Proceedings of the Eighteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment
(AIIDE 2022)

The Nonsense Laboratory

Allison Parrish,[1] **Jenny Goldstick,**[2] **Tim Szetela**[3]

[1] New York University
[2] Independent Researcher
[3] Princeton University
aparrish@nyu.edu, mail@jennygoldstick.com, tim@timszetela.com

Abstract

We present the Nonsense Laboratory, a series of playful web-based tools for manipulating the way words sound and the way words are spelled. The Laboratory's various playful interfaces give users an opportunity to adjust, poke at, mangle, curate, rearrange and elaborate the sounds of words in written text. The project makes use of Pincelate, a code library and machine learning model for phoneme-to-grapheme and grapheme-to-phoneme tasks in English, trained on the CMU Pronouncing Dictionary. The goal of the project is twofold: first, to make possible new ways of manipulating spelling (akin to playing a musical instrument or modeling with clay); and second, to demystify machine learning by providing an intuitive, friendly interface to a machine learning model. We briefly describe the design and implementation of the Nonsense Laboratory's five interfaces, and situate the project in our own research concerning design, literary studies, linguistics, and machine learning.

Introduction

A vital component of language is the way words sound. Works that exercise this desire for linguistic play include nonsense poems like Lewis Carroll's *Jabberwocky*, sound-focused authors like Dr. Seuss, and imaginary languages in the worlds of fantasy, science fiction and games—all of which enjoy enduring popularity in contemporary pop culture. As author Ursula K. Le Guin states, "The sound of the language is where it all begins and what it all comes back to. The basic elements of language are physical: the noise words make and the rhythm of their relationships. [...] Most children enjoy the sound of language for its own sake. They wallow in repetitions and luscious word-sounds and the crunch and slither of onomatopoeia; they fall in love with musical or impressive words and use them in all the wrong places. . . ." (Le Guin 2015).

The English language—with its rich vowel system and crunchy phonotactics—affords especially expressive phonetic play. Perhaps counterintuitively, these affordances are made more rich by the inconsistent spelling rules of English, in which words are never spelled quite like how they sound. To be sure, these rules are a source of constant frustration for English learners (children and adults alike). But as English readers and writers, the strategies we develop to bridge the gap between sound and spelling are also a source of endless creative possibility (Davies 1987). When we let ourselves play loose with grade school rules, we can use the way words are spelled to be expressive ("yaaaay!"), to inhabit personas ("you're a wizard, 'arry!"), and to invent imaginary worlds ("Bilbo Baggins" goes to "Rivendell"...).

```
noon

nein   threwe

nine   thein   three

knich   thike

six
```

Figure 1: Example poem from *Compasses*

The *Nonsense Laboratory*[1] is a series of playful web-based tools for manipulating the way words sound and the way words are spelled. Powered by Pincelate—a code library and machine learning model to spell English words from the way those words sound, and vice versa—the Laboratory's various playful interfaces give users an opportunity to adjust, poke at, mangle, curate, compress and elaborate the sounds of words in written text.

The Nonsense Laboratory is a continuation of our research into creative manipulation of English spelling with machine learning. This research culminated in the publication of several literary works, including *Compasses* (Parrish 2019) and *10,000 Apotropaic Variations* (Parrish 2020). (See figure 1 for an example of a poem from *Compasses*.) We published descriptions and source code for the techniques used to produce these works, with the goal of making the techniques more accessible to poets and creative writers with a non-technical background. However, we found that the technical nature of the research—requiring an understanding of computer programming, machine learning, and phonetics—presented too high a barrier for this audience. The

[1] Available online at https://experiments.withgoogle.com/nonsense-laboratory

Nonsense Laboratory is an attempt to overcome these barriers by offering a web-based interface that is friendly and playful, but also makes available as much functionality from the original code as possible.

