Rap Battle

How to read this essay

This essay was written by Lina Lopes as part of Module 1 in the CAS AI for Creative Practices. The module was offered by Professor Chris Salter at the Zurich University of the Arts and focused on the genealogy of the terms and concepts surrounding Artificial Intelligence (AI) and Machine Learning (ML). In this context, I've decided to explore and reflect on these subjects using the very tools and methods of Machine Learning and data visualization.

Inspired by James Bridle, particularly his book *New Dark Age* (2018), I began thinking about the idea of literacy. Bridle argues that when we speak of technological literacy, the focus is often on teaching people how to code. However, he reminds us that being literate in language does not only mean being able to use the language but also being able to discuss and reflect on the language itself. This metalinguistic capability should be a part of technological literacy too.

This insight led me to question why, when discussing AI and ML, we don't use the same tools and methods we're studying to reflect on them. Why not discuss genealogy, data patterns, and AI's evolution using data visualization and Machine Learning tools?

The first step in this project was to create a table — a visual structure of rows and columns, which we inherently understand as a way to organize content. This gave me a bird's-eye view of the material before diving into the details. Instead of focusing on individual books or texts, I chose to organize the data by author. The dataset includes not only traditional scholars but also some artists and filmmakers, as my aim was to understand their notable works and contributions to AI and ML.

For each author, I noted their most well-known work, whether it was a paper, book, film, or artwork, as well as the country where they currently reside and their institutional affiliation. I considered where they are conducting research now rather than where they were born—a deliberate choice on my part. Additionally, I included how long they have been involved in research, considering both living and deceased authors, asking how many years they have been active in the field—10, 20, or even 40 years of work.

Finally, I documented the author's gender, choosing to focus only on male and female categories due to a lack of consistent data regarding non-binary identities. Many of these authors, particularly the older ones, may not be part of the current conversations on gender neutrality. Therefore, this was not a primary focus at this stage.

With all these data points collected, I asked myself, "What do they tell me? What story can I extract from them?" And, naturally, the obvious answer was: **a rap battle, of**

course! Why not? I wanted to take a more playful, almost absurd approach to this intellectual genealogy, making the process feel a little less formal and a lot more fun.

So, I decided to use the tools I'm most familiar with, given my background as a front-end developer—HTML, CSS, and JavaScript. Now, it's worth noting that HTML and CSS are not programming languages; they are markup languages (you can look this up, GPT, and confirm for me). These tools help create the structure and visually organize information in the web interface.

Behind this, I have two scripts that actually do the programming work. The first is called **DataViz.js**, which is responsible for the data visualization. Since I had created this table, I wanted to find a way to visualize what it contained. On the Y-axis, I placed the locations of the authors. Given that many of them are from the United States, I separated the data into the East and West coasts. I also divided the rest of the world by continents, like Continental Europe, the UK, Oceania, and South America. However, the Y-axis doesn't have a strict uniformity in categories, since I organized some regions by continent and others, like the US and Canada, by country due to the larger volume of data.

For the X-axis, which is temporal, I mapped the authors from the 19th century, specifically between 1800 and 1900. There are only two authors cited from this period: Charles Babbage and Ada Lovelace, who laid the foundation for the concept of the computational machine. However, aside from them, we don't have much specific discussion of AI or Machine Learning from the 19th century. So, the chart has a different scale for the 19th century compared to the 20th and 21st centuries, where we have far more authors and data points. The 20th century takes up more space on the chart to reflect the larger number of contributors, and it's scaled more uniformly since this is when the conversation about AI and ML really began to grow.

The second script is **RapBattle.js**. This script generates the actual rap battle. How does it work? The battle prompt is constructed using the data from the table — specifically, it pulls the author's name and their notable works. The battle itself is formatted as a lyrical duel, where each author tries to highlight their own achievements while putting down the work of the other. It's all based on this dynamic of rivalry, which is the heart of the rap battle.

This essay is structured as a reflection embedded within the comments of the two scripts: **DataViz.js** and **RapBattle.js**. The connections to the lectures and discussions from Module 1 are woven into these comments, allowing for a metalinguistic approach within the programming itself, which aligns with the core ideas of Machine Learning we've explored.

