The AIIDE-17 Workshop on
Experimental AI in Games
WS-17-19

Poetic Sound Similarity Vectors Using Phonetic Features

Allison Parrish
Interactive Telecommunications Program
New York University
New York, NY 10003
aparrish@nyu.edu

Abstract

A procedure that uses phonetic transcriptions of words to produce a continuous vector-space model of phonetic sound similarity is presented. The vector dimensions of words in the model are calculated using interleaved phonetic feature bigrams, a novel method that captures similarities in sound that are difficult to model with orthographic or phonemic information alone. Measurements of similarity between items in the resulting vector space are shown to perform well on established tests for predicting phonetic similarity. Additionally, a number of applications of vector arithmetic and nearest-neighbor search are presented, demonstrating potential uses of the vector space in experimental poetry and procedural content generation.

Introduction

Research in psychology and psycholinguistics shows that—with apologies to Saussure—the sign is *not* arbitrary: words and sounds in language trigger synesthetic associations beyond pure denotation (Ramachandran and Hubbard 2001). Although a consistent and universal system of sound symbolism may not exist, patterns of sound inside a text nevertheless attract the the attention of those who encounter them, and thereby become important touchstones in the interpretation and negotiation of meaning:

> [C]entral words (in theme or imagery) strengthen the importance of a sound repetition; and the sound pattern, in its turn, focuses attention to the relations between these words (in meaning, or imagery). [...] Focusing patterns do not merely make important words conspicuous, or link disparate images, they may also reshuffle the emphasis of importance of words in a text, thereby imposing upon the poetic text an additional principle of order: that of sound relations. (Hrushovski 1980)

Alliteration, assonance, consonance and rhyme are all figures of speech deployed in literary writing in order to create patterns of sound, whether to focus attention or to evoke synesthetic associations. These figures are all essentially techniques for introducing *phonetic similarity* into a text, whether that similarity is local to a single word, or distributed across larger stretches of text. Indeed, the presence of phonetic similarity has long been considered a criterion for classifying a text as "poetic," and researchers have deployed quantification of phonetic similarity for a wide range of statistical analyses of literary style and quality (Skinner 1941; Altmann 1964; Roazzi, Dowker, and Bryant 1993; Kao and Jurafsky 2012).

The research presented in this paper concerns not just how to quantify phonetic similarity in a text, but also how to *characterize* and *manipulate* it. Inspired by recent results in vector representations of semantics based on word distribution (Bengio et al. 2003, Mikolov, Yih, and Zweig 2013, etc.) which allow for semantic relationships to be expressed using vector operations, I present a method for embedding text in a continuous vector space such that stretches of text that are phonetically similar are represented by similar vectors. This embedding affords not just a simple and consistent way to calculate phonetic similarity but also allows for phonetic relationships to be characterized as vector operations. In the remainder of the paper, I explain the model itself and give experimental results alongside several creative applications of vector operations in a phonetic similarity space.

The source code for this procedure, along with code to replicate the experiments is available online.[1] Available in the same location is a file containing pre-computed vectors for the CMU Pronouncing Dictionary.

Background and related research

One of the earliest systems for calculating phonetic similarity is Soundex, first used to classify and disambiguate personal names in studies of the United States Census in the 1930s (Stephenson 1974). Soundex-like systems that calculate phonetic similarity based on orthography alone are used in informatics applications such as information retrieval and spell-check (Philips 1990; Tissot, Peschl, and Fabro 2014). Approaches to quantifying phonetic similarity that specifically involve phonetic features have been used by linguists to study synchronic language variation (Ladefoged 1969), diachronic language change (Nerbonne 2010), and sound patterning in phonology (Mielke 2012).

Hahn and Bailey (2005) present a computational model for word similarity that uses measures of edit distance with sub-phonemic features weighted with regard to syllabic

[1] https://github.com/aparrish/phonetic-similarity-vectors/

Phone	Features	Phone	Features	Phone	Features
AA	bck, low, unr, vwl	F	frc, lbd, vls	P	blb, stp, vls
AE	fnt, low, unr, vwl	G	stp, vcd, vel	R	alv, apr
AH	cnt, mid, unr, vwl	HH	apr, glt	S	alv, frc, vls
AO	bck, lmd, rnd, vwl	IH	fnt, smh, unr, vwl	SH	frc, pla, vls
AW	bck, cnt, low, rnd, smh, unr, vwl	IY	fnt, hgh, unr, vwl	T	alv, stp, vls
AY	cnt, fnt, low, smh, unr, vwl	JH	alv, frc, stp, vcd	TH	dnt, frc, vls
B	blb, stp, vcd	K	stp, vel, vls	UH	bck, rnd, smh, vwl
CH	alv, frc, stp, vls	L	alv, lat	UW	bck, hgh, rnd, vwl
D	alv, stp, vcd	M	blb, nas	V	frc, lbd, vcd
DH	dnt, frc, vcd	N	alv, nas	W	apr, lbv
EH	fnt, lmd, unr, vwl	NG	nas, vel	Y	apr, pal
ER	cnt, rzd, umd, vwl	OW	bck, rnd, smh, umd, vwl	Z	alv, frc, vcd
EY	fnt, lmd, smh, unr, vwl	OY	bck, fnt, lmd, rnd, smh, unr, vwl	ZH	frc, pla, vcd

