

Project Report - Simon Zouki

Personal Views

After finishing the Computer Vision course, which was my first formal introduction to the field, I felt like the field is fascinating, as it has a lot of different applications in numerous fields and, as mentioned by Dr. Wu in class, includes many philosophical schools within the field. In fact, some people believe that the way going forward for the field is very dependent on Deep Learning, similarly to the revolution in NLP, while others believe that research on the traditional methods should be constantly ongoing, and each of the groups have their reasons and perspectives. The latter group believes that Deep Learning has limitations that outweigh its benefits, one problem being the fact that the models are black-box and are barely explainable, and I feel like this group resonates most with me. I don't feel like creating gigantic models on huge amounts of data without understanding what's happening under the hood, without curating datasets that are well documented, and without considering the other lines of research that are more theoretically sound is the way to go. This came to my attention throughout the NLP course when we were reading LLMs papers which have a theoretical and mathematical basis, and the design choices were made and justified by "it seems to work better this way". I believe that as designers of models that have huge impacts on the economy and on people, we ought to have a better understanding and better explanations of our choices and designs. Also, a paper I read by Bender et al. [1], covered all of the

different problems facing the development of LLMs, and it made sense to me why we would want to steer the Computer Vision field away from this direction.

With all that being said, I felt like this course was a perfect introduction to Computer Vision, as it helped us understand the theoretical basis of the methods applied for most of the important tasks like Color Detection for example, while also giving us the chance to apply what we have learned through the homeworks, which were extremely beneficial. The last lectures included a brief overview of some Deep Learning methods, notably those used to do object detection as they seem to be the better performing. Overall, I now have solid knowledge of many topics in the field, especially those we had a homework on, and going forward, I definitely want to delve deeper into some more advanced topics. Sadly, I won't be at Northwestern University when the Advanced Computer Vision course will be given, so I will have to pursue the advanced topics on my own. More specifically, an area that interests me is a discussion point we once had in class about the importance of segmentation and the different techniques that could be employed to do the task, and how some research groups believe that segmentation isn't essential. Also, object tracking is an interesting topic I want to learn more about, as it seems like it's become an increasingly important component of the computer vision field due to the fact that self-driving cars are becoming more popular, as well robotics where robots have an integral part of having to view and process the environment around them.

Project Description

For the project, we wanted to choose a topic which would include many computer vision tasks which we learned in class throughout the quarter, as we wanted to test the different possibilities in a real life situation which would help us gain a better understanding of the algorithms and methods. In addition, we wanted to choose a topic that interests both Gautam and I, and since we are both Basketball players and we are both decently good shooters, we have both been through the process of learning how to properly shoot a basketball. In my case, the motivation for learning how to shoot was when I was 15, I only played soccer while all my friends played basketball, and I had always wanted to be part of the games. Also, being in 2 varsities in my high school was important to have better college prospects. It took me around 1 year of training to perfect my shot, and having the help of a shooting coach/assistant would have helped tremendously. Indeed, shooting a basketball properly is nothing short of art, and most people who watch or play the sport on the highest level know that. Some of the main principles to keep in mind when shooting a basketball are the following:

- Proper grip on the ball
- Eyes focused on the target or rim
- Smooth and controlled release
- Follow-through with the shooting hand
- Adequate arc on the shot trajectory
- Consistent and balanced body alignment
- Proper footwork and positioning
- Tight elbow with 90 degrees angle

Because the shot has so many different aspects it is important to have something or someone helping with the tracking of these different metrics. One might ask why good performance in basketball, and more specifically in shooting, is such an important subject. The reason is that it provides numerous physical, mental, and social benefits that can positively impact the players' overall well-being and personal development. Basketball helps individuals improve their physical fitness by enhancing cardiovascular endurance, agility, coordination, and strength. Learning how to shoot not only enhances hand-eye coordination but also develops fine motor skills and body control. Furthermore, basketball and shooting skills contribute to mental development by promoting discipline, concentration, strategic thinking, and decision-making under pressure. It helps individuals develop resilience, perseverance, and a growth mindset as they strive to improve their shooting accuracy and overall performance. In addition to the physical and mental benefits, learning basketball and shooting can foster social connections and teamwork. It provides individuals with the opportunity to engage in competition that could be friendly and more fierce in a professional setting, build relationships with other players as well as coaches, and also develop the communication skills that allows them to thrive within a team. Last but not least, basketball is the source of many families, and performing well on the highest level plays a huge difference in their quality of life. We also know that NBA players earn millions of dollars every year, and that the NBA brings in around 10 Billion US dollars annually in revenue (2022) [2]. Stephen Curry is an example of a player earning millions of dollars because of his shot (amongst other skills), as he is known to be the best shooter in history, as he signed a 4 year 215M\$ contract [3]. Since Stephen Curry has a huge

