

Playing with Pop Culture: Writing an Algorithm To Analyze and Visualize Lyrics From the Musical “Hamilton”

Erik Hinton
Wall Street Journal
1211 Ave. of the Americas
New York, N.Y. 10036
erik.hinton@dowjones.com

Joel Eastwood
Wall Street Journal
1211 Ave. of the Americas
New York, N.Y. 10036
1-212-416-2531
joel.eastwood@wsj.com

ABSTRACT

In this paper, we discuss the interactive visualization project “How Does ‘Hamilton,’ the Non Stop, Hip-hop Broadway Sensation Tap Rap’s Master Rhymes to Blur Musical Lines?” and the rhyme-pattern analysis algorithm that made the piece possible. This interactive extracts and describes the rhyme patterns interlaced throughout Lin-Manuel Miranda’s Broadway hit “Hamilton,” analyzing their higher-order structure to produce a genealogy of rap influence in the popular musical. The final section of the interactive allows readers to input their own lyrics and view the algorithmic categorization of their submission’s rhymes. The project was created for the Wall Street Journal and published online at <http://graphics.wsj.com/hamilton/>. An adapted portion of the project also appeared in the print edition. This paper explains the motivations behind the project, and the research and development of the user-interaction models, a unique visual language for rhyme patterns and the algorithm that produces them. We then use this process to suggest methods for getting news readers to play with pop-cultural analysis as a tactic to create strong feedback loops of engagement with news interactives that are not specifically games.

CCS Concepts

Human-centered computing: Visualization → Visualization application domains → Information visualization

Keywords

Information visualization; audio; lyric analysis; rhyme schemes; rap; hip-hop; musical theater; pop culture; Hamilton

1. INTRODUCTION

Pop-cultural analysis is rarely fun and even less-frequently interactive. Despite the play inherent to its media—the dance floors and drum circles of music, the social-media and water cooler perusal of TV—pop culture is stuck in the shapes of the review and the thinkpiece in the news. Around the edges, you

might find quizzes matching your personality to Disney princesses and infographics about the color palette of “Breaking Bad,” but these forms are generally didactic and static. The reader rarely has an opportunity to submit input, other than responding to queries and filtering views. We set out to produce a “Hamilton” project that would inform readers while encouraging them to play with our data, and to mirror, in our interactive project, the culture of fun that constantly remixes the source musical, producing “Hamilton” fan art, annotations on lyrics website Genius.com, outdoor musical numbers during the ticket lottery dubbed “Ham4Hams,” and many other attendant phenomena responsible for the musical’s stratospheric success.

But the trick of play, of rich interactivity, is procedural generation: true and deep response to a user’s inputs. We knew from the beginning that we couldn’t simply sit down with the lyrics to Hamilton and color-code the rhymes. Even if we created a joyful experience of this static color-coding, the feedback loop would be incomplete if readers couldn’t use our tools to analyze the rhymes of their own favorite lyrics. By closing the loop, the project becomes generative, and readers can take the lessons suggested in the body of the piece and find new connections between Hamilton and hip-hop history and beyond. In order to close the loop, we had to create an algorithm for automatic rhyme detection.

Though there have been many attempts to develop rhyme-mining algorithms [1][2] (some which we only discovered after publication [3]) all of them mismatched our requirements. Most of the prior art fell into two groups: 1) algorithms to determine whether or not two words rhyme and to identify patterns of rhyming words or 2) algorithms to determine abstract statistics on the density, quality, frequency and complexity of rhymes in rap lyrics. The first approach was insufficient because we needed to be able to highlight the syllable-level patterns. We required a finer-grained approach to accommodate the complex sub-rhymes and near rhymes of rap and “Hamilton.” The second approach was not useful because it didn’t group rhymes; it merely reduced over their statistics.