Table 1: Arpabet phone to feature mapping.

structure. Bradlow et al. (2010) introduce a vector space model for phonetic similarity, though the object of their study is not individual words but entire languages and their model parameters are drawn from a mixture of phonetic features and suprasegmental characteristics.

In the realm of computational creativity and procedural content generation, a great deal of research has used phonological information to condition and constrain text generation techniques. Most common are poetry generators that take lexical information about stress into account when generating metrically-constrained verse, or use phonological information to identify stretches of text that rhyme (see, e.g., Das and Gambäck 2014 for a good example of a generator that uses both). Approaches that take into account phonetic features beyond meter and rhyme, or use a continuous vector space to model phonetic similarity, are more difficult to find. Hervás, Robinson, and Gervás (2007) use a measurement of alliteration, calculated as a percentage of identical phonemes across a stretch of text, as part of a "fitness function" in a poem-generating genetic algorithm. Bay, Bodily, and Ventura (2017) use Euclidean distance between points on the International Phonetic Alphabet chart to estimate similarity between vowel segments in the service of detecting slant rhymes, but do not use a similar vector space for other types of phonemes.

Model

The overall goal is to create a vector space $\mathbb{R}$ for a data set where individual observations are embedded such that phonetically similar observations have similar vectors. In order to do this, a function $Prp(b)$ must be defined that maps an observation *b* to a corresponding vector. It must hold that for observations b_1 and b_2, the result of $similarity(Prp(b_1), Prp(b_2))$ increases according to the phonetic similarity of b_1 and b_2. For the purposes of the method being described, valid observations for $Prp(b)$ are sequences of Arpabet phonemes.

The technique for accomplishing this mapping is described in this section. At a high level, the technique can be described as extracting *interleaved bigrams of phonetic features*.

Phonetic features and context

Linguists have long recognized that certain pairs of phonemes in spoken language have a greater degree of similarity than other pairs. Patterns in allophony and diachronic sound change suggest that certain properties of phonemes, such as their manner and place of articulation, can be used as dimensions for reasoning about this perceived similarity (Austin 1957). Distinctive feature theory (Halle and Chomsky 1968) states that phonemes are discriminated from one another solely based on a small set of properties (such as [+/- voice] to distinguish voiced consonants from voiceless consonants; [+/- round] to distinguish rounded vowels from unrounded vowels), and that phonemes fall into categories of similarity based on shared features. Several analyses of similarity between languages and dialects have used quantification of shared features as a measure of phonetic similarity (Ladefoged 1969; Bradlow et al. 2010).

Drawing on this insight, it is clear that any embedding for phonetic similarity should be based on a feature-level analysis. For example, the words "back" (Arpabet transcription `/B AE K/`) and "pack" (`/P AE K/`) are intuitively more similar to one another than they are to "hack" (`/HH AE K/`), owing to the underlying similarity of the phonemes `/B/` and `/P/` (which share both a manner and place of articulation). Measurements of phonetic similarity that draw primarily or solely on orthography or phoneme-level transcriptions (Nelson and Nelson 1970; Philips 1990; Hervás, Robinson, and Gervás 2007; Tissot, Peschl, and Fabro 2014) fail to capture this intuition.

The method presented here assumes that each English phoneme can be mapped to a set of corresponding phonetic features. I use the mapping suggested in the specification for X-SAMPA (Kirshenbaum 2001), adapted to the Arpabet transcription scheme. The mapping of phonemes to features is shown in table 1.

Kirshenbaum's features are largely theory-agnostic, but are based primarily on articulatory properties of the phonemes (i.e., the physical configuration of the vocal tract observed when producing the sound). However, articulation of speech is not discrete, and individual features are not atomic but are influenced in their manifestation by the articulations that coincide with, precede and follow them

$$
\begin{aligned}
Prp(\texttt{BEG}, \texttt{R}, \texttt{IH}, \texttt{NG}, \texttt{END}) &= (F(\texttt{BEG}) \times F(\texttt{R})) \cup (F(\texttt{R}) \times F(\texttt{IH})) \cup (F(\texttt{IH}) \times F(\texttt{NG})) \cup (F(\texttt{NG}) \times F(\texttt{END})) \\
&= (\{beg\} \times \{alv, apr\}) \cup (\{alv, apr\} \times \{fnt, smh, unr, vwl\}) \\
&\quad \cup (\{fnt, smh, unr, vwl\} \times \{nas, vel\}) \cup (\{nas, vel\} \times \{end\}) \\
&= \{(beg, alv), (beg, apr), \\
&(alv, smh), (alv, fnt), (alv, unr), (alv, vwl), \\
&(apr, smh), (apr, fnt), (apr, unr), (apr, vwl), \\
&(smh, vel), (smh, nas), (fnt, vel), (fnt, nas), \\
&(unr, vel), (unr, nas), (vwl, vel), (vwl, nas), \\
&(vel, end), (nas, end)\}
\end{aligned} \tag{1}
$$

