Applications of graph theory to an English rhyming corpus

Morgan Sonderegger*

University of Chicago, Department of Computer Science, 1100 East 58th Street, Chicago, IL 60637, USA

Abstract
How much can we infer about the pronunciation of a language – past or present – by observing which words its speakers rhyme? This paper explores the connection between pronunciation and network structure in sets of rhymes. We consider the rhyme graphs corresponding to rhyming corpora, where nodes are words and edges are observed rhymes. We describe the graph G corresponding to a corpus of \(~12000\) rhymes from English poetry written c. 1900, and find a close correspondence between graph structure and pronunciation: most connected components show community structure that reflects the distinction between full and half rhymes. We build classifiers for predicting which components correspond to full rhymes, using a set of spectral and non-spectral features. Feature selection gives a small number (1–5) of spectral features, with accuracy and \(F\)-measure of \(~90\%\), reflecting that positive components are essentially those without any good partition. We partition components of \(G\) via maximum modularity, giving a new graph, \(G'\), in which the “quality” of components, by several measures, is much higher than in \(G\). We discuss how rhyme graphs could be used for historical pronunciation reconstruction.

Keywords: Rhymes, graph theory, complex networks, poetry, phonology, English

1. Introduction

How can we reconstruct what English sounded like for Pope, Shakespeare, or Chaucer? Pronunciation reconstruction traditionally involves triangulation from several sources; one crucial type of data is rhyming verse [1]. Because rhymes are usually between words with the same endings (phonetically), we might infer that two words which rhyme in a text had identically-pronounced endings for the text’s author. Unfortunately, this reasoning breaks down because of the presence of “half” rhymes. Consider the following rhymes, from poetry written by William Shakespeare around 1600.\(^1\)

(a) But kept cold distance, and did thence remove,
   To spend her living in eternal love.
(b) And deny himself for Jove,
    Turning mortal for thy love.
(c) But happy monarchs still are fear’d for love:
    With foul offenders thou perforce must bear,
    When they in thee the like offences prove:
    If but for fear of this, thy will remove;
(d) And pay them at thy leisure, one by one.
    What is ten hundred touches unto thee?
    Are they not quickly told and quickly gone?

\(^1\)“A Lover’s Complaint” (a,e), Love’s Labour Lost IV.3 (b), “The Rape of Lucrece” (c), “Venus and Adonis” (d,f).

Preprint submitted to Computer Speech and Language March 10, 2010
(e) Which fortified her visage from the sun,
    Whereon the thought might think sometime it saw
    The carcass of beauty spent and done:
    Time had not scythed all that youth begun,
(f) What bare excuses makest thou to be gone!
    I'll sigh celestial breath, whose gentle wind
    Shall cool the heat of this descending sun:

We write \( x:y \) when a rhyme is observed between \( x \) and \( y \), and \( x \sim y \) if \( x \) and \( y \) have the same ending (in a sense made more precise below). One intuitively knows, from experience with songs or poetry, that if \( x \sim y \) then it is possible to rhyme \( x \) and \( y \), and that usually, \( x:y \) implies \( x \sim y \). If we assume that \( x:y \Rightarrow x \sim y \) always, then from Examples (a)–(c),

\[
\text{love} \sim \text{Jove} \sim \text{remove} \sim \text{prove}
\]

and from Examples (d)–(f),

\[
\text{one} \sim \text{gone} \sim \text{sun} \sim \text{done} \sim \text{begun}
\]

However, it turns out that not all words in the first group of rhymes were pronounced the same for Shakespeare, while all words in the second group were.\(^2\)

Because of the uncertainty in the implication \( x:y \Rightarrow x \sim y \), in pronunciation reconstruction rhyming data is only used together with other sources, such as grammar manuals and naive spellings \([1]\). But these sources are expensive and limited, while rhyming data is cheap and plentiful. If we could somehow make the implication stronger, rhyming data could stand on its own, making reconstruction significantly easier.

This paper attempts to strengthen the implication in two ways: first, by building classifiers to separate half (e.g. (a)–(c)) from full (e.g. (d)–(f)) groups of rhymes, based on the groups’ rhyme graphs; second, by breaking groups of rhymes into smaller and more full groups, based on the structure of their rhyme graphs. Although the long-term goal of this project is to infer historical pronunciation, this paper uses recent poetry, where the pronunciation is known, to develop and evaluate methods. We first (Sec. 2–3) introduce rhyme graphs, outline the corpus of poetry used here, and describe its rhyme graph, \( G \). In Sec. 4, we build classifiers for components of \( G \), using a set of features which reflect components’ graph structure. We then (Sec. 5) partition components into smaller pieces, giving a new graph \( G’ \), and evaluate the quality of rhymes in \( G’ \) versus \( G \).

2. Data

2.1. Rhyming corpora

Rhyming corpora have traditionally been used in two ways by linguists interested in phonology.