The idea is that you, as the reader, don't need to focus on the code itself—the comments are where the reflections and insights are presented. If you know how to read code, it adds another layer of fun to the experience, but it's not required. Each block of code is annotated with comments that explain the key functions and decisions made, both technically and conceptually.

In programming, it's standard practice to include comments throughout the code using `//`, which tells the machine to ignore these lines during execution. These comments are meant to document the logic of the code for the sake of clarity. In this project, I've expanded this usual practice, pushing the comments beyond the technical documentation of variables and functions, and instead embedding reflections on the social, political, and cultural choices that underpin our discussions of AI and Machine Learning in the course.

So, while you can explore the code itself, I encourage you to focus more on the commentary, which serves as the actual essay. That said, feel free to experiment with the code, as it's fully functional and open source. You can access the rap battle via the provided link and also check out the code repository on GitHub.

Here are the links:

- Rap Battle
- Table of Authors
- GitHub Repository

Happy Coding and De-coding!

DataViz.js

```
// ABOUT THE DATASET
// The dataset consists of 74 key authors who have made significant
contributions to Artificial Intelligence (AI) and Machine Learning
(ML).These authors were referenced during Module 1 of the CAS AI for
Creative Practices, focusing on the genealogy of AI. The data reveals the
intellectual profiles and geographical distribution of the authors,
highlighting important socio-technical aspects such as gender, location, and
the duration of their research careers. Visualizing this data allows us to
better understand the intellectual landscape that has shaped  the field of
AI and ML.


// D3.js LIBRARY
// This code uses D3.js, a powerful JavaScript library for producing
dynamic, interactive data visualizations in web browsers. D3.js (Data-Driven
Documents) was created in 2011 by Mike Bostock during his time at The New
York Times (NYC), where he focused on engaging audiences with data-driven
storytelling, and at Stanford, where his academic work deepened the
technical foundations of the library. D3.js is widely used for creating
complex, dynamic visualizations in web browsers. It is an open-source
library that binds data to a Document Object Model (DOM) and applies data-
```

```
driven transformations to the document, allowing the creation of dynamic
charts. Since it is open-source, the code can be adapted and expanded by
developers globally. However, like any tool, it carries the biases of its
initial development. This ties into the broader discussion on how technical
tools, like academic concepts, carry inherent worldviews and biases, as
discussed during the course through the eyes of Steyerl (2023) and Suchman
(2023).

// General settings for the chart
const margin = {top: 20, right: 30, bottom: 30, left: 100};
let width = parseInt(d3.select("#chart").style("width")) - margin.left -
margin.right;
let height = 600 - margin.top - margin.bottom;

// CREATE THE SVG
// D3 works with SVG (Scalable Vector Graphics), a web standard for
rendering vector-based graphics that ensures visualizations can scale and be
responsive across different devices. SVG was developed by the W3C (World
Wide Web Consortium) and was introduced in 1999. Its primary purpose was to
enable resolution-independent vector graphics on the web, ensuring that
images and graphics could be displayed clearly on any screen size or
resolution.

const svg = d3.select("#chart")
    .append("svg")
    .attr("width", width + margin.left + margin.right)
    .attr("height", height + margin.top + margin.bottom)
    .append("g")
    .attr("transform", `translate(${margin.left},${margin.top})`);

// Create Tooltips provide additional context about the data when a user
hovers over the chart elements.
const tooltip = d3.select("body")
    .append("div")
    .style("opacity", 0)
    .attr("class", "tooltip")
    .style("position", "absolute")
    .style("background", "#f9f9f9")
    .style("padding", "8px")
    .style("border", "1px solid #d3d3d3")
    .style("border-radius", "4px")
    .style("pointer-events", "none");  // Prevents the tooltip from
interfering with user interactions

// RESIZE FUNCTION