Figure 1: Example properties for the word "ring" /R IH NG/

(Lisker and Abramson 1967; Browman and Goldstein 1992; Hillenbrand, Clark, and Nearey 2001). As an attempt to model this fact, my procedure for phonetic similarity embedding is based not on individual phonetic features but on *interleaved bigrams* of features, as described below. (Using bigrams of features also helps to capture the sequencing of phonemes, in accordance with the intuitive notion that stretches of language are phonetically similar if their sounds occur in similar or identical orders.)

Feature engineering

Each observation in the data set should consist of a sequence of Arpabet phonemes. (These can be individual words, such as the phonetic transcriptions found in the CMU Pronouncing Dictionary, or transcriptions of longer stretches of text.) The function *Prp* to give the set of properties for observation *b* of length *n* is calculated like so:

$$Prp(b) = (F(p_1) \times F(p_2)) \cup \cdots \cup (F(p_{n-1}) \times F(p_n))$$

where $F(i)$ evaluates to the set of phonetic features corresponding to phoneme *i* (given in table 1) and p_i is the phoneme at position *i* in the observation.

Anchoring and order. As part of the feature extraction process, two special pseudo-phonemes, BEG and END, are added to each observation to respectively anchor the beginning and ending of the observation. The pseudo-phoneme BEG has one corresponding phonetic feature, *beg*, and the pseudo-phoneme END has one corresponding feature, *end*. Figure 1 shows an example feature extraction for the word "ring" (transcribed in Arpabet as R IH NG).

In order to capture the similarity of words regardless of ordering of phonemes (e.g., "tack" /T AE K/ and "cat" /K AE T/ are considered to be similar words, despite the mirrored order of their phonemes), the procedure calculates properties for the reversed order of the phonemes in an observation as well. (For the sake of brevity, the properties obtained from the reversed procedure are omitted from figure 1).

Dimensional reduction. The set of all properties in the data set is the union of each observation's properties. After the set of properties for the data set is determined, the number of times each property occurs in each observation is recorded. The end result is a matrix of size $(m \times n)$ where *m* is the number of observations and *n* is the number of unique properties found. The value at location (i, j) in the matrix is the number of times feature *j* occurs in observation *i*.

When using this method with all 133,852 entries in version 0.7b of the CMU Pronouncing Dictionary, 949 unique properties (i.e., 949 unique interleaved feature bigrams) are found across the entire data set, yielding a matrix of size $(133852, 949)$. To make this number more manageable, the PCA class from Scikit-learn (Pedregosa et al. 2011) is deployed to perform a principal components analysis on the matrix, reducing the number of dimensions to 50.

Applications and experiments

The resulting embeddings have several interesting characteristics, illustrated by the experiments and poetic applications given in this section.

Phonetic similarity

As a baseline for subsequent experimentation, the vector space resulting from the method described above was evaluated for its ability to predict phonetic similarity, by using methodology and data from Vitz and Winkler (1973). In the experiments conducted by Vitz and Winkler, small groups of L1 American English speakers were asked to score the phonetic similarity of a "standard" word with a list of comparison words on a scale from 0 (no similarity) to 4 (extremely similar). Several experiments were performed with different standard words and comparison words. Vitz and Winkler compare the elicited scores to the results from their own procedure for determining phonetic similarity (termed "Predicted Phonemic Distance," shortened here as PPD).

In my experiment, I calculated the cosine similarity between the vector corresponding to the standard word and the vectors corresponding to each of the comparison words in three of Vitz and Winkler's experiments. Figure 2 shows the result of this procedure with data from Vitz and Winkler's experiment 3 alongside their original results for that experiment.[2]

[2]The scale for Vitz and Winkler's "predicted phonemic distance" ($PPD(w)$) starts at 0 and goes up to 1. In order to maintain