To this end, we developed our own algorithm that identified rhyming syllables of all qualities and attempted to cluster them into optimal families of similar sounds. The algorithm relied on speech phonetic research to quantify rhyming and sounding similar, as well as graph and genetic algorithms to cluster and optimize the subsequent rhyme families. Finally, we created an interactive chart template to display the rhymes output by the algorithm. Though this was simply a display tool for the analyzed data, it had to accomplish the difficult task of visualizing the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Computation + Journalism 2016, Sept. 30 – Oct. 1, 2016, Stanford, California, United States.

Copyright 2015 Erik Hinton and Joel Eastwood

predominantly auditory and temporal phenomena of rhyme, while not sacrificing ease of understanding. It had to present a complex topic as simply as possible while inviting readers to play with the data it was charting.

Following in this paper, we will present the details of both the algorithm and the user-interaction design. We will describe the relationship of previous work, in both algorithmic rhyme analysis and pop-cultural news interactives, as well as the unique challenges and solutions of the “Hamilton” project. Finally, we will summarize our learnings and suggest a strategy for future interactive news projects covering pop-culture.

2. BACKGROUND AND RELATED WORK

To familiarize ourselves with the rich history of hip-hop and rap composition, we read a number of texts on the subject. Paul Edwards’ “How to Rap: The Art and Science of the Hip-hop MC,”[4] and “The Rap Year Book: The Most Important Rap Song from Every Year since 1979, Discussed, Debated, and Deconstructed,”[5] by Shea Serrano, Arturo Torres, and Ice-T, provided essential background on the different schools, styles and evolutionary moments in rap. To understand the phonetic properties of vowels and consonants, we drew from the papers “Acoustic Characteristics of the Vowel Systems of Six Regional Varieties of American English,”[6] by Cynthia G. Clopper, David B. Pisoni, and Kenneth De Jong, as well as “Consonant Identification in Consonant-vowel-consonant Syllables in Speech-spectrum Noise,”[7] by David L. Woods, E. William Yund, Timothy J. Herron, and Matthew A. I. Ua Cruadhlaioich.

We studied several online projects that had attempted similar rhyme analysis, including Mike Lin’s “B-Rhymes Rhyming Dictionary”[2] and Eric Malmi’s “Algorithm That Counts Rap Rhymes and Scouts Mad Lines.”[1] After the publication of our project, we were informed of a similar attempt to programmatically identify rhymes by Hussein Hirjee and Daniel G. Brown, which is detailed in their paper “Using Automated Rhyme Detection to Characterize Rhyming Style in Rap Music.”[3]

A number of other publications have created their own visualizations of rap lyrics. Vox Media highlighted the rhyme patterns of prolific rappers in Estelle Caswell’s animated video “Rapping, deconstructed: the best rhymers of all time.”[8] Writing in Tech Insider, Gus Lubin identified the rhymes in excerpts of “Hamilton” in the article “The rhymes in Lin-Manuel Miranda’s ‘Hamilton’ are just insane.”[9] Finally, several others have experimented with combining audio and visuals to create interactive explorations of pop culture. One notable example is the online publication Polygraph, which has published pieces such as “How Music Taste Evolved,”[10] an animated, musical line chart of the Billboard top 100 from 1990 onward.

3. IDENTIFYING RHYME IN TEXT

3.1 Types of Rhyme

In order to train a computer to systematically identify rhymes, we first had to learn to define rhymes ourselves. In our earliest drafts, we listened to the first verse of the opening song of “Hamilton” and circled rhyming words by hand. It quickly became apparent that the rhyme structures of “Hamilton” specifically, and the genres of rap and hip-hop generally, defied easy categorization. Consider the difference between perfect and imperfect rhymes. In a perfect rhyme, different consonant prefixes are followed by an identical vowel and consonant sound. For example, the words

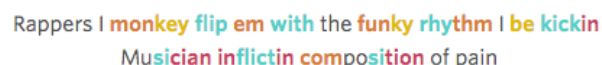
“scholar” and “squalor” rhyme perfectly with each other because, despite the different spellings, they both share an “AA” vowel sound and “ER” ending.