Implementation

Model

The Nonsense Laboratory is implemented with a machine learning model called Pincelate (Parrish 2021). Pincelate consists of a pair of recurrent neural network models: one that performs English language grapheme-to-phoneme conversion, and another that performs English language phoneme-to-grapheme conversion. Both models are trained on the CMU Pronouncing Dictionary (Carnegie Mellon Speech Group 2014). The primary goal of this model is to facilitate artistic research in the poetics of nonsense words and neologisms. Like the phonetic models in our previous research (Parrish 2017), Pincelate represents phonemes not as atomic values but as a set of distinctive phonetic features (Halle and Chomsky 1968), each of which is assigned a probability at every timestep. This architecture is intended to make it possible for the model to produce plausible spellings even when presented with blended or impossible phonemes, which is important for the stated use case of the model. We most directly exploit this architecture in the "Sequencer" component of the Nonsense Laboratory, but its benefits are visible in the other components as well.

Pincelate was originally developed in Python using the TensorFlow (TensorFlow Developers 2022) and Keras (Chollet et al. 2015) frameworks. For the Nonsense Laboratory, we ported the model to JavaScript using TensorFlow.js. All model inference is performed on the client side. The site itself is developed in JavaScript using the Svelte web framework (Svelte Developers 2022).

Components

The Nonsense Laboratory is a web application that includes five components, described below.

Mixer The *Mixer* (see figure 2) takes advantage of the underlying vector representation of sound and spelling in Pincelate, allowing users to apply the equivalent of a "blend" or "crossfade" operation between words (based on their sounds and spelling). Supplied with two (or more) words, the interface invents a new word combining the sounds and letters of the source words. Internally, the code uses the hidden state of Pincelate's phoneme-to-grapheme model to represent the sound and spelling of a word, and then "injects" the mean of the vector values into the model before decoding orthography from the RNN. (This is the same technique we used in *Compasses*.) Some example blends that the interface produces:

- january + december = dismuery
- allison + parrish = palrison
- kitten + puppy = cuppey
- artificial + intelligence = intelificel

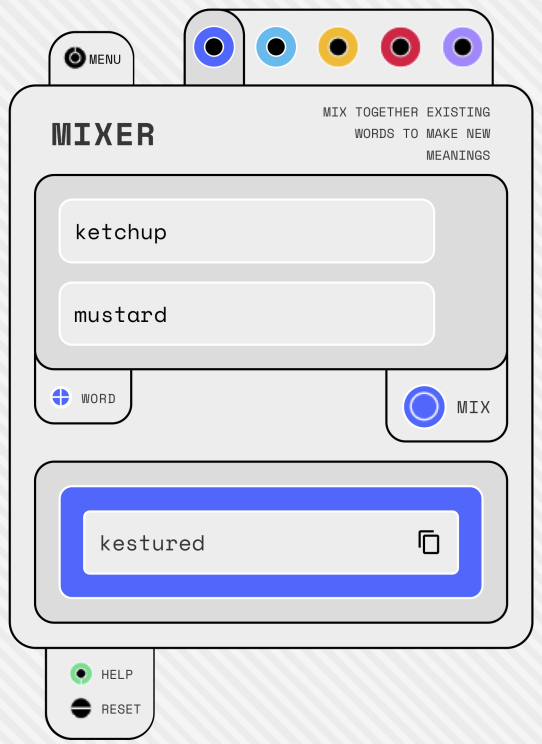

Figure 2: Mixer interface (mobile view)

Mouthfeel Tuner The *Mouthfeel Tuner* (figure 3) rewrites a text so that the words feel different in your mouth. The interface presents a text area for user input, alongside a number of sliders with impressionistic or onomatopoeic labels. Adjusting these sliders transforms and respells the text so that the phonetics of the words in the input are drawn closer to those indicated in the label. For example, the phrase *How doth the little crocodile improve his shining tail* with the "mpbfmpb" slider turned up becomes *Houf douft the lipple propopyl improv phiph fhinemif pale*. The phrase *interactive digital entertainment* with the "ahhh" slider turned up becomes *antaractive dagaital andortanement*. Multiple sliders can be engaged if desired.

The underlying code performs the equivalent of "logit warping" in the output layer of the grapheme-to-phoneme leg of the Pincelate model, emphasizing or attenuating the probabilities associated with each phonetic feature before decoding. For example, In the "mpbfmpb" example, the phonetic features associated with the lips (i.e., the bilabial and labiovelar) are emphasized. The "headcold" slider attenuates nasal phoneme features, while the "eee" feature boosts phonetic features associated with high front vowels (as in words like *heat* and *keep*).

