Ranking Super Smash Characters

Zach Gormley, Jeffrey Hilton, Pearce Keesling, Trayson Keli'i, Garrett Wilhelm
CS 478, Winter 2019 Department of Computer Science
Brigham Young University

Abstract

This paper seeks to explore the ability of machine learning models to predict the efficacy of different Super Smash Bros. characters across the different generations of games in the franchise. To date there are 5 generations of the game. Across all of the generations there are consistent attributes that allow comparison between different characters no matter the generation from which they originate. The newest generation of this game, Smash Ultimate, is new enough that players have not settled upon consistent rankings for the different characters. As such our goal is to predict those rankings by evaluating the attributes which were found to be significant features in previous generations. The different models which were evaluated where a decision tree, K nearest neighbors, and multi-layer perceptrons. Of these models, the MLP was found to be the most effective, managing an accuracy of 80%. For further exploration, an unsupervised clustering algorithm was also run on the characters to help identify trends and interesting relationships.

1 Introduction

1.1 Problem Description

The Super Smash Brothers series maintains a strong presence in the esports community and is praised for its large, diverse roster of characters. Professional players and amateurs alike are constantly creating lists of character rankings to determine which character would perform the best in a tournament setting. A character's rank represents the relative advantage of using one character over any other in a fight of two equally matched players, and a character tier is a grouping of characters who are ranked similarly, usually listed in letter grades. We aim to run different machine learning algorithms on the rankings and tiers of the 133 characters in previous Super Smash Brothers games, then use the best learned model on the 72 characters in the sequel, Super Smash Brothers Ultimate to predict what their rankings should be. In the future, we will compare our machine-learned rankings with the official rankings when the game stops receiving updates.

1.2 Data Sources

The bulk of our data is retrieved from the Super Smash Brothers wiki `www.ssbwiki.com`. This is a community driven database for all of the Super Smash Brothers games. Another source of information that we're hoping to consider comes from SmashBoard user KuroganeHammer who is the curator of a fan-made collection of more in-depth information for the characters `http://kuroganehammer.com`. For our initial tests we have chosen not to include this data because it proved to be very difficult to unify across characters. There is also a lot of missing data, especially in the newest generation of the game. For some of the attributes, we wanted to incorporate information that was not purely measurable in the game, such as character archetype. For this we had to rely on our own knowledge of the games and experiments conducted by our team members. Finally, we collected tier information and individual character rankings and tier lists from the following sources to use as potential class labels:

- SmashBoards
- SmashBackroom (via SSBwiki)
- RankedBoost
- HTC eSports
- And a collection of other tier lists (including lists made by professional players) published by IGN

1.3 The Dataset

The data that we are using is the damage output of each character's movesets. Each of the characters has roughly the same set of moves. This allows us to compare categories of attacks across the different characters. An important step in this comparison is to preprocess the data so that they are uniformly measurable. For example, some characters perform multiple successive attacks for the same player input. It is important to combine those attacks together so that it can be compared to another character with only one attack for that player input. Our current dataset consists of 133 training instances, the rosters from games 1-4, with 42 attributes per instance, and one class label–the character's predicted tier. The test set, comprised of the roster of Super Smash Brothers Ultimate, will contain 72 instances, for a combined total of 205 instances. There is a combination of continuous and nominal data. The

labels were created from standardizing the individually collected tier and ranking lists, averaging together the standardized scores for each instance, re-standardizing the averaged values to retain a standard deviation of one, and then finally discretizing the values into 6 even width bins to represent a universal standard tier list: S (Superior), A, B, C, D, and F.

The 42 features are: Weight (nominal), Archetype(nominal), Recovery(nominal), Jab(continuous), Jab Type(nominal), FTilt(continuous), FTilt Type(nominal), UTilt(continuous), UTilt Type(nominal), DTilt(continuous), DTilt Type(continuous), Dash Attack(continuous), Dash Attack Type(nominal), FSmash(continuous), FSmash Type(nominal), USmash(continuous), USmash Type(nominal), DSmash(continuous), DSmash Type(nominal), NAir(continuous), Nair Type(nominal), Fair(continuous), Fair Type(nominal), BAir(continuous), BAir Type(nominal), UAir(continuous), UAir Type(nominal), Dair(continuous), DAir type(nominal), FThrow(continuous), BThrow(continuous), UThrow(continuous), DThrow(continuous), NeutB(continuous), NeutB Type(nominal), SideB(continuous), SideB Type(nominal), UpB(continuous), UpB Type(nominal), DownB(continuous), DownB Type(nominal), Tier. Actual Instance: Mario med, hybrid, good, 8, melee, 13, melee, 10, melee,12, melee, 11, melee, 17, melee, 19, melee, 17, melee, 12.5, melee, 13, melee, 13, melee, 10.5, melee, 4, melee, 12, 26, 19, 19, 7, ranged, 7, ranged, 15, melee, 14, melee, B. Because the data is so rich in attributes and attribute values, we've attached a reference to our in-progress dataset and the tier information to give you a better idea of what we've actually collected:

- Attributes
- Tiers

Note that, for most classification problems, having only 133 training instances with so many attributes for each instance would be problematic. However, the problem we're solving is unique in that the universal set of instances is described in full with the 205 characters across all games (including Ultimate). And with how unique each character is, the more detail we can capture in each instance, the better. Thus the need for a somewhat larger set of attributes and possible attribute values.

2 Methods

Since there are many different machine learning models that each perform differently on a given dataset, we used a variety of different models to find out which one performed the best on our data.

2.1 Decision Tree

2.2 KNN

Algorithm

The K Nearest Neighbors algrithm works by colleting a group of data points as a kernel and then comparing new data points against those points to find its closest neighbors. Each new point's nearest K neighbors are found and then each of the output classes for those neighbors is given a vote that is scaled based on their distance from the new point. The largest vote wins.

Results

When KNN was run on the Super Smash Bros. dataset the accuracy was underwhelming with a best-acheived accuracy of 25%. Different values of k were tested, and k=5 was found to be the best performing. However, although the accuracy was rather low, when the confusion matrix was evaluated it was discovered that the model was usually close to identifying the correct classification. Below is a chart showing the confusion matrix for the KNN model run on the dataset.

	S	A	B	C	D	F
S	0	3	3	2	1	0
A	5	4	2	0	4	0
B	4	1	4	2	4	1
C	3	2	6	1	2	2
D	0	1	1	2	5	2
F	1	0	0	0	3	3

As can be seen in the chart above, the highest numbers are along the diagonal axis. This suggests that although the prediction was wrong, it was close to the correct classification. In further investigation it would be interesting to switch out KNN for RBF and compare the Mean Squared Error with other regression models.

2.3 K-Means

Because K-Means groups similar instances into clusters, it seemed like the oerfect model to create a group of character rankings. Before using this model, we decided upon the k value, that is to say the number of clusters the model will produce. Most Super Smash Brothers tier lists contain six tiers, which act as clusters of characters that perform the same in tournament settings. As such, we decided to use k=6 as a baseline of our clustering, as this would emulate a novel tier list. We also tested k=10 and k=15, but did not test any higher values because the model would begin to create clusters with only one character, which is not statistically significant for this problem. While there are many performance metrics for K-Means, we used Silhouette scoring as our primary accuracy metric. Because Silhouette scoring measures the ratio of dissimilarity between an instance and members of the nearest cluster over the dissimilarity of the instance compared to members of its same cluster, a Silhouette score close to one is desired. The following table includes the total Silhouette score for our tested k values.

k	Silhouette Score
6	0.07394
10	0.07561
15	0.10935

Surprisingly, our K-Means model performed poorly, as evident by the Silhouette scores close to zero. These low scores indicate a weak total clustering, and that most instances fell on the border of two clusters.

2.4 MLP

The features used to solve the problem and details on how you gathered and represented the features, including critical decisions/choices made along the way

3 Initial Results

Your initial results with your initial model.
The iterative steps you took to get better results (improved features and/or learning models)

4 Final Results

Clear reporting and explanation of your final results including your training/testing approach

5 Conclusions

Conclusions, insights

6 Future Work

As our primary difficulty in solving this problem was our data, it would be worthwhile to spend more time on refining our dataset. It is possible that including more detailed data, such as launch and frame data for each attack, would produce more accurate results. Another possible improvement would be the use of different models. Ensembles of different learning models or of the same learning models with different hyperparameters might produce more accurate results. Deep learning might be able to learn the complicated, higher-order relations between character attack types and damage, as well as how raw data, such as speed and weight, affect a character's ranking.

References

[1] "Smashwiki". textitSmashwiki, 2019, https://www.ssbwiki.com/. Accessed Feb 2019.