In diachronic phonology, collections of rhymes are traditionally a key tool for pronunciation reconstruction, (e.g. \([3, 4, 8]\) for English); in this case the focus is on full rhymes, which indicate identity between (parts of) words. In synchronic phonology, rhyming corpora have been used for Japanese song lyrics \([9]\), Romanian poetry \([10]\), English song lyrics \([11, 12]\), and English poetry \([13, 14, 15]\).\(^3\) In these cases, the focus is on half rhymes (see below, Sec. 2.2), which reflect speakers’ intuitions about phonological similarity.

Our use of rhyming corpora differs in several ways. First, we are interested in both half and full rhymes. Second, we consider rhyming corpora primarily from the perspective of (applied) graph theory, rather than a linguistic framework. Most importantly, previous work has focused on small subsets of rhymes (usually individual rhymes), or the local structure of a corpus; our focus is on global structure, as reflected in the corpus’ rhyme graph.

Our corpus consists of rhymes from poetry written by English authors around 1900.\(^4\) The contents of the corpus, itemized by author, are summarized in Table 1. Poetry was obtained from several online

\(^2\) General sources on pronunciation around 1600 are (in order of accessibility) \([2, 3, 4]\); contemporary phonetic transcriptions (e.g., \([5, 6, 7]\)) provide direct evidence.

\(^3\)However, none of the English poetry corpora are electronically available.

\(^4\)We use poetry from c. 1900 rather than the present day due to practical considerations. First, rhyming poetry has become
Poet # Rhymes (10^1) Sources
A.E. Housman (1859–1936) 1.52 [16, 17, 18, 19]
Rudyard Kipling (1865–1936) 2.60 [20, 21, 22]
T.W.H. Crosland (1865–1924) 0.60 [23]
Walter de la Mare (1873–1956) 1.74 [24]
G.K. Chesterton (1874–1936) 1.29 [25]
Edward Thomas (1878–1917) 0.52 [26]
Rupert Brooke (1887–1915) 1.05 [27]
Georgian Poets (c. 1890) 3.07 [28]

Table 1: Summary of authors of rhymes used in the corpus. “Georgian Poets” are contributors to the Georgian Poetry anthologies (1912–22) [29].

<table>
<thead>
<tr>
<th>Word</th>
<th>IPA</th>
<th>Short RS</th>
<th>Long RS</th>
</tr>
</thead>
<tbody>
<tr>
<td>bat</td>
<td>bæt</td>
<td>æt</td>
<td>æt</td>
</tr>
<tr>
<td>cement</td>
<td>sɛnt</td>
<td>ɛnt</td>
<td>ɛnt</td>
</tr>
<tr>
<td>England</td>
<td>ɛŋ.gland</td>
<td>ɛŋl@nd</td>
<td>ɛŋland</td>
</tr>
</tbody>
</table>

Table 2: Examples of short and long rhyme stems. ‘IPA’ indicates a word’s pronunciation (from CELEX) in International Phonetic Alphabet transcription.

databases; most were public-domain sources, one (Twentieth Century English Poetry (Chadwyck-Healey); [30]) is available through university libraries.

Poems were first hand-annotated by rhyme scheme, then parsed using Perl scripts to extract rhyming pairs. All rhyming pairs implied by a given rhyme scheme were counted, not just adjacent pairs of words. For example, the rhymes counted for (c) above were love:prove, prove:remove, and love:remove, rather than only the first two pairs.

To simplify automatic parsing, we standardized spelling as necessary, for example counting learned and learn’d as the same word.5 We also associated each spelling with its most frequent pronunciation; for example, all instances of wind were recorded as [wInd], corresponding to the most frequent (noun) pronunciation.

2.2. Pronunciations, rhyme stems

We took pronunciations for all words in the corpus from CELEX [31], a standard electronic lexicon of British English, as pronounced c. 1988 (in Everyman’s English Dictionary; [32]). Using 1988 norms for pronunciations around 1900 is an approximation, but a relatively good one, as standard British pronunciation (“RP”) has changed relatively little over this period [33]. Importantly, rhyming data is only affected by the relative pronunciation of words, so that among changes between 1900 and 1988, only mergers and splits (see below, Sec. 6.1) would affect rhyme quality. The mergers and splits in the RP noted by Wells [33] all affect small sets of words. In examining rhyme graphs for 1900 data using CELEX pronunciations, we only noticed inconsistencies for a few words.

We first define what we mean by “rhyme” and “rhyme stem”. Two different definitions of the English rhyme stem (RS) are often used; we call these the short rhyme stem and long rhyme stem, and consider both in this paper. A word’s short rhyme stem is the nucleus and coda of its final syllable, and its long rhyme stem is all segments from the primary stressed nucleus on; Table 2 gives examples. Short and long rhyme stems were found for all words in the corpus, again using CELEX.

less popular over the twentieth century, making it more difficult to find enough rhymes for a sizable corpus. Second, much recent poetry is still under copyright, making it harder to obtain electronic versions of poems.

5Spelling variants in poetry can indicate different intended pronunciations, a possibility we are abstracting away from.
### Table 3: Examples of full rhymes and half rhymes, for short and long rhyme stems. For example, the short rhyme stems of *travel* and *gobble* are identical (full rhyme), but their long rhyme stems are not (half rhyme). It is not possible for two words to have identical long rhyme stems (full rhyme) but different short rhyme stems (half rhyme).