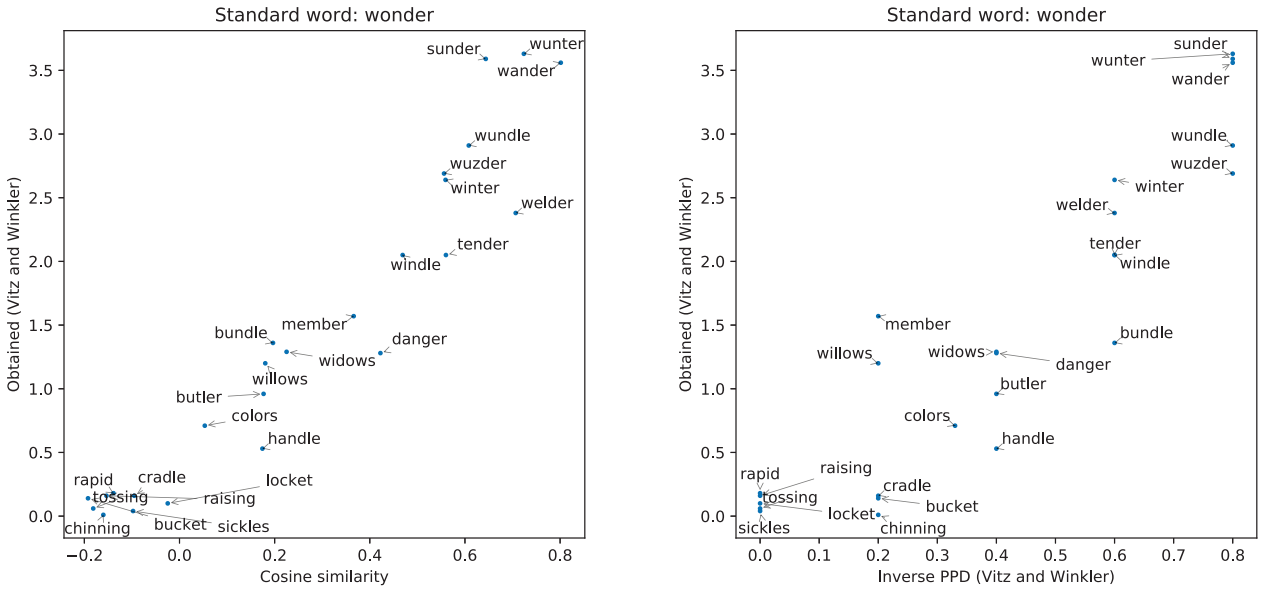

Figure 2: Scatter plots of vector space cosine similarity versus obtained average similarity (Vitz and Winkler 1973) (left) and "predicted phonemic distance" versus obtained average similarity (right).

In experiment 3, cosine similarity from the vector space outperforms Vitz and Winkler's metric, with a correlation of 0.958 (compared to 0.917 for PPD). Results for experiment 2 are also favorable, with a correlation of 0.776 compared to PPD's correlation of 0.795. However, in results for experiment 1, which concerns monosyllabic words, vector space cosine similarity significantly underperforms, with a correlation of 0.646 compared to PPD's correlation of 0.941. Overall, these results demonstrate that a vector space constructed according to the method described in this paper reflect commonsense intuitions about phonetic similarity, though the result for experiment 1 suggests that a fruitful next step is to fine-tune the feature engineering for better accuracy with short sequences of phonemes.

Arithmetic and analogy

For the following applications, the procedure described in the Model section was used to calculate an embedding $\mathbb{R}$ for all of the entries in the CMU Pronouncing Dictionary 0.7b (Carnegie Mellon Speech Group 2014). A function $Vec(t)$ evaluates to the vector associated with token t, and a function $Nn(v)$ returns the nearest neighbor in $\mathbb{R}$ of vector v. Experimentation with these operations and vector arithmetic revealed surprising phonetic relationships latent in the vector space, analogous to the semantic relationships observed in word distribution vector spaces (Mikolov, Yih, and Zweig 2013). In the applications below, nearest-neighbor lookups were performed with the Annoy Python package (Bernhardsson 2017) using simple Euclidean distance as the distance metric.

No	Operation	Result
1	$Nn(Vec(\texttt{sub}) + Vec(\texttt{marine}))$	"submarine"
2	$Nn(Vec(\texttt{miss}) + Vec(\texttt{sieve}))$	"missive"
3	$Nn(Vec(\texttt{fizz}) + Vec(\texttt{theology}))$	"physiology"
4	$Nn(Vec(\texttt{curiously}) - Vec(\texttt{lee}))$	"curious"
5	$Nn(Vec(\texttt{wordsworth}) - Vec(\texttt{word}))$	"disregards"
6	$Nn(Vec(\texttt{ingredient}) - Vec(\texttt{reed}))$	"insignia"

Table 2: Sample sound arithmetic results

As shown in table 2, vector addition and subtraction in $\mathbb{R}$ model the analogous phonetic relationships between words. Adding the vector of one word to the vector of another yields a word in $\mathbb{R}$ that combines the phonetic features of both. Likewise, subtracting the vector of one word from another yields a word in $\mathbb{R}$ that has the phonetic features of the first word but lacking the features of the second.