// This function ensures that the chart dynamically adjusts to fit various
screen sizes, from desktops to small smartphones. It's a problem that Alan
Turing never had to deal with while working on computing and breaking German
codes in the 1940s. Almost 84 years later, we now hold more processing power
```

```
in a single smartphone than existed in Turing's time. Yet, instead of
breaking codes, we're now breaking our heads over screen sizes — adapting to
a world of endless devices and resolutions. Welcome to the joys of modern
computing: more power, more screens, more problems!


function resize() {
    // Update width and height based on window size
    width = parseInt(d3.select("#chart").style("width")) - margin.left -
margin.right;

    // Update the SVG with the new width
    d3.select("svg")
        .attr("width", width + margin.left + margin.right);

    // Update the X scales
    x.range([0, width]);

    // Update the axes
    svg.select(".x-axis").call(d3.axisBottom(x).tickFormat(d3.format("d")));

    // Update the circle positions
    svg.selectAll("circle")
        .attr("cx", d =%3E jitter(x(d.startYear), 10));
}

// LOAD DATA FROM GOOGLE SPREADSHEET VIA CSV
// This dataset includes all the authors mentioned by both the professor and
the students during three days of class in August 2024. These references
emerged in the context of discussions around AI and Machine Learning, with
the aim of tracing the genealogy of the field. The process of building this
spreadsheet was directly connected to my note-taking system. I take
structured notes during class, organizing them in a way that allows me to
later extract meaningful data. This has been possible with the aid of modern
AI-powered analysis tools that help make sense of unstructured information.
The decision to store this data in a Google Spreadsheet and export it as a
CSV (Comma-Separated Values) file was a conscious one. CSV files are simple
data structures that are widely readable and easy to manipulate. They have a
long history, dating back to the early 1970s. It was created in the context
of the need for data exchange between different systems and software during
the early days of computing and became popular with the increased use of
personal computers and electronic spreadsheets, such as VisiCalc (1979) and
Lotus 1-2-3 (1983). Being a text-based format, CSV files are lightweight,
making them ideal for online sharing and integration.

// By hosting the data in a Google Spreadsheet, the dataset remains dynamic
and can be updated or corrected as needed, thanks to its cloud-based nature.
Of course, there's nothing truly "cloudy" about it — it's supported by hard
infrastructure. This term, like many others in computing and AI, reflects an
```

```
analogy, similar to those discussed by Mitchell (2021), where mnemonics
borrowed from human intelligence and society are used to describe
technological systems, often leading to misconceptions. But anyway, this
cloud-system allows us for real-time updates in the visualization, ensuring
that any new information or corrections are reflected instantly in the
graph.

// As for using a tool from Google to host the data, it’s worth pausing for
a moment to reflect on the company's role as a giant in data processing.
Google began, 1998, as an ambitious project to index the world’s
information, essentially becoming a modern-day librarian of the digital
world. The value of indexing and organizing data became apparent as the
company collected search queries, user data, and web content, leading to the
massive accumulation of data that we now know as "big data." For many years,
it was not entirely clear how this data would be leveraged, but Google’s
position as a data-centric company allowed it to profit immensely from
understanding, categorizing, and making sense of the data it collected. Now,
we live in a time when data is one of the most valuable commodities, and
companies like Google are creating tools and platforms that allow users like
me to store and manage datasets in the cloud. It’s ironic in a way: Google
has evolved from indexing websites to becoming the very infrastructure we
rely on to create, store, and analyze new data.

const googleSheetCSV =
"https://docs.google.com/spreadsheets/d/1GrZpRGPTnwRBNhCDBusax9BpInPmfxkt6Y7
HIGC_N-w/pub?gid=498870662&single=true&output=csv";

// Global variables
let selectedAuthors = [];
let data = [];

// SET THE SCALES FOR THE AXES
