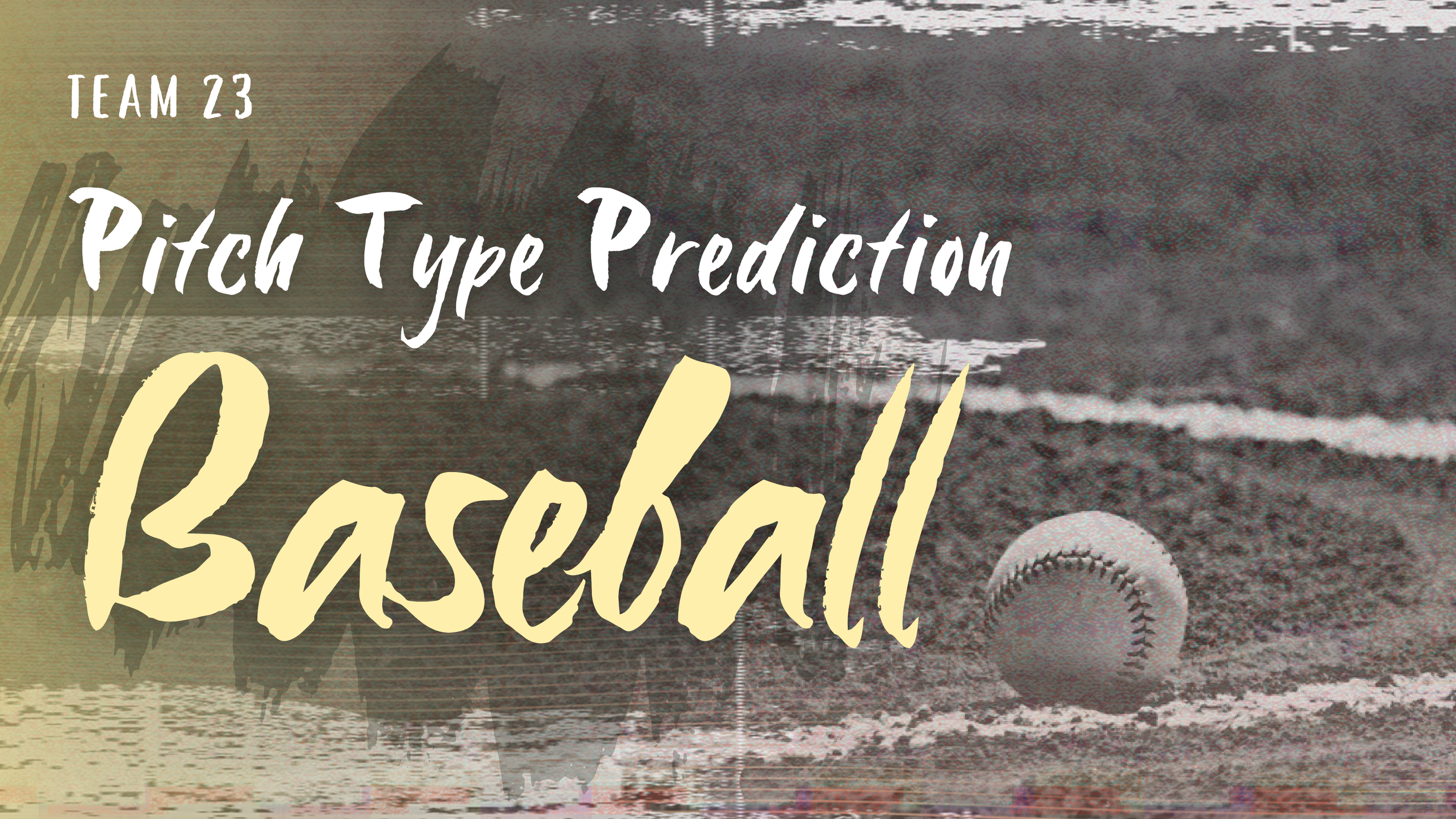
TEAM 23

Pitch Type Prediction

Baseball

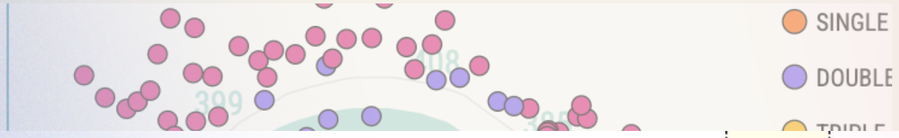


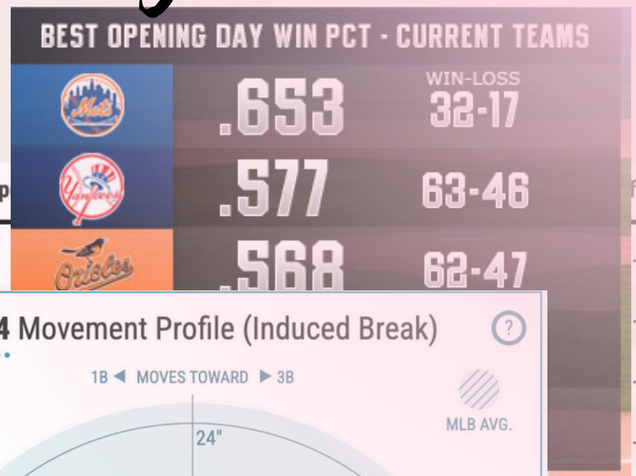

Using RNN model

Outline

- Introduction and Background
- Methodology
- Experimoents
- Results
- Discussion and Conclusion

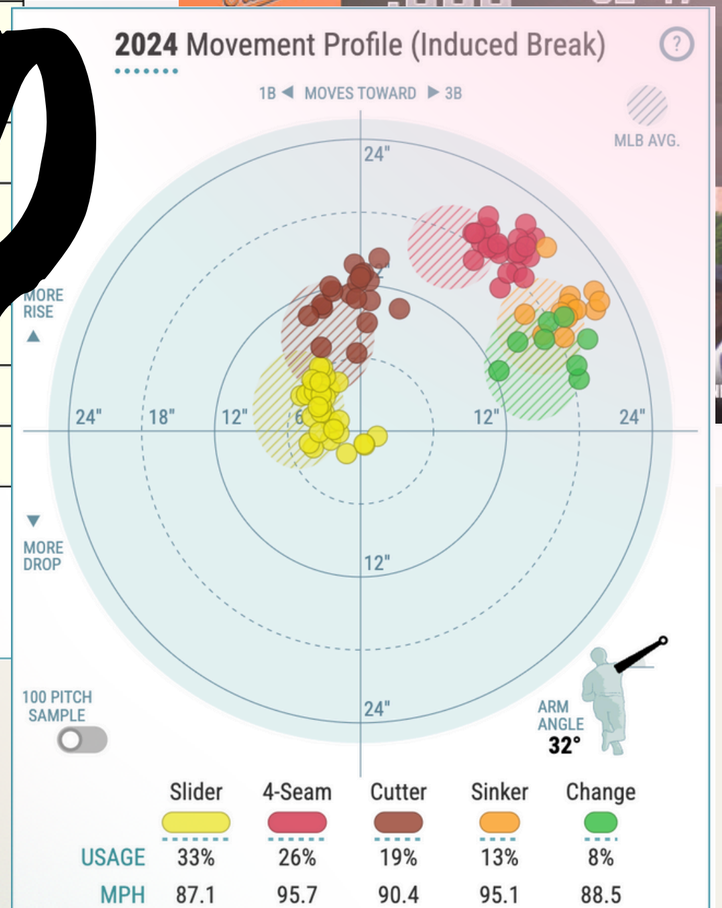
Analytics Nowadays ?





Rk.	Player	Year	BF	K%	BB%	wOBA	xwOBA	LA Sweet-Sp	ff %	Swing %
1	Skenes, Paul	2024	514	33.1	6.2	.247	.248	27.8	.7	49.8
2	Glasnow, Tyler	2024	522	32.2	6.7	.258	.254		.9	48.4
3	Skubal, Tarik	2024	753	30.3	4.6	.246			.9	51.8
4	Steele, Justin	2024	555	24.3	6.7	.271			.7	51.5
5	Wheeler, Zack	2024	787	28.5	6.6	.256			.5	51.2
6	Sale, Chris	2024	702	32.1	5.6	.260			.0	49.0
7	Crochet, Garrett	2024	595	35.1	5.5	.280			.1	53.2
8	Ryan, Joe	2024	538	27.3	4.3	.282	.264		.6	50.9
9	Greene, Hunter	2024	611	27.7	9.3	.265	.271		29.7	50.6

Texture Map
Transaction History
Visual Pitch Report
Zone Swing Profile



1. Data Processing

jldbc/**pybaseball**

Pull current and historical baseball statistics using Python (Statcast, Baseball Reference, FanGraphs)

44 Contributors | 392 Used by | 1k Stars | 339 Forks

Package Documentation

Player-Specific Queries

For a player-specific statcast query, pull pitching or batting data using the `statcast_pitcher` and `statcast_batter` functions. These take the same `start_dt` and `end_dt` arguments as the statcast function, as well as a `player_id` argument. This ID comes from MLB Advanced Media, and can be obtained using the function `playerid_lookup`. The returned columns match the set above, but filtered to rows for that specific pitcher or batter. A complete example:

```
# Find Clayton Kershaw's player id
from pybaseball import  playerid_lookup
from pybaseball import  statcast_pitcher
playerid_lookup('kershaw', 'clayton')
  name_last name_first  key_mlbam key_retro  key_bbref  key_fangraphs  mlb_played_first  mlb
0   kershaw    clayton     477132  kersc001  kershcl01           2036            2008.0

# His MLBAM ID is 477132, so we feed that as the player_id argument to the following functic
kershaw_stats = statcast_pitcher('2017-06-01', '2017-07-01', 477132)
kershaw_stats.groupby("pitch_type").release_speed.agg("mean")
pitch_type
CH    86.725000
CU    73.133333
FF    92.844622
SI    94.515385
SL    87.962381
Name: release_speed, dtype: float64
```

Input: name, date

```python
input_last_name = "shota"
input_first_name = "imanaga"
input_start_date = '2024-04-01'
input_end_date = '2024-11-02'
# Look up player ID
pitcher_id = playerid_lookup(input_first_name, input_last_name).key_mlbam[0]

# Output path for the CSV
output_root = 'input_all_data_2024.csv'

# Fetch all data for the pitcher in the given date range
data = statcast_pitcher(input_start_date, input_end_date, player_id=pitcher_id)

# Save the complete dataset to a CSV file
data.to_csv(output_root, index=False, encoding='utf-8')

print(f"All data successfully saved to {output_root}!")
```