Items (1), (2) and (4) in table 2 show that some varieties of regular English derivational morphology can be expressed as vector operations in $\mathbb{R}$. Items (3), (5) and (6), on the other hand, show that the relationship between operands and the resulting nearest neighbor can be approximate, capturing a word that *feels* like the result of the operation, maintaining as close an order of phonemes to the original as possible.

The use of vector arithmetic also shows that offsets between vectors can capture phonetic relationships between *classes* of words. These relationships can be described in terms of analogy: w_a is to w_b as w_c is to w_d, where w_d is

consistency across charts, I have instead used $1 - PPD(w)$ in figure 2. The correlation calculations also reflect this inversion (i.e., correlation numbers are positive rather than negative as they originally appeared in Vitz and Winkler).

No	w_a	w_b	w_c	w_d
1	decide	decision	explode	explosion
2	glory	glorify	liquid	liquefied
3	bite	bitten	shake	shaken
4	leaf	leaves	calf	calves
5	foot	feet	tooth	teeth
6	automaton	automata	criterion	criteria
7	four	fourteen	nine	nineteen
8	light	slide	lack	slag
9	whisky	whimsy	frisky	flimsy
10	could	stood	calling	stalling

Table 3: Sample sound analogies ($w_a : w_b :: w_c : w_d$)

given using the following expression:

$$Nn(Vec(w_b) - Vec(w_a) + Vec(w_c))$$

Table 3 shows a few sample results from such analogies. As shown in items (1–7), vector offsets are able to capture and reproduce English regular and irregular morphology on phonetically similar items, to the extent that these morphological processes can be described using phonetics alone, even when the morphology is non-productive (as in item 7). Items (8–10) show that the analogical process can also operate on more "poetic" sound correspondences, finding pairs of words whose phonetic relationship is similar to the relationship between any other arbitrary pair, even when the difference between the words is an individual feature, not an entire phoneme.

Sound symbolism "tinting"

By representing each word as its corresponding vector in the CMU Pronouncing Dictionary phonetic space $\mathbb{R}$, an entire text can be conceptualized as a vector of size (m, n) where *m* is the number of words in the text and *n* is the dimensionality of the vector space. The resulting text vector can be manipulated using techniques analogous to those used to manipulate audio or images. For example, in the same way that an image can be digitally "tinted" by adding a constant value to the red, green and blue components of each pixel, a text can have its sound symbolism "tinted" by adding a constant to the vector for each word in the text, and then rewriting the text by finding the closest word in $\mathbb{R}$ for each vector.

In many studies of synaesthesia and sound symbolism, the word *kiki* is synesthetically associated with sharp, spiky shapes, while the word *bouba*[3] is synesthetically associated with round, oblong shapes (Ramachandran and Hubbard 2001; D'Onofrio 2014). Drawing on this research, I attempted to rewrite Robert Frost's celebrated poem *The Road Not Taken* (Frost 1916) using sound symbolism tinting to produce a "spiky road" version and a "round road" version. Table 4 shows the result. The first column of the table shows the original text, while the second column shows the result of adding $Vec(\texttt{kiki}) \times 0.8$ to each word, and the third column shows the result of adding $Vec(\texttt{babu}) \times 0.8$ to each word. In both the second and third columns, the underlying vectors have been transformed back into words using the $Nn(v)$ function described above.

[3]The word *babu* is substituted for *bouba* in this example, as there is no entry for *bouba* in the CMU Pronouncing Dictionary.

> Sweet hour of prayer, sweet hour of prayer it was the hour of prayers. In the hour of parting, hour of parting, hour of meeting hour of parting this. With power avenging, ... His towering wings; his power enhancing, in his power. His power. Thus: the blithe powers about the flowers, chirp about the flowers a power of butterfly must be with a purple flower, might be the purple flowers it bore. The petals of her purple flowers where the purple aster flowered, here's the purple aster, of the purple asters there lives a purpose stern! A sterner purpose fills turns up so pert and funny; of motor trucks and vans, and after kissed a stone, an ode after Easter. And iron laughter stirred, O wanderer, turn; oh, wanderer, return. O wanderer, stay; O Wanderer near. Been a wanderer. I wander away and then I wander away and thence shall we wander away, and then we would wander away, away O why and for what are we waiting. Oh, why and for what are we waiting, why, then, and for what are we waiting?

Figure 3: Sample random walk through lines of poetry in Project Gutenberg, based on sound similarity.

In this experimental application, no further attempt has been made to normalize the grammaticality of the text post-transformation, although every word found in the texts is a "valid" English word (in the sense that it occurs in the CMU Pronouncing Dictionary). Leaving aside for future research the specific claim on the ability to induce sound symbolism, I find the results of this experiment very promising. Although the two transformed texts appear to be nonsense, they are nevertheless immediately recognizable as variations on the original in which consistent yet aesthetically distinct phonetic transformations have been applied.

