Will Schuckenbrock and Jack Schwanewede

CSC 362: Data Visualization

Dr. Katy Williams

May 14, 2025

Crossovers in *Super Smash Bros. Ultimate*

Introduction

Super Smash Bros. Ultimate is a 2018 platform fighting game developed by Nintendo. This is the fifth entry into the *Super Smash Bros*. franchise, which started in 1999, and it features a larger roster than any of its predecessors. The franchise features characters from several other popular video games and was originally created as a "Nintendo All-Stars" game but quickly became much more than that as a celebration of video games beyond just Nintendo. These types of crossovers have become increasingly common in gaming, with other games collaborating with multi-media companies such as *Fortnite*, *Dead by Daylight*, and *LEGO Dimensions*. Extending all of these crossovers to other franchises would create a massive web of collaborations, ranging from video games, to TV shows, to movies, to comics, to even luxury clothing brands. We wanted to visualize a fragment of that massive nexus, based on the franchise that arguably started it all. Our visualization depicts the crossovers between *Super Smash Bros. Ultimate* (2018) and some of the many video game franchises that make up its roster. We created a Force Directed Graph using HTML, CSS and JavaScript to visualize this, allowing users to interact with and see information about the types of franchises featured in *Smash*, and their relationships with one another.

Data

In order to create our visualization, we searched for a dataset that had information on video game franchise collaborations, specifically with *Super Smash Bros. Ultimate*. Unfortunately, there was no existing dataset that had the data we were looking for, so we had to create the dataset by scratch. The first step we took was identifying all the video game franchises we wanted to include. We had a total of 57 franchises, one being *Super Smash Bros. Ultimate*, 39 franchises that feature a playable fighter, and the remaining 18 franchises having some other form of crossover.

After we selected the franchises of interest, we created a 57 x 57 combination matrix in Google Sheets to classify whether the franchises had collaborated with one another. We used data taken from Crossover Wiki (https://fictionalcrossover.fandom.com/wiki/Crossover_Wiki) to complete the matrix. This was done manually, and we only counted direct collaborations or crossovers within a video game (at our discretion). For example, a character or item appearing in another franchise would count as a collaboration, whereas a marketing campaign outside of a video game would not (two games featured on a cereal box, for example).

We then made a chart containing every franchise featured, its genre, its first-party status (relative to Nintendo), and the estimated total of copies sold for all games in that franchise. This number is not exact, since many developers are not fully transparent in their total copies sold, so we estimated based on several other sources online. Next, we asked ChatGPT to combine the two CSVs, and asked Perplexity to add a “neighborCount” attribute (number of crossovers), giving us our updated .json file. These prompts are attached in the appendix. Tables 1 and 2 below depict screenshots of the CSV we made.

Table 1: Sample of Correlation Matrix CSV

	Smash Bros Ulti	Super Mario	Donkey Kong	Yoshi	Wario	Legend of Zelda	Metroid	Kirby
Smash Bros Ultimate	NA	Z	Z	Z	Z	Z	Z	Z
Super Mario	Z	NA	X	X	X	X	X	X
Donkey Kong	Z	X	NA	X	X	X	X	X
Yoshi	Z	X	X	NA	X	X		X
Wario	Z	X	X	X	NA	X	X	X
Legend of Zelda	Z	X	X	X	X	NA		X
Metroid	Z	X	X		X		NA	X
Kirby	Z	X	X	X	X	X	X	NA
Star Fox	Z	X	X		X			
Pokemon	Z	X	X		X			
Mother/Earthbound	Z	X						X
F-Zero	Z	X						X
Ice Climber	Z	X	X		X			
Fire Emblem	Z	X			X			X
Game & Watch	Z	X	X		X	X		
Kid Icarus	Z	X			X	X		X
Metal Gear	Z	X		X				
R.O.B.	Z	X			X	X		X
Sonic	Z	X	X	X		X		
Pikmin	Z	X	X		X	X		
Animal Crossing	Z	X	X	X	X	X	X	X
Mega Man	Z	X						
Wii Series	Z	X			X	X	X	
Punch-Out	Z	X	X		X			
Pac-Man	Z	X	X					
Xenoblade	Z	X				X		
Duck Hunt	Z	X			X			
Street Fighter	Z							
Final Fantasy	Z	X						
Bayonetta	Z	X				X	X	
Splatoon	Z	X		X	X	X		
Castlevania	Z							
Persona	Z							
Dragon Quest	Z	X						
Banjo-Kazooie	Z		X					
Fatal Fury	Z							