Raw data structure

```
     pitch_type   game_date  release_speed  release_pos_x  release_pos_z  \
0            FF  2024-04-01           92.4           2.36           5.68
1            FF  2024-04-01           93.5           2.19           5.69
2            FF  2024-04-01           92.7           2.35           5.59
3            FF  2024-04-01           93.3           2.38           5.67
4            FS  2024-04-01           84.1           2.50           5.43
...         ...         ...            ...            ...            ...
2585         FF  2024-09-22           90.1           2.54           5.52
2586         FF  2024-09-22           91.3           2.51           5.49
2587         SI  2024-09-22           88.3           2.68           5.32
2588         FF  2024-09-22           90.9           2.60           5.44
2589         FS  2024-09-22           83.0           2.64           5.36

        player_name  batter  pitcher                    events  \
0     Imanaga, Shota  453568   684007                       NaN
1     Imanaga, Shota  453568   684007                       NaN
2     Imanaga, Shota  453568   684007                field_out
3     Imanaga, Shota  663898   684007                       NaN
4     Imanaga, Shota  663898   684007                       NaN
...              ...     ...      ...                       ...
2585  Imanaga, Shota  702358   684007                       NaN
2586  Imanaga, Shota  702358   684007                strikeout
2587  Imanaga, Shota  696285   684007                       NaN
2588  Imanaga, Shota  696285   684007                       NaN
2589  Imanaga, Shota  696285   684007  grounded_into_double_play

         description  ...  n_thruorder_pitcher  \
0       called_strike  ...                    1
1                ball  ...                    1
2       hit_into_play  ...                    1
3                foul  ...                    1
4     swinging_strike  ...                    1
...               ...  ...                  ...
2585             ball  ...                    3
2586  swinging_strike  ...                    3
```

```python
data = pd.read_csv('input_all_data_2024.csv')
data = data.iloc[::-1].reset_index(drop=True)

print(data)
```

Methodology

Data Preprocessing

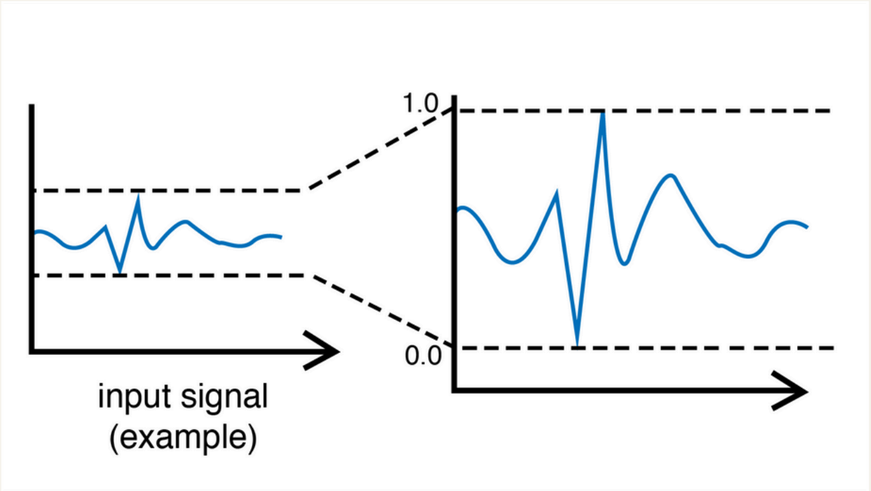
Step 1: Normalization

Some of numerical features were scaled to a range [-1, 1] using a MinMaxScaler.

RELEASE_SPEED

LAUNCH_ANGLE

HIT_DISTANCE



Data Preprocessing

Step 2: One-Hot Encoding

Categorical features were converted into binary vectors using one-hot encoding.

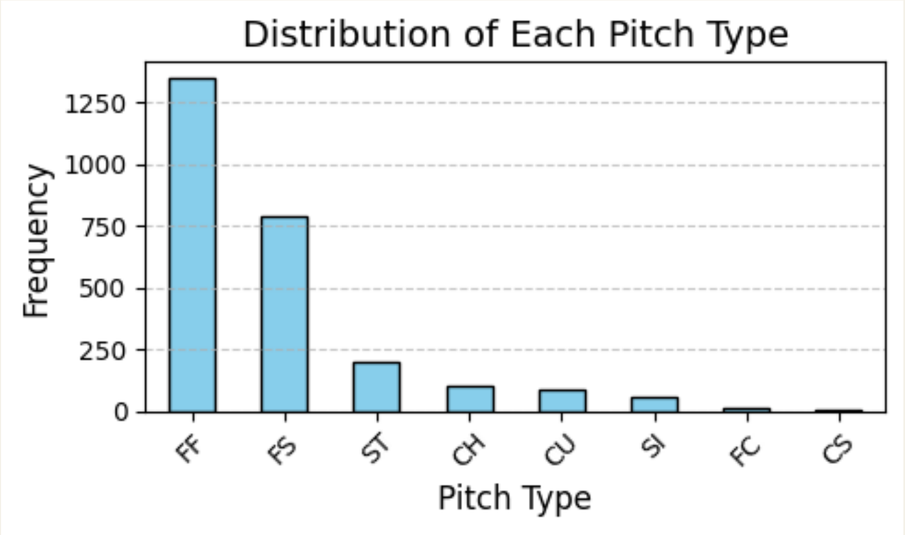
- pitch_type
- strikes
- balls
- zone

Feature	Original Value	One-Hot Encoding
pitch_type	Fastball	[1, 0, 0, 0]
strikes	2	[0, 0, 1]
zone	3	[0, 0, 1, 0, 0]

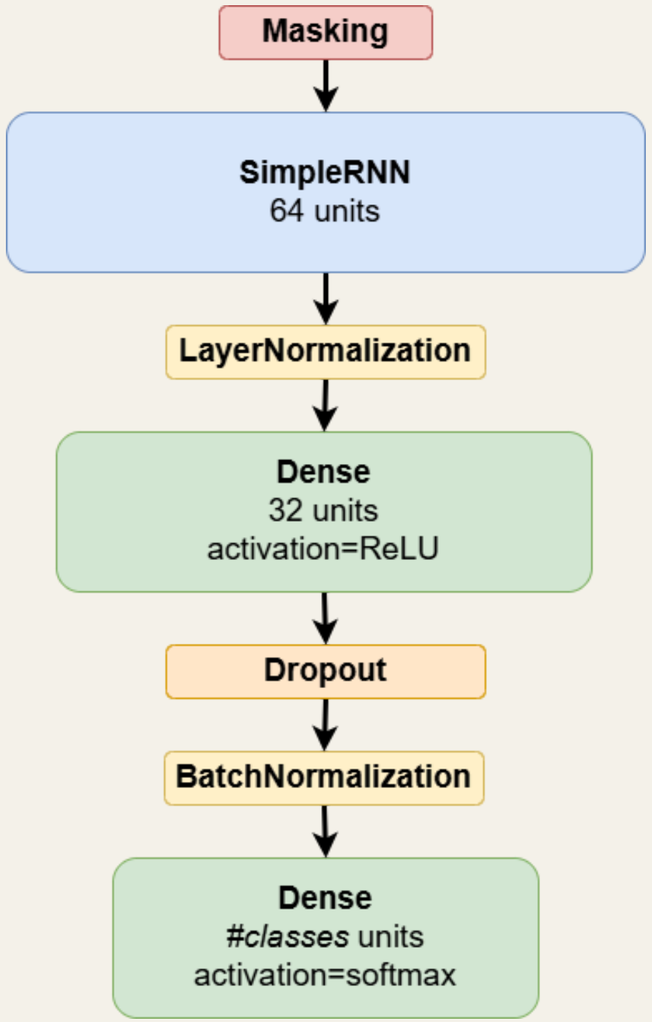
Data Preprocessing

Step 3: Adding Frequency Features

Added additional features based on the frequency of different pitch types.

