Using an Autoencoder for Statistical Anomalies: A Kobe Bryant Analysis

Sebastian Algharaballi-Yanow

Cognitive Science Department, University of California, San Diego, 9500 Gilman Dr, La Jolla, 92093, California, USA.

1 Abstract

This project showcases the development of a custom autoencoder deep learning algorithm that is then used to detect anomalies or unusual patterns in Kobe Bryant's regular season and playoff statistics from the 2008-2009 NBA season. After importing, cleaning, and preprocessing two datasets that contained Kobe's advanced stats for that season (including field goal percentage, +/-, and 18 other indicative basketball statistics), I developed the autoencoder using a PyTorch encoder and decoder architecture. The encoder was designed with linear layers followed by ReLU activation, while the decoder also utilized linear layers with the addition of a sigmoid activation function. After training the algorithm with a mean-squared error (MSE) loss function and an Adam optimizer, I analyzed the encoded representations of the input data, compared the reconstruction errors of the encoded data with a predefined threshold, and evaluated the model using standard machine learning metrics (precision, recall, accuracy, and F1-score). These were the steps that ultimately led to being successful in discovering outliers during this season of Kobe Bryant's career.

2 Introduction

Basketball legend Kobe Bryant was one of the most influential and successful athletes of all time. His NBA record 20 seasons with the Los Angeles Lakers was full of historic and one-of-a-kind accomplishments, including 5 NBA championships, an 81-point scoring game, 18 All-Star appearances, an NBA MVP award, an NBA hall-of-fame inductee, and so much more [1]. His career spanned a total of 1346 NBA games played from 1996 to 2016, housing some of the most impressive statistical findings in the history of the league. Bryant was known as a fierce competitor on the court, a

player that possessed an all-around skill set that grants him the designation of one of the greatest players to ever play the game. Throughout his career, he excelled at scoring the ball in all forms possible, setting up his teammates for scoring opportunities through assists, playing stellar defense while recording rebounds, steals, and blocks, and affecting the overall game in such a positive way that his team was often on the winning side in each game that he played. Of course, every basketball player has their ups and downs over the course of an overall season. Even the most prominent NBA talents have stretches of games where their stat lines fall short of what that player is capable of, and instances where players continually outperform their season averages for varying seasonal stretches.

The phenomenon of athletic slumps and moments of excellence has been studied throughout the NBA. One researcher explored the well-known "sophomore slump" typically seen in second-year NBA players, and was able to discover that there are elements of both randomness and concrete instances where players in their second overall season experience sudden decreases in statistical performance [2]. Another group determined that players tend to slump a lot more after signing a large, long-term contract, and were also able to find that players possess statistical boosts when in the last year of their contract and working towards a salary increase (also known as a "contract year") [3]. An additional researcher found that peak player performance occurred with 3 days between recently played games, and dipped when teams had to travel across the country in a short amount of time between games [4]. Regardless the matter, NBA players are prone to performance peaks and dips at any given time, and there are a multitude of reasons why this may be the case.

Kobe Bryant's 2008-2009 NBA campaign was one of his best, capped off by an NBA Finals victory and subsequent Finals MVP. Many consider this season to be one of the most well-rounded and complete of Bryant's career, a season in which he played a total of 105 regular season and playoff games. I want to take a deeper dive into this masterful season and analyze whether Bryant was able to outperform his outstanding averages at certain points in the season. I also want to determine if there was stretches during the season where Bryant fell short of his standards, and connect the dots as to why he might have faced particular ups or downs throughout the campaign.

One of the best ways to perform this task is through the power of deep learning. Autoencoders, a deep learning algorithm, have shown tremendous potential in uncovering patterns and extracting meaningful information from complex datasets. Autoencoders are neural networks designed to learn efficient representations of the input data by encoding it into a lower-dimensional latent space and then decoding it back to its original form. In the NBA world, autoencoders have been used to track and uncover player movement on the court [5]. They've also been used to uncover offensive basketball techniques in latent court space to develop offensive game plans that are proven to lead to on-court success [6]. In the context of analyzing Kobe Bryant's performance during the 2008-2009 NBA season, autoencoders can help us identify distinct trends in his statistical performance. By feeding the extensive dataset of Bryant's game-by-game statistics into an autoencoder, we can train the model to learn the underlying structure and representation of his performance. This can reveal hidden anomalies that might not be immediately apparent through manual analysis.