Table 2: Sample of Franchise Data CSV

Franchise	Estimate Copies sol	Nintendo	Primary Genre
Smash Bros Ultimate	76.85	X	Fighting
Super Mario	909.38	X	Platformer
Donkey Kong	92	X	Platformer
Yoshi	29.34	X	Platformer
Wario	22.81	X	Party
Legend of Zelda	170.35	X	Action/Adventure
Metroid	21.6	X	Metroidvania
Kirby	48	X	Platformer
Star Fox	11.67	X	Shooter
Pokemon	480	X	RPG
Mother/Earthbound	1.58	X	RPG
F-Zero	5.85	X	Racing
Ice Climber	1.5	X	Platformer
Fire Emblem	20	X	Tactical RPG
Game & Watch	43.4	X	Arcade
Kid Icarus	3.07	X	Shooter
Metal Gear	61.1	X	Action/Adventure
R.O.B.	1.32	X	Puzzle
Sonic	182.13		Platformer
Pikmin	7.2	X	Strategy

The new .json file is split into two parts: nodes and links. Nodes contain all information about a franchise found in the chart: its name, copies sold in millions, its genre, and its first-party status. Each link represents a crossover as seen in our combination matrix. A link is made up of three parts. The source and the target represent the two franchises collaborating—there is no difference between a source and a target. A link's "type" represents its connection to the *Super Smash Bros.* universe. Links between *Smash* and a franchise with a playable character in *Smash* are type "Z". Links between *Smash* and a franchise with any other type of crossover, such as having an Item, Spirit, Mii Costume, Stage, or Assist Trophy, are denoted as type "Y." Finally, any collaboration between two non-*Smash* franchises are type "X". While collaborations are bidirectional, there are no repeated links, as only one copy of a link is needed for the visualization. The data did not need any additional cleaning, since we directly created it to suit our visualization and tasks.

Paper Prototype

To create a prototype, we took several franchises' logos as nodes and used colored lines as edges. We placed *Smash* in the middle, since it is the central focus of our visualization, and put several other nodes surrounding them. We sized them somewhat proportionally to their total copies sold, and drew the links if they had a crossover. We also colored the links depending on their type (red for *Smash* fighter, blue for *Smash* other, black for between other franchises). This visualization is depicted below in Figure 1.

Figure 1: Prototype Visualization of *Smash Ultimate* Crossovers

This visualization has the core concepts of our final visualization, such as having nodes dependent on sized, colored links, and more. We also planned to have the tooltip highlight information about the game, such as the copies sold and game name. For our final visualization, we planned to add alt text to the graph and use colorblind friendly colors. For this reason, we will not actually use logo images for the nodes, as this may limit colorblind accessibility and also be a complicated implementation. We will also allow for zooming in the window by preventing distortion and maintaining graphical quality.

Task Analysis:

For our task analysis, we wanted our users to be able to use the visualization to find out what franchises collaborated with each other, as well as learn more about each franchise and sort

based on several attributes. Using the framework provided in the rubric, we created three tasks for the piloting and testing sessions.