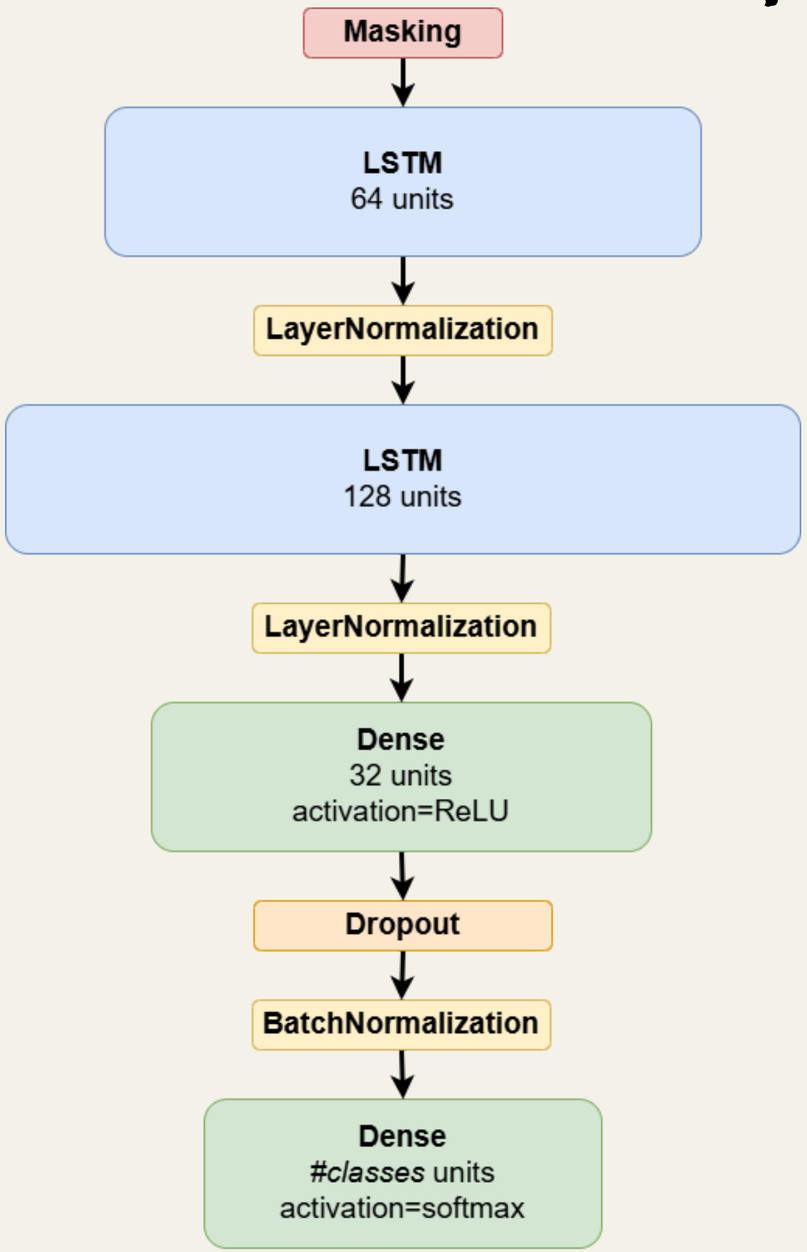


Model Design: RNN



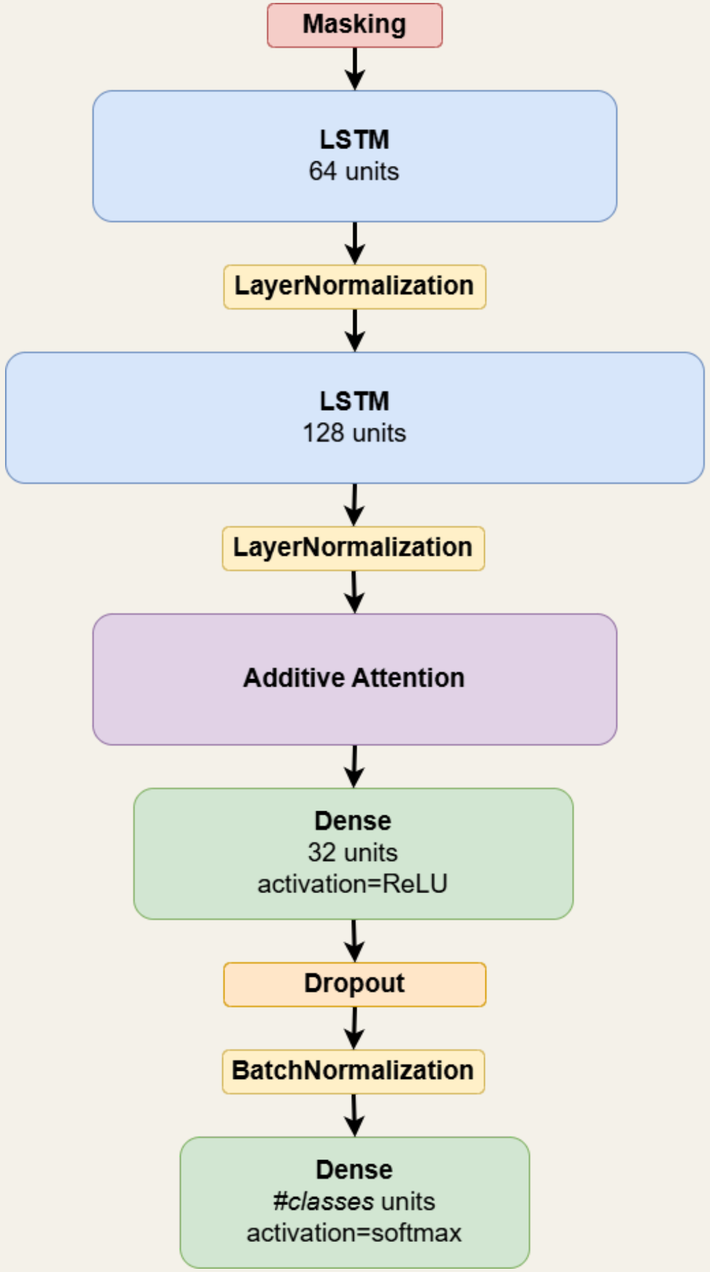
- Masking layer ensures padded values in each sequence are ignored
- Dropout layer randomly drops units to prevent overfitting
- LayerNormalization layer normalizes across features for each timestep
- BatchNormalization layer normalizes across the batch for each feature

Model Design: LSTM



- Stacked LSTM enable the model to learn complex, hierarchical sequential patterns
- It captures low-level temporal dependencies in the 1st layer and refines them into higher-level abstractions in the 2nd layer.

Model Design: LSTM with Attention

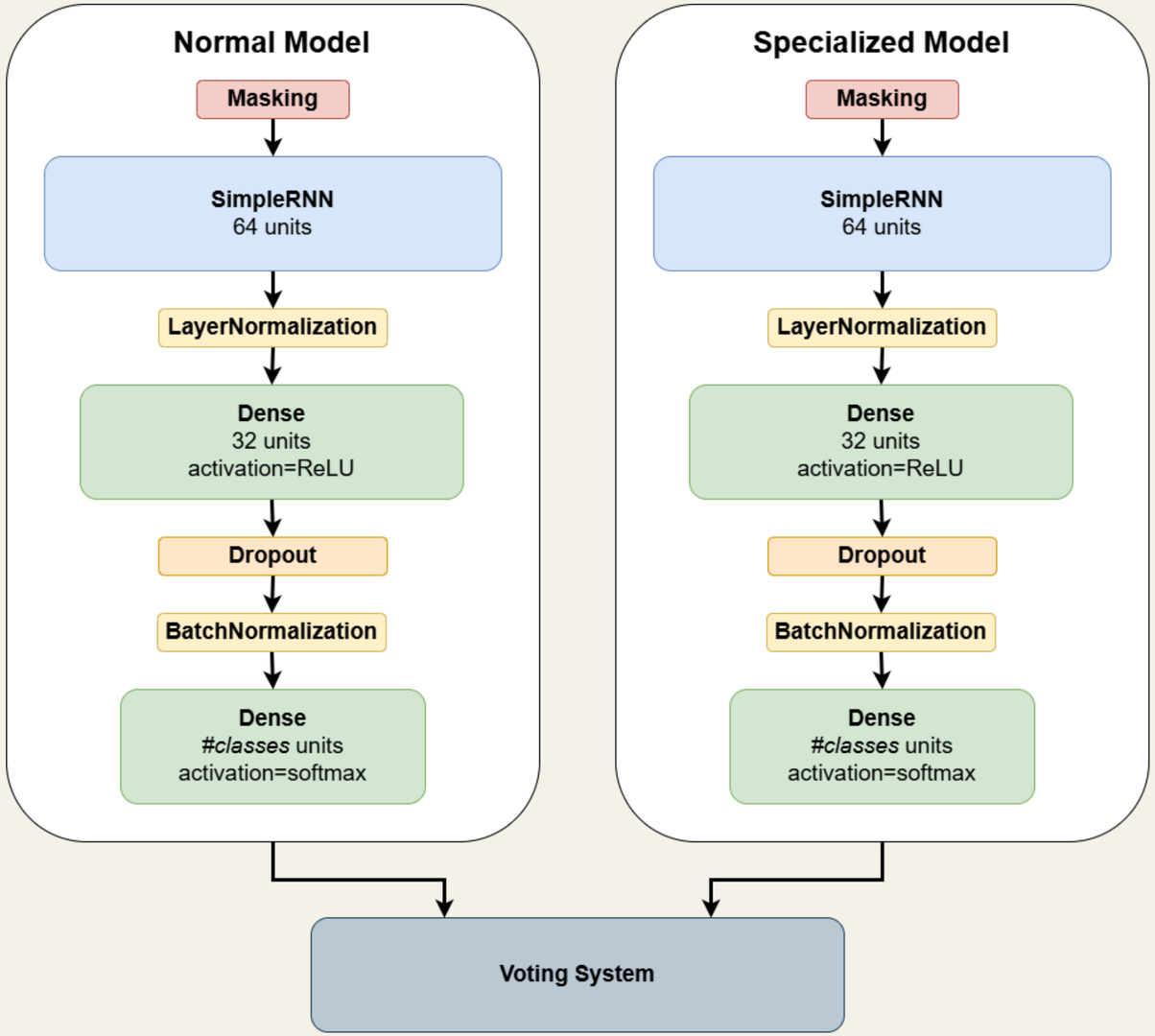


- Attention layer helps the model to focus on relevant timesteps in the input sequence

$$C = \sum_{t=1}^{T} a_t \cdot o_t$$

where a_t is the attention weights,
o_t is the output of the LSTM layer at timestep t,
C is the output of the Attention layer

Model Design: Specialist Model



- Normal model focuses on predicting fastball pitch types
- Specialized model focuses on predicting off-speed pitch types
- Voting system weights the predictions from both models and outputs the final prediction


Experiments

Data Preprocessing

To prepare the data for time-series modeling, we experimented with two methods for generating subsequences:

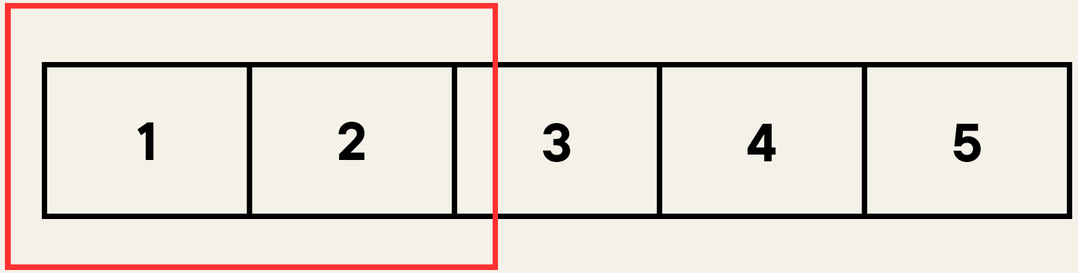
- Batter-Based
- Date-Based

Data Slicing: Take an example

input: min = 2, max = 3

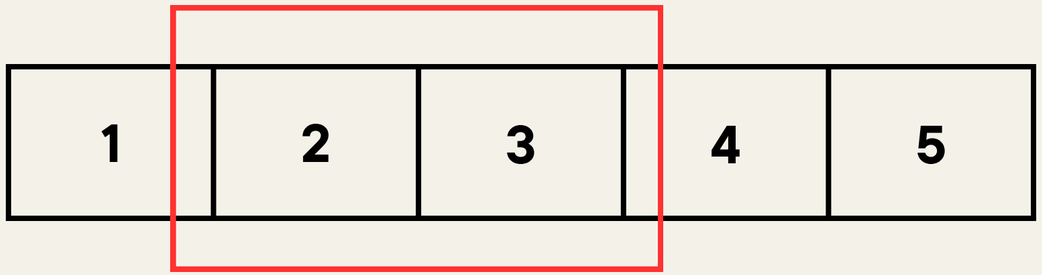
Data Slicing

length = 2



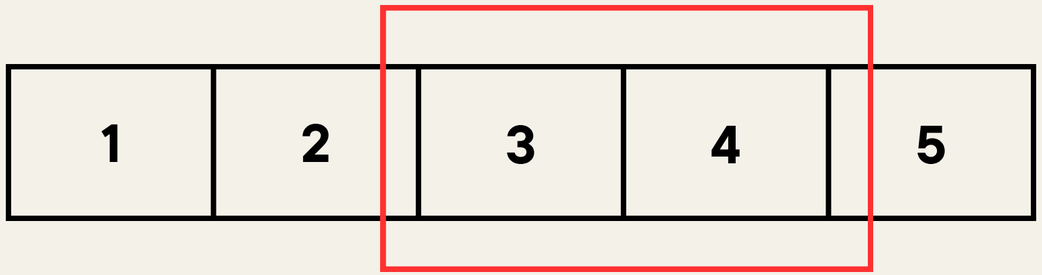
Data Slicing

length = 2



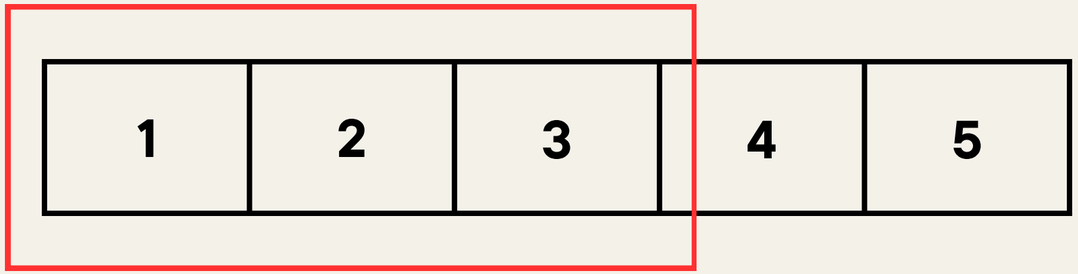
Data Slicing

length = 2



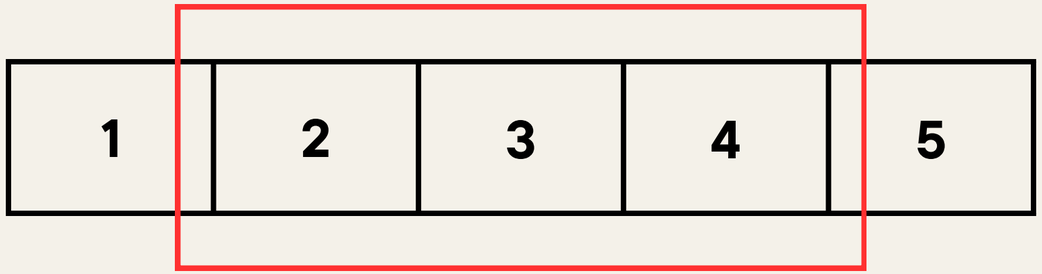
Data Slicing

length = 3



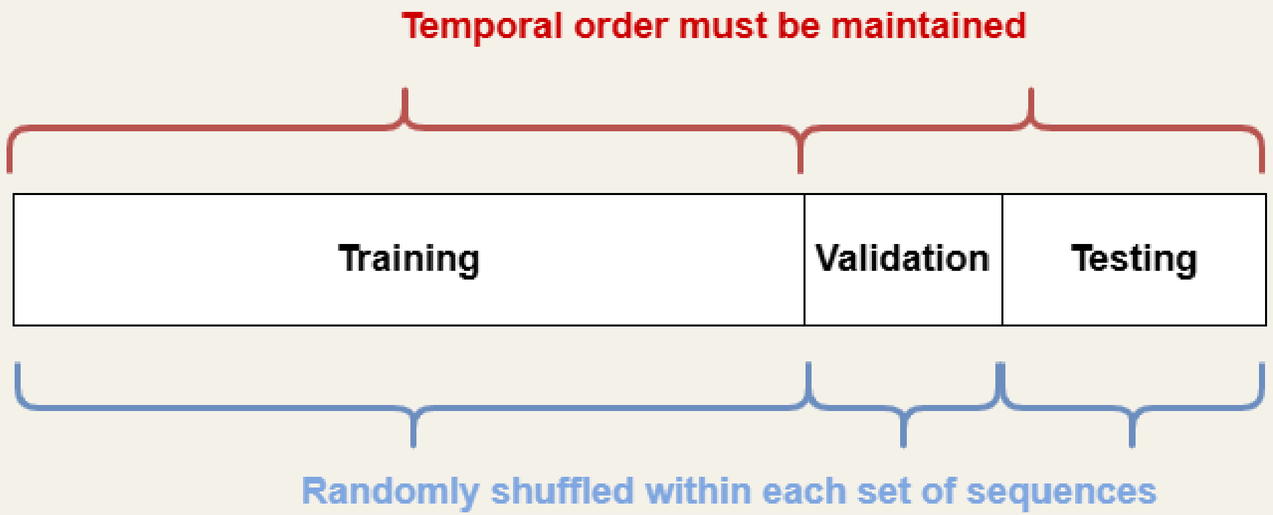
Data Slicing

length = 3



Data Splitting

- Data is split with the ratio, training:validation:testing = 0.64:0.16:0.2
- The datasets {training, validation} and {Testing} are split without breaking their temporal order
- Subsequences in each of the above dataset are randomly shuffled to prevent the model from mistakenly learning temporal relations between these subsequences