In contrast, an imperfect rhyme—a category that includes oblique and partial rhyme—will only partially match sounds, such as matching vowel sounds but different consonant endings. For example, the words “sweat” and “trek” partially rhyme because they share a similar “EH” vowel sound, despite having different consonant prefixes and suffixes. The words “ton” and “tin” are another imperfect rhyme, though in the opposite way: they share the same consonant prefixes and suffixes, but have completely different vowel sounds.

Imperfect rhymes are one of the defining traits of rap and hip-hop lyrics, so it is essential to consider them in any analysis of the rhyme schemes of these genres. The problem is complicated even further because rappers will pronounce or stress words in unusual ways to create rhymes that would not exist with a typical pronunciation of the word.

Another consideration is that rap and hip-hop lyrics will often combine rhyming sounds together across multiple words. As a result, it is not enough to simply compare two words and see if they rhyme; one must also consider adjacent words, as a multisyllabic rhyme can combine multiple words in different orders to produce unexpected rhyme patterns.

For example, when rapper Nas writes these words in this verse from his song “N.Y. State of Mind,” seen in Figure 1, he is rhyming the word “monkey” with “funky.”



Rappers I **monkey** flip em with the **funky** rhythm I be **kickin**
Musician **infectin** composition of pain

Figure 1: Color-coded lyrics from “N.Y. State of Mind”

However, the words “musician” and “composition” rhyme as well. In addition, the “com-“ in “composition” partially rhymes with the “mon-” and “fun-” sounds, as they share a similar vowel and consonant ending.

3.2 How the Algorithm Works

Once we began pondering the complexity of these rhyme patterns, we realized it was not enough to check if two words rhymed; instead, we needed to break words into their component syllables and compare those for rhyme. As such, we began writing an algorithm that could break each word into syllables and translate those into a regular phonetic language. We do this with the assistance of the CMU Pronouncing Dictionary with syllable annotations added by Susan Bartlett, Grzegorz Kondrak and Colin Cherry [11].

Their dictionary translates each word into a series of syllables, each of which is composed of a series of phonemes, written in a phonetic transcription code called the “Arpabet.” In order for two syllables to rhyme, they must share a similar vowel sound. The CMU Pronouncing Dictionary assigns each syllable one of 15 different vowel sounds. Any two syllables that have the same vowel sound, as represented by the Arpabet character, are potential rhymes. Two syllables with vowels that do not exactly match but sound similar are weaker potential rhymes.

We tested and refined the algorithm by plugging in lyrics from “Hamilton” and other hip-hop verses, and then comparing the output of the program to the rhymes we could hear while listening

to the song. Through this process we learned that vowels with similar resonant frequencies, which are often produced in similar parts of the mouth, are more likely to be rhymed together, as they are similar enough that a rapper can bend their sounds to produce syllables that rhyme. For instance, when “Hamilton” writer Lin-Manuel Miranda has the character Aaron Burr rhyme the word “Scotsman” with the words “dropped in,” he is able to do so because the “ə” vowel of “man” has very similar formant frequencies to the “ɪ” vowel of “in.”

Stress is another key part of rhyme. Rhyme tends to be heard most strongly between two stressed or emphasized syllables. We devised a scoring system, by which two matching syllables are given the highest score if they are both stressed. As such, the highest score goes to two stressed syllables. A pair of syllables with a single stressed syllable receives a lower score, as it is less likely to be an obvious rhyme. The lowest score is given to two unstressed syllables.

Finally, we look at the consonant sounds that come after the vowel in a given syllable to determine if two syllables rhyme. If two syllables share a similar vowel sound and the same suffix phonemes, we classify them as a rhyme as there is an extremely high chance they sound similar. As a visual aid to understanding this process, Figure 2 is the table of rhyme scores for the line “Forgotten spot in the Caribbean.” Higher scores are more likely to rhyme.