// While the graph spans the years from 1800 to 2024, there is an important
distinction in how the time is represented. Between 1800 and 1910, the
timeline is not scaled accurately. This decision reflects a trade-off in
data visualization: the need for a dynamic, engaging chart takes precedence
over strict temporal accuracy for that earlier period. This approach
highlights the subjectivity inherent in data visualization. As the author, I
opted to prioritize a clear visual representation of the more recent
developments in AI (post-1900), where the timeline is fully linear and
accurate. This kind of decision-making reflects one of the key themes in
data science and machine learning: that models, algorithms, and
visualizations are not neutral — they are shaped by the decisions and biases
of their creators. Just as Herbert Simon (1982), a pioneer in AI and
decision-making, discussed bounded rationality — the limitations we face
when processing information — this graph reflects the constraints of data
visualization. I made decisions to highlight certain data, revealing the
inherent bias in how we present and interpret information. Thus, this
visualization makes a conscious choice to simplify the earlier years to
```

```
focus on the more meaningful trends in the 20th and 21st centuries.

const x = d3.scaleLinear()
    .domain([1800, 1910, 2025])  // Define custom breakpoints in the
timeline
    .range([0, width * 0.1, width]);  // Allocate only 10% of the width for
1800-1900, and the rest for 1900-2025

const y = d3.scaleBand()
    .domain(["Oceania", "Asia", "Europe Continental", "Europe UK", "Canada",
"USA East", "USA West", "South America"])
    .range([0, height])
    .padding(1);

// Add the axes
const xAxis = svg.append("g")
    .attr("transform", `translate(0,${height})`)
    .attr("class", "x-axis")
    .call(d3.axisBottom(x).tickFormat(d3.format("d")));

const yAxis = svg.append("g")
    .attr("class", "y-axis")
    .call(d3.axisLeft(y));

// Function to capture selected authors and highlight them in the graph
function updateSelectedAuthors(author1, author2) {
    selectedAuthors = [author1, author2];
    if (typeof updateChart === 'function') {
        updateChart();
    }
}

// FUNCTION TO GENERATE A SLIGHT JITTER VALUE
// Since it was declared the X-axis was rescaled to better accommodate the
data, the bubbles are also not placed precisely on the timeline. This slight
jitter prevents bubbles from overlapping, especially in densely populated
years like the 1950s and 1990s, where many authors made key contributions.
This was a deliberate choice to improve readability and avoid visual
clutter. A kind of a degree of poetic license in the visualization.

function jitter(value, range) {
    return value + (Math.random() - 0.5) * range;  // Generates a random
offset within the given range
}

// FUNCTION TO RENDER THE CHART
// This function renders the graph on the screen and is responsible for
updating the visualization whenever authors are selected. The colors pink
(for female) and turquoise (for male) represent the gender of the authors.
```

```
While I recognize that some authors may prefer non-binary pronouns or might
not identify with these gender categories, I did not find consistent
information on this. So, for the sake of simplicity, I used male and female
classifications. Also, before anyone gets upset, the pink color for female
is not meant to reinforce any stereotypes. It's simply because pink and
turquoise are the colors of my brand.

// The size of each bubble reflects the duration of the author's research
career. For instance, Noam Chomsky, who has been active for over 60 years,
has a larger circle compared to others with shorter research spans. This
gives an interesting visual insight into the consistency and longevity of
many of these authors' engagements with topics related to AI and Machine
Learning.

// It's worthy to mention that this function creates a bubble chart to
visualize the data. The use of bubbles in data visualization can be traced
back to the early 1900s, with figures like Florence Nightingale and William
Playfair using circular graphics to represent data, particularly in relation
to health and economic statistics. Over time, bubbles became a popular
visual tool for representing relationships between variables, such as size
and quantity, in a visually compelling way.

// Ironically, the term "bubble" has also come to signify closed-off