skillset, taking a player who primarily relies on his shot would help put things into perspective. JJ Reddick is a shooting guard who is a great shooter without having great defense, physicality, finishing, amongst other skills. He earned 23.5M\$ in 2018 in one of the final years of his career [4].

Thus, we present our project, which is a Computer Vision Tool to do “Basketball Shot Analysis”, to help basketball players and aspiring players to learn how to shoot or improve their shots by tracking some of the metrics mentioned above that are essential to having a good basketball shot. This will require tracking the body landmarks of players, the basketball, the rim, and the trajectory of the shot and implementing a certain logic to track the movements. We will also develop a web interface to let the users upload their video and get the relevant information to their shot. We will also be implementing a feature that lets the user submit 2 videos, their shot and the shot of a professional basketball player, and they will see a table that includes the comparison of the different metrics in the shot.

Project Design

Basketball and Rim Detection

The first part we worked on was the detection of the ball and the rim, as their detection is essential for the rest of the project. We had many options to choose from when it comes to doing the detection. One popular approach is to employ feature-based methods, where distinctive visual features on the basketball and rim, such as corners or

edges, are detected and tracked throughout a video sequence. One of these methods would be the template-matching method we used in the last MP we had for the class, which worked quite well to detect the face of the individual who was posing in front of the camera. Other techniques that could have been used are the Scale-Invariant Feature Transform (SIFT) or Speeded-Up Robust Features (SURF). Another option is to utilize object detection algorithms, such as the You Only Look Once (YOLO) or Single Shot MultiBox Detector (SSD), to detect and track the basketball and rim as bounding boxes within each frame of a video. These algorithms are trained on large annotated datasets and can provide efficient and accurate object localization. Since these models have big capacity and are trained to detect many classes and many bounding boxes within the same image, fine-tuning them for a certain number of epochs could be enough to have a good detection model, while also being flexible to some of the changes that might happen in our frames. Also, we had the option to use simpler models like smaller CNNs or RNNs that could be a good fit for the task we have, as the smaller capacity means that we could train them from scratch, although a very big challenge would be to find a dataset big enough to train such models from scratch. The first thing we tried was to use template-based matching to detect the basketball and rim (similar to what we did in MP#7), but this was problematic for many reasons, mainly because it was hard to adapt to all of the possibilities of images and angles using this technique. For example, the recording of the video can happen for different sides and from different angles, also, it might be happening with different backgrounds and different light conditions, some of which will make all of the scenery hard to work with the template-base matching (high overlap of colors in background with the ball). Also,

when the ball enters the rim, the matching seems to break. For these reasons, we decided to work with YOLO models for many reasons. Modern object recognition algorithm YOLO functions in real-time and offers quick and effective tracking performance. By partitioning the image into a grid and forecasting bounding boxes and class probabilities for each grid cell, it can concurrently detect many objects within a frame, including the basketball and rim. This makes it possible to locate the things of interest with accuracy and precision. YOLO does not rely on explicitly preset features or object descriptors, in contrast to fundamental computer vision approaches like feature-based methods. Instead, during the training phase, it learns to detect objects directly from data. Because of this, YOLO is more resistant to changes in appearance, lighting, and perspective. It excels in tracking objects because it can handle objects of various scales, orientations, and sizes. Additionally, YOLO is effective enough to process video frames in real-time, which makes it appropriate for applications that demand quick tracking results. This is especially useful in situations like live sports broadcasting or real-time analytics where the basketball and rim are moving quickly or where low-latency tracking is required.