Random walks through poem space

The previous experiments and applications have used single words as the embedded units in the vector space. However, any stretch of Arpabet phonemes can be embedded. As an example of embedding longer sequences of phonemes, I found embeddings for a large subset of all lines of poetry in Project Gutenberg (http://www.gutenberg.org). For each line in the poetry corpus, an Arpabet transcription composed of the concatenated phonetic transcriptions of every word was used as a single observation in the matrix. (Lines containing words that are not found in the the CMU Pronouncing Dictionary were excluded from the embedding.)

The text in figure 3 is the result of performing a "random walk" through the resulting vector space. Starting with a randomly selected item from the vector space, the procedure finds the nearest neighboring vector that hasn't already been selected, and appends the text of the corresponding vector to the output. The procedure repeats, jumping from one nearest neighbor to the next, until a text of a particular desired length is reached. This radical composition technique results in a text that smoothly moves from one pattern of sounds to the next, evoking sound symbolism in uncanny ways. The procedure flattens semantic, syntactic and stylistic differences

Baseline	$Nn(Vec(w) + Vec(\texttt{kiki}) \times 0.8))$	$Nn(Vec(w) + Vec(\texttt{babu}) \times 0.8))$
Two roads diverged in a yellow wood	To roads diverged in a yellow woodke	To roads barboursville in a yellow would
And sorry I could not travel both	And sarti i gokey knotty keevil booth	And bari i koba knob bava both
And be one traveler long I stood	And be one traveler long i stookey	And bobby one bosler long i stobaugh
And looked down one as far as I could	And loci down one as far as i gokey	And jukebox doub one as fahd as i koba
To where it bent in the undergrowth	Tuckey kiwi eat bent in the undergrowth	To bowell it bondt in the bogard
Then took the other as just as fair	Then kupek the other as keast as fichera	Babu bocook the bother as bobst as fair
And having perhaps the better claim	And having perhaps the becky claim	And having bopper the bobbette claim
Because it was grassy and wanted wear	Geeky eat was keesee and wanted kiwi	Babic it zabawa basi and wambaugh bowell
Though as for that the passing there	Though as for that the kiki giguere	Though as for bogacz the babu babu
Had worn them really about the same	Heeg worn them keeley kabuki the safekeeping	Hob worn them bali abboud the same
And both that morning equally lay	And booth that morning equally lay	And both bogacz booming equally lay
In leaves no step had trodden black	In teves know techie heeg teagarden blackie	In bob's know bobbette hob babar blob
Oh i kept the first for another day	Oh i khaki the thirsty for another ghee	Oh i bobcat the first for bother ba
Yet knowing how way leads on to way	Fekete knowing how way tiegs on tuckey way	Bobbette knowing baja way bob's on to way
I doubted if I should ever come back	I tiki if i shooed keever come bacchi	I bowed if i should ever kebab bob
I shall be telling this with a sigh	I shall be leckey keith withey a sigh	I shall bobby babu bob with a seibu
somewhere ages and ages hence	Somewhere keizai and keizai hence	Babu bobby's and bobby's hence
two roads diverged in a wood and I	To roads diverged in a woodke and i	To roads barboursville in a would and i
I took the one less traveled by	I kupek the one leckey keevil bacchi	I bocook the one babu traveled ba
and that has made all the difference	And that has pigue all the defrance	And bogacz has baydhabo all the difference

Table 4: Sample sound symbolism tinting with Robert Frost's *The Road Less Taken*.

in favor of phonetic cohesion, while retaining the local-level grammatical coherence of the lines themselves.

Conclusion

I have described a method for embedding stretches of text in a continuous vector space according to their phonetic similarity. The procedure can be fruitfully applied to create vector spaces consisting of individual words or vector spaces with larger stretches of text, such as lines of poetry. Vector spaces generated with this technique have been shown to have useful and surprising properties related to their ability to represent phonetic relationships between words. They have also been shown to be an effective and original tool for unconventional manipulation and composition of poetic language.

Potential applications

Phonetic similarity vector embeddings based on phonetic features have the potential to be a useful tool for researchers and practitioners in various fields.

One potential area is computational stylistics and aesthetics. The model of poetry quality in Kao and Jurafsky (2012), for example, includes a variable for phonetic similarity, defined as the presence of assonance and consonance. However, their calculation was programmed to find *identical* phonemes, and only identical phonemes within a limited window (nine syllables). As a consequence, their analysis fails to account for how phonemes with shared features are perceived as similar, and how phonetic similarity functions not just between adjacent words but as a stylistic attribute that structures an entire text (Fónagy 1961; Hrushovski 1980).