Respeller The *Respeller* (figure 4) is the orthographic counterpart of the Mouthfeel Tuner: it rewrites a text, trying to retain its sound while omitting certain letters. The interface presents a text area for user input alongside a list of letters in the English alphabet. Tapping or clicking on those letters toggles the active state of the letter. When the "Apply"

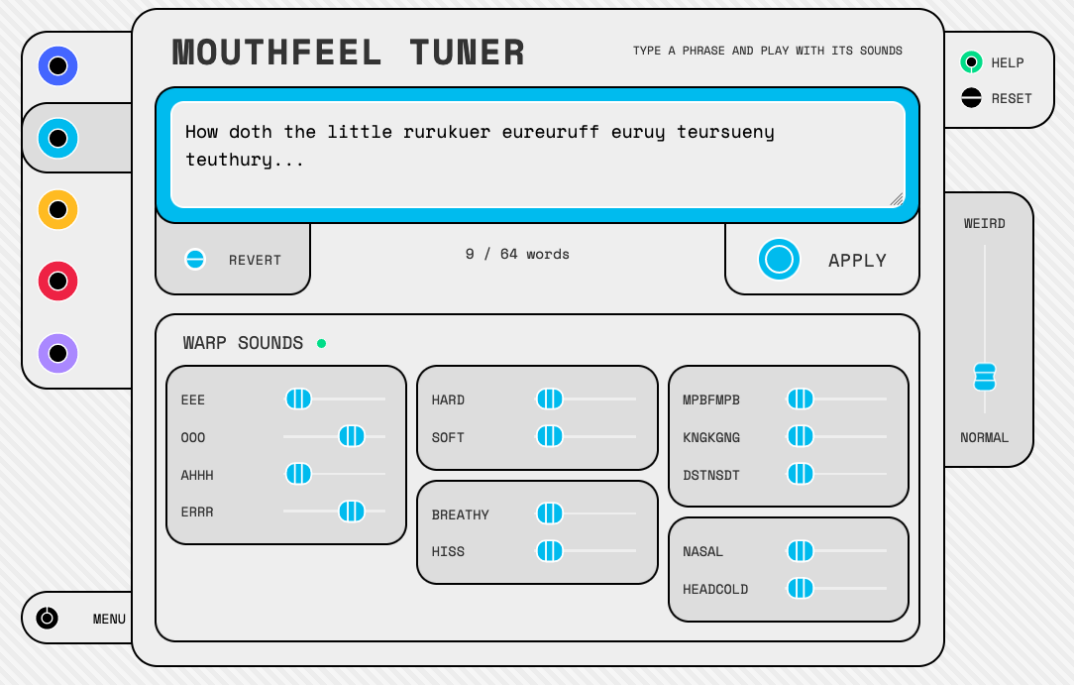

Figure 3: Mouthfeel Tuner interface (desktop view)

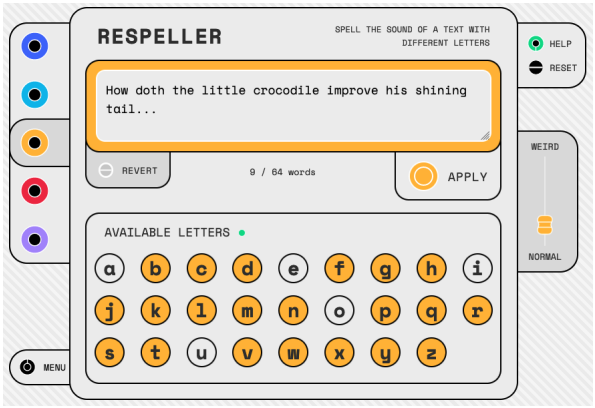

Figure 4: Respeller interface (desktop view)

button is activated, the text in the text area is rewritten, such that any of the letters that are set as inactive are not used, but the sound of the words is as close as possible to the original words. For example, the phrase *How doth the little crocodile improve his shining tail*, with all vowels (A, E, I, O, U) set to inactive, becomes *Hhw d'th th lyttl cr'ch-dy-dly ymprv hyss shyn-kyng t'l*. The phrase *interactive digital entertainment* with only "T" set to inactive becomes *inceraclive digidal enerdanemen'*.