<table>
<thead>
<tr>
<th>Short RS</th>
<th>Full rhyme</th>
<th>Half rhyme</th>
</tr>
</thead>
<tbody>
<tr>
<td>Long RS</td>
<td>parting:darting</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>travel:gobble</td>
<td>parted:portrait</td>
</tr>
</tbody>
</table>

Once a definition has been chosen, each word has a unique rhyme stem. A *rhyme* is a pair of two words, \( w_1 \) and \( w_2 \), observed in rhyming position in a text.\(^6\) Assuming that a definition of the rhyme stem has been chosen, the rhyme is *full* if the rhyme stems of \( w_1 \) and \( w_2 \) are the same, and *half* otherwise.\(^7\) Table 3 shows examples of full rhymes and half rhymes between pairs of bisyllabic words, for both rhyme stem definitions.

### 3. Rhyme graphs

#### 3.1. Notation

We first introduce formalism for associating a rhyming corpus with a weighted graph, the *rhyme graph*. The general idea of using graph theory to study sets of rhymes has to our knowledge been proposed once before, in the 1970s by Joyce [34, 35], with a somewhat different formalism and smaller scope than here.\(^8\)

A rhyming corpus consists of a set \( R \) of rhymes, defined as (unordered) pairs of words. We write a rhyme between words \( v_i \) and \( v_j \) as \( \{v_i, v_j\} \). Let \( V \) be the set of all word types observed in some pair, and let \( n = |V| \). Let \( n_{ij} \) be the number of times the rhyme \( \{v_i, v_j\} \) is observed, and assume \( n_{ii} = 0 \) (there are no self-rhymes). Let \( d_i \) be the *degree* of \( v_i \):

\[
d_i = \sum_j n_{ij}
\]

Let \( a_i \) be the number of edges connected to \( v_i \): \( a_i = |\{v_j \mid n_{ij} > 0\}| \).\(^9\)

We associate with a rhyme corpus two types of weighted graph \( G = (V,E,W) \), the *rhyme graph*, where \( E = \{ \{v_i, v_j\} \mid v_i, v_j \in V, n_{ij} > 0 \} \), and \( w_{ij} \) is the weight of the edge between words \( w_i \) and \( w_j \):

1. Unnormalized weights: \( w_{ij} = n_{ij} \)
2. Normalized weights: \( w_{ij} = n_{ij} / \sqrt{d_i d_j} \).

Let \( d'_i \) be the *weighted degree* of node \( i \): \( d'_i = \sum_j w_{ij} \)

We use “vertex” and “word” interchangeably, and use “edges” to mean “pairs of vertices \( \{w_i, w_j\} \) such that \( w_{ij} \neq 0 \)”. By “component of \( G \)” we mean “connected component of \( G \)”:

- By “component of \( G \)” we mean “connected component of \( G \)”:
- a subgraph in which any node can be reached from any other node (via some path consisting of positive-weight edges), and to which no additional nodes or edges from \( G \) can be added while preserving this property.

\(^6\)We assume that the rhyme scheme, which specifies which words in a stanza rhyme, is known. For our corpus, rhyme schemes were coded by hand.

\(^7\)These are also called “perfect” and “imperfect” rhymes.

\(^8\)Joyce considers the late Middle English poem *Pearl* (2222 rhymes), uses directed rather than undirected graphs, and shows several components of the rhyme graph for *Pearl*.

\(^9\)In the unweighted case, where \( n_{ij} = 0 \) or 1, \( a_i \) would be the degree of \( v_i \).
3.2. The rhyme graph \textbf{G}

Parsing the corpus gave 12387 rhymes (6350 distinct types), consisting of pairs of 4464 words. About half these words (2024) only occur in one rhyme. Types which only appear once in a corpus are often called \textit{hapax legomena} (or \textit{hapaxes}). We sanitized the data using two steps, which empirically seem to make the structure of the rhyme graphs more transparent.

1. All rhymes including hapaxes were excluded from the corpus, for a total of 10363 rhymes (4326 types) between 2440 words.

2. We removed all components of fewer than 6 words (after removing hapaxes) leaving 9388 rhymes (4059 types) between 1785 words.

A component size cutoff is motivated by practical concerns. The spectral features used below (Sec. 4) are measures of the quality of the best partition of a component into two non-trivial (\(\geq 2\) vertices), connected subgraphs. For components of size \(< 4\) there are no non-trivial partitions, so some size cutoff is needed. The choice of 6 is arbitrary. Below (Sec. 4.3), we consider the effect of varying the component size cutoff on performance.

We denote the rhyme graph corresponding to the corpus as \(G\). \(G\) has 70 connected components; we call this set \(C\). To visualize the graphs corresponding to rhyme data, we use GraphViz, an open-source graph visualization package [36, 37]. In all graphs shown here, colors indicate pronunciations of final syllable vowels, a necessary but not sufficient condition for full rhyme (for both long and short rhyme stems); and the \(\{i, j\}\) edge is labeled with \(n_{ij}\). Because the whole graph is too large to show, we illustrate what \(G\) looks like through some example components.