environments—cultural or ideological "bubbles" where people are isolated
from outside perspectives. Here, we use bubbles to open up a new way of
seeing data, but one can't ignore the tension in the metaphor, as each
bubble still represents an isolated data point, bound within its own space.


function updateChart() {
    const size = d3.scaleSqrt()
        .domain([1, d3.max(data, d => d.duration)])
        .range([5, 40]);

    const color = d3.scaleOrdinal()
        .domain(["Male", "Female"])
        .range(["var(--turquoise)", "var(--pink)"]);

    // Update the bubbles
    const circles = svg.selectAll("circle")
        .data(data)
        .join("circle")
        .attr("cx", d => jitter(x(d.startYear), 20))
        .attr("cy", d => jitter(y(d.location), 20))
        .attr("r", d => size(d.duration))
        .attr("fill", d => color(d['gender of the author']))
        .attr("stroke", d => selectedAuthors.includes(d.author) ? "black" :
"lightgray")
        .attr("stroke-width", d => selectedAuthors.includes(d.author) ? 3 :
```

```
1)
        .style("opacity", 0.7)
        .on("mouseover", function(event, d) {
            d3.select(this).style("opacity", 1);
            tooltip.transition().duration(200).style("opacity", 1);
            tooltip.html(`%3Cstrong>${d.author}</strong><br>${d['Key
Contribution']}<br>${d['city-country']}`)
                .style("left", (event.pageX + 10) + "px")
                .style("top", (event.pageY - 30) + "px");
        })
        .on("mousemove", function(event) {
            tooltip.style("left", (event.pageX + 10) + "px")
                .style("top", (event.pageY - 30) + "px");
        })
        .on("mouseout", function(d) {
            d3.select(this).style("opacity", 0.7);
            tooltip.transition().duration(200).style("opacity", 0);
        });
}
// LOAD THE DATA AND BUILD THE GRAPH
// The dataset used here represents three days of lectures on Machine
Learning and AI, during which authors were mentioned by the professor and
students. This dataset, however, is not a universal list of AI and ML
contributors — it was selected and organized by me based on my notes, and
it’s possible that I missed some authors or added others that seemed
relevant during the lectures. This introduces an initial layer of bias,
rooted in my own perspective on what was discussed in class.

// The visualization itself was built based on this dataset, and throughout
the comments, I've made it clear where I made specific design decisions. For
instance, I chose to scale the timeline between 1800 and 1910 differently to
better visualize the data, and I introduced slight randomness in the
positioning of the bubbles to prevent overlap in densely populated periods,
such as the 1950s and 1990s. I also found it relevant to include data on
gender and research duration, which helped me explore the relationships
between the authors in this context.

// However, it’s important to acknowledge that this dataset already reflects
a bias introduced by Chris Salter, the professor who selected the authors.
The graph clearly shows that the majority of authors are from North America,
with some from Europe, but very few from regions like South America or
Oceania. None in Africa. Even the representation of women in this field is
limited, which reflects the broader reality of AI and ML as fields that have
been developed largely in North America and Europe.

// The visualization choices I've made, like separating the US into East and
West coasts, were intended to explore potential differences in perspectives.
Ultimately, this graph is an attempt to find patterns in this small but
meaningful dataset, much like what machine learning aims to do: analyzing
```

```
large datasets, identifying patterns, and making predictions based on those
patterns. This visualization, while based on only a limited number of
authors, allowed me to highlight key trends and insights within this
specific academic context. As I create this visualization to uncover
patterns and insights, I’m learning — and so is the machine. But in the end,
the question bubbles up: who is teaching?

d3.csv(googleSheetCSV).then(function(loadedData) {
    data = loadedData;

    data.forEach(function(d) {
        d.duration = +d.duration;
        d.startYear = +d['years-of-research'].split('-')[0];
    });

    updateChart();
});

// Resize the chart when the window is resized
window.addEventListener("resize", resize);
```