The Respeller works in a manner very similar to the Mouthfeel Tuner. In this case, the output layer of the phoneme-to-grapheme model is modified at each timestep, such that the probability of any deactivated letter in the interface is set to zero. The model must therefore proceed as best as it can to spell the word's phonology as though the graphemes in question did not exist.

Sequencer The *Sequencer* (figure 5) spells a word from virtual mouth movements that users put in sequence, similar to the interface of a drum machine. The interface consists of a grid, where the columns are timesteps (i.e., individual phonemes), and the rows are phonetic features. An active grid square indicates that the given phonetic feature should be present at the given timestep. Tapping or clicking the "Play" button spells the word created by this sequence of phonetic features. Although the underlying model tracks several dozen phonetic features, we judged it to be cumbersome and intimidating to show all supported features to users when they first arrive at the interface. To mitigate this problem, the interface includes several "accordion" widgets so that less intuitive and less common phonetic features are hidden by default, and can later be revealed at the user's discretion.

Pincelate's phoneme-to-grapheme model is trained on entries from the CMU Pronouncing Dictionary, where each phoneme is represented with the values of a lookup table that associates each phoneme with its set of phonetic features. This amounts to a 2D array, where each timestep is a column and each row is set to 1 if a phonetic feature is present, and 0 otherwise. The code for the Sequencer interface constructs a similar array to use as input for Pincelate's phoneme-to-grapheme model, using the data from the Sequencer's grid. The model then decodes a spelling from that array.

Word Explorer Finally, the *Explorer* (figure 6) presents a Google Maps-like interface that encourages users pan

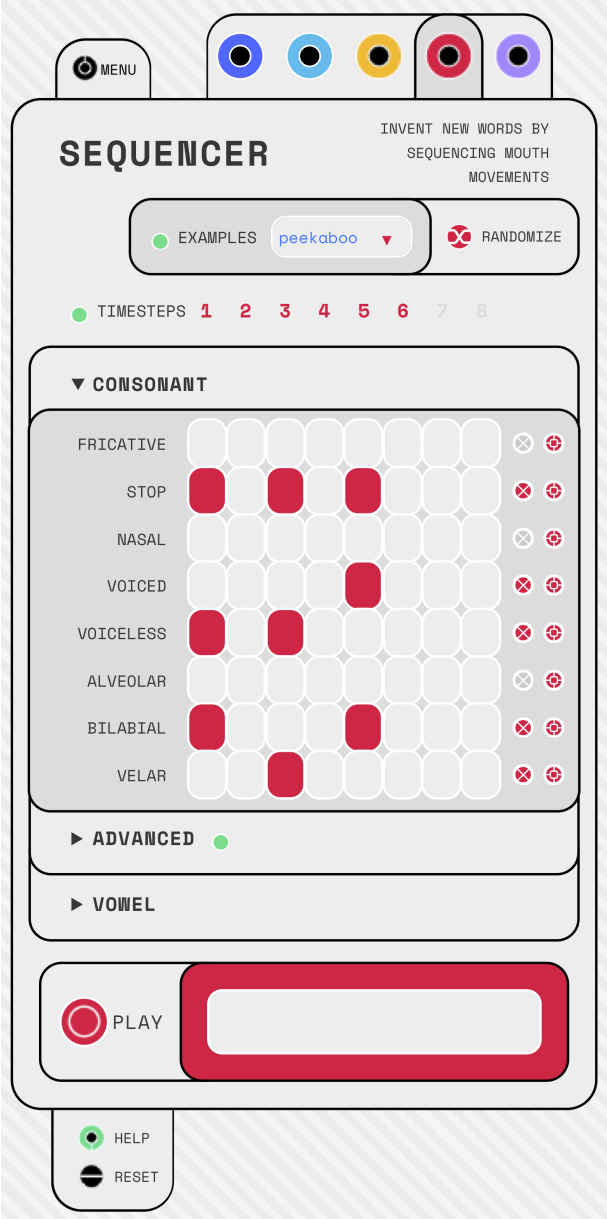

Figure 5: Sequencer interface (mobile view)

around an endless field of nonsense words, in which neighboring words have similar phonetics. Some of the words on the map are "real" words, and users can search for those words in order to discover nonsense words that sound alike. The "Jump" button moves the user to a random location in this phonetic space.

