels and toes. After evaluating our three models using MSE, we found support for our hypothesis. Overall, the models were able to detect coordinate locations of players’ toes and heels. There is room for improvement, as these predictions somewhat deviated from the true locations. From Figures 8 through 10, we noticed that the estimated key points appeared around the lower half of the athletes’ bodies, indicating that the models were able to identify the correct part of the image where the joints are located. However, the key points wavered from the true locations of the toes and heels. The pre-trained DeepPose model we implemented is a simplified version of the model proposed by Toshev and Szegedy (2013), which does not include the cascade of pose regressors. Thus, the model was able to identify rough locations of joints, but lacked the ability to refine these predictions without the presence of the cascade, which would have considered smaller sub-images that are localized around the key points.

**Figure 5. DeepPose Training and Validation Loss.**

**Figure 6. Modified AlexNet Training and Validation Loss.**

**Figure 7. Modified ResNet50 Training and Validation Loss.**

**Figure 8. DeepPose predicted coordinate locations.**

**Figure 9. Modified AlexNet predicted coordinate locations.**

**Figure 10. Modified ResNet50 predicted coordinate locations.**

Note: the sample of images generated by the modified ResNet50 model is different from those of the other two models because we had to resize the images to 224 x 224 in order to run the pre-trained ResNet50 model.

We decided to use MSE, as this is a regression problem. But, in the future, we could create a custom loss function that combines binary cross entropy and weighted MSE. This would force the network to both accurately identify key points and correctly distinguish between toes and heels through the evaluation metric. We could also add a probability associated with each x and y coordinate to represent our confidence - all manually annotated points would have a corresponding probability of one, as we are 100% certain that these points are correctly identifying either toes or heels. Since we decided to disregard images with less than four annotated points, we were not able to use weighted MSE because all of our probabilities were the same. Nevertheless, we could potentially impute the x and y coordinates for images with less than four points and randomly assign a low probability to represent our lack of confidence with regards to these points. The subsequent models would be trained on all these points, but would place less weight on the data with lower probabilities. This would introduce noise into our models, helping with regularization and preventing overfitting.

**Discussion and Conclusions**

While our models are currently able to provide estimations of key joint locations, they are limited by several factors. Because we manually created the key point annotations, the quality of our labeling likely impacted model performance - for example, if we incorrectly marked a few toe and heel positions, then the models trained on these points could potentially learn incorrect patterns. Several people worked on manually annotating the images, which introduced human error in addition to different annotating styles. Also, many of the images were blurry due to the angles at which they were taken, which made it difficult to visually assess the locations of toes and heels, especially when players were captured in mid-motion or many athletes were clumped together. Furthermore, another limitation of our project is the insufficient data used to train the models - the data points generated from roughly 1,000 annotated images is still relatively small compared to the millions of parameters in the deep neural networks.

Future work to improve our project could include increasing our dataset count and implementing the full version of the DeepPose model that uses the cascade of pose regressors for better performance. There are also alternative neural network models that might demonstrate better performance in joint estimation including: OpenPose, High-Resolution Net, DeepCut, Regional Multi-Person Pose Estimation, PoseNet, and DensePose. Some of these approaches, like DeepCut, may function better because they employ a bottom-up approach, which would allow for immediate joint estimation in multi-person images without the need for preliminary bounding box detection.

**References**

[1] “NFL Players - Players by Team, NFL Player Search.” *CBSSports.com*, https://www.cbssports.com/nfl/players/.

[2] Staff, SVG. “NFL Partners with Amazon Web Services to Strive towards Better Player Health, Safety.” *Sports Video Group*, 6 Dec. 2019, https://www.sportsvideo.org/2019/12/06/nfl-partners-with-amazon-web-services-to-strive-towards-better-player-health-safety/#:~:text=Amazon%20Web%20Services%20%28AWS%29%20and%20the%20National%20Football,NFL%20will%20together%20shape%20the%20future%20of%20football.

[3] *NFL Head, Neck and Spine Committee’s Concussion Diagnosis and ...* https://static.www.nfl.com/image/upload/v1597677293/league/dhfywrwelyfsivvia3ie.pdf.