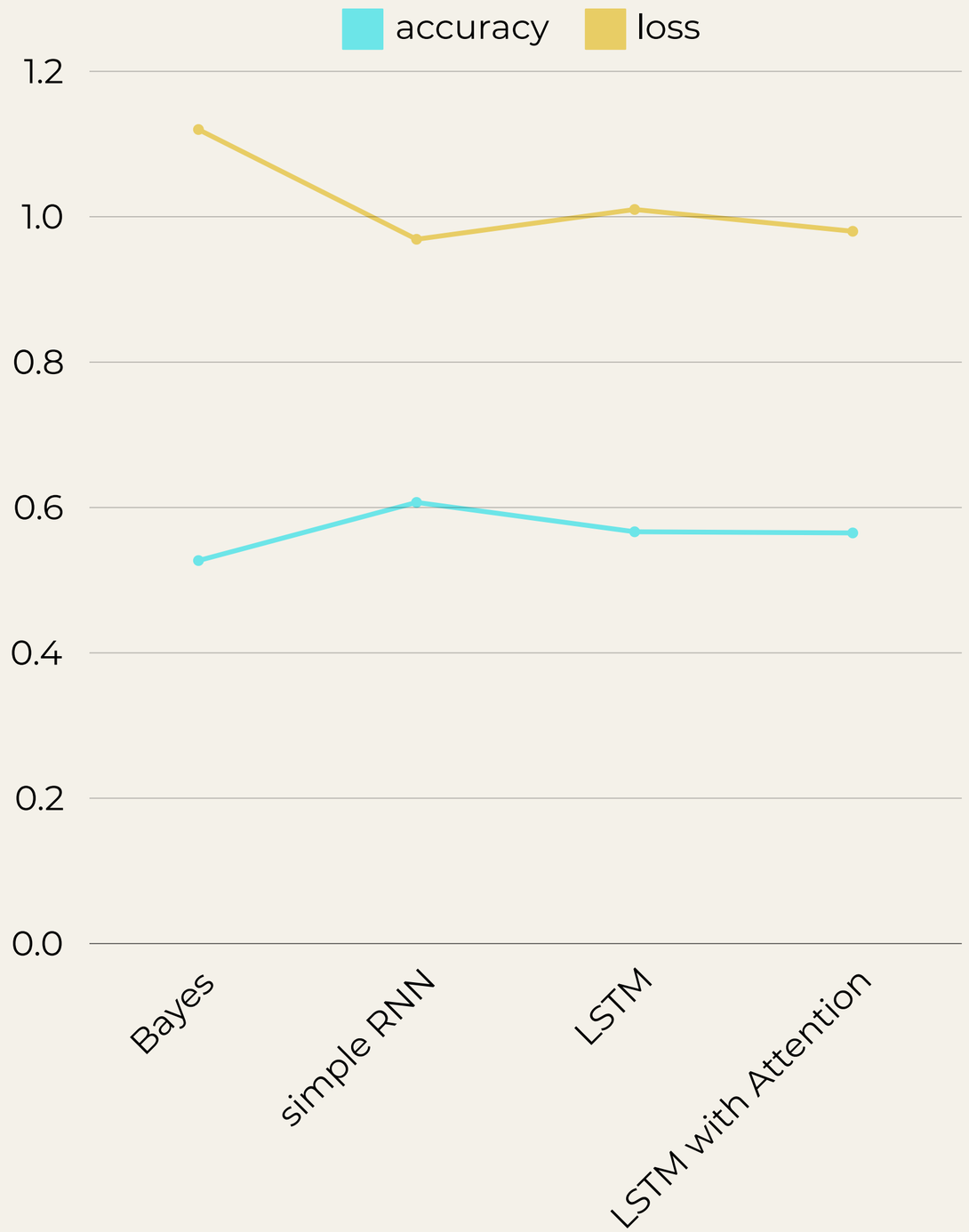

Results

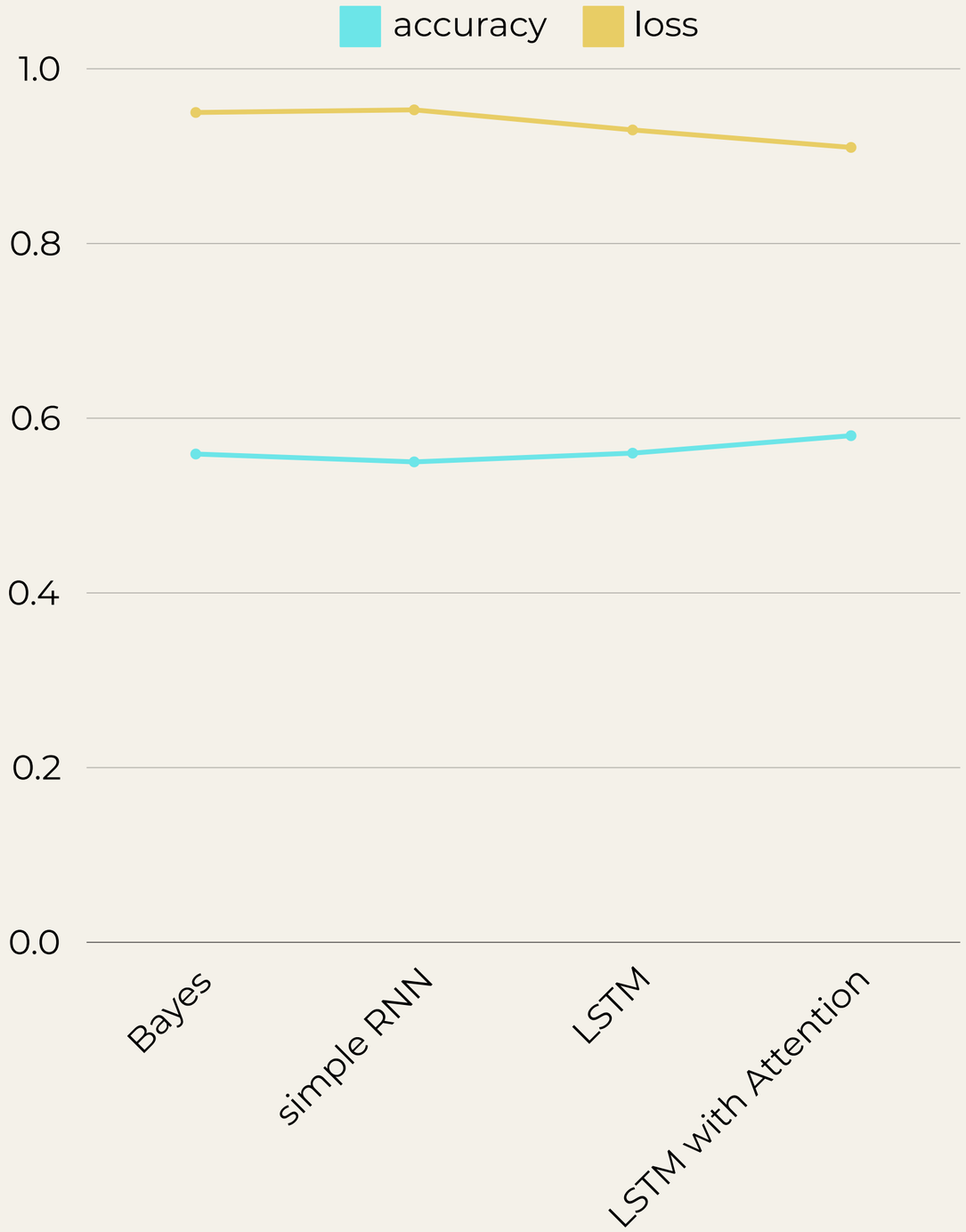
STARTING PITCHER

STARTING PITCHER

STARTING PITCHER

SHOTA IMANAGA





STARTING PITCHER

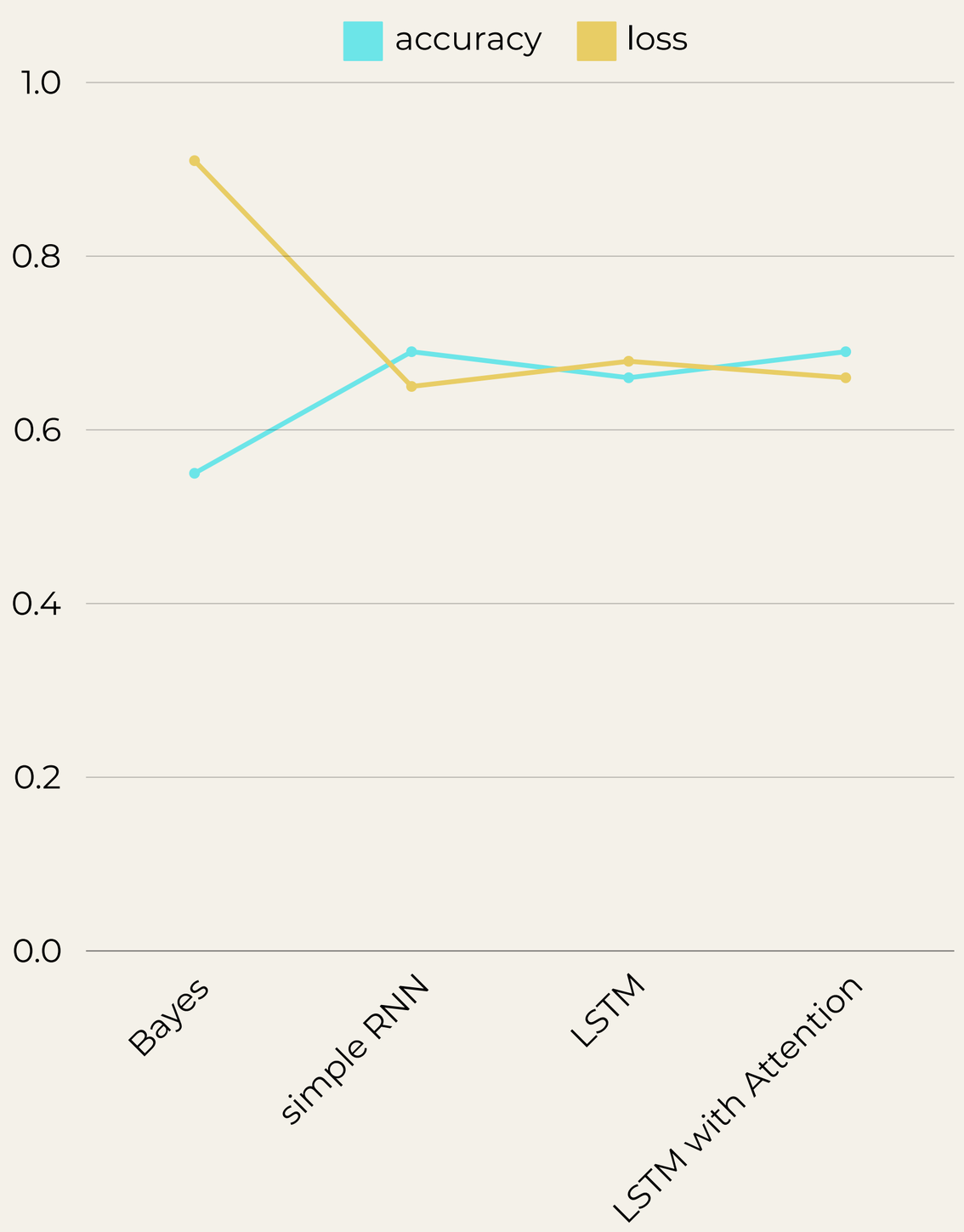
STARTING PITCHER
STARTING PITCHER

HUNTER GREENE

RELIEF PITCHER

RELIEF PITCHER
RELIEF PITCHER

MASON MILLER



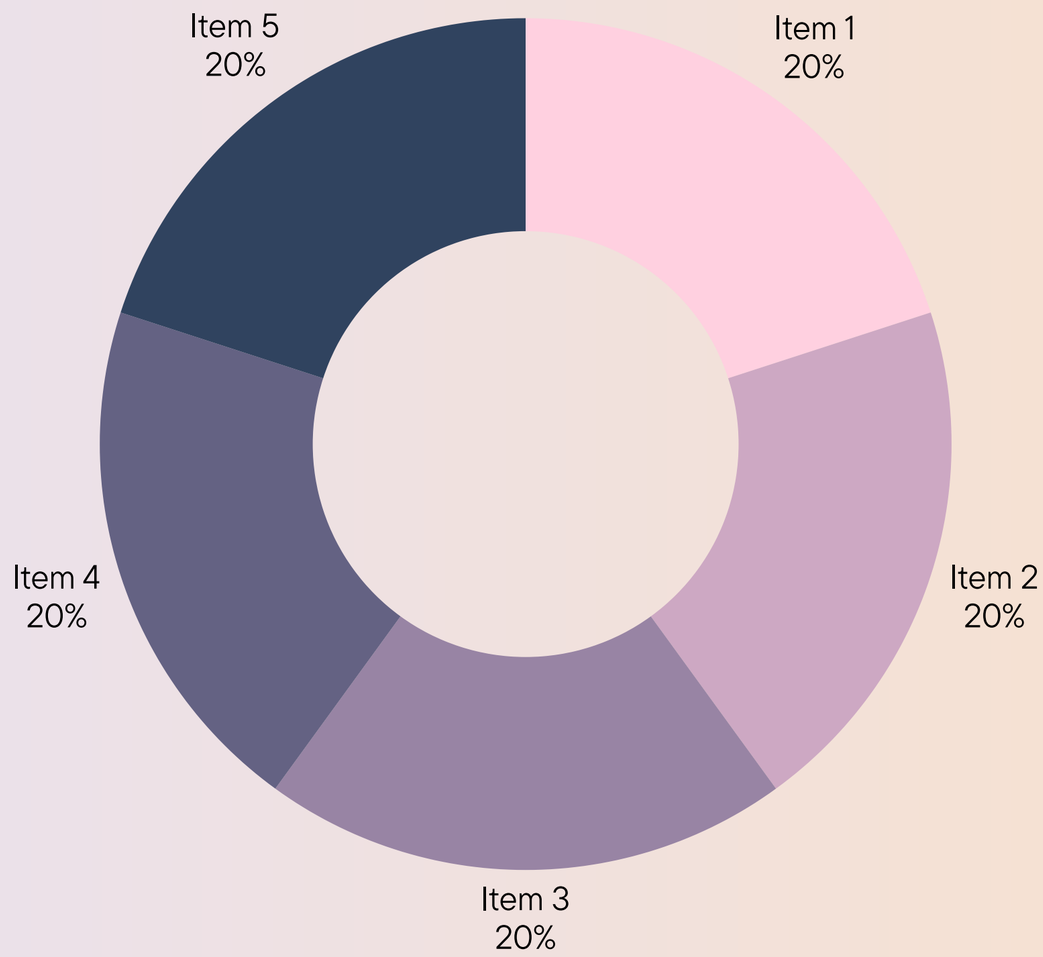
Our accuracy of predicting pitch type in relief picther

75%

FAIL TO SEE THE BENEFITS OF DIVERSITY BECAUSE OF A LACK OF INCLUSIVE PRACTICES

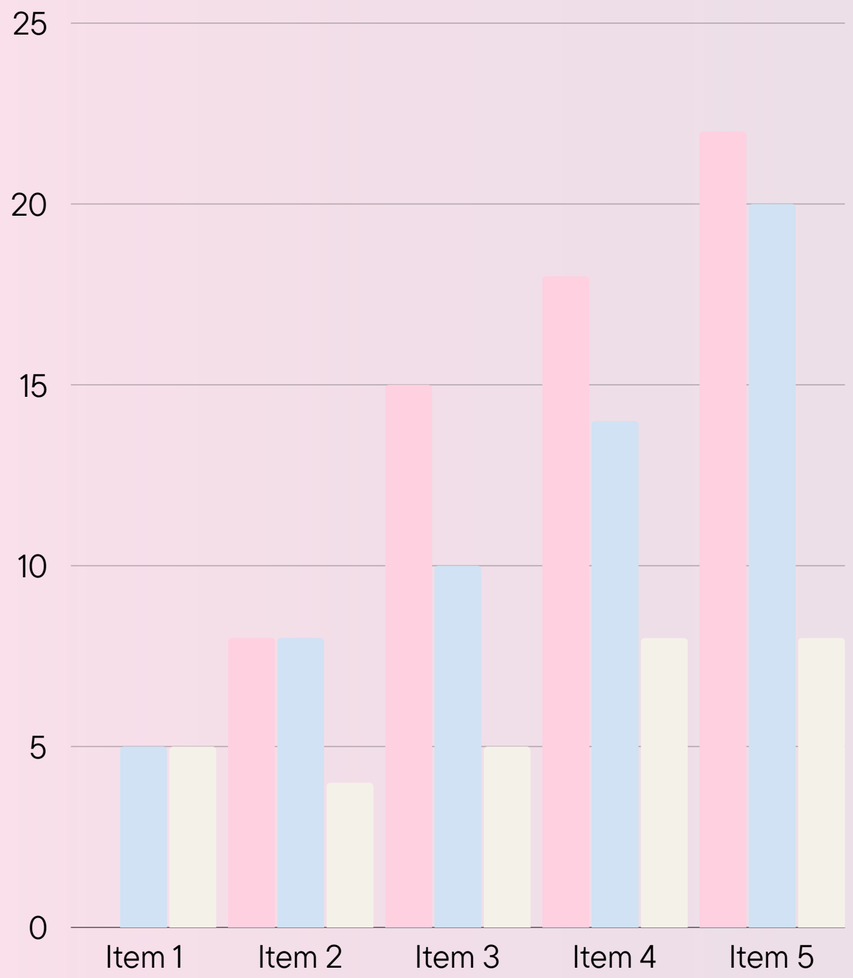
CONFRONTING UNCONSCIOUS BIAS

Awareness is the first step to addressing unconscious bias.