To generate the Explorer word grid, we calculated the RNN's hidden state vector in Pincelate's phoneme-to-grapheme model for several hundred words, then used a combination of umap-learn (McInnes et al. 2018) and RasterFairy (Klingemann 2022) to map the vectors of those words to a 2D grid where words with similar vectors will be located near each other. We then used the same interpolation code as used in the Mixer interface to generate new nonsense words *between* those existing words on the grid, giving the appearance of a continuous space of nonsense words. Because this process is computationally intensive, the grid data for the Explorer is pre-calculated (rather than being generated on the client, as is the case for the other interfaces).

Visual Design and User Experience

The Nonsense Laboratory was developed for Google Arts and Culture, whose stated audience is "anyone, anywhere." We entered the design process with a firm conviction that the kind of language play that Pincelate facilitates is inherently fun and intuitive, even to an audience with this scope. Nevertheless, we judged that this audience might have some trepidation when approaching both the subject of the tool (nonsense words) and its implementation (machine learning), both of which are technical and esoteric in nature. Our challenge, then, was to design an interface intuitive and inviting enough to encourage quick experimentation, while also exposing as much of the expressive potential of the model as possible. Toward this end, all of the interfaces in the Laboratory are either pre-populated with example input, or (as with the Sequencer) have preset examples that the user can select. Each interface shares a common design language that helps to communicate to users how and where they can interact (e.g., inputs, outputs, potential user actions, and hotspots that activate help messages when tapped or hovered over). Each interface also has a "Help" modal that describes in plain language the function of the interface in more detail, and offers some examples of things to try. (See figure 7 for an example.)

The structure and organization of the Laboratory changed dramatically during development. The original proposal was a single-screen application that would show all of the interfaces simultaneously (see 8). We rejected this idea early in the design process, for two reasons. First, we judged that the design would not be viable on mobile devices, which have a smaller amount of usable screen area. Second, we determined after a round of prototyping that the single-screen interface would require dramatically minimizing the amount of visible user interface and explanatory text, which was ultimately incompatible with our goal of making the Laboratory feel intuitive and inviting. As a consequence, we decided to implement each interface on its own page, with a common menu interface widget to make it possible to switch between interfaces. Each interface also was assigned an individual color treatment to emphasize this distinction.

The original design also included an interface called the "Phonetic Editor" (also visible in figure 8), which had the combined functionalities of what became the Respeller and the Mouthfeel Tuner. Both the Respeller and the Mouthfeel Tuner make use of the same underlying function call in the Pincelate model software, a fact which conditioned the original design. However, during testing with users, we discovered that the presence of both orthographic and phonetic was overwhelming to users, making the interface less approachable. Our solution was to split the interface into two: one interface (the Respeller) for modified spelling, and another interface (the Mouthfeel Tuner) for modified phonetics. In