[4] Mack CD;Kent RW;Coughlin MJ;Shiue KY;Weiss LJ;Jastifer JR;Wojtys EM;Anderson RB; “Incidence of Lower Extremity Injury in the National Football League: 2015 to 2018.” *The American Journal of Sports Medicine*, U.S. National Library of Medicine, https://pubmed.ncbi.nlm.nih.gov/32485114/.

[5] FOX Sports. “NFL Looking into High Number of Lower Extremity Injuries.” *FOX Sports*, FOX Sports, 4 Mar. 2020, https://www.foxsports.com/stories/nfl/nfl-looking-into-high-number-of-lower-extremity-injuries.

[6] Toshev, Alexander, and Christian Szegedy. “DeepPose: Human Pose Estimation via Deep Neural Networks.” *CVF Open Access*, 1 Jan. 1970, https://openaccess.thecvf.com/content\_cvpr\_2014/html/Toshev\_DeepPose\_Human\_Pose\_2014\_CVPR\_paper.html.

[7] Park, Sieun. “Implementing DeepPose Baseline Model with Keras.” *Medium*, Analytics Vidhya, 7 Sept. 2020, https://medium.com/analytics-vidhya/implementing-deeppose-baseline-model-with-keras-67f8c8ab63c1.

[8] Hebbar, Nachiketa. “Transfer Learning with Keras(Resnet-50).” *Chronicles of AI*, Chronicles of AI, 16 July 2021, https://chroniclesofai.com/transfer-learning-with-keras-resnet-50/.

[9] Odemakinde, Elisha. “Human Pose Estimation with Deep Learning - Ultimate Overview in 2022.” *Viso.ai*, 12 Dec. 2021, https://viso.ai/deep-learning/pose-estimation-ultimate-overview/.

[10] “NFL 1st and Future - Impact Detection.” *Kaggle*, https://www.kaggle.com/c/nfl-impact-detection/overview.

[11] “The Leading Training Data Platform for Data Labeling.” *Labelbox*, https://labelbox.com/.

[12] Cochard, David. “Yolov5 : The Latest Model for Object Detection.” *Medium*, Axinc-Ai, 26 May 2021, https://medium.com/axinc-ai/yolov5-the-latest-model-for-object-detection-b13320ec516b.

[13] “Introduction.” *YOLOv5 Documentation*, https://docs.ultralytics.com/.

[14] Anwar, Aqeel. “Difference between Local Response Normalization and Batch Normalization.” *Medium*, Towards Data Science, 7 Apr. 2021, https://towardsdatascience.com/difference-between-local-response-normalization-and-batch-normalization-272308c034ac.

[15] Ayyüce Kızrak, Ph.D. “Comparison of Activation Functions for Deep Neural Networks.” *Medium*, Towards Data Science, 7 Jan. 2020, https://towardsdatascience.com/comparison-of-activation-functions-for-deep-neural-networks-706ac4284c8a.

[16] Jeremy Jordan. “Setting the Learning Rate of Your Neural Network.” *Jeremy Jordan*, Jeremy Jordan, 29 Aug. 2020, https://www.jeremyjordan.me/nn-learning-rate/#:~:text=For%20learning%20rates%20which%20are%20too%20low%2C%20the,%22bounce%20around%22%20and%20even%20diverge%20from%20the%20minima.

[17] Kaushik, Aakash. “Understanding Resnet50 Architecture.” *OpenGenus IQ: Computing Expertise & Legacy*, OpenGenus IQ: Computing Expertise & Legacy, 21 July 2020, https://iq.opengenus.org/resnet50-architecture/.

[18] Konda, Sailaja. “Improving the Performance of Resnet50 Graffiti Image Classifier with Hyperparameter Tuning in Keras.” *Medium*, Towards Data Science, 8 Apr. 2020, https://towardsdatascience.com/improving-the-performance-of-resnet50-graffiti-image-classifier-with-hyperparameter-tuning-in-keras-dbb59f43c6f7.