1. Compare values: **Which franchise has more crossovers, *Minecraft* or *Pokémon*?**

   Correct answer: *Minecraft* (15), *Pokémon* (7)

This task can be completed using either the force directed graph or the dropdown to the side. The user can find the Minecraft and Pokemon nodes, either using the alphabetical dropdown search or using nodes' color and size to find the nodes manually. From there, they can select each node and see the number of crossovers listed in the tooltip. Alternatively, the user can use the dropdown to sort all nodes by number of crossovers and compare the values listed there.

2. Range**: What is the range of the amount of copies sold?**

   Correct answer: 908 million (909.38-1)

This task is optimally completed with the dropdown menu, where a user can sort all nodes by number of copies sold. They could then determine the range by subtracting the lowest sales value from the highest sales value. Alternatively, they could find the largest and smallest node on the FDG, use the tooltip to see exactly how many copies each node has sold, and determine the range through those values.

3. Order**: Order the franchises from least to greatest based on the number of copies sold.**

This can be done through the dropdown menu. A user can sort all nodes by their number of copies sold ascending, displaying an ordered list that they can scroll or arrow through.

These three tasks are very reliant on our dropdown menus, but the force directed graph allows users to complete a variety of other tasks. Users can **discover**, **present**, and **enjoy** the visualization, as these tasks are intrinsic to an interactive and visually appealing graph. Since nodes tend to "cluster" into groups based on similar crossovers, users can also **browse** a region of the graph, finding unknown nodes with a known location. Similarly, users can **locate** specific nodes; if the user knows the first party status and a rough estimate on the sales of a game, its node will not be difficult to find. Additionally, after our feedback sessions we added a feature where nodes could be searched for using the dropdown menus. Sorting alphabetically allows users to see an "index" of sorts with all the games, and clicking or entering on a franchise in that menu highlights the node's location on the graph. Arrowing through this menu also displays the tooltip and highlights the node with a stroke on the outside. Finally, since every node in the graph has a tooltip, users can **explore** and **identify** nodes through a variety of metrics.

Piloting

For our pilot, we used the following script, modeled off of the example given in the rubric.

"Speaker: We are evaluating our visualization and are asking you, the participant, to complete some tasks using the visualization and then provide feedback about the visualization and experience. As a reminder, we are evaluating the visualization, not you as a participant, so you don't need to worry about being "right" as you complete these tasks. There are three tasks, followed by a brief feedback session. The whole pilot session should take under 10 minutes. Do you consent to participate? [Wait for yes]

Speaker: Thank you for agreeing to participate. We will start with the three tasks. Please ‘think aloud’ as you complete the task, meaning voice what you are thinking as you work through the task.

1. **Compare values: Which franchise has more crossovers, *Minecraft* or *Pokémon*?**
2. **Range: What is the range of the amount of copies sold?**
3. **Order: Order the franchises from least to greatest based on the number of copies sold.**

Speaker: “That is the end of the third task. For this last bit, we welcome any feedback you may have about the visualization or about your process for completing the tasks.”
[Allow participant to speak first, then informal discussion]

In this pilot, the participant initially used the node colors to search for Minecraft and Pokémon nodes in the FDG. They were aware of what kind of crossover the franchises had, so they searched for a pink link (pink at the time of piloting). They wondered if they could sort by pink links, which was not implemented at that time. They found the Minecraft and Pokémon nodes in the graph and compared the number of crossovers from the node tooltips to answer Minecraft with 15, more than Pokémon with 7.

For the second task, finding the range of copies sold, they immediately used the order dropdown to sort by copies sold, ascending. They scrolled through and found the range, from Balloon Fight (1 M) to Super Mario (909.38 M). This order was the same order that we asked for in the third task, so they already had completed that.

For feedback, the participant suggested that we give the user the ability to sort by edge types.Specifically, the participant wanted to be able to sort by franchises that have a fighter in

Smash, or nodes with (at the time) pink edges. Also, we were curious about whether the participant gained any useful information from the neighbor feature, where clicking on a node highlights only the neighbors. They did not use this, but when we showed it to them they suggested that we make it so the only highlighted links are the direct node to neighbor links. At that point, it had all links shared by the current nodes and neighbors (Ex. *Mario* collab with *Zelda* and *Tetris*, should only show links from *Mario* and not the *Zelda*/*Tetris* link).