By comparing Bryant's actual statistics with the reconstructed data generated by the autoencoder, we can quantify the deviation from his expected performance levels, and reconnect large, positive deviations as peaks in performance while considering strong, negative deviations as a slump. We can then take a closer look at the games where Bryant faced performance anomalies and correlate the patterns to key factors such as playing through an injury, facing a weaker (or stronger) opponent, or even going through the usual structure of an NBA schedule, where continual games on one's home court could lead to a peak in performance, while a long stretch with continuous travel and road games could lead to worse performance.

3 Methods

3.1 Data Collection

Before developing the structure of the autoencoder, I needed to collect high quality data from each of Kobe Bryant's games from the 2008-2009 NBA season. It was crucial to obtain his most advanced statistics to ensure the results from this experiment are valid and can potentially be used for analyzing future seasons. Advanced basketball stats are publicly available information, and are easily accessible on Basketball Reference, a database that houses a large amount of basketball statistics [7]. Basketball Reference collected a precise game-log corresponding to each game that Bryant played during this season, including both regular season and playoff games. Their data consisted of advanced basketball statistics such as opponent played, field goal, three-point, and free throw percentages, rebounding and defensive numbers, and even the +/- measurement, a statistic that takes the overall minutes a player was on the floor and reflects how many total points their team scored or conceded throughout the total time that player was in the game. This was the ideal data to extract and begin working with in conjunction with my autoencoder, as these are statistics that ultimately shed light on Bryant's performance each and every night he was playing. It was also helpful that Bryant played in all 82 regular season games, in addition to all 23 playoff games. This meant that I had a large mass of game data for my autoencoder to process, which would ultimately improve the results of my experiment once completed.

3.2 Data Processing

I used the Pandas and Ski-Kit learn libraries from the Python programming language for all of my necessary data processing steps. Bryant's regular season and playoff data were stored in separate .csv files, so the first step before anything else was to extract these files on to my personal device and read them into Python. Then, in order to gather both sets of data into one dataframe, I used the "concat()" method. Having one dataframe with all of the game statistics would make it a lot easier to eventually train my autoencoder, so this was a very important step.

Next, I needed to drop the columns that were not going to be useful when implementing the autoencoder algorithm. The data consisted of several unneeded columns, such as Kobe's age by day each game, the date of games played, and total minutes played. After performing this step, my dataframe consisted of 105 rows (total games

played in 2008-2009), along with 20 columns, each representing a crucial statistic that can be used to determine Bryant's performance in each of his 105 games.

After this, the next step was to perform normalization on the numerical columns using the Min-Max scaling technique from Ski-Kit learn. By normalizing the data, you ensure that all the features the autoencoder is being trained on have similar scales, which prevents certain features from dominating the learning process simply because they have larger values. This is especially the case regarding basketball, since scoring 38 points while having 8 assists should be considered a successful game, even though the values of 38 and 8 are far apart from each other. Normalization also makes sure that the data falls within a specific range. The Min-Max technique ensures that all of my data points are between 0 and 1. This can help the autoencoder model to learn more efficiently, as it becomes less sensitive to outliers and extreme values. After normalization, the autoencoder can truly focus on capturing meaningful patterns while also reducing the impact of noisy data. To accomplish this, I imported the "MinMaxScaler" class from Ski-Kit's preprocessing module, and selected each of the 20 columns in my dataset to normalize. I then created a copy of the original dataframe to ensure that I didn't corrupt or change the statistics gathered from the database. Finally, I applied the he "fit_transform()" method on the normalized dataframe, specified each of the 20 columns to normalize, and reassigned the normalized columns back to the original table. This ensured that the normalization changes are reflected in the dataset I would be using to train my autoencoder. My new dataframe was still 105x20, and each column was now on the same scale for the algorithm to train on. However, normalizing each column led to 5 of Kobe's games producing at least one "nan" value in one or more of his statistics for that game. This is quite common when applying normalization techniques, as some extreme values may lead to certain statistics being divided by very small ranges, resulting in the appearance of "nan" values. Of course, having multiple missing values in my dataframe meant that my eventual model would not be able to train properly. To mitigate this issue, I decided to drop all rows that held a missing value, since statistics for 5 games, considering I would still have a total of 100 games to work with, would not drastically affect my model in the long run. My final dataframe ended up being 100 rows of games x 20 columns of statistics, which was an excellent amount of data to successfully train my autoencoder.