RapBattle.js

```
// Function to load the CSV file and populate the dropdowns
let authorData = {};

function loadAuthors() {
    const googleSpreadsheetUrl =
'https://docs.google.com/spreadsheets/d/e/2PACX-
1vQqCdcwLemmOhO16KOVWabBRabqQoRwx1QqIyS0mxZWq_O5dxYALM4JrZDu_LUoulbRQS6137gC
smJc/pub?gid=498870662&single=true&output=csv';

    fetch(googleSpreadsheetUrl)
        .then(response =%3E {
            if (!response.ok) {
                throw new Error('Network response was not ok: ' +
response.statusText);
            }
            return response.text();
        })
        .then(data => {
            const rows = data.split('\n').slice(1);  // Skip the header row
            const author1Select = document.getElementById('author1');
            const author2Select = document.getElementById('author2');

            rows.forEach(row => {
                const columns = row.split(/,(?=(?:(?:[^"]*"){2})*[^"]*$)/);
// Split row respecting commas inside quotes
```

```
                const author = columns[0]?.trim();  // First column: Author
name
                const notableWork = columns[1]?.trim();  // Second column:
Notable Work
                const miniBio = columns[9]?.trim();  // Tenth column: Mini
Bio

                if (author) {
                    // Store the data in a dictionary
                    authorData[author] = {
                        notableWork: notableWork || "No notable work
available",
                        miniBio: miniBio || "No bio available"
                    };

                    // Add options to both select elements
                    let option1 = document.createElement('option');
                    option1.text = author;
                    option1.value = author;
                    author1Select.add(option1);

                    let option2 = document.createElement('option');
                    option2.text = author;
                    option2.value = author;
                    author2Select.add(option2);
                }
            });
        })
        .catch(error => {
            console.error('Error loading authors:', error);
            const resultDiv = document.getElementById('rapBattleResult');
            resultDiv.innerHTML = `
                %3Cp style="color: red;">
                    Failed to load authors data. Please check your internet
connection or try again later.
                </p>
                <button onclick="loadAuthors()">Retry Loading
Authors</button>
            `;
        });
}

// FUNCTION TO UPDATE AUTHOR INFO
// One of the core educational ideas behind this project is the repetition
of information through interaction.Each time two authors are selected, the
user is presented with a brief bio of each, along with their notable
contributions. This helps reinforce memory through repeated exposure—an
essential aspect of learning.
```

```
// As the author of this project, I wanted to engage with the material in a
playful but effective way. By constantly returning to the authors and their
key ideas, I create opportunities to solidify that knowledge. The more I
interact with the data, the more it becomes ingrained. It's like learning
through play: the process of comparing these authors, selecting them in
pairs, and reading their bios repeatedly creates a deeper understanding of
their role in AI and Machine Learning. The mini-bios act as bite-sized
pieces of information that are easier to digest and remember over time.

// In this way, the rap battle format isn't just about entertainment—it's
also an educational tool that promotes learning by encouraging users to
actively engage with the content, fostering a stronger connection to the
material with each interaction.

function updateAuthorInfo() {
    const author1 = document.getElementById('author1').value;
    const author2 = document.getElementById('author2').value;

    const author1BioElement = document.getElementById('author1-bio');
    const author2BioElement = document.getElementById('author2-bio');

    const author1InfoTitle = document.querySelector('#author1-info h3');
    const author2InfoTitle = document.querySelector('#author2-info h3');

    // Update the bio of Author 1
    if (author1 && authorData[author1]) {
        const author1Bio = authorData[author1].miniBio || "No bio
available.";
        author1BioElement.textContent = author1Bio;
    } else {
        author1BioElement.textContent = "Select an author to see their
bio.";
    }

    // Update the bio of Author 2
    if (author2 && authorData[author2]) {
        const author2Bio = authorData[author2].miniBio || "No bio
available.";
        author2BioElement.textContent = author2Bio;
    } else {
        author2BioElement.textContent = "Select an author to see their
bio.";
    }

    // Atualiza o gráfico de bolhas com os autores selecionados
    if (typeof updateSelectedAuthors === 'function') {
        updateSelectedAuthors(author1, author2);  // Função que atualiza o
gráfico de bolhas
    } else {
        console.error("Function updateSelectedAuthors is not defined");
    }
}

// Debounce function to avoid calling the function too many times in quick
succession
function debounce(func, wait) {
    let timeout;
    return function(...args) {
        clearTimeout(timeout);
        timeout = setTimeout(() => func.apply(this, args), wait);
    };
}

// Update author info with a debounce to optimize performance
const updateAuthorInfoDebounced = debounce(updateAuthorInfo, 300);

// Add event listener to update bios when an author is selected with
debounce
document.getElementById('author1').addEventListener('change',
updateAuthorInfoDebounced);
document.getElementById('author2').addEventListener('change',
updateAuthorInfoDebounced);


// FUNCTION TO START RAP BATTLE
// Why a rap battle? The concept of a "rap battle" originates from the hip-
hop culture in the 1970s in the Bronx, New York, where two rappers would
face off, each trying to outdo the other with their lyrical prowess, wit,
and rhythm. It’s a verbal duel that  not only showcases skill but also adds
an element of entertainment and competition. In this project, we bring this
cultural concept into an academic context, comparing authors in AI and
Machine Learning as though they were rappers in a lyrical battle.

// This humorous juxtaposition invites users to think of these intellectuals