\textbf{Common component types.} Many components consist entirely (Fig. 1(c)) or mostly (Fig. 1(d)) of words with a single rhyme stem. When components contain more than one rhyme stem, they often consist of two or more dense clusters largely corresponding to different rhyme stems, with relatively few edges between clusters. For example, the components in Figs. 1(a)–1(b) consist of two well-defined clusters, and Fig 5(c) shows a component with 10 clusters, discussed further below. These two types of components make up much of the graph, though some are not as clean as the examples shown. In components made of several clusters, the clusters seem to correspond to words primarily sharing a single rhyme stem (and connected by full rhymes), with edges between these clusters corresponding to half rhymes. This intuition is confirmed below (Sec. 4–5) in the classification and partitioning tasks.

\textbf{Less common component types.} Two other types of components occur, though less frequently. Components such as Fig. 2(a) contain many edges corresponding to half rhymes between words with similar spellings, or \textit{spelling rhymes} [1]. Components such as Fig. 2(b) contain many edges which correspond to full rhymes if a special literary pronunciation is used for one word (usually ending in a suffix).\textsuperscript{10} For example, reading -\textit{ness} as [\text{nEs}] would make all half rhymes in Fig. 2(b) full, and reading -\textit{ity} as [\text{itaI}] would make rhymes such as \textit{sanity}:\textit{fly} full (using short rhyme stems). We call such cases \textit{poetic pronunciation conventions} (PPCs). PPCs often (as in Fig. 2(b)) result in spelling rhymes, but not always (as in \textit{sanity}:\textit{hi}).

3.3. Summary

We have introduced rhyme graphs, the corpus, and its rhyme graph \(G\). We found that most of its components consist of one or more well-defined clusters, reflecting the real pronunciation of words, while the rest reflect poetic usage. We now make practical use of the relationship between structure and pronunciation seen in most components of \(G\).

\textsuperscript{10}In most cases, these pronunciations are “literary” in the sense that they have not been used in colloquial English for centuries [8].
Figure 1: Examples of common component types from G. (a), (b) Full-rhyme clusters connected by half rhyme edges. (c) Single full rhyme cluster. (d) Full-rhyme cluster including a small percentage of outliers. In these and subsequent rhyme graph components, colors correspond to the final vowel, as listed in CELEX.
4. Classification

Some components of $G$ contain mostly words corresponding to a single rhyme stem; other components do not. In this section, we attempt to predict which group a given component falls into, using features derived from its graph structure. We first describe the feature set and a binary classification task implementing the intuitive notion of component “goodness”, then train and evaluate several classifiers for this task over the components of $G$. We find that the “goodness” of a component – corresponding to whether it contains half-rhymes – can be largely determined from graph structure alone, independent of any information about pronunciation or spelling.

4.1. Feature set

Let $G = (V, E, W)$ be a connected component of $G$, where $W$ are the unnormalized weights ($w_{ij} = n_{ij}$), with $n$ nodes ($v_1, \ldots, v_n$), and $m$ edges, corresponding to $\{v_i, v_j\}$ pairs with $w_{ij} > 0$. Define $n_{ij}$, $d_i$, $w_{ij}$, $d'_i$, and $a_i$ as above (Sec. 3.1). In addition, we need a matrix of distances, $d$, between nodes in a component. Because higher weight should correspond to smaller distance, we use

$$d_{ij} = \begin{cases} 1/w_{ij} & \text{if } w_{ij} \neq 0 \\ \infty & \text{otherwise} \end{cases}$$

We define 17 features describing the structure of $G$, of two types. Non-spectral features are properties of graphs (e.g. diameter, maximum clique size) often used in research on social networks [38, 39], and complex networks more generally [40, 41]. Spectral features are based on the eigenvalues of the Laplacian of $G$, which are intimately related to $G$’s structure [42].

4.1.1. Non-spectral features

We first define some non-spectral properties of graphs often used in network research.

Let $g$ be the matrix of geodesics using distances $d$: $g_{ij}$ is the length of the shortest path between vertices $i$ and $j$. The vertex betweenness centrality of $v \in V$ is the percentage of geodesics that include $v$.

Considering $G$ as an unweighted graph (with $\{v_i, v_j\} \in E$ if $w_{ij} > 0$), the clustering coefficient of $v_i \in V$ is the number of edges between neighbors of $v$, divided by the total number possible:

$$C(v_i) = \frac{|\{v_j, v_k \in E : v_i \sim v_j \sim v_k\}|}{|\{v_j, v_k \in E : v_i \sim v_j \sim v_k\}|}$$

\[\text{(a)}\]

\[\text{(b)}\]

Figure 2: Examples of less common component types from $G$. (a) Spelling rhymes. (b) Poetic pronunciation convention.

\[\text{(a)}\]

\[\text{(b)}\]

A larger feature set including 18 other non-spectral features was initially tried, but did not give better classification performance. The non-spectral features used here are the most predictive features from the larger set.
We define 10 non-spectral features:

- **mean/max nzd degree**: Mean/maximum of the $a_i$, divided by $n - 1$.
- **edge rat**: $\frac{m}{n(n-1)/2}$, the fraction of all possible edges present.
- **max clique size nzd**: Fraction of vertices in the largest clique of $G$.
- **max vtx bcty**: Maximum vertex betweenness centrality.
- **radius**: The minimum eccentricity of a vertex, where eccentricity($v_i$) = $\max_{v_j \in V} g_{ij}$.
- **ccoeff**: Mean clustering coefficient for vertices in $V$.
- **log size**: $\log(n)$

In addition, **log size** is z-scored across components, and **diameter** is divided by its maximum value across components. With these normalizations made, no non-spectral feature transparently depends on the absolute size of a component. This makes the non-spectral features more comparable to spectral features, which are intuitively related to the “shape” of a graph, not its size.