Based on the feedback, we made the following changes:

- Add a legend for intuitive understanding, instead of being in the text
- Adding a filter feature, to filter the type of links (*Smash* Fighter crossover, other crossover, between franchise crossover), and the type of node (Nintendo vs Third Party)
- Removed extra links in the neighbor selection feature, only show direct node-to-neighbor links
- Added a "search" feature, allowing users to use the menus to find and select nodes

Testing

We used the same script above as in the testing after we made our changes. We had three participants: the first being a Biology/Anthropology double major, the second being a Communications major, and the third being Undecided. Participant 1 completed the first task by using the order menu, clicking on the nodes using the search feature and looking at the tooltip to see which franchise had more crossovers (Minecraft). For the second task, they began to search for the range by clicking on the nodes, but quickly realized they could use the order menu to sort by copies (descending). For the final task, they switched descending to ascending to properly sort the games.

Participant 2 started by searching through the graph, looking for Minecraft and Pokémon without a specific strategy (did not use the legend). However, they navigated to dropdown instead of sorting by crossovers, which led them to the correct answer. Since they were already at the dropdown, they continued to use it to sort by copies sold (descending) to find the range, then switched it to ascending to finish task 3. Participant 3 also started with the graph, but discovered that they could drag the nodes as well. There was some confusion about how to populate the order dropdown and was using the crossover (neighbors) list instead, which took longer to locate both games since they themselves were not neighbors. Eventually, they found and compared the tooltips. For Task 2, the participant found the range by clicking on the *Smash* node, knowing it was neighbors with all, and found the range using the neighbor dropdown. For the final task, they attempted to reuse this strategy before quickly moving to the order dropdown and sorted in the correct way.

We had a few interesting pieces of feedback. One participant wanted a more direct search feature, to which we explained our method of sorting alphabetically and clicking. Participants also reported that they did not use the legend to complete tasks. They also said using the menu and dropdowns was much easier for the questions we asked, and one participant said that the FDG was a bit overwhelming and threatening.

To summarize, we were surprised by the variety of ways users completed the tasks provided. Participant 3 used a completely novel way of completing Task 2 that we had not anticipated, so it was pleasant to see that the visualization can be piloted in a variety of ways depending on user preference. One final change we made was to the order menu, making sure it populated automatically on RenderVis and not just when the dropdowns are selected, in order to prompt more attention towards the menu and encourage participants to experiment with the drop

downs or search for games that way. There was a little bit of confusion about what the neighbor list was and how it operated, so adding an explanation of how the dropdown works in future iterations of the site would be helpful. It was touched upon in the heading text, but may not have been clear (or participants did not have enough time to familiarize themselves and read).Lastly, the legend can be a little out of view on some devices, but this seems to be a hardware-specific limitation, as the only solution would be to reduce font size below what is acceptable for accessibility purposes.

Final Visualization:

Figure 2: Final Visualization of *Smash Ultimate* Crossovers

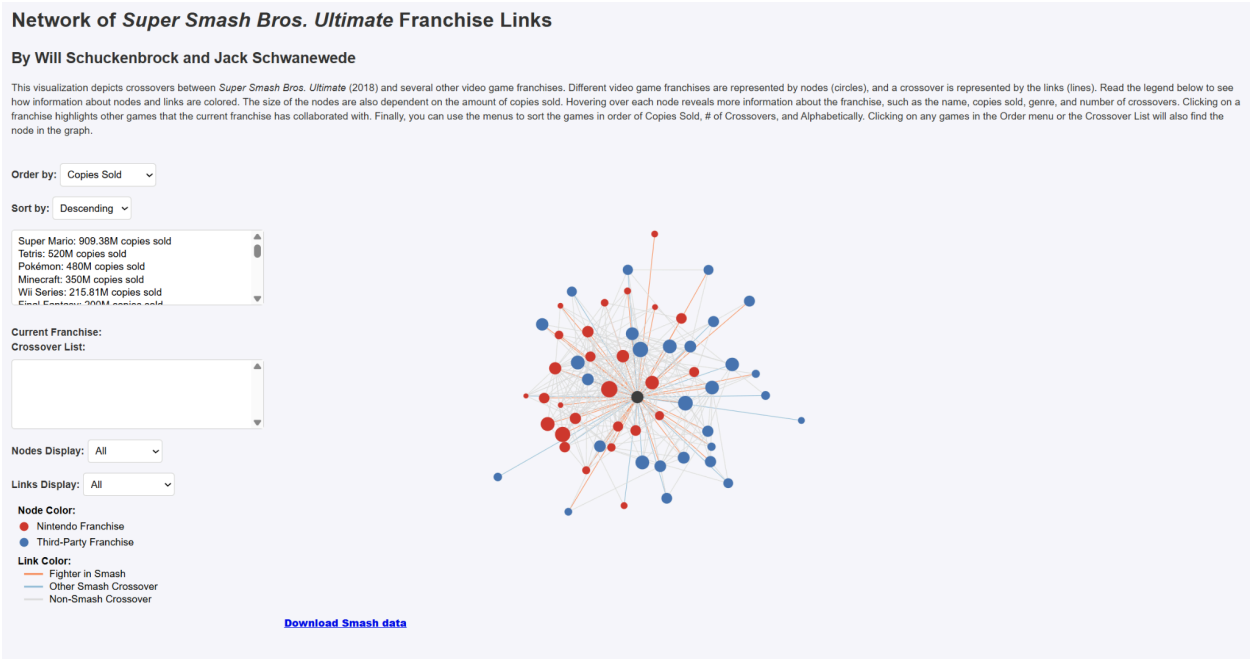


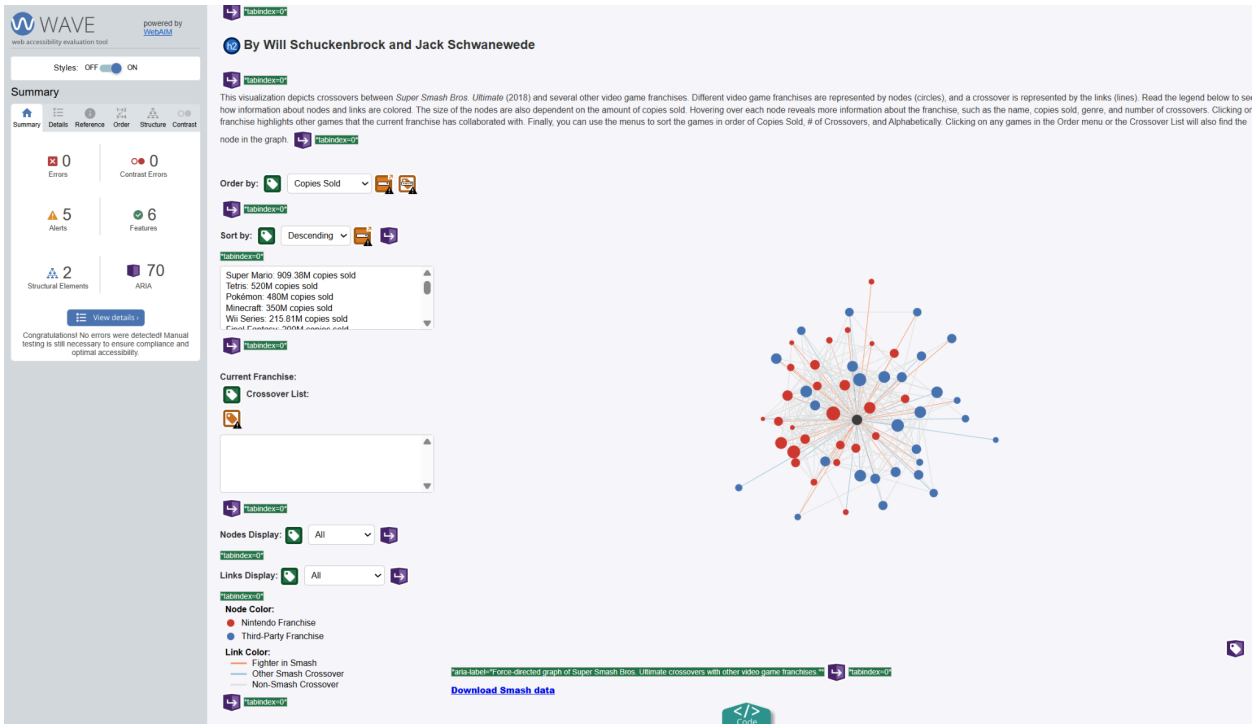
Figure 2 depicts our final visualization, after our final changes. This can also be found at the following link: chickenbrothbowl.github.io