3.3 Autoencoder Implementation (with PyTorch)

After my data processing steps, I was ready to fully develop and train my autoencoder. For my task, I decided to split the data into 80% training and 20% testing. This meant that 80 random games from Kobe's season would be used for training the algorithm, and a random set of 20 games would be used for evaluating the model. If my model was built strong enough, 80 games would give the algorithm an excellent opportunity to truly learn the patterns from this season of Bryant's career and successfully detect any anomalies within the 20 games I would be using to test it on. Next, I needed to set the dimensions of the encoding and decoding layers. This is an hyperparameter specific to autoencoders. It affects the model's ability to capture meaningful patterns in the data, which, in turn, plays a crucial role in the overall performance of the algorithm. In my case, I initially set the encoding dimension to 15. This value was determined based

on the complexity of the dataset and the desired level of compression. By choosing a higher encoding dimension, the model would have more capacity to capture intricate patterns in Kobe Bryant's performance, potentially leading to a more accurate representation of his complex basketball statistics. In my initial test run, I was hopeful that a value of 15 would strike a balance between capturing essential features and avoiding excessive complexity. I was now ready create the autoencoder class with a neural network. As with all autoencoder structures, my model consisted of an encoder and a decoder, both implemented as sequential neural networks using the respective PyTorch container. The encoder contains a linear layer that reduces the input dimension to the encoding dimension and applies a ReLU activation function. The decoder performs the reverse operation, taking the encoded representation and reconstructing the original input dimensions using a linear layer and a sigmoid activation function. I then defined my loss function using a mean squared error metric (MSE), which would measure the difference between the reconstructed outputs and original inputs. I also added the Adam optimizer in order to optimize the parameters of the model. Finally, I created a training loop that iterated over mini-batches of my data. For each mini-batch, the inputs are fed through the autoencoder, and the loss is computed using the criterion defined earlier. The optimizer performs the backward pass and updates the model's parameters, and the average loss for the epoch is calculated and stored in the "loss_values" list. The reconstruction loss using MSE was also calculated throughout this process, which provided a measure of how well the autoencoder was reconstructing the inputs. Upon training completion, the encoded representations of the training and testing data are obtained by passing the respective tensors through the encoder part of the autoencoder. These encoded representations are then converted back to NumPy arrays, which would be used when further analyzing for anomalies and making conclusions on Kobe's performance throughout the testing games. After carefully implementing the above steps, I was ready to experiment and evaluate the success of my model.

4 Experiment

I began my evaluation process by generating a graph that displays both of my loss functions. Upon initial analysis, we can confidently see that the regular loss, along with the reconstruction loss, followed a very similar path in converging towards the gradient at the end of the training process. This was an excellent sighting. I now knew that the model was successfully learning and capturing the underlying patterns in Kobe Bryant's season statistics. The convergence of both losses indicates that the model was able to minimize the difference between the original input and the reconstructed output, and that the learned representation was able to capture the essential features and patterns of Kobe's performance. It also demonstrates that the autoencoder was able to effectively compress the data into a lower-dimensional representation and then reconstruct it with minimal loss of information. With this promising evaluation, I could proceed with confidence to assess additional aspects and performances of my autoencoder.