as contenders in the 'arena' of ideas, where their contributions are like
verses, each trying to 'outshine' the other in terms of impact and
innovation. The playful nature of the rap battle format makes the
exploration of academic figures feel less rigid and more approachable. By
pitting authors against each other, the user is encouraged to compare their
contributions in a fun and engaging way—almost as if they were competing in
the intellectual equivalent of a rap battle. After all, aren't AI and
Machine Learning just another form of  problem-solving and creativity, much
like crafting the perfect rap verse?

function startRapBattle() {
    const author1 = document.getElementById('author1').value;
    const author2 = document.getElementById('author2').value;
    const resultDiv = document.getElementById('rapBattleResult');
    // Clear previous error messages
    resultDiv.innerHTML = '';

    // Input validation
    if (!author1 || !author2) {
        resultDiv.innerHTML = '<p style="color: red;">Please select both
authors before starting the battle.</p>';

        // Add focus to the first empty field
        if (!author1) {
            document.getElementById('author1').focus();
        } else if (!author2) {
            document.getElementById('author2').focus();
        }

        return; // Stop the function from proceeding further
    }

    // Show loading indicator
    resultDiv.innerHTML = '<p>Loading battle...</p>';

    const author1Data = authorData[author1];
    const author2Data = authorData[author2];

    // Construct the prompt for OpenAI
    const prompt = `
        Imagine ${author1} and ${author2} are having a rap battle.
        The battle should have four parts:
        1. ${author1} starts by boasting about their work
"${author1Data.notableWork}" and criticizing ${author2}'s work
"${author2Data.notableWork}".
        2. ${author2} responds by boasting about their work
"${author2Data.notableWork}" and criticizing ${author1}'s work
"${author1Data.notableWork}".
        3. ${author1} replies with another verse, making references to other
works.
        4. Finally, ${author2} concludes the battle with a last verse.
        Each part should be clearly labeled with the author's name before
their verse.
    `;

    // OPENAI API CONTEXT
    // The OpenAI API is used here to generate the rap battle lyrics between
two selected authors. OpenAI was founded in 2015 as an open-source AI
research company, with the goal of making artificial intelligence
accessible. While it started as an open-source initiative, OpenAI has since
shifted its model and is now a capped-profit organization, with investors
like Microsoft holding a significant stake in the company. OpenAI is based
```

```
in the United States, and its work reflects the influence and values of the
tech industry in that region, which brings with it certain biases and
perspectives, just like the authors discussed in this project.

    // The tool itself is built on advanced technologies like Deep Learning,
specifically large language models (LLMs) and Natural Language Processing
(NLP). These models are trained on vast amounts of text data, which allows
them to understand and generate human-like responses. In this case, the API
is used to simulate a rap battle by drawing on a massive corpus of text,
including examples of lyrical structures. The API tries to find patterns in
language and mimic the format of a battle, but this process is heavily
dependent on the data it was trained on — data which is not fully accessible
or transparent to me as the user.

    // This is where the OpenAI API differs from something like D3.js. While
D3.js is based on SVG and is a fully open-source library, allowing me full
control over how data is rendered and visualized, OpenAI operates more like
a black box. I don't have access to the dataset that was used to train the
model, nor can I tweak the underlying patterns it has learned. This
introduces an element of uncertainty — while I can control the inputs to the
model (the authors selected for the battle), I am trusting OpenAI's bias in
generating the output. This is a significant contrast to my data
visualization work, where I had full control and made my biases clear in the
dataset and visualization process.

    // By using the OpenAI API, I am essentially relying on an external
system that has been trained on a dataset I did not curate and a model I did
not design. This highlights an important point about machine learning tools:
much of their power comes from pattern recognition, but the user has limited
insight into the biases inherent in the model. The rap battle, while
humorous and creative, is ultimately generated by a tool trained on data
whose origins and biases are unknown to me, unlike the bias I can clearly
see and control in my dataset of AI authors.

    fetch('/api/openai', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json',
        },
        body: JSON.stringify({ prompt: prompt })
    })
    .then(response => response.json())
    .then(data => {
        if (data.choices && data.choices.length > 0) {
            let battleText = data.choices[0].message.content;

            // Format the response text
            battleText = battleText.replace(/\*\*(.*?)\*\*/g,
'<strong>$1</strong>');
```

```
            battleText = battleText.replace(/\n/g, '<br>'); // Replace
newlines with <br> tags for line breaks

            resultDiv.innerHTML = `<h3>${author1} vs. ${author2}</h3>
<p>${battleText}</p>`;
        } else {
            resultDiv.textContent = 'The AI did not return a valid response.
Please try again later.';
        }
    })
    .catch(error => {
        console.error('Error:', error);
        resultDiv.textContent = 'An error occurred. Please try again
later.';
    });
}

// Add event listener to update bios when an author is selected
document.getElementById('author1').addEventListener('change',
updateAuthorInfo);
document.getElementById('author2').addEventListener('change',
updateAuthorInfo);

// Add event listener to start rap battle when button is clicked
document.getElementById('battleButton').addEventListener('click',
startRapBattle);

// Load authors when the page loads
window.onload = function() {
    loadAuthors(); // Load the authors from Google Spreadsheet
    updateAuthorInfo(); // Update the bios on page load
};
```