Here are some behaviors employees identified as showing biases and microaggressions.

Visualize complicated and dense information with graphs and charts. These are visual aids that help add more context to the topic you are discussing.



DRIVING GROWTH THROUGH D&I

Practicing diversity and inclusion increases talent retention, employee satisfaction, and productivity.

See how businesses have grown based on their commitment to diversity.

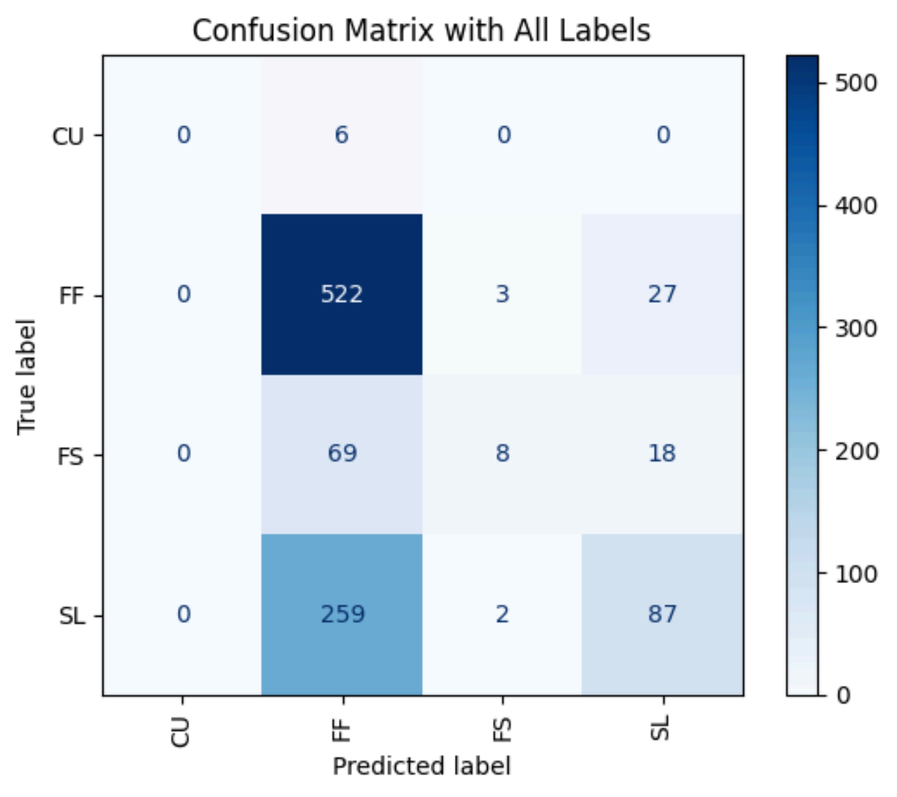
Visualize complicated and dense information with graphs and charts. These are visual aids that help add more context to the topic you are discussing.

Disscusion & Conclusion

1. Accuracy and Loss Are Not Enough to Evaluate Models.

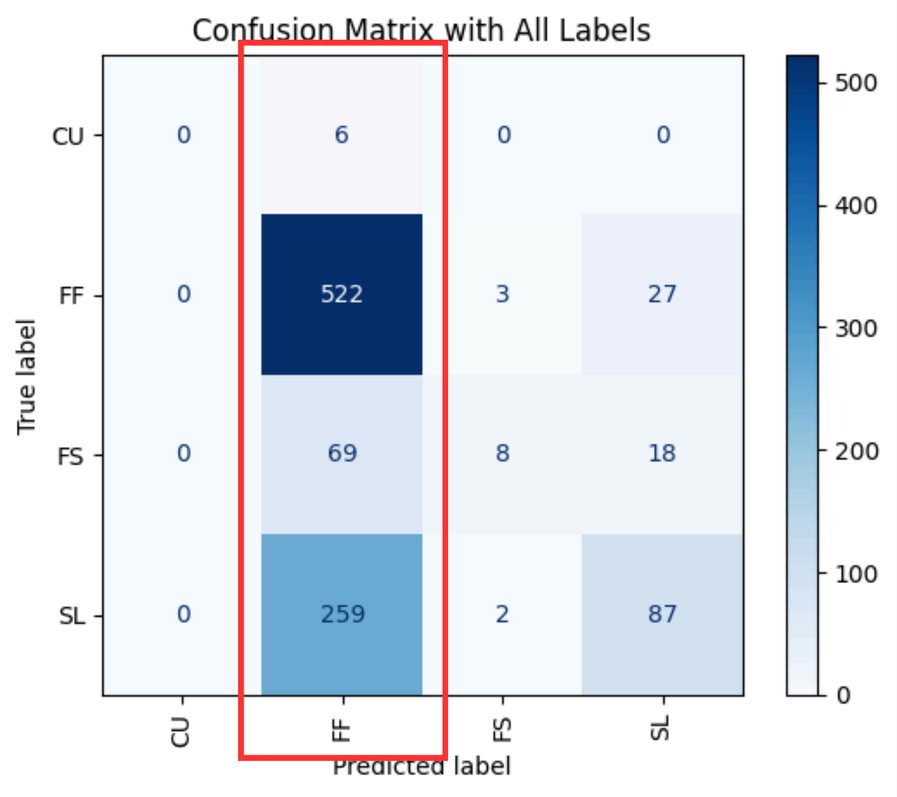
1. Accuracy and Loss Are Not Enough: Confusion Matrix

Hunter Greene: 61.68%



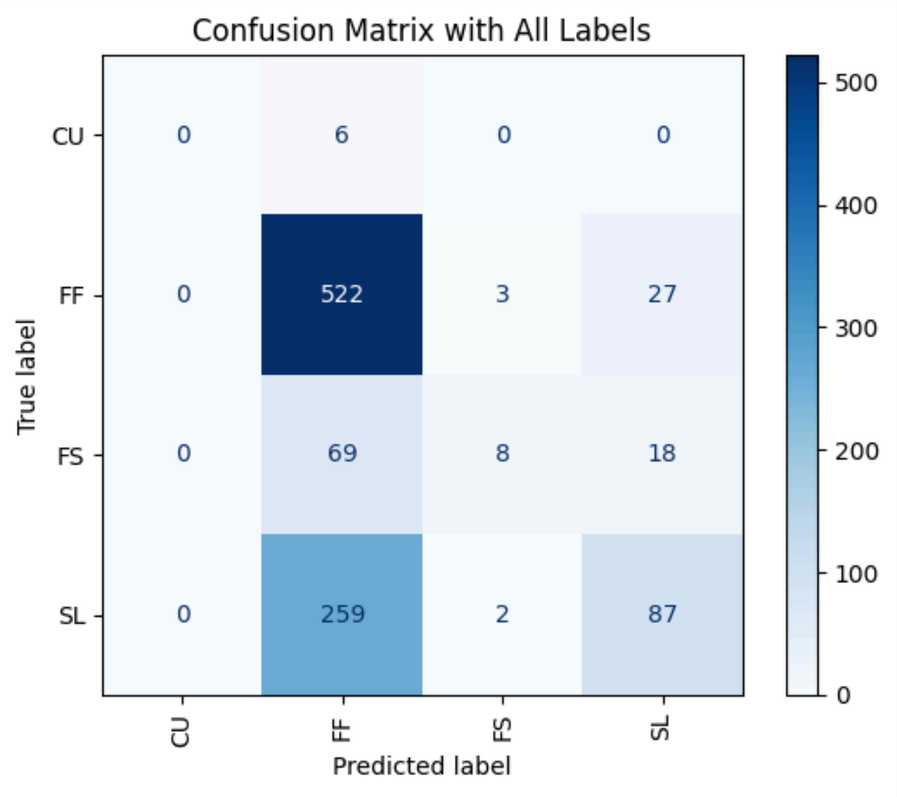
1. Accuracy and Loss Are Not Enough: Confusion Matrix

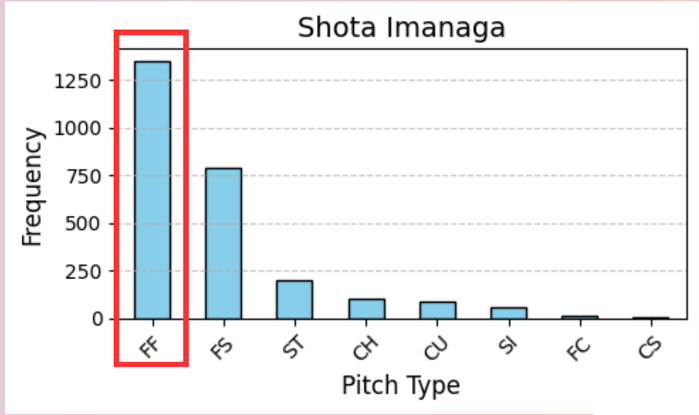
Hunter Greene: 61.68%

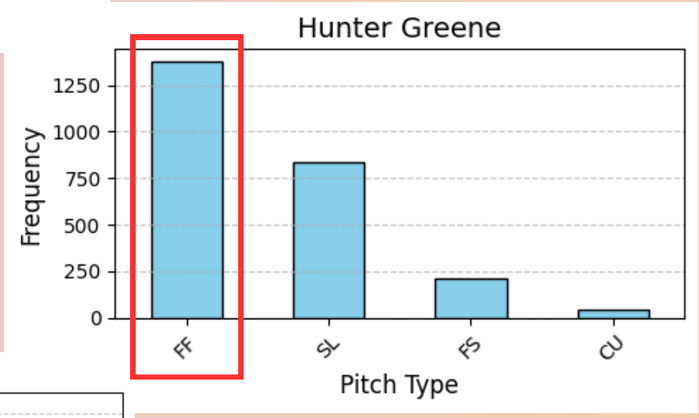


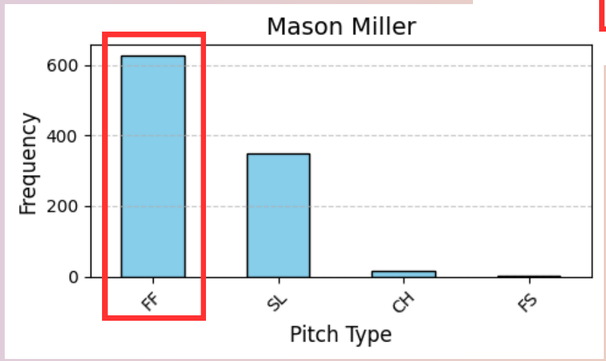
1. Accuracy and Loss Are Not Enough: Confusion Matrix