An analysis of sound similarity in a text based on a vector embedding, on the other hand, can easily account for these factors. Moreover, with a vector representation, rule-based schemes for phonetic analysis can be supplanted with potentially more powerful machine learning techniques for classification, regression and clustering. A model with the same goal as Kao and Jurafsky could (for example) compute how the use of sound similarity in a poem is focused or diffuse, or calculate a "phonetic footprint" for individual poems, authors and genres. Kao and Jurafsky ultimately conclude that phonetic similarity "did not have significant predictive value" in determining whether a poem was most likely written by a professional or amateur poet, but I believe that a more nuanced characterization of sound similarity with regard to poem quality could be reached using these techniques.

Likewise, research in the field of computational creativity surrounding the generation of lyric and poetic text could benefit from a robust vector representation of phonetic similarity. For example, both Ghazvininejad et al. (2016) and Malmi et al. (2016) use sophisticated vector-based representations of semantics as part of the first pass of the text-generation process, then use a system of rules to filter the resulting text for words and lines that adhere to a meter and have internal rhymes or end rhymes. At the very least, a phonetic vector embedding of words and lines could serve as a way to narrow the search space in these procedures for further rule-based filtering on rhyme and meter. A more ambitious application of phonetic vectors would be to conceptualize a poem as consisting of both a phonetic component and a semantic component, or a combined representation of both, and with these components apply vector operations to transform the underlying semantics into a surface text that exhibits the desired phonetic characteristics. (One potential process for performing this kind of phonetic "style transfer" is hinted at in the "sound symbolism tinting" application in the previous section.)

The original impetus for this research was to enable innovative ways, in the tradition of sound poets like Hugo Ball and Kurt Schwitters, to "arrange words, not according to their semantic meanings, but according to their phonetic valences" (Bök 2009). I am particularly interested in using this vector embedding model as a part of augmented and mixed-initiative interfaces for writing. In the spirit of Amiri Baraka's "expression scriber" (Baraka 1965), I envision a system in which poems can be composed expressively using analog inputs to stretch, distort, sharpen and (dis)color the sound of a text in real-time.

Limitations and future research

There are several important limitations to this research described in this paper. The first is empirical verifiability. While the model presented here was intended from the beginning for aesthetic rather than "practical" ends (i.e., computer-generated poetry), evaluating the model on existing experimental data for phonetic similarity proved invaluable. As my research in this area continues, I plan to seek out (or create) larger and more robust data sets for evaluating phonetic similarity and phonetic relationships between words.

The number of dimensions (fifty) used in the dimensionality reduction process was determined through a process of trial and error as a reasonable compromise between computational efficiency and maintaining the predictive integrity of the vectors. The dimensionality reduction technique itself (Principle Components Analysis) was selected primarily for its speed and convenience. (In particular, Scikit-learn's `IncrementalPCA` implementation made it possible to prototype and implement the techniques in the paper on inexpensive consumer hardware.) It is quite likely that more sophisticated techniques for reducing dimensionality, such as a manifold learning algorithm like Isomap (Tenenbaum, De Silva, and Langford 2000), would yield superior results. Future iterations of this research will establish a baseline of phonetic similarity performance against which different dimensionality reduction techniques can be evaluated.

Another problem is the burden of feature engineering. While the features in the model described in this paper perform admirably on experimental and poetic tasks, the model can only be improved by "tweaking" the features. Ideally, a machine learning algorithm would be able to learn features for the similarity vectors in an unsupervised fashion. Research in distributional learning of phonological rules (Calamaro and Jarosz 2015) and probabilistic rhyme detection (Hirjee and Brown 2010) may provide a way forward in this area.

Finally, the experiments and applications of the model in this paper relied solely on words in the CMU Pronouncing Dictionary. Ideally, the model would be able to cope with out-of-vocabulary words and other stretches of text. One possible first step in this task is to train a neural network to learn similarity vectors for raw sequences of characters based on the orthography of sequences already found in the vector space.

References

Altmann, G. 1964. The measurement of euphony. In *Teorie Verše. I, Sborník Brněnské Versologické Konference, 13.-16. Května 1964*.

Austin, W. M. 1957. Criteria for Phonetic Similarity. *Language* 33(4):538–544.

Baraka, I. A. 1965. Technology & Ethos. *Raise, Race, Rays, Raze: Essays Since 1965* 155–158.

Bay, B.; Bodily, P.; and Ventura, D. 2017. Text Transformation Via Constraints and Word Embedding. In *Proceedings of the 8th International Conference on Computational Creativity (ICCC'17)*.

Bengio, Y.; Ducharme, R.; Vincent, P.; and Jauvin, C. 2003. A neural probabilistic language model. *Journal of machine learning research* 3(Feb):1137–1155.

Bernhardsson, E. 2017. Annoy: Approximate Nearest Neighbors in C++/Python optimized for memory usage and loading/saving to disk.

Bök, C. 2009. When Cyborgs Versify. In *The Sound of Poetry/The Poetry of Sound*. Chicago: University of Chicago Press. 129–141.