The next step was to choose the dataset we were going to use as we needed to train or fine-tune a yolo model, which would need at least a couple thousands of annotated images. It was difficult to find a dataset that is fitting to our exact use case, which is someone shooting a basketball in an isolated setting. Finding such a dataset would have been perfect but we couldn't find an exact match. On a side note, we considered cultivating our dataset but it would have taken a lot of time and effort for a very small part of the project. So, we chose to go with an open-sourced dataset on "roboflow"

which includes around 3,000 images that are annotated for the presence of basketballs, people, and rims. We will only use basketball and rim in our analysis for now, although the use of people detection might be useful for later improvements in our product. We first tried to train a model from scratch, but it didn't do a good job generalizing as the dataset is very small and the model's capacity is very large, which would require a big dataset to get good results on the validation and test sets. The obvious alternative is fine-tuning the model. We chose YoloV5 [5], which is the state of the art model for Yolo models. Other options would have been choosing an earlier version of Yolo, but we didn't find any reasons to do so. Also, we considered a smaller version of Yolo called Tiny-Yolo, since it is faster than the standard Yolo model, but after trying to fine-tune on this model, we found that the model trained had much lower accuracy on the test set compared to YoloV5. We will see the training results in the next section.

The main model was trained with the following hyperparameters:

- Momentum: 0.95
- Weight Decay: 0.0005
- Epochs: 40
- Batch Size: 16
- Optimizer: SGD
- IOU threshold: 0.2 (for evaluation)

These are some of the hyperparameters, and the rest can be seen in our code. We also had 3 warmup epochs with a momentum of 0.8, which is logically lower due to the fact that we are warming up our model.

Body Landmarks and Angle Detection

After training the model to detect the basketball and the rim, we had to implement the detection of the different landmarks on the shooter's body to be able to track the important metrics specified above. To do this, we considered using "open pose" and "media pipe" but we ended up going for the latter because the former's main implementation is using C++ and we found that incorporating the library with our application, which is based on Python. Implementing using media pipe was much better in terms of installation, setting it up in the environment, and ease of running in our python code. We wrote code to detect the main landmarks on the shooter's body which include but are not limited to the shooter's right and left elbows, hands, knees, shoulders, feet, and their head. After doing so, we calculated the angle for the knee by taking the intersection of the (foot, knee) and (knee, hip) segments and getting the angle at the intersection, and the angle for the elbow by taking the intersection of the (shoulder, elbow) and (elbow, hand) segments and getting the angle at the intersection.

Shot Detection Logic

Given the video as input, we take it in our code and loop over the different frames and apply the previous sections as well as the logic we are going to describe in this section. Basically, for every frame, we run the mediapipe pipeline as well as the object detection model. This gives us the coordinates of the detection boxes at every time step, from which we choose only the boxes corresponding to the highest probability of a basketball and the highest probability of a rim. Also, we get the coordinates of the body landmarks as well as the angles for the knees and elbows, which we later showcase in the video.

When it comes to the logic we implemented, we had many possibilities at every step and at every detection we were trying to make. We used variables to track the state at the different stages of the shot: release started, release ended (shot start), shot tracking between shot start and shot end, shot end, score/miss, reset variables and start with the next shot. Starting with the release start, we chose to start it when the ball's y coordinate is higher than the shoulder's y coordinate, since it is illegal to dribble higher than shoulder height. Other options would have been when both hands are close together and touching the ball, but we found the first choice to be better when experimenting, since keeping track of both hands when taking a video for the left or right side is difficult and this breaks the logic in some instances. After having the shot release started, the shot release ends when the ball moves away from the shooter's hand. At this instant, we register the shot angle, by taking the angle of the last 2 coordinates with the x-axis. This is an important measure to determine the quality of a shot. After that, we start with the shot tracking, where we track every coordinate of the ball which will be later used for trajectory fit. We also track if the ball is still moving close to the basket, and when it gets near the rim (or inside the basket) and starts moving away and is away by a certain threshold, we know that the shot has ended. If it moved upwards from the rim for example, we know it is a miss. If it can be detected inside the rim or right under the basket, we know that the shooter scored the shot. This is where we stop tracking the coordinates of the shot and look to reset the shot, which happens when the hands of the player touch the basketball again, which means that the player retrieved the ball and will soon shoot again, which will also be tracked.

An important detail is that the knee angles and elbow angles are only relevant between release start and release end, as the minimums of these values should be in a certain range. For example, the elbow angle should be between 80 and 95 degrees.