Hunter Greene: 61.68%



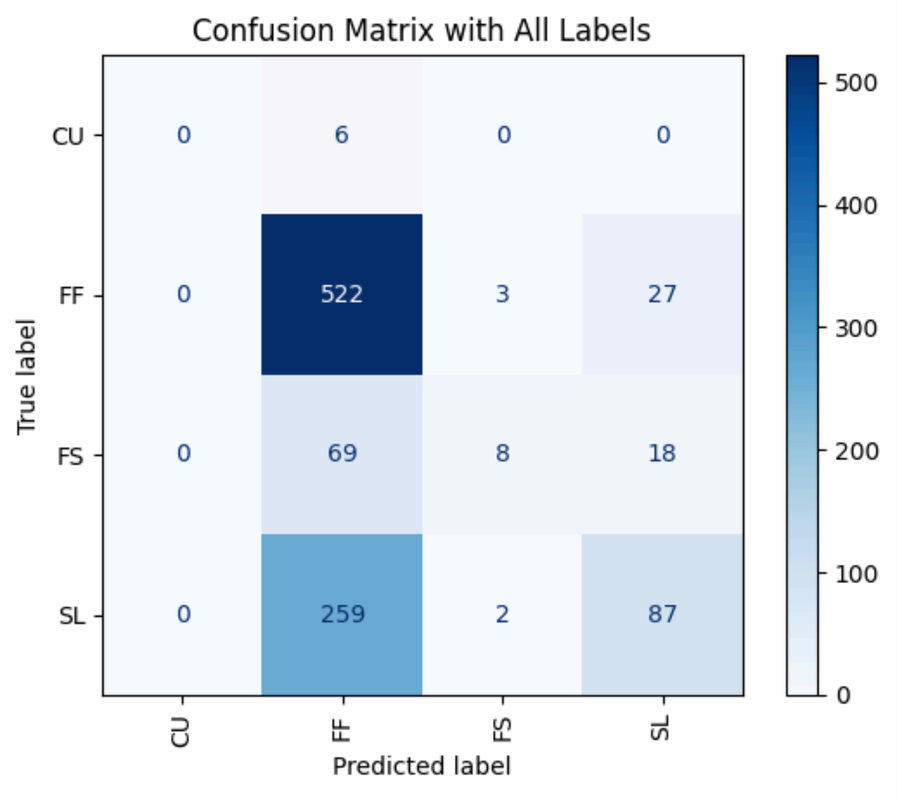






1. Accuracy and Loss Are Not Enough: Confusion Matrix

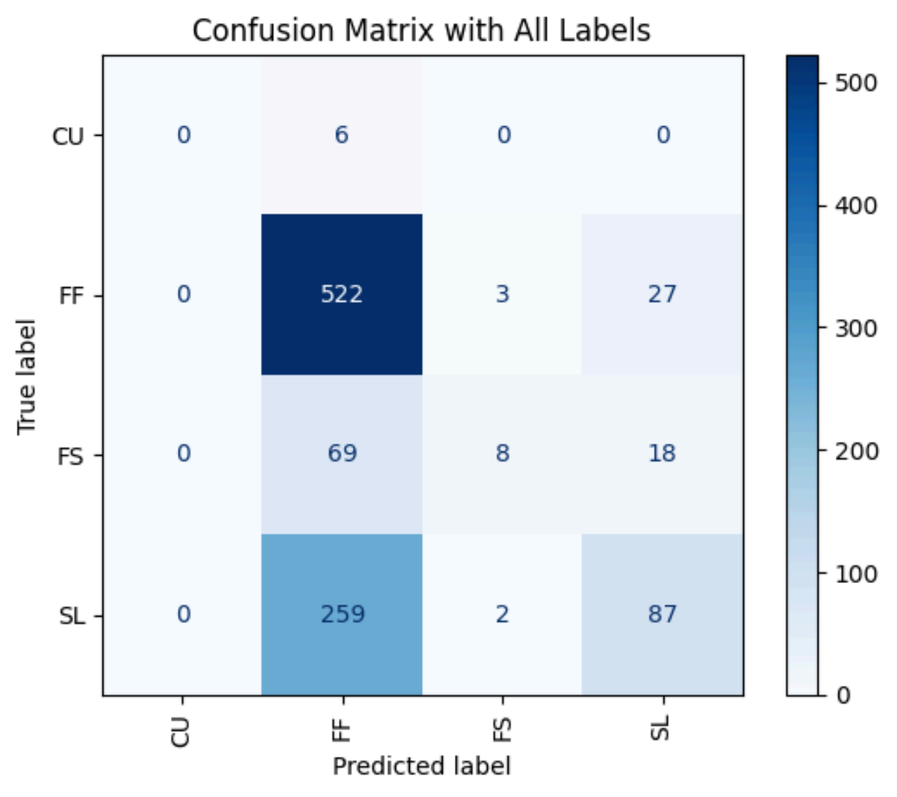
Hunter Greene: 61.68%



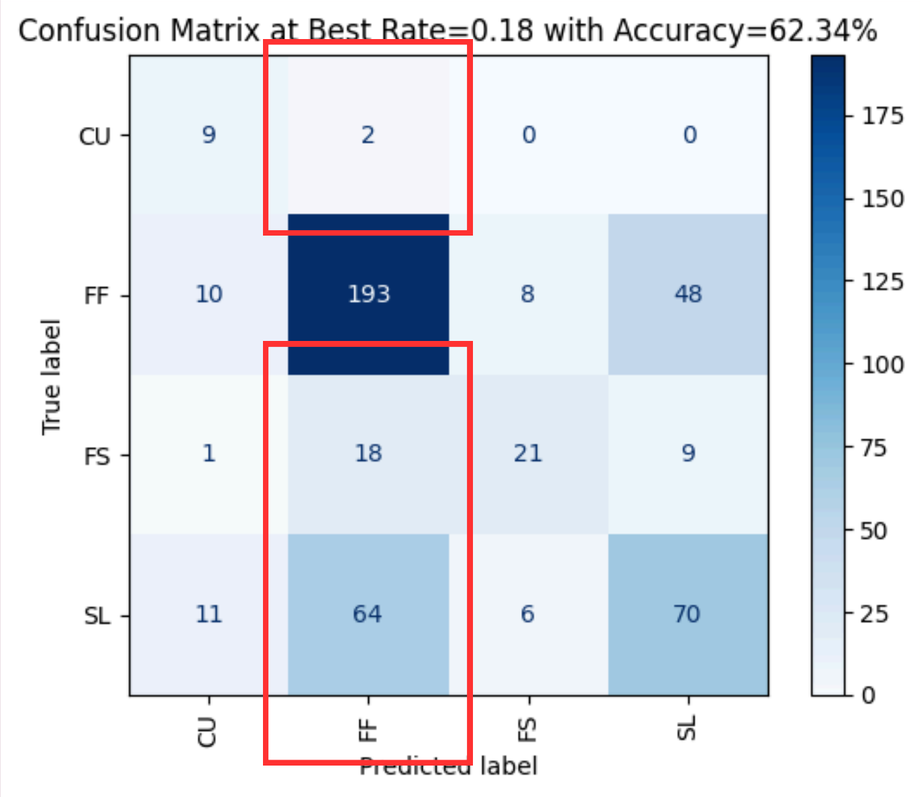
Adding Specialist Model?