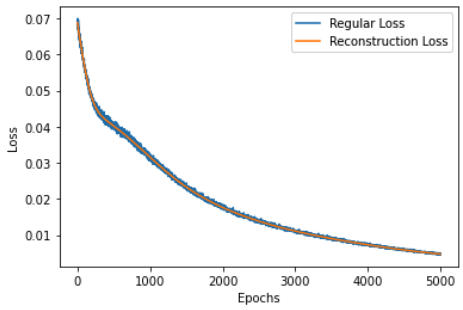

Fig. 1 Regular vs Reconstruction Loss.

Next, I wanted to analyze the distribution of reconstruction errors. These errors represent the discrepancies between Bryant's original data and the reconstructed data produced by the autoencoder model. After plotting a histogram of the reconstruction errors on the test set, I noticed that the highest frequency of errors came around the 0.005 mark. This, once again, indicates that the autoencoder is able to reconstruct the input data with very high accuracy, and suggests that the model is capturing the crucial patterns needed to detect anomalies in Kobe's season. The periodic spikes at values like 0.02 and 0.0125 could potentially indicate specific games where the model struggled to accurately classify the game as an anomaly or not. It could also be related to certain statistical patterns that are more challenging for the model to capture. This was definitely an area that needed more exploration, but overall, the visualization reinforced the notion that the model was generally performing well in reconstructing Kobe Bryant's 2008-2009 campaign.

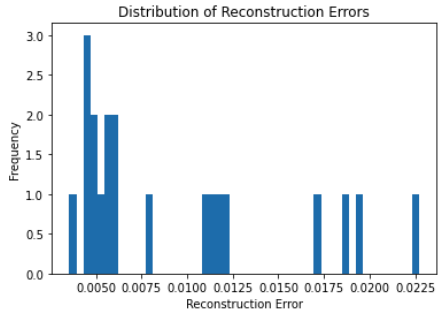

Fig. 2 Distribution of Reconstruction Errors.

My next step was to implement the "elbow method" on my test set. This is a commonly used technique for determining the optimal number of clusters in unsupervised learning, particularly in the context of k-means clustering. However, the elbow method can also be applied in other scenarios, including anomaly detection with autoencoders. It can be used to identify a suitable threshold for distinguishing between normal and

anomalous data points based on reconstruction errors. The goal is to plot the reconstruction error against different threshold indices (or cutoff values) and observe the change in the rate of decrease in the error. The elbow point in the plot represents a significant change or "bend" in the graph. By selecting a threshold at the elbow point, you're more likely to find a balance where the model can effectively detect anomalies while controlling the number of false positives. After careful integration of this step, I determined that my elbow point occurred between the threshold index (test games) of 15 and 17, which is where the highest change in the distance between reconstruction errors occurred. This would be the boundary point that the model would be using to classify games as anomalies.

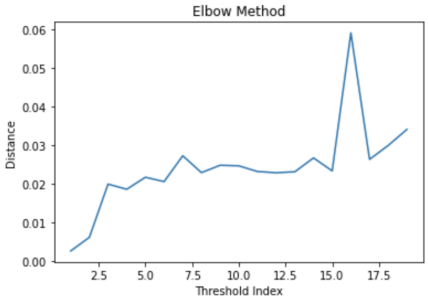

Fig. 3 Elbow Method Graph.