	F-ER	G-AA-T	T-AH-N	S-P-AA-T	IH-N	DH-AH	K-ER	IH	B-IY	AH-N
F-ER	N/A	0	0	0	0	0	1	0	0	0
G-AA-T	0	N/A	0	6.25	0	0	0	0	0	0
T-AH-N	0	0	N/A	0	1.56	0	0	0	0	1
S-P-AA-T	0	6.25	0	N/A	0	0	0	0	0	0
IH-N	0	0	1.56	0	N/A	0	0	0	0	1
DH-AH	0	0	0	0	0	N/A	0	0	0	0
K-ER	1	0	0	0	0	0	N/A	0	0	0
IH	0	0	0	0	0	0	0	N/A	0	0
B-IY	0	0	0	0	0	0	0	0	N/A	0
AH-N	0	0	1	0	1	0	0	0	0	N/A

Figure 2: Table of rhyme scores for lyrics “Forgotten spot in the Caribbean.”

3.3 Scoring Syllable Pairs

The scoring process for suffix sounds is almost identical to the process for vowel sounds. Identical suffixes receive the highest score. Suffixes that sound very similar, such as “m” and “n”, receive a lower score. We determined how closely suffixes sounded to each other primarily by examining how frequently listeners mistake one consonant sound for another, as described in the paper “Consonant identification in consonant-vowel-consonant syllables in speech-spectrum noise,” by David L. Woods [7]. We initially attempted to group rhymable similar consonant sounds based on phonetic properties, such as place of articulation, manner of articulation, and voicing, but, based on internal testing, it proved a poorer index of rhymability.

Once our algorithm could decompose lyrics into their component syllables and measure every syllable against every other, it could assign every syllable pair a rhyme score based on a product of these factors: vowel similarity; stress and syllable similarity; and stress. After scoring, the algorithm filters out rhymes based on a custom filter we devised through trial and error. Highly scored

pairs that were rarely actual rhymes included very commonly occurring English words, such as “then” and “an.” Another source of unwanted rhymes were rhymes that occur very far apart without similar rhymes between them. This false positive became more common when analyzing a larger body of lyrics, as a greater number of syllables increase the chances of a match through sheer coincidence. We also filtered out rhymes that were merely the repetition of two words, occurring outside of any larger rhyme patterns. Finally, we filtered out low-scoring rhymes that don’t appear to be in patterns with other rhymes, and therefore are likely unintentional or add little to the overall rhyme pattern.

3.4 Clustering and Ordering

This process results in an undirected graph, with the syllables as the nodes, in which edges are weighted based on rhyme quality. To determine the discrete rhyme families, we run a Markov Cluster Algorithm against the graph. Though the results of this clustering are not intuitively predictable, the results are better than our original efforts to greedily recursively group syllables into rhyme families, resolving disputes with some gross measure of family similarity. Finally, we can visualize the results. The syllables are represented as squares, colored into rhyme families with each family resting on its own line. We sort the families for clean visual display by running a simulated annealing pass against their order, scoring against how indexically near to each other successive syllables’ rhyme families are. This problem is similar, though not identical to, a traveling salesman problem. For instance, you would want to order the rhyme pattern “CABACABAC” as “ABCBABCBA”, for the most logical and pleasurable display, as seen in Figure 3.

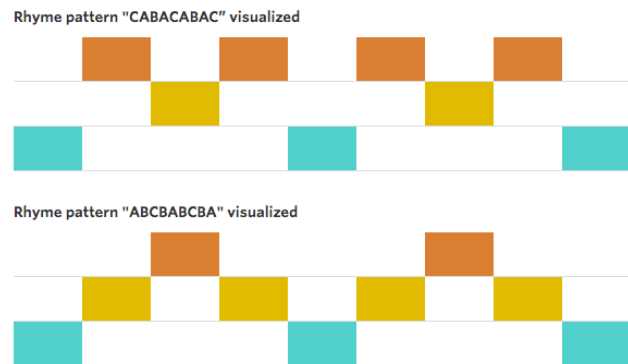


Figure 3: Sorting a rhyme pattern for logical display.