### 4.1.2. Spectral features

We outline the relationship between the eigenvalues of $G$ and measures of how “cuttable” $G$ is, then define features based on this connection.

**Graph Laplacians.** There are several ways to define the Laplacian of a graph, which in general yield “different but complementary information” about its structure [43]. We use three versions here.

Let $A$ be the adjacency matrix of component $G$. Let $N$ and $D'$ be the diagonal matrices with diagonal elements $a_i$ and $d'_i$, respectively. The unweighted, unnormalized Laplacian is

$$L_{00} = N - A$$

The weighted, unnormalized Laplacian is

$$L_{10} = D' - W$$

The weighted, normalized Laplacian is

$$L_{11} = D'^{-1/2}L_{10}D'^{-1/2}$$

However it is defined, the Laplacian’s eigenvalue spectrum is closely related to many properties of the underlying graph ($G$); the study of this connection is **spectral graph theory** [42]. Graph eigenvalues are essential because they figure in a variety of bounds on quantities of interest about the underlying graph, such as finding the “best” partition, most of which are provably NP-hard to compute. Our spectral features are based on several such bounds.

It can be quickly checked that $L_{00}$, $L_{10}$ and $L_{11}$ (a) are positive semi-definite, which implies their eigenvalues are real and positive (b) have smallest eigenvalue 0. Let $\lambda_{00}$, $\lambda_{10}$, and $\lambda_{11}$ be the second-smallest eigenvalues. Let $\lambda_{10}'$ be the largest eigenvalue of $L_{00}$, and denote $\mu_{00} = \frac{2\lambda_{10}'}{\lambda_{00} + \lambda_{10}}$.

These eigenvalues can be used to bound several measures of the “cuttability” of a graph. Let $E(S, \bar{S})$ be the set of edges between a subset $S \subset V$ and its complement, and define $\text{vol}(S) = \sum_{v \in S} a_i$. The **Cheeger constant** of $G$ is

$$h_G = \min_{S \subset G} \frac{|E(S, \bar{S})|}{\min(\text{vol}(S), \text{vol}(\bar{S}))}$$

(1)

Intuitively, the Cheeger constant corresponds to a bipartition of $G$ which balances two conflicting goals: make the two pieces as equal in size as possible, and separated by as few edges as possible.\(^{12}\) It can be

\(^{12}\)The Cheeger constant is also called “conductance”.

8
shown [42] that
\[ \frac{\lambda_{00}}{2} \leq h_G \leq \sqrt{1 - (1 - \lambda_{00})^2} \] (2)

where \( \delta \) is the minimum unweighted degree of a vertex (\( \min a_i \)).

For the weighted case, define \( E(S, S') \) to be the sum over weights of edges between \( S \) and \( S' \), and let \( \text{vol}(S) = \sum_{i \in S} d'_i \). The Cheeger constant is then defined as in (1). Lower [44, p. 32] and upper [45, p. 10] bounds on \( h_G \) using \( \lambda_{10} \), analogous to the unweighted case in (2), are:
\[ \frac{\lambda_{10}}{2} \leq h_G \leq \sqrt{1 - (1 - \frac{\lambda_{10}}{\delta})^2} \] (3)

where \( \delta \) is the minimum weighted degree of a vertex (\( \min d'_i \)).

Similarly, \( h_G \) can be bounded using \( \lambda_{11} \):
\[ \frac{\lambda_{11}}{2} \leq h_G \leq 2\sqrt{\lambda_{11}} \] (4)

Finally, we consider a different measure of the geometry of \( G \). For the unweighted version of \( G \) (adjacency matrix \( N \)), given a subset \( X \subset V \), define the vertex boundary \( \delta(X) \) to be the vertices not in \( X \), but adjacent to a vertex in \( X \). Then for any subset, the ratio between its “perimeter” and “area” can be bounded from below [42]:
\[ \frac{\text{vol}(\delta X)}{\text{vol}(X)} \geq \frac{1 - (1 - \mu_{00})^2}{1 + (1 - \mu_{00})^2} \] (5)

Intuitively, if the perimeter/area ratio can be lower-bounded for all \( X \), as in a circle, there is no good cut of \( G \).