Accessibility:

We also implement numerous accessibility functions, as suggested by WebAIM's WAVE Web Accessibility Evaluation browser extension. Many tasks can be accomplished non-visually due to our tabbing system. Users can use the tab key to iterate through nodes, with each tabbed node displaying its screen reader-legible tooltip. Users can also press enter when on a node to select that node, highlighting its neighbors both visually and within the neighbor dropdown menu in the top left corner of the visualization. This allows for a node's edges to be interpreted even non-visually. The backspace key allows a user to deselect a node, returning all nodes back to their original state. The user can navigate through the dropdown menus with the arrow key, using the enter and backspace keys to manipulate the graph while within the dropdown. The sorting and filtering menus can also be tabbed and arrowed through in order to edit the appearance of the menus and graph. Our extensive tabbing system allows the visualization to be piloted entirely by keyboard, assisting both visually and mechanically impaired users.

Other accommodations have been made for the sighted but visually impaired. As previously mentioned, all text in the visualization is screen reader-legible, but it also meets the requirements suggested by WebAIM for font size, as all text is at minimum 12px/9px in size. Apart from text, our red-blue color palette was selected using ColorBrewer 2.0, with great care taken in choosing a coloring system that would be easy for colorblind users to interpret. While the visualization is dynamic, it does not contain any fast flashing that could trigger epileptic users. While users utilizing accessibility mode would not be able to drag the visualization, very little context is lost when operating the visualization purely with accessibility functions. Also, tabbing through each node may take a longer time, since there are 58 total nodes. Nonvisual or hard-of-sight users would not be able to determine the first-party status of a node, as this

information is encoded solely through color. Apart from this specific setback, all other information is accessible through the dropdown or node tooltips.



Personal Reflection

I knew since seeing the Les Miserables graph near the beginning of the semester that I wanted to make an interactive visualization. I think it adds a layer of engagement that is very difficult to reproduce with a static graph. It wasn't difficult for my partner and I to think of a topic once we decided on our basic format, since we share an interest in video games. Creating the dataset for our visualization was a challenge that I was initially hoping to avoid, but it ended up being quite beneficial. Since we created the dataset it came equipped with all the data we needed and nothing more – there was no data cleaning required. We were also very familiar

with the data and how it was stored, since we authored it. This provided an excellent platform to start our vis, and we got a rough draft of the code done in just a few hours.

Jack and I worked exceptionally well together throughout the project, but I felt this was best exemplified in the interview process. Across our 4 interviews I took the note-taker role for 3 of them and by the end, we had the process down to a science. It was intriguing to see where interviewees succeeded, failed, or solved a problem in a unique way.

Overall, I feel like I got a lot out of the process - this is definitely the best-looking visualization I've made in the class, and I feel like you can see the effect of the multistep process in how polished the vis is.

