**NNextPitch:** A Novel Deep Learning Approach to Major League Baseball Pitch Recognition

Author: Brenton Graham

**Problem Statement & Motivation**

Organizations across all major sports in North America have been forced to adopt data-driven approaches to decision making in order to stay current and competitive in recent decades. Major League Baseball is commonly recognized as the first major sports league to widely embrace analytics. In fact, an entire domain of sports-based analytics, termed *sabermetrics*, is dedicated to baseball-specific statistics and analysis. As such, a wealth of high-resolution public information and unexplored opportunities exist within the world of baseball.

Baseball enthusiasts would agree that success within baseball relies heavily on *the game within the game*. Discovering and exploiting small advantages within the game can pay large dividends with respect to realizing desired outcomes. Here, we focus on developing a deep learning-based method for pitch recognition using videos from Major League Baseball games. The motivation behind this is two-fold. First, a reliable pitch predictor can be advantageous to hitters in real-time at-bats. Second, pitch prediction capabilities of a deep learning model can provide pitchers with insight into how predictable they are and how they might be able to better hide their pitches. In light of these motivations, an emphasis will be placed on model interpretability.

**Related Works**

*Note:* Multiple reports focusing on MLB pitch prediction exist (in the form of published journal articles, theses, and online projects). However, this project will leverage new thoughts, approaches, and capabilities that should make this work distinguishable. These contributions will be discussed in the next section.

*Well-Cited Articles*

The articles below are well-cited journal articles that focus on binary and multi-class pitch prediction. These articles tend to use tabular data and traditional machine learning classifiers such as LDA and SVMs. Most look at MLB-level data rather than pitcher-specific data. Feature importance is provided by most which will be taken into consideration when selecting features.

1. Ganeshapillai, Gartheeban and John V. Guttag. “Predicting the Next Pitch.” (2012). [[Link](https://www.semanticscholar.org/paper/Predicting-the-Next-Pitch-Ganeshapillai-Guttag/e455030bd945ceffcbf2fc99bb12271ee9c013ff)]
2. Hoang, P., et al. 2015. A dynamic feature selection based LDA approach to baseball pitch prediction. In Trends and applications in knowledge discovery and data mining, 125–37. Springer, New York, NY. [Link]
3. Sidle, Glenn and Tran, Hien. ‘Using Multi-class Classification Methods to Predict Baseball Pitch Types’. 1 Jan. 2018: 85 – 93. [[Link](https://content.iospress.com/articles/journal-of-sports-analytics/jsa171)]
4. Hamilton, Michael et al. “Applying Machine Learning Techniques to Baseball Pitch Prediction.” International Conference on Pattern Recognition Applications and Methods (2014). [[Link](https://www.semanticscholar.org/paper/Applying-Machine-Learning-Techniques-to-Baseball-Hamilton-Hoang/c56a36aebb401fe41b34991c8a0933c0ba142caf)]

*Deep Learning Approaches*

These are a couple of approaches I found that use deep learning. They use different approaches/architectures (including an encoder-decoder framework and a LSTM RNN) than I plan to use and use tabular data for their feature sets. Results from these suggest that deep learning can outperform traditional methods.

1. No Pitch Is an Island: Pitch Prediction With Sequence-to-Sequence Deep Learning [[Link](https://community.fangraphs.com/no-pitch-is-an-island-pitch-prediction-with-sequence-to-sequence-deep-learning/)]
2. C. -C. Yu, C. -C. Chang and H. -Y. Cheng, "Decide the Next Pitch: A Pitch Prediction Model Using Attention-Based LSTM," 2022 IEEE International Conference on Multimedia and Expo Workshops (ICMEW), Taipei City, Taiwan, 2022, pp. 1-4 [[Link](https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9859411)]

*An Approach Using Video Data (A Thesis)*

This was a unique take on the pitch prediction problem. The author uses OpenPose to extract pitcher pose data from video at the pitcher’s “set position”. These data are used as input to a Random Forest classifier. Limitations here are sample size (n < 100) and the fact that the author only uses data from one frame (snapshot) in each pitch.

1. USING PITCH TIPPING FOR BASEBALL PITCH PREDICTION, Brian Ishii [[Link](https://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=3865&context=theses)]

**Methods**

The novelty of the method I am proposing comes from the following components. First, I plan to use both tabular, situational game data and video data as input features. The only deep learning methods that I have seen only use tabular data. Meanwhile, the only video-based prediction method I’ve seen uses traditional machine learning (RF), uses prohibitively small sample sizes, and does not leverage the time-series data that is available before a pitch is released (i.e., the author only uses a snapshot image from each video).

1. *Data Collection*: I’ve developed a python script that leverages Selenium and BeautifulSoup to scrape and download any baseball video from baseballsavant.mlb.com. Their website includes videos for *every* pitch thrown in the MLB since 2018. I plan to choose a subset of pitchers from a recent season and scrape videos/tabular data for every pitch they had thrown. The pictures on the next page show the effectiveness of the web scraper; I believe that using this will be the key to retrieving substantial sample sizes and getting quality results. I can send over this file if requested.

*Note:* I will need to figure out some sort of pipeline or cloud storage solution to collect/extract the data considering the large file sizes.

Graphical user interface, text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

1. *Feature Extraction (Video2Data Pipeline)*: The feature extraction method that I am going to use *most closely* resembles methods used by Ishii [7], which incorporates the use of the OpenPose package to extract positional features from video. Again, a major distinguishing factor between our work is that I will make use of the time-series data available while Ishii only looked at pose estimations from a single time point. The pipeline produces the following.

A baseball player prepares to throw a baseball

Description automatically generated with medium confidence

Chart, scatter chart

Description automatically generatedA baseball player holding a bat

Description automatically generated with low confidence

1. *Candidate Models*
   1. CNN: I plan to use a CNN that leverages an appropriate architecture. Each pitch will have 13 associated “images” (one for each body part followed). Additional time-series information might be tracked as well (such as the angle between the throwing wrist and elbow). For each image I will separately extract features with convolutional layers and then convert into a flattened vector. I will then concatenate these latent features together (along with the tabular features) and make predictions using fully connected layers with dropout/regularization.
   2. XGBoost for baseline comparison (potentially difficult for time-series)
   3. I plan to use three separate input configurations for each model (tabular + video, video only, and tabular only)
2. *Evaluation*: I plan to use accuracy, AUC (if binary classification), and MCC (if multiclass classification) for evaluation metrics
3. *Interpretability:* I want to use something like Grad-CAM to interpret the video/image data. Below is an image I extracted from a medical article [[link](https://www.nature.com/articles/s41591-020-0870-z)] I found that looks at ECG traces. I think that I can apply this in a similar fashion.

Chart, histogram

Description automatically generated

**Timeline Update**

|  |  |
| --- | --- |
| Topic | Timeline (time to completion) |
| Data Collection/Scraping | Done |
| Feature Extraction | Done |
| Modeling/Evaluation | Working On |
| Reporting | Somewhat Started |

**Not for MS Requirement**

Updates are in yellow