Based on these relationships between a graph’s eigenvalues and its “cuttability”, we define 7 spectral features corresponding to the bounds in (2)–(5):

- cut lower bound 1: \( \frac{\lambda_{00}}{2} \)
- cut upper bound 1: \( \sqrt{1 - (1 - \frac{\lambda_{00}}{2})^2} \)
- cut lower bound 2: \( \frac{\lambda_{10}}{2} \)
- cut upper bound 2: \( \sqrt{1 - (1 - \frac{\lambda_{10}}{\delta})^2} \)
- cut lower bound 3: \( \frac{\lambda_{11}}{2} \)
- cut upper bound 3: \( 2\sqrt{\lambda_{11}} \)
- subset perim/area bound: \( \frac{1 - (1 - \mu_{00})^2}{1 + (1 - \mu_{00})^2} \)

4.2. Experiments

We now describe a binary classification task involving this feature set, as well as subsets of these features.

**Binary classification task.** For both short and long rhyme stem data, we wish to classify components of the rhyme graph as “positive” (consisting primarily of true rhymes) or “negative” (otherwise). As a measure of component goodness, we use the percentage of vertices corresponding to the most common rhyme stem, denoted Most Frequent Rhyme stem Percentage (MFRP). The pronunciation of rhyme stems was manually determined from CELEX, as discussed above (Sec. 2.2). Intuitively, if a component is thought of as made up of vertices of different colors corresponding to different rhyme stems, the MFRP is the percentage of vertices with the most common color.\(^{13}\) For example, the component in Fig. 1(b) has MFRP=52.9 (corresponding to

\(^{13}\)A node’s color in the rhyme graph components shown here corresponds to its final vowel nucleus, not its rhyme stem. However, within an individual component, there is almost always at most one rhyme stem with a given final vowel nucleus, so that each color can be thought of as representing one rhyme stem. We use colors to represent final vowels, rather than rhyme stems, because the number of different rhyme stems (>50) makes it impractical to represent each one by a visually distinct color.
Table 4: Information gain of features for short and long rhyme stem classification. (Dashed line divides spectral and non-spectral features.) Feature subsets given by Correlation-based Feature Selection are marked with asterisks. The maximally predictive features are cut lower bound 1 for short RS data and subset perim/area bound for long RS data.

<table>
<thead>
<tr>
<th>Short RS</th>
<th>Info. Gain</th>
<th>Long RS</th>
<th>Info. Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>cut upper bound 1*</td>
<td>0.57</td>
<td>subset perim/area bound*</td>
<td>0.54</td>
</tr>
<tr>
<td>cut lower bound 1*</td>
<td>0.57</td>
<td>cut lower bound 2*</td>
<td>0.53</td>
</tr>
<tr>
<td>subset perim/area bound*</td>
<td>0.54</td>
<td>cut lower bound 3*</td>
<td>0.50</td>
</tr>
<tr>
<td>cut upper bound 3</td>
<td>0.49</td>
<td>cut upper bound 3</td>
<td>0.50</td>
</tr>
<tr>
<td>cut lower bound 3*</td>
<td>0.49</td>
<td>cut lower bound 1*</td>
<td>0.46</td>
</tr>
<tr>
<td>cut lower bound 2</td>
<td>0.45</td>
<td>cut upper bound 1</td>
<td>0.46</td>
</tr>
<tr>
<td>cut upper bound 2*</td>
<td>0.42</td>
<td>cut upper bound 2</td>
<td>0.38</td>
</tr>
<tr>
<td>max nzd degree</td>
<td>0.29</td>
<td>mean nzd degree</td>
<td>0.25</td>
</tr>
<tr>
<td>mean nzd degree</td>
<td>0.26</td>
<td>edge rat</td>
<td>0.28</td>
</tr>
<tr>
<td>edge rat</td>
<td>0.26</td>
<td>max vtx bcty*</td>
<td>0.24</td>
</tr>
<tr>
<td>radius*</td>
<td>0.23</td>
<td>diameter</td>
<td>0.23</td>
</tr>
<tr>
<td>mean shortest path</td>
<td>0.22</td>
<td>mean shortest path</td>
<td>0.22</td>
</tr>
<tr>
<td>max clique size nzd</td>
<td>0.22</td>
<td>diameter</td>
<td>0.21</td>
</tr>
<tr>
<td>diameter</td>
<td>0.22</td>
<td>max clique size nzd</td>
<td>0.20</td>
</tr>
<tr>
<td>log size</td>
<td>0.19</td>
<td>max nzd degree</td>
<td>0.19</td>
</tr>
<tr>
<td>max vtx bcty*</td>
<td>0.18</td>
<td>log size</td>
<td>0.17</td>
</tr>
<tr>
<td>ccoeff*</td>
<td>0.18</td>
<td>ccoeff*</td>
<td>0.15</td>
</tr>
</tbody>
</table>

Feature Sets. Given that we have a large number of features relative to the number of components to be classified, and that many features are strongly correlated, we are in danger of overfitting if the full feature set is used. For each of long and short RS data, we define two smaller feature sets.

The first is the subset given by Correlation-based Feature Selection (CFS) [47], a standard used feature-selection strategy in which a subset is sought which balances features’ predictiveness with their relative redundancy. We used the implementation of CFS in Weka [48], an open-source machine learning package. Table 4 shows the optimal CFS subsets for long and short RS data.

The second subset used, for each of the short and long RS datasets, is simply the most predictive feature: cut lower bound 1 for short RS data and subset perim/area bound for long RS data.