Analysis After tracking

After tracking the shot, we fit a trajectory for the ball. We do so by doing a best fit of the ball trajectory during shot tracking with a quadratic function of the form ax^2+bx+c , which returns a curve fit of the shot which can be later analyzed, and it is generally recommended to have a higher arc on the shot, as seen in Stephen Curry's shot. Curry's shot is a good baseline for the perfect shot and he holds many shooting records including most 3 pointers made in NBA history. Also, for every shot, we return the following:

- Knee Angle on release
- Elbow Angle on release
- Miss/Hit
- Trajectory
- Release Angle
- Release Time

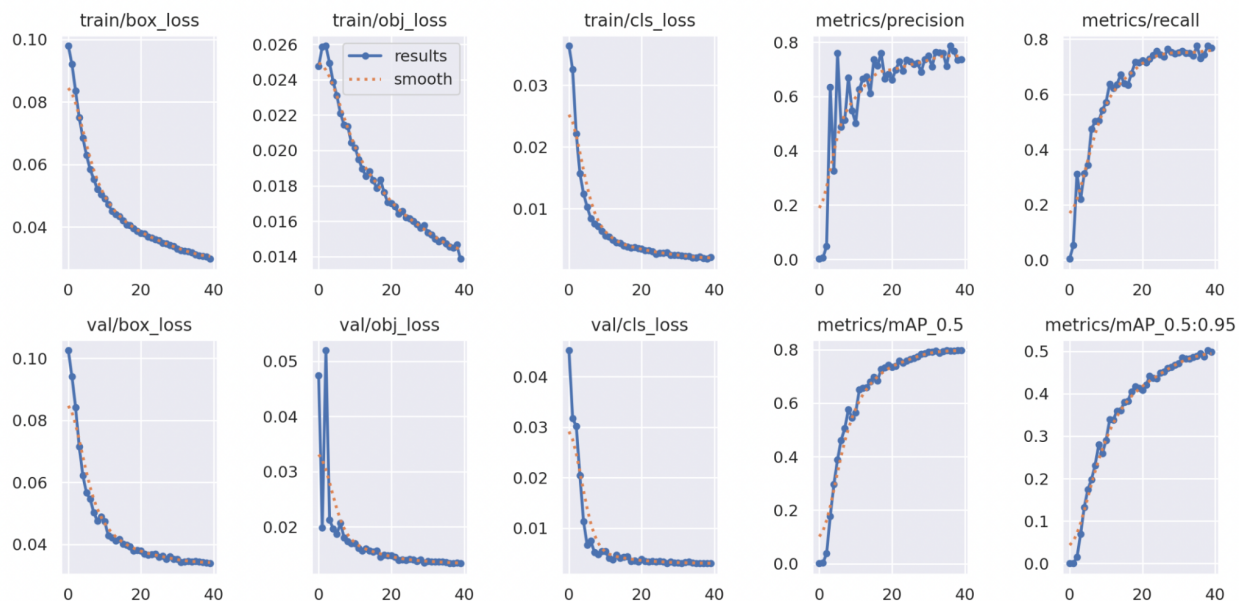
Web Interface

For the web interface, we used a python library called "streamlit" for many reasons. "Streamlit" is a Python library that focuses on the creation of web applications for users to interact with as well as plots and dashboards, which is perfect for our use case as we

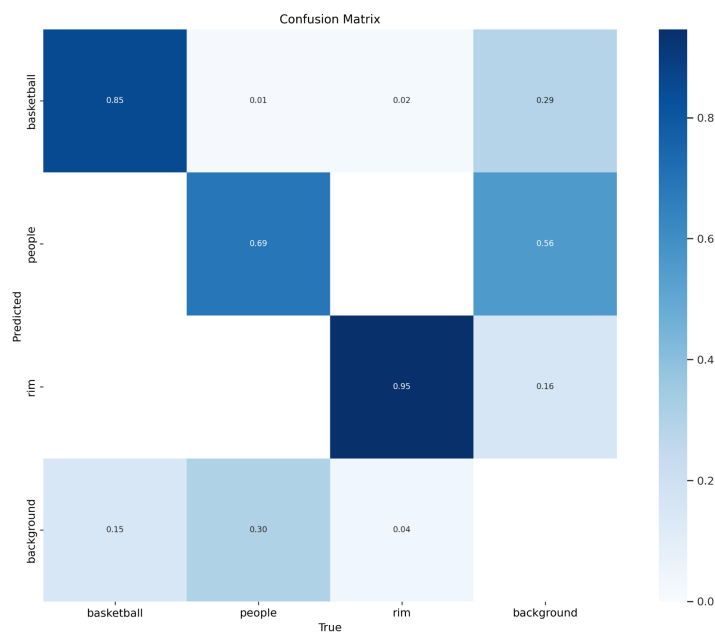
want to show tables and images related to the shots the users want to track. It also offers an API that is easy to use and easy to adapt with our code, allowing us to implement our Python scripts by importing the code from different classes and functions within the same directory and later implement it as part of the web application. We have two interfaces that are available for users: the first one is an interface where the user can submit a video of one or multiple shots, and we return a table containing entries for every shot with the details mentioned at the top of the page. The second one is an interface where the user can submit a video of them shooting, and another video of a professional basketball player shooting, and he will get a table comparing the metrics of their shot and the metrics of the professional player's shot. The interface can be seen in the YouTube video we submitted.