Now that I defined my anomaly threshold, I could graph the reconstruction errors against the testing games and precisely visualize Kobe's games that were considered anomalies. The sample indices (test games) that were present above the threshold line indicated an anomaly game. Since I could easily connect the indicies back to the testing games, I was ready to carefully analyze the graph. At the end of this process, the indicies in which the model predicted as anomalies were games 3, 4, 6, 12, 15, 16, and 17. These were the specific games in which the model predicted Bryant's performance as a deviation from this traditional outputs:

- Testing game 3: Lakers at Mavericks. November 11, 2008. 27 total points. 10/20 shooting (50%), 0/2 three-pointers (0%), 7/8 free throws (87.5%), 1 offensive rebound (ORB), 3 defensive rebounds (DRB), 1 assist, 1 steal, 0 blocks, 1 turnover, 1 foul. The Lakers won this game by 7 points.
- Testing game 4: Lakers vs Kings. November 23, 2008. 24 total points. 10/20 shooting (50%), 0/1 three-pointers (0%), 4/4 free throws (100%), 1 ORB, 3 DRBs, 6 assists, 3 steals, 1 block, 3 turnovers, 2 personal fouls. The Lakers won this game by 10 points.
- Testing game 6: Lakers at Kings. December 9, 2008. 28 total points. 9/25 shooting (36%), 3/8 three-pointers (37.5%), 7/7 free throws (100%), 2 ORBs, 2 DRBs, 3 assists, 0 steals, 0 blocks, 2 turnovers, 2 personal fouls. The Lakers lost this game by 12 points.

- Testing game 12: Lakers vs Wizards. January 22, 2009. 11 total points. 4/11 shooting (36.4%). 1/3 three-pointers (33.3%), 2/2 free throws (100%), 0 ORB, 4 DRBs, 5 assists, 0 steals, 0 blocks, 0 turnovers, 2 fouls. The Lakers won this game by 20 points.
- Testing game 15: Lakers vs Thunder. February 10, 2009. 34 total points. 13/26 shooting (50%), 0/3 three-pointers (0%), 8/8 free throws (100%), 3 ORBs, 4 DRBs, 1 assist, 0 steals, 0 blocks, 3 turnovers, 1 personal foul. The Lakers won this game by 7 points.
- Testing game 16: Lakers at Thunder. February 24, 2009. 36 total points. 13/21 shooting (61.9%), 0/3 three-pointers (0%), 10/13 free throws (76.9%), 0 ORB, 4 DRBs, 5 assists, 1 steal, 0 blocks, 4 turnovers, 0 personal fouls. The Lakers won this game by 14 points.
- Testing game 17: Lakers at Bucks. April 1, 2009. 30 total points. 10/19 shooting (52.6%), 1/2 three-pointers (50%), 9/9 free throws (100%), 1 ORB, 7 DRBs, 4 assists, 3 steals, 0 blocks, 4 turnovers, 1 personal foul. The Lakers won this game by 6 points.

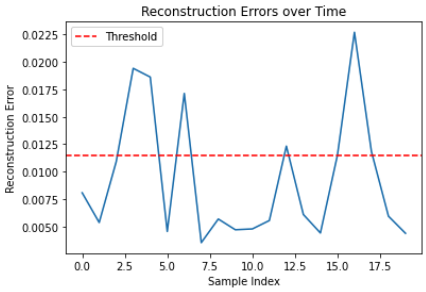

Fig. 4 Anomaly Games (Above Threshold Line).

As shown above, the games that the model predicted as outliers were very, very accurate. Kobe averaged 26.8 points, 5.2 rebounds, 4.9 assists, 1.5 steals, 0.5 blocks, 2.6 turnovers, 46.7 shooting percentage, 35.1 three-point percentage, and a 85.6 free throw percentage during this season [8]. In the first two testing games, Bryant exceeded his season averages across the board, especially though his field goal percentages and defensive statistics. These two games took place within the first twelve of the season, a point in time where one would imagine Bryant being at his freshest state after coming out of the four month offseason with plenty of rest. His performance fell in the following two training games, mainly seeing dips in his overall scoring output and field goal shooting. Upon further analysis, it is accurate to say that these dips in performance could be alluded to the difficult NBA schedule at that point in time. The loss against the Kings came at a time where the Lakers were playing a span of six games in nine nights, a schedule that seemed to be very taxing on the superstar of the team. The poor output against the Wizards could be correlated with the fact that this was the second game of a back-to-back game for the Lakers. Kobe did not play as many minutes in that game (only 28), yet the Lakers still came out of that