Classifiers. There are 33 positive/37 negative components for long rhyme stems, and 39 positive/31 negative components for short rhyme stems. As a baseline classifier Base, we use the classifier which simply labels all components as positive.

We use three non-trivial classifiers: k-nearest neighbors, Classification and Regression Trees [49] and Support Vector Machines [50]. We used Weka’s versions of these classifiers, with the following settings:

- **KNN-5/KNN-10**: Classify by 5/10 nearest neighbors, using Euclidean distance.
- **CART**: Binary decision tree chosen using minimal cost-complexity pruning, with five-fold internal cross validation (numFoldsPruning=5), minimum of 2 observations at terminal nodes.
- **SVM**: Support vector machine trained using sequential minimal optimization [51]. All features normalized, $C = 1$ (complexity parameter), linear homogeneous kernel.
Table 5: 10-fold cross-validated accuracies (percentage correct) and F-mea-
sures for several classifiers over components of G, for short (top) and long (bottom) rhyme stems. For each rhyme stem type, CFS subset and most predictive feature given in Table 4.

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base</td>
<td>KNN-5</td>
</tr>
<tr>
<td>All features</td>
<td>55.7</td>
<td>88.6</td>
</tr>
<tr>
<td>CFS subset</td>
<td>55.7</td>
<td>87.4</td>
</tr>
<tr>
<td>Most pred. feature</td>
<td>55.7</td>
<td>87.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base</td>
<td>KNN-5</td>
</tr>
<tr>
<td>All features</td>
<td>47.1</td>
<td><strong>85.7</strong></td>
</tr>
<tr>
<td>CFS subset</td>
<td>47.1</td>
<td>82.9</td>
</tr>
<tr>
<td>Most pred. feature</td>
<td>47.1</td>
<td>85.7</td>
</tr>
</tbody>
</table>

Results. Table 5 shows classification results using 10-fold cross-validation, stratified by dataset (short vs. long RS), feature set, and classifier. Classifiers’ performances are given as accuracies (percentage of instances correctly labeled) and F-measures (harmonic mean of precision and recall).

Unsurprisingly, in all cases the non-trivial classifiers perform better than the baseline classifier. Performance using either feature subset is significantly better than for the full feature set, suggesting overfitting or badly predictive features. Performance (using non-trivial classifiers) is better for short rhyme stems than for long rhyme stems, though the difference is only statistically significant for F-measures. This could be taken to argue that the corpus’ rhymes are better described using short rhyme stems.

It is striking that across classifiers, performance measures, and rhyme stem types, performance using only one feature (the most predictive feature) is not clearly worse than performance using a subset of features given by a feature selection procedure (CFS). For both long and short rhyme stems, the most predictive feature (and the first several most predictive features generally, as seen in Table 4) is spectral. Classifying components then comes down to a single feature which can be interpreted in terms of graph structure: more cuttable components (lower \( \lambda \)) are classified as negative, while less cuttable components (higher \( \lambda \)) are classified as positive.

4.3. Sensitivity analysis

To construct the dataset used in the classification experiments, two free parameters were fixed: the threshold MFRP (\( \text{threshMFRP} \)), and the component size cutoff (CSC). Varying \( \text{threshMFRP} \) changes which components have positive labels, while changing (CSC) changes the number of components in the dataset. We now briefly check how sensitive the experimental results summarized in Table 5 are to varying these parameters, to make sure that the particular values chosen are not responsible for the good performance of our classifiers. We re-ran all experiments for the Most Predictive Feature condition, changing the dataset by varying one of \( \text{threshMFRP} \) and CSC at a time.

For both short and long rhyme stems, for all classifiers, two points are important: no classifier achieves its best performance at \( \text{threshMFRP}=0.85 \), and there is a range of values including \( \text{threshMFRP}=0.85 \) within which performance varies little.

\(^{14}\) \( p = 0.21 \) for accuracies, \( p = 0.01 \) for F-measures, Wilcoxon paired rank-sum test.

\(^{15}\) Which feature was most predictive changed as \( \text{threshMFRP} \) and CSC were changed, but was always a spectral feature.
Figure 3: 10-fold cross-validated accuracies and $F$-measures for several classifiers over components of $G$, for short and long rhyme stems, as $\text{threshMFRP}$ is varied; the component size cutoff is fixed at 6. Only the most predictive feature is used for classification.

Figure 4: 10-fold cross-validated accuracies and $F$-measures for several classifiers over components of $G$, for short and long rhyme stems, as the component size cutoff is varied; $\text{threshMFRP}$ is fixed at 0.85. Only the most predictive feature is used for classification.
As discussed above (Sec. 3.2), CSC must be at least 4 for spectral features to make sense for all components; we consider \( \text{CSC} \in [4, 10] \). Fig. 4.3 shows accuracies and \( F \)-measures for classification on data resulting from values of CSC in this range, with \( \text{threshMFRP} \) kept fixed at 0.85. For short rhyme stem experiments, accuracies for non-trivial classifiers range between 81%-90%, and \( F \)-measures range between 77%-92%. For long rhyme stem experiments, accuracies range between 61%-90%, while \( F \)-measures range between 51%-89%. The same two points hold as for the \( \text{threshMFRP} \) case: no classifier achieves its best performance at CSC=6, and there is a range of values including CSC=6 within which performance varies little.