Results and Analysis

For the trained Yolo V5 model, the results of the fine-tuning can be seen in the images below:



As seen in the figures, the train and val loss decreased significantly at the end of the training epochs, and all the relevant training metrics we wanted to improve scored much better towards the end of training and started showing signs of a plateau, especially for the validation set metrics. We can also confusion matrix for the training set as defined by the IOU threshold mentioned before:



Knowing that we only care about basketballs and rims, the 0.85 and 0.95 values are pretty good for our predictions. Also, after investigating where the model fails, we find that the images are very unclear, unlike our problem statement, where the ball and rims are much clearer, since videos will mostly be taken from our mobile phones' cameras which have very high resolution. Other figures can be seen in our GitHub repository, such as the PR curve, F1 curve, and labels EDA plots.

After running the model on some of the videos we recorded ourselves, and some either shooting videos we got from the Web, we saw that our model was performing really well, especially for the video that we took in the indoor basketball courts at Northwestern University, as the background was easily discernible from the ball and the rim. We were satisfied with the results and decided to move on to the other parts of the project.

When it comes to the rest of the logic, it was being tested as we went through the different implementations and we couldn't have any metrics to track how well our system was doing, but we had around 10 videos we were constantly testing that included shots from different angles and different sides, and we only moved forward to different components when the component we were on worked for all of the videos. Specific examples for videos can be seen in the YouTube video and a demo can be run by cloning the GitHub repo which includes all of the details necessary to run the system on any machine that has Python running. We also examined the metrics for some of the videos to validate the results, and we saw that the angles matched what we wanted, the number of shots taken was always identical to the one seen in the videos, the hit/score measure was always correct, and after trying to shoot with different angles and trajectories, we saw the system adapt properly.

When it comes to the trajectory fitting, the results were also satisfactory and one example can be seen in the web interface results shown below.

For the web interface, which is seen in the YouTube demo, and was presented in the presentation in class, everything was working smoothly as we wanted it to. A screenshot of the interface can be seen below. In the video, I can be seen shooting 3

shots and the results can be seen in the table and one of the shot traces can also be seen.