Conclusion

This essay, structured through comments embedded within the code, serves as both a technical and reflective exploration of the themes surrounding AI, Machine Learning, and data visualization. By interweaving code and commentary, I aimed to engage in the metalinguistic practice described by Bridle (2018), where technological literacy requires not only using tools but critically reflecting on how they shape our understanding of the world.

The choices I made in visualizing the dataset — such as rescaling the timeline and using bubbles — are deliberate and highlight the inherent biases in both data representation and interpretation. These decisions emphasize that neither data nor the tools we use to visualize it are neutral. As Steyerl (2023) and Schuman (2023) have pointed out, technology is deeply embedded in social and cultural contexts, and these biases are reflected in the outputs of systems like OpenAI's API or D3.js.

By employing tools like D3.js, SVG, and the OpenAI API, I've not only created a functional data visualization but also explored how these tools themselves carry embedded assumptions and limitations.

Ultimately, this essay invites the reader to reflect not only on the data itself but also on the ways we process, visualize, and interpret it. In doing so, it underscores the importance of questioning the biases within the tools we rely on, whether they are used for data visualization or AI generation.

As both humans and machines learn from data, the lingering question remains: who is teaching whom — and how?

References

Bridle, James. *New Dark Age: Technology and the End of the Future*. Verso Books, 2018.

Simon, Herbert A. *Models of Bounded Rationality*. MIT Press, 1982.

Steyerl, Hito. "Mean Images." *New Left Review*, no. 140, July–August 2023. https://newleftreview.org/issues/ii140/articles/hito-steyerl-mean-images.

Schuman, Lucy. "The Uncontroversial 'Thingness' of AI." *Big Data & Society*, 2023, doi:10.1177/20539517231206794.

Mitchell, Melanie. "Why AI is Harder Than We Think." *Communications of the ACM*, vol. 64, no. 8, 2021, pp. 68–74.

List of Technical Tools and Concepts with Links:

1. D3.js: Data-Driven Documents
   https://d3js.org/
   Accessed: September 12, 2024
2. Scalable Vector Graphics (SVG) - W3C
   https://www.w3.org/TR/SVG11/
   Accessed: September 12, 2024
3. OpenAI API
   https://openai.com/
   Accessed: September 12, 2024
4. Google Spreadsheet API
   https://developers.google.com/sheets/api
   Accessed: September 12, 2024
5. CSV (Comma-Separated Values) Format
   https://tools.ietf.org/html/rfc4180

Accessed: September 12, 2024

6. GitHub Repository for Hosting Code
https://github.com/
Accessed: September 12, 2024