4. USER INTERFACE DESIGN

We knew that a crucial and difficult step in realizing the Hamilton interactive would be creating a compelling visual language for rhyme patterns. The design would have to be immediately understandable, abstractable to higher-order observations about rhyme pattern similarity and enjoyable to play with. After a long evolution, we settled on the format presented in Figure 4, in which each rhyme family has its own line as well as its own color. Individual syllables are represented as diamonds, lyrical fragments appear above their corresponding diamonds, the analyzed song plays in time with a rhythmic animation of the depicted rhymes, and users can click specific diamonds to hear the sound sampled there. The lyrics appear again beneath the visualization, with the rhyming syllables colored.

The double-encoding of the rhyme families by line and by color allowed for the simple visualization of patterns of alternation between two, three, or even more families of rhyme. The synchronized lyrics and music further allowed readers to intuitively understand the graphic, solving earlier problems in which it was hard to associate lyrics with their graphical representations. Finally, the on-demand playback of individual syllables and the placement of the full verse allowed readers to, at their own pace, explore the manifold ways in which rappers and Hamilton cast members bend and change words to create rhymes.



Figure 4: Final presentation of the opening verse of “Hamilton.”

There were many pieces of information we had to exclude to achieve the simplicity of presentation. We decided against encoding the quality of each rhyme, which presented as noise and distracted from the higher-order patterning. Early versions of the algorithm also produced metrics such as stress, rhyme density, rhyme complexity and multi-syllabic of rhyme. All of these suffered the same fate as rhyme quality. We could not find a way to include them without over-taxing readers.

5. MAKE YOUR OWN

The final piece of the interactive was the “Make Your Own” section, where we allowed readers to paste in any lyrics they wanted, visualizing them with the analysis of our algorithm. The first challenging in creating this section was porting our original algorithm code from server-side Python to client-side Javascript. We anticipated a large reader response and we wanted custom lyric analysis to be fast and seamless, without the overhead of server-side processing of an uncachable and CPU-intensive algorithm.

Next, we had to solve the design challenge of training our readers on what type of inputs would be most interesting to analyze. Much of the algorithm was designed with rap lyrics rather than pure poetry in mind, resulting in higher-quality results for hip-hop and pop. Fortunately, solving this problem allowed us to solve the problem of rap representation. Throughout the piece, space-constraints forced us to limit our examples from the history of hip-hop. By including “easy buttons” that filled in the Make Your Own section with some classic rap verses, we were able to suggest user behavior and surface some our favorite lyrics that we couldn’t include in the main piece (Figure 5).

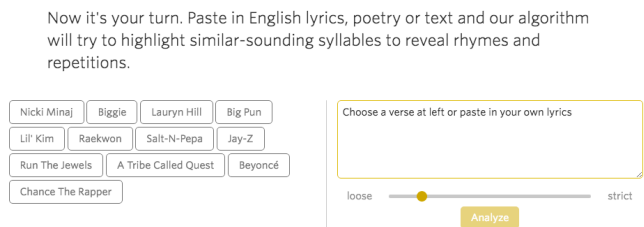


Figure 5: The culminating “Make Your Own” section, which visualizes sample verses or reader submissions.

6. READER RESPONSE

We were pleasantly surprised by the volume of user responses to the piece and overwhelmingly positive feedback. The piece received approximately five times as many online views as we aim for with a typical interactive graphic. This was particularly gratifying as the project was a pop-culture interactive on a news website that receives a majority of its traffic from audiences interested in financial and political coverage.

Readers spent much longer on our Hamilton page than on an average article or even a similar interactive project. We saw many users discussing the project on social media, including screenshots of our algorithm analyzing their personal poetry or lyrics and those of their favorite musicians. Each of these produced entirely organic opportunities for their communities to engage with our analysis and technology. Several users tweeted to indicate that the full interactive of the piece could be used to teach students about poetry, an unintended but desirable outcome.

The wide appeal of the project crossed demographic and interest-group lines, attracting attention from linguistics and poetry academics, theatre fans, and hip-hop enthusiasts. Rich interactivity and careful selection of topic allowed many different groups of readers to see their passions and curiosities reflected in our project. Lin-Manuel Miranda, the creator and star of “Hamilton,” tweeted his praise (Figure 6) for the interactive, applauding the care taken to showcase foundational but often-overlooked rappers such as Big Pun and Rakim in a major publication such as the Wall Street Journal.