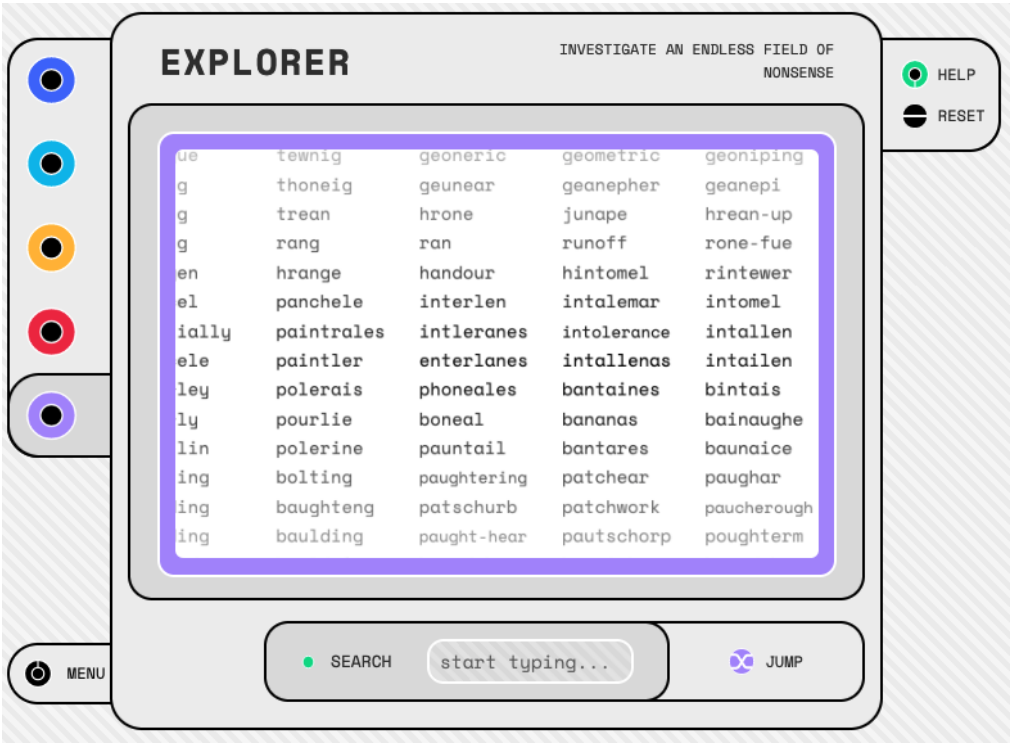

Figure 6: Explorer interface (desktop view)

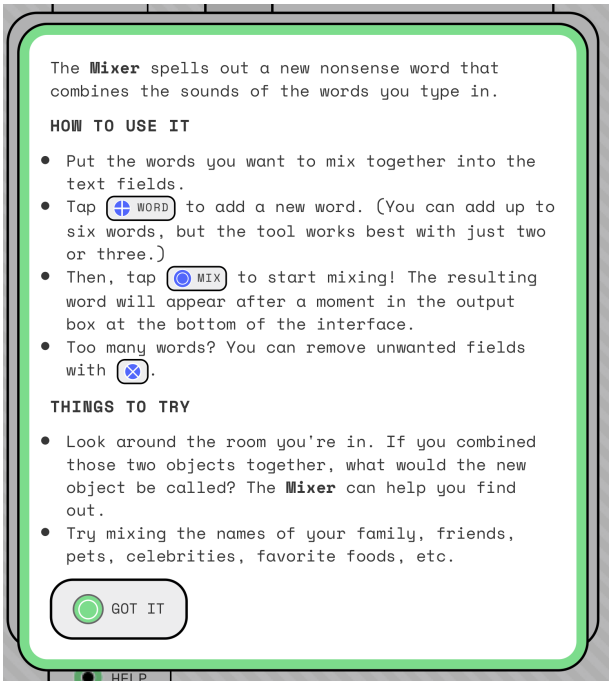

Figure 7: Screenshot of a "Help" modal dialog

our judgment, this solution is a good compromise: no functionality is lost, and the split allows for more user interface elements and explanatory text specific to the functionality in question.

Our guiding principle for the visual design was "playful, but not childish." The primary design cues we drew from were mid-20th century analogue control panels and synthesizers. We tempered the hi-tech feeling of these influences with bright colors, bold lines, generous spacing and sizing of the user interface elements, and subtle motion design that responds to user interaction. The end result, in our estimation, is a visual design that resembles neither a child's toy nor a highly technical tool, yet still invites play while not foreclosing on more practical uses. In designing for playful tools, we also looked to game design as inspiration. Designer Will Wright describes the potential of games and toys "like the way a telescope or microscope recalibrates our eyesight"—or more simply "an amplifier for the player's imagination" (Wright 2007). In Nonsense Laboratory, we aim to encourage a similar sort of creativity in exploring language and the way words sound.