Bradlow, A.; Clopper, C.; Smiljanic, R.; and Walter, M. A. 2010. A Perceptual Phonetic Similarity Space for Languages: Evidence from Five Native Language Listener Groups. *Speech communication* 52(11-12):930–942.

Browman, C. P., and Goldstein, L. 1992. Articulatory phonology: An overview. *Phonetica* 49(3-4):155–180.

Calamaro, S., and Jarosz, G. 2015. Learning General Phonological Rules From Distributional Information: A Computational Model. *Cognitive Science* 39(3):647–666.

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

Das, A., and Gambäck, B. 2014. Poetic machine: Computational creativity for automatic poetry generation in Bengali. In *5th International Conference on Computational Creativity, ICCC*.

D'Onofrio, A. 2014. Phonetic Detail and Dimensionality in Sound-shape Correspondences: Refining the Bouba-Kiki Paradigm. *Language and Speech* 57(3):367–393.

Fónagy, I. 1961. Communication in Poetry. *WORD* 17(2):194–218.

Frost, R. 1916. *Mountain Interval*. New York, NY, USA: Henry Holt and Company.

Ghazvininejad, M.; Shi, X.; Choi, Y.; and Knight, K. 2016. Generating Topical Poetry. In *EMNLP*, 1183–1191.

Hahn, U., and Bailey, T. M. 2005. What makes words sound similar? *Cognition* 97(3):227–267.

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

Hervás, R.; Robinson, J.; and Gervás, P. 2007. Evolutionary assistance in alliteration and allelic drivel. *Applications of Evolutionary Computing* 537–546.

Hillenbrand, J. M.; Clark, M. J.; and Nearey, T. M. 2001. Effects of consonant environment on vowel formant patterns. *The Journal of the Acoustical Society of America* 109(2):748–763.

Hirjee, H., and Brown, D. G. 2010. Using Automated Rhyme Detection to Characterize Rhyming Style in Rap Music. *Empirical Musicology Review* 5(4).

Hrushovski, B. 1980. The Meaning of Sound Patterns in Poetry: An Interaction Theory. *Poetics Today* 2(1a):39–56.

Kao, J., and Jurafsky, D. 2012. A computational analysis of style, affect, and imagery in contemporary poetry. In *NAACL Workshop on Computational Linguistics for Literature*, 8–17.

Kirshenbaum, E. 2001. Representing IPA Phonetics in ASCII. https://web.archive.org/web/20160304092234/http://www.kirshenbaum.net/IPA/ascii-ipa.pdf.

Ladefoged, P. 1969. The Measurement of Phonetic Similarity. In *Proceedings of the 1969 Conference on Computational Linguistics*, COLING '69, 1–14. Stroudsburg, PA, USA: Association for Computational Linguistics.

Lisker, L., and Abramson, A. S. 1967. Some Effects of Context On Voice Onset Time in English Stops. *Language and Speech* 10(1):1–28.

Malmi, E.; Takala, P.; Toivonen, H.; Raiko, T.; and Gionis, A. 2016. DopeLearning: A Computational Approach to Rap Lyrics Generation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, 195–204. San Francisco, California, USA: ACM.

Mielke, J. 2012. A phonetically based metric of sound similarity. *Lingua* 122(2):145–163.

Mikolov, T.; Yih, W.-t.; and Zweig, G. 2013. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of NAACL-HLT*, 746–751.

Nelson, D. L., and Nelson, L. D. 1970. Rated acoustic (articulatory) similarity for word pairs varying in number and ordinal position of common letters. *Psychonomic Science* 19(2):81–82.

Nerbonne, J. 2010. Measuring the diffusion of linguistic change. *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 365(1559):3821–3828.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Philips, L. 1990. Hanging on the metaphone. *Computer Language* 7(12 (December)).

Ramachandran, V. S., and Hubbard, E. M. 2001. Synaesthesia–a window into perception, thought and language. *Journal of consciousness studies* 8(12):3–34.

Roazzi, A.; Dowker, A.; and Bryant, P. 1993. Phonological abilities of Brazilian street poets. *Applied Psycholinguistics* 14(4):535–551.

Skinner, B. F. 1941. A Quantitative Estimate of Certain Types of Sound-Patterning in Poetry. *The American Journal of Psychology* 54(1):64–79.

Stephenson, C. 1974. Tracing Those Who Left: Mobility Studies and the Soundex Indexes to the U.S. Census. *Journal of Urban History* 1(1):73–84.

Tenenbaum, J. B.; De Silva, V.; and Langford, J. C. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290(5500):2319–2323.

Tissot, H.; Peschl, G.; and Fabro, M. D. D. 2014. Fast Phonetic Similarity Search over Large Repositories. In *Database and Expert Systems Applications*, 74–81. Springer, Cham.

Vitz, P. C., and Winkler, B. S. 1973. Predicting the judged "similarity of sound" of English words. *Journal of Verbal Learning and Verbal Behavior* 12(4):373–388.