Acknowledgements

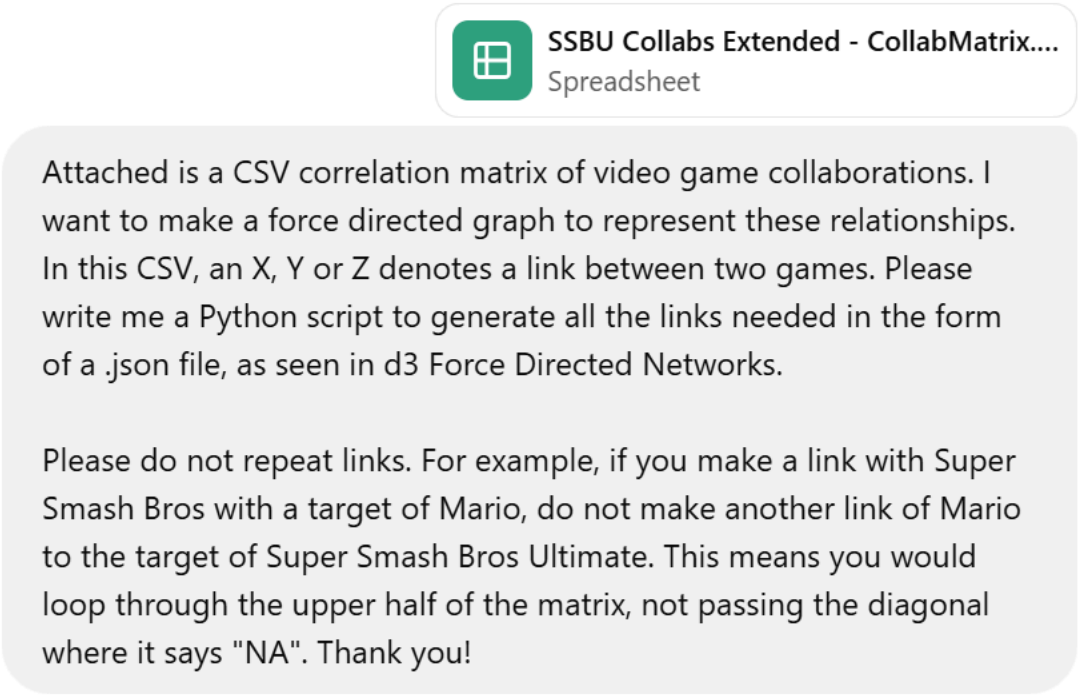
Generative AI (ChatGPT, Perplexity) was used to assist our development of the project, in agreement with the policies we talked about in class. Such uses are clarified below in the Appendix for our data cleaning and processing steps, and clarified in the code for helping troubleshoot and assist with bugs or features that were difficult to implement. For these features, we also ensured that we understood the workings and process in which the code works, and did not copy/paste entire sections. In this project, AI was used to supplement and augment our abilities, such as reducing "busy work" time, and was not used as a crutch on which we relied.

Lastly, we'd like to acknowledge and thank our participants, as well as Dr. Williams for all of her assistance and guidance throughout this semester. This final project was the culmination of several lab assignments, in class activities, homeworks and more, and we are proud of what we have made and learned in the past few months.

On my honor I have neither given nor received unauthorized information regarding this work, I have followed and will continue to observe all regulations regarding it, and I am unaware of any violation of the Honor Code by others.

Appendix

ChatGPT Data Cleaning:



- Generated updated .json file with nodes and links information

Perplexity Data Cleaning Prompt:

“Attached is a json file of data for super smash bros, with nodes and links. I want to make a neighbor attribute for each node based on how many links they have.

Can you create a neighbor count for the whole json file?

- Generated Python file to loop through links and add neighborCount attribute to all nodes

ChatGPT Debugging:

ChatGPT was used on a few occasions to aid with the debugging process. This would usually involve providing GPT with the code segment being debugged as well as a detailed breakdown of what isn't working and ideas on why.