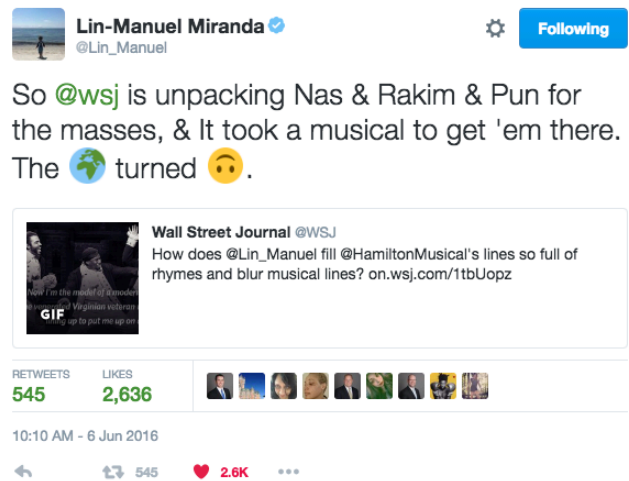


Figure 6: “Hamilton” creator Lin-Manuel Miranda’s tweet.

7. CONCLUSIONS

Deep user engagement with interactive news demands both rewarding interactivity and full user feedback. Merely changing views, sorting, filtering, and displaying static data, no matter how aesthetically pleasing the design, only supports shallow behavior. To close the loop between news piece and user, true interactivity must be achieved, bought at the cost of time-and-labor-intensive interface design, and client-facing algorithms and systems, generating novel responses to user input. “How Does ‘Hamilton,’ the Non Stop, Hip-hop Broadway Sensation Tap Rap’s Master Rhymes to Blur Musical Lines?” was a success because it provided a playful interface for our analytic conclusions and allowed users to experiment with our technology, generating and sharing their own conclusions.

REFERENCES

- [1] Malmi, Eric. "Algorithm That Counts Rap Rhymes and Scouts Mad Lines." Mining for Meaning. N.p., 13 Feb. 2015. Web.
- [2] "B-Rhymes Blog." B-Rhymes. Web. <<http://www.b-rhymes.com/blog>>.
- [3] Hirjee, Hussein, and Daniel G. Brown. "Using Automated Rhyme Detection to Characterize Rhyming Style in Rap Music." Empirical Musicology Review 5.4 (2010): 121-45. Web.
- [4] Edwards, Paul. How to Rap: The Art and Science of the Hip-hop MC. Chicago: Chicago Review, 2009.
- [5] Serrano, Shea, Arturo Torres, and Ice-T. The Rap Year Book: The Most Important Rap Song from Every Year since 1979, Discussed, Debated, and Deconstructed.
- [6] Clopper, Cynthia G., David B. Pisoni, and Kenneth De Jong. "Acoustic Characteristics of the Vowel Systems of Six Regional Varieties of American English." The Journal of the Acoustical Society of America J. Acoust. Soc. Am. 118.3 (2005): 1661-76.
- [7] Woods, David. L., E. William Yund, Timothy J. Herron, and Matthew A. I. Ua Cruadhlaioich. "Consonant Identification in Consonant-vowel-consonant Syllables in Speech-spectrum Noise." The Journal of the Acoustical Society of America J. Acoust. Soc. Am. 127.3 (2010): 1609. Web.
- [8] Caswell, Estelle. "Rapping, Deconstructed: The Best Rhymers of All Time." YouTube. Vox Media, 19 May 2016. <<https://www.youtube.com/watch?v=QWveXdj6oZU>>.
- [9] Lubin, Gus. "The Rhymes in Lin-Manuel Miranda's 'Hamilton' Are Just Insane." Tech Insider. 7 Jan. 2016. Web.
- [10] "How Music Taste Evolved." Polygraph. Web. 22 July 2016. <<http://polygraph.cool/history/>>.
- [11] Bartlett, Susan, Grzegorz Kondrak and Colin Cherry. "On the Syllabification of Phonemes." NAACL-HLT 2009.