1. Accuracy and Loss Are Not Enough: Confusion Matrix

Simple RNN: 61.68%

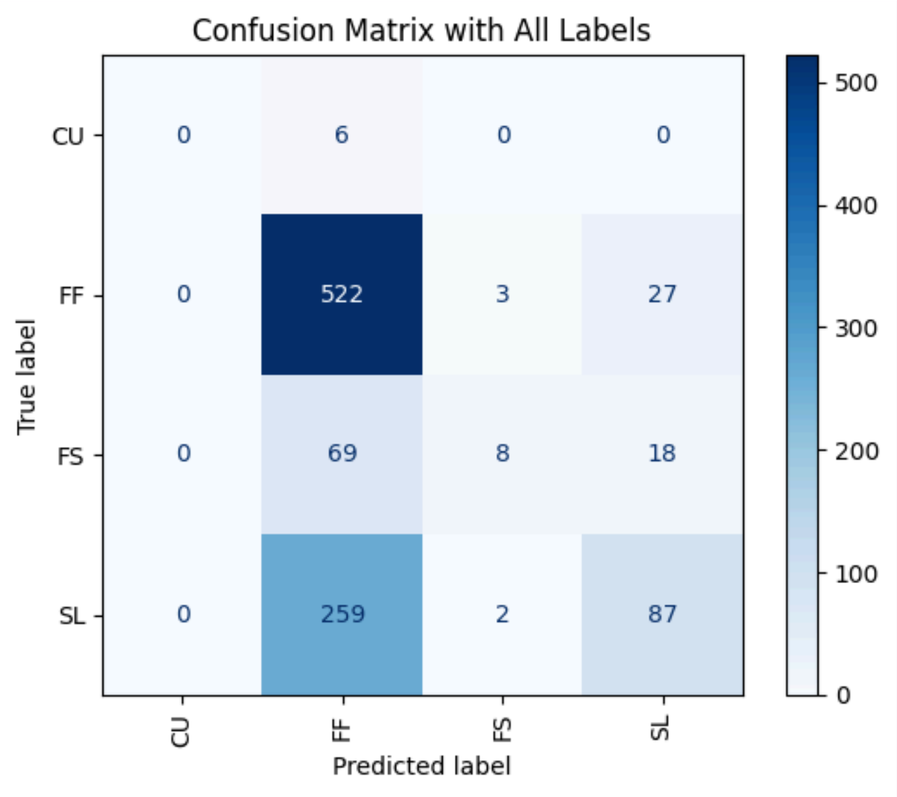


Adding Specialist Model: 62.34%

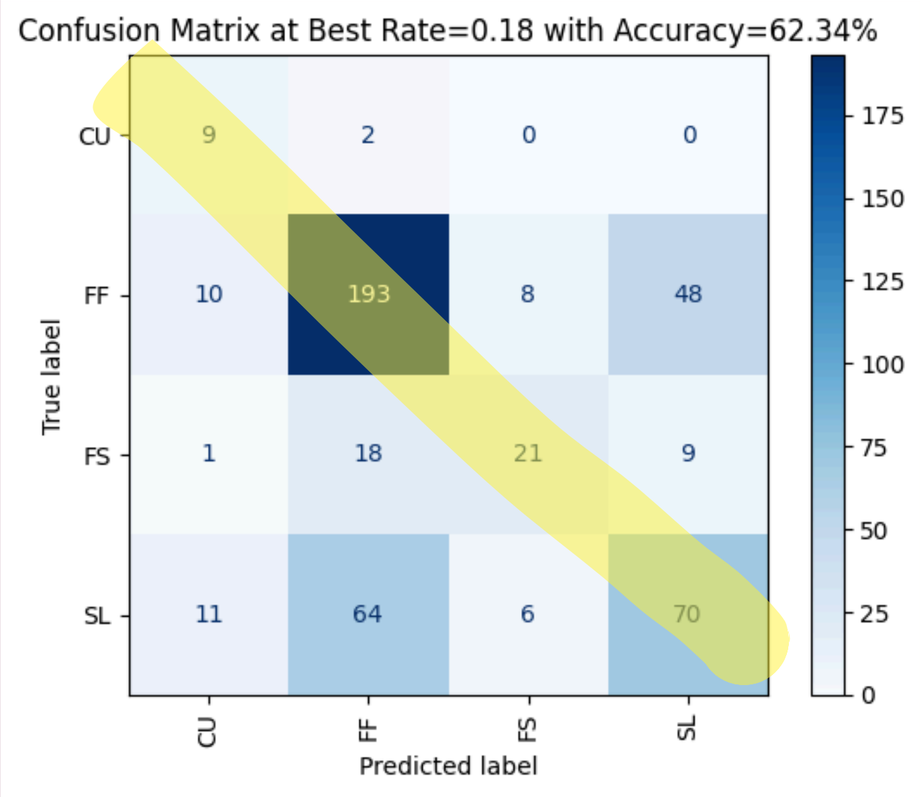


1. Accuracy and Loss Are Not Enough: Confusion Matrix

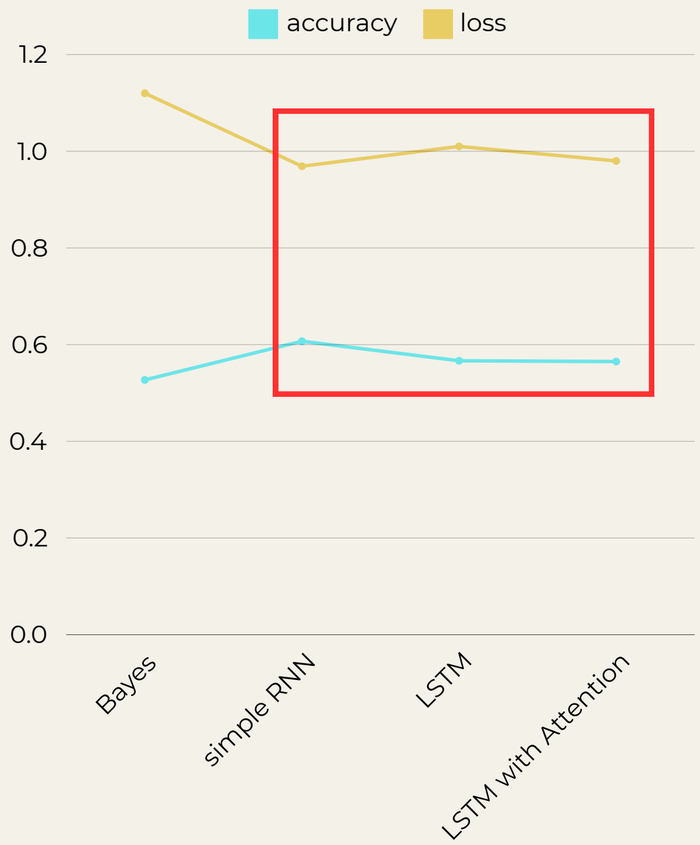
Simple RNN: 61.68%



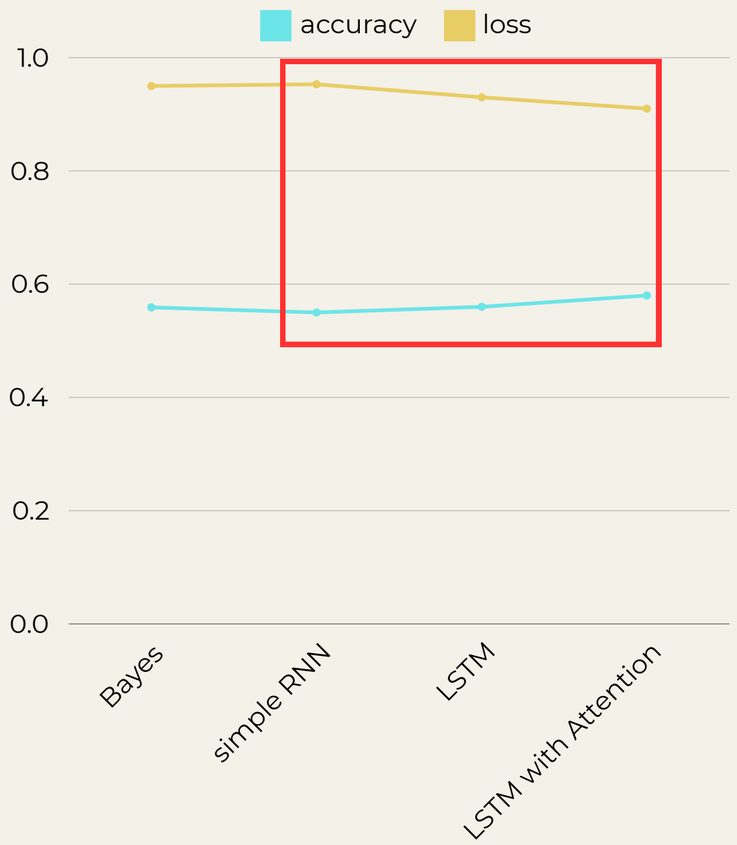
Adding Specialist Model: 62.34%



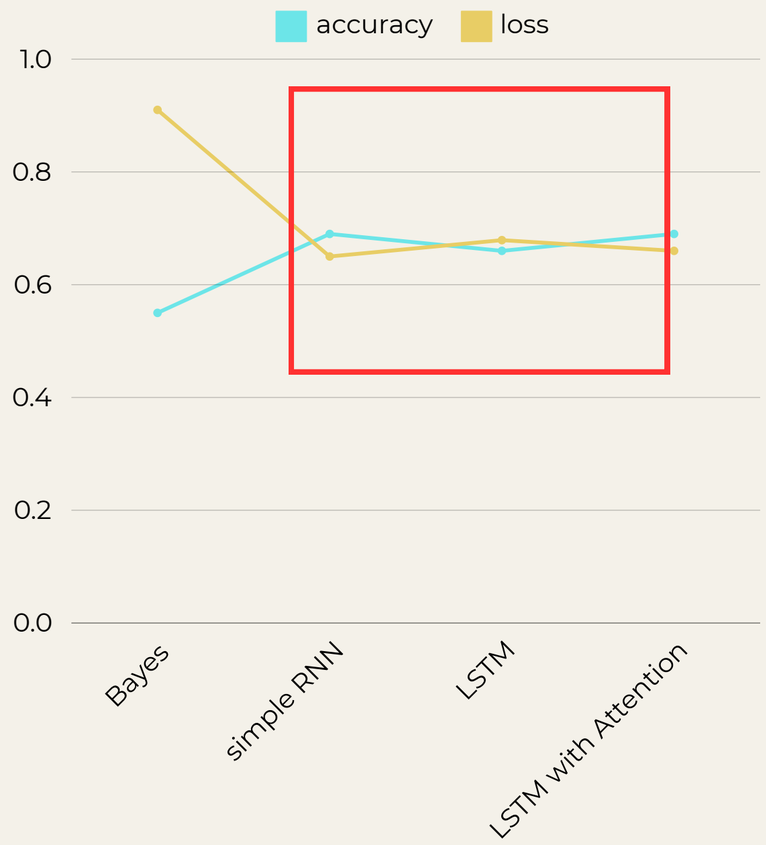
2. LSTM and LSTM+Attention Do Not Outperform on Short Sequences



SHOTA IMANAGA



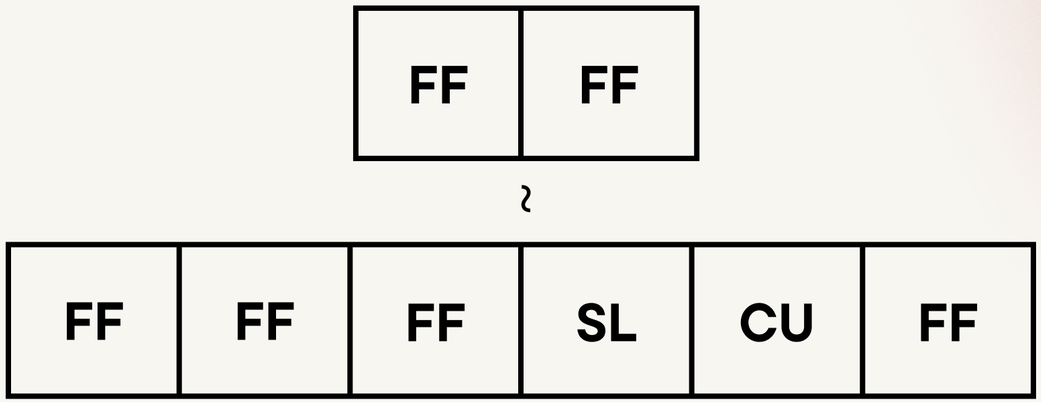
HUNTER GREENE



MASON MILLER

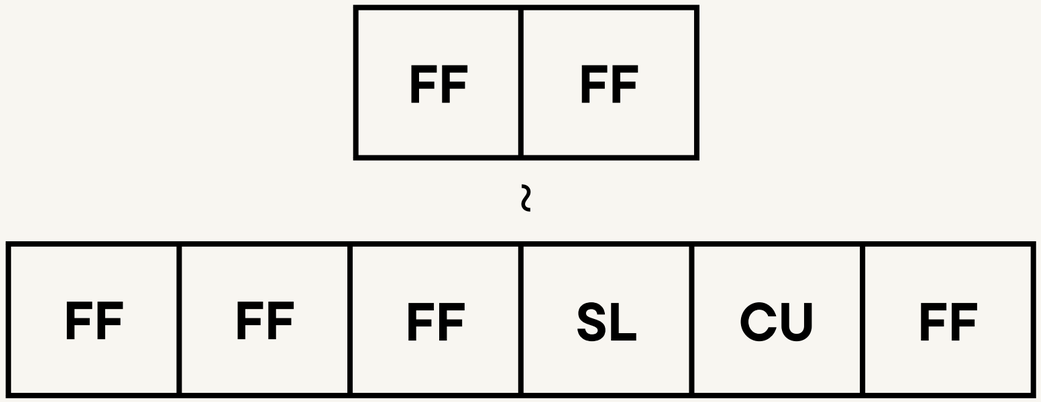
2. LSTM and LSTM+Attention Do Not Outperform on Short Sequences

(1) LONG-TERM MEMORY ADVANTAGE A NOT FULLY UTILIZED



2. LSTM and LSTM+Attention Do Not Outperform on Short Sequences

(1) Memory advantage a not fully utilized



(2) Bias accumulation of Attention Layer

Conclusion:

Our model outperforms Bayesian Baselines

Conclusion:

Our model outperforms Bayesian Baselines

Our model performs BETTER for pitchers with fewer pitch types

Conclusion:

Our model outperforms Bayesian Baselines

Our model performs BETTER for pitchers with fewer pitch types

But for pitchers with more pitch types, our model still remains solid

Practical Value of our Model:

Provides a powerful tool to analyze the pitching tendencies

Pitchers

Coaches

Catchers


THANKS FOR LISTENING
TEAM23