game with a win, also proving that his competition that night was not the greatest. Therefore, a stellar Bryant performance was not needed to secure a victory, potentially being another explanation for the fall in statistics that the model captured. Bryant's performance once again took a massive increase in each of the final three testing games, boasting statistics that were highlighted by his high scoring numbers, stellar defense and free throw shooting, and assisting the ball to his teammates. Bryant may have felt the urge to bring his play up yet another notch due to the postseason approaching at the time of these testing games, with the Lakers trying to secure the top seed going into the playoffs. My model had definitely passed the eye test, but I wanted to run a series of evaluation metrics to finalize the idea that I had created a successful model. I calculated the accuracy, precision, recall, and F1-score in consideration of the predicted and true labels of Kobe's games (anomaly versus normality). The model achieved scores of 100% in each of these categories, proving that, once again, this model was more than capable of anomaly detection throughout Kobe's 2008-2009 season.

5 Conclusion

To conclude, this project was created in an effort to explore the capabilities of deep learning autoencoders and develop a system that can be used to track and label a professional basketball player's performance over the course of an entire season. At the end of my study, it was determined that, with proper data collection, cleaning, model architecture, and fine-tuning, an autoencoder can be successfully deployed on Kobe Bryant's large mass of games from the 2008-2009 NBA season and, with 100% evaluation metrics, detect his anomaly matches. As with any machine learning project, there are many extensions one could take this experiment. It is quite possible to gather Bryant's entire career data and run this study in an effort to detect his anomaly games throughout the course of his 20 seasons. You could also extend this project to other professional sports and use my model architecture to classify anomaly games in non-basketball athletes. Autoencoders continue to become more and more powerful in the machine learning world as time passes, so the possibilities are truly endless.

This project could be the start of a new development for the NBA and the data analytic world as a whole, providing teams and analysts with invaluable information for on-court decision making, to go along with precise reasoning behind a player's increase or decrease in performance over a period of time. I hope that deep and machine learning continue to evolve to the point where my project could act as a reference to how these amazing technologies could transform the era of statistical-based player analysis to new heights. I plan to continue researching ways to further improve my model in an effort to eventually deploy it on even larger datasets. I wish to be at the forefront of the integration of these cutting-edge technologies and propel the field to unprecedented heights.

References

[1] Daubs, K.: Kobe Bryant's Best Accomplishments For Each Season Played: A Complete Break Down Of Career Honors And Awards. Fadeaway World. https://tinyurl.com/yc5etaac (2021)

[2] Marcin, T.: NBA News: The Data Behind The 'Sophomore Slump,' If It Exists At All. International Business Times. https://www.ibtimes.com/nba-news-data-behind-sophomore-slump-if-it-exists-all-1818298 (2015)

[3] White, M., Sheldon, K.: The contract year syndrome in the NBA and MLB: A classic undermining pattern. Springer. https://link.springer.com/article/10.1007/s11031-013-9389-7 (2013)

[4] Steenland, K., Deddens, J.: Effect of travel and rest on performance of professional basketball players. PubMed. https://pubmed.ncbi.nlm.nih.gov/9381060/ (1997)

[5] Acuna, D.: Unsupervised modeling of the movement of basketball players using a Deep Generative Model. University of Toronto. http://www.cs.toronto.edu/~davidj/projects/unsupervised_modeling_using_a_DGM.pdf (2015)

[6] Chao, Y., Chen, W., Peng, J., Hu, M.: Learning Robust Latent Space of Basketball Player Trajectories for Tactics Analysis. IEEE ICME. https://ieeexplore.ieee.org/document/9859960 (2022)

[7] Basketball Reference. Kobe Bryant 2008-09 Game Log. https://www.basketball-reference.com/players/b/bryanko01/gamelog/2009 (accessed on 1st June 2023)

[8] Statmuse. Kobe Bryant 2008-09 Stats. https://www.statmuse.com/nba/ask?q=kobe+bryant+2008-09+stats (2023)