Select a tab

Shooting Analysis

Select a tab

Shooting Analysis

Shooting Analysis Application

Upload your shooting video

Drag and drop file here

Limit 200MB per file • MP4, MOV

Browse files

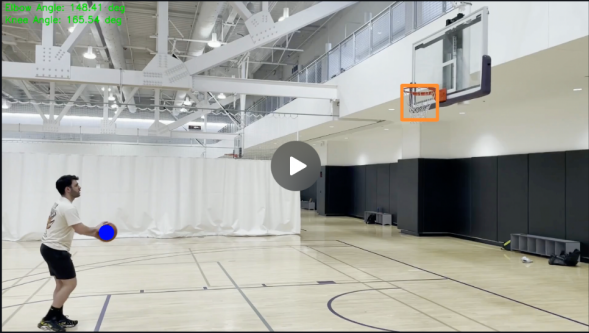
simon.mp4

10.0MB

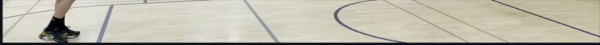
Processed Video

Elbow Angle: 148.41 deg

Knee Angle: 165.54 deg



Table

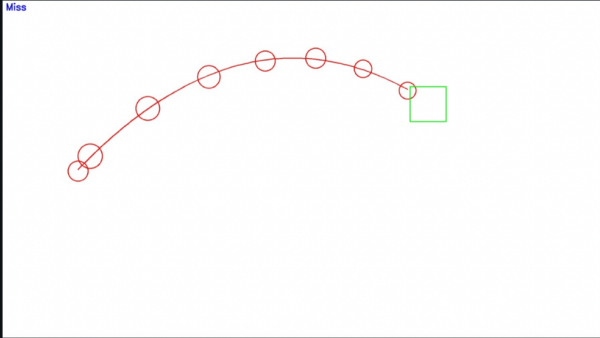


Table

	Shooting Time (Sec)	Release Angle (Deg)	Make or Miss	Knee Angle (Deg)	Elbow Angle (Deg)
Shot 1	0.8	50.5	Miss	155.785	84.977
Shot 2	1.1	42.255	Miss	145.141	81.04
Shot 3	1.0	40.257	Miss	149.657	78.086
Averages	0.97 sec	44.34 Deg	0.0% Shooting Percent	150.19 Deg	81.37 Deg

Shot Traces

Miss



Overall, we were really satisfied with the results as they performed well on an array of videos that are very diverse and different from one another. Some limitations are going to be discussed in the future work section.

Remarks and Future Work

After finishing the project work, we had some ideas for future work that could ameliorate the system and solve some of the limitations we had.

The first thing is to create our own curated dataset that would fit perfectly for our use case. Having a big dataset where the images look exactly like the ones we would submit to our application, we would be able to fine-tune or even train a bigger model that wouldn't lose track of the ball or the rim during the shooting motion. This happens scarcely in our test data, and we implemented some logic to take care of these exceptions by taking the last found coordinate for the basketball and/or the rim, but having a better model would definitely help and make the system better.

Also, in the logic, having a way to detect if the shooter is left-handed or right-handed would help as we could perfect the tracking logic when this information is available. Also, if the shooter is left-handed it would be recommended for them to track the shot from their left side, and vice-versa for a right-handed shooter, as it would help us in tracking the elbow angle of the shooting hand and would help the model have a clearer view of the model, without the non-shooting hand hiding some of it.

In addition, after taking the advice of many shooting coaches, we could give recommendations for how to improve the shot given the metrics we extracted. This might be a little challenging since there is no absolute consensus for the metrics, but

having these options or letting the user input the ranges he desires to abide by would help. We tried to remediate this by giving the players the options to compare their shots to another shot which would help them in knowing what parts to change in their shots to have a better form.

Also, in some cases, the media pipe library ceases to work as it should and causes some minor problems in the logic. We will have to research how to fix this as the library is giving as is and there is no clear way to improve it. Maybe applying some image for the image would help in detecting the landmarks better.

Course Feedback

I found the course to be very beneficial, as it built up from the basic computer vision techniques, and later covered some of the more advanced techniques, which helped in assimilating the material gradually. Also, I believe that the homeworks helped a lot in learning the concepts, as we had to understand everything in detail to be able to have a proper implementation. Also, the homeworks description had a very good level of detail, and the fact that we needed to write a report helped us summarize everything we learned into words. In my opinion, we should have had 2 more assignments for the last few subjects discussed in class, even if they were optional, as having such assignments would have also helped in understanding the material we learned later in the course better, especially that the material felt harder.

Also, the flexibility we had in the project was great and made us something we found fun and interesting, while also having a large array of Computer Vision techniques to choose from, instead of being limited to just a few ones.

Finally, I really enjoyed Dr. Wu's approach to learning as he focused on teaching us the theoretical basis and the intuition behind the algorithms. Also, he always gave us real-life examples that would help us better grasp and understand the topics discussed.

References

- [1] Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the dangers of stochastic parrots. *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. <https://doi.org/10.1145/3442188.3445922>
- [2] Ozanian, M. (2022, October 29). *NBA Team values 2022: For the first time in two decades, the top spot goes to a franchise that's not the Knicks or Lakers*. Forbes. <https://www.forbes.com/sites/mikeozanian/2022/10/27/nba-team-values-2022-for-the-first-time-in-two-decades-the-top-spot-goes-to-a-franchise-thats-not-the-knicks-or-lakers/?sh=d630031ccecf>
- [3] Janie McCauley | The Associated Press. (2021, August 7). *Four more years: Steph Curry finalizes \$215m extension with warriors*. NBA.com. <https://www.nba.com/news/four-more-years-stephen-curry-finalizes-215m-extension-with-warriors>
- [4] Forbes Magazine. (n.d.). *J.J. Redick*. Forbes. <https://www.forbes.com/profile/jj-redick/?sh=25dc47e91d0d>
- [5] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. <https://doi.org/10.1109/cvpr.2016.91>