Conclusion

We present the Nonsense Laboratory as an example of creating a playful tool for working with a specialized machine learning model, intended for a general audience. We hope that it can serve as a model for other tools that are not necessarily aimed at straightforward applications of AI in the arts and media, but at making it possible for a general audi-

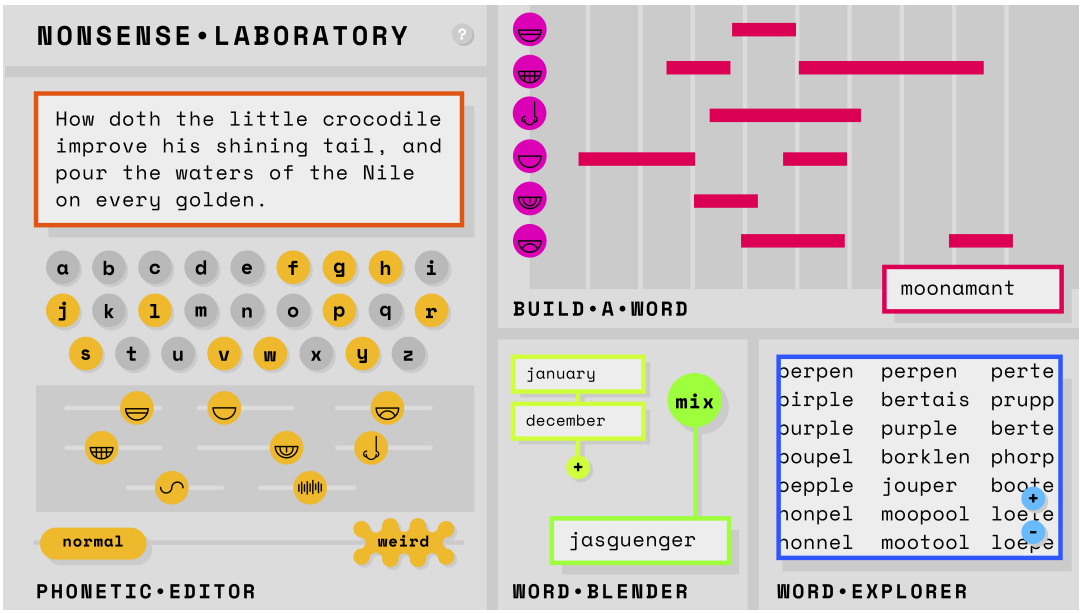

Figure 8: Initial visual design proposal for the Nonsense Laboratory

ence to explore the affordances and poetic possibilities that a particular machine learning model offers.

Acknowledgements

We thank Google Arts and Culture and Artists + Machine Intelligence for providing funding and logistical support for this work. Our special gratitude is given to Holly Grimm, who performed the initial work to port Pincelate to JavaScript, and Hannah Andrews, our tireless producer at Google Arts and Culture.

References

Carnegie Mellon Speech Group. 2014. The CMU Pronouncing Dictionary 0.7b. http://www.speech.cs.cmu.edu/cgi-bin/cmudict. Accessed: 2022-08-01.

Chollet, F.; et al. 2015. Keras. https://keras.io. Accessed: 2022-08-01.

Davies, E. E. 1987. Eyeplay: On Some Uses of Nonstandard Spelling. *Language & Communication*, 7(1): 47–58.

Halle, M.; and Chomsky, N. 1968. *The Sound Pattern of English*. Harper & Row.

Klingemann, M. 2022. RasterFairy-Py3. https://github.com/Quasimondo/RasterFairy. Accessed: 2022-06-03.

Le Guin, U. K. 2015. *Steering the Craft: A Twenty-first-century Guide to Sailing the Sea of Story*. Houghton Mifflin Harcourt. ISBN 978-0-544-61161-0.

McInnes, L.; Healy, J.; Saul, N.; and Großberger, L. 2018. UMAP: Uniform Manifold Approximation and Projection. *Journal of Open Source Software*, 3(29): 861.

Parrish, A. 2017. Poetic Sound Similarity Vectors Using Phonetic Features. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 8.

Parrish, A. 2019. *Compasses*. Andreas Bülhoff.

Parrish, A. 2020. *10,000 Apotropaic Variations*. Bad Quarto.

Parrish, A. 2021. Pincelate. https://github.com/aparrish/pincelate/. Accessed: 2022-06-02.

Svelte Developers. 2022. Svelte. https://github.com/sveltejs/svelte. Accessed: 2022-06-03.

TensorFlow Developers. 2022. TensorFlow. https://www.tensorflow.org/. Accessed: 2022-06-03.

Wright, W. 2007. Spore, Birth of a Game. https://www.ted.com/talks/will_wright_spore_birth_of_a_game. Accessed: 2022-08-08.