Classification performance can change significantly as \( \text{threshMFRP} \) and CSC are varied, especially for experiments using long rhyme stems. However, performance is not optimal for the fixed values of \( \text{threshMFRP} \) and CSC used above; and for most experiments, the change in performance when these parameters are varied near their fixed values is small. We can thus conclude that the good performance obtained in Sec. 4.2 is not a result of the particular values used for \( \text{threshMFRP} \) and CSC.

4.4. Summary

We have found that spectral features are more predictive of component goodness than non-spectral features; and that although different spectral features in principle provide independent information about components, classifiers using a single spectral feature have 85-90% accuracy, significantly better than a baseline classifier, and in line with classifiers trained on an optimized subset of features. We have also shown that this performance is not an artifact of the particular values chosen for two free parameters used to construct the dataset. For both short and long rhyme stems, the single spectral feature corresponds to how “cuttable” each component is. We have thus confirmed the intuition that by and large, the bad components are those for which a good partition exists. We now see whether such good partitions can be used to increase the quality of the dataset itself.

5. Partitioning

For each connected component of \( G \), we would like to find the best partition into several pieces. The optimal number of pieces is not known beforehand, and if no good partition exists, we would like to leave \( C \) unpartitioned. This general problem, called graph partitioning or community detection, is the subject of much recent work (see [52, 53] for reviews). In this section, we apply one popular approach, modularity maximization, to the connected components of \( G \), resulting in a subgraph \( G' \subset G \). We show that by several measures for comparing graph partitions in general and rhyme graphs in particular, \( G' \) represents “better” data than \( G \), relative to the gold standard of 1-1 correspondence between rhyme stems and components.

5.1. Modularity

Many community detection algorithms attempt to find a partition which maximizes a measure of partition quality. Intuitively, given a hypothesized partition of a graph into subsets, we measure how connected the vertices in each subset are to each other (relative to the rest of the graph), versus how connected we would expect them to be by chance given no community structure, and sum over subsets. The most commonly-used formalization of this idea is modularity, a measure introduced by Newman & Girvan [54].

Consider a partition \( \mathcal{P} \) of a graph \( G = (V, E) \) (unweighted) with \( n \) vertices, \( m \) edges, and adjacency matrix \( A \). Consider a random graph \( G' = (V, E') \), in which there are \( m \) edges, vertices have the same degrees \( \{a_i\} \) as in \( E \), and the probability of an edge being placed between \( i \) and \( j \) is proportional to \( a_i a_j \). The difference between the observed and expected number of edges between \( i \) and \( j \) is then

\[
\frac{A_{ij}}{m} - \frac{a_i a_j}{2m^2}
\]

This quantity is summed over all pairs of vertexes belonging to the same community:

\[
Q(G, \mathcal{P}) = \sum_{\{i,j\}} \left( \frac{A_{ij}}{m} - \frac{a_i a_j}{2m^2} \right) \delta(P_i, P_j)
\]
where $P_i$ and $P_j$ are the communities of vertices $i$ and $j$, and $\delta$ is the Kronecker delta. $Q(G, P)$ is called the modularity of $G$ under partition $P$.

In the weighted case, given $G = (V, E, W)$ and partition $P$, let $d_i = \sum_{j \sim i} w_{ij}$ and $m' = \sum_{i,j} w_{ij}$. By analogy to (6), modularity is defined as

$$Q(G, P) = \sum_{i,j} \left( \frac{w_{ij}}{m'} - \frac{d_i d_j}{2m'^2} \right) \delta(P_i, P_j)$$

(7)

5.2. Modularity maximization

Given the weighted graph $G$ corresponding to a connected component of a rhyme graph, we would like to find the partition $P^*$ which maximizes modularity: $P^* = \arg\max_P Q(G, P)$. Let $Q^* = Q(G, C^*)$. Because the trivial partition (where $P = \{G\}$) has modularity 0, $Q^* \geq 0$. It can be shown that modularity is always less than 1 [52]. Thus, $Q^* \in [0, 1]$.

An exhaustive search for $P^*$ intuitively seems hard due to the exponential growth of possible partitions to be checked, and is in fact NP-complete [55]. However, in practice very good approximation algorithms exist for graphs of the size considered here [53]. Such algorithms find a partition $P$ approximating $P^*$, which increases modularity by $\Delta Q$. Because the trivial partition has modularity 0 and $Q^* \leq 1$,

$$0 \leq \Delta Q \leq Q^* \leq 1$$

(8)

The algorithm used here is a variant of Simulated Annealing (SA), adapted for modularity maximization by Medus et al. [56]. Here modularity acts as the “energy” (with the difference that modularity is being maximized, while energy is usually minimized), graph vertices are “particles”, transferring a vertex from one subset to another is a “transition”, and the state space is the set of possible partitions. In addition, every state (partition) includes exactly one empty subset. This does not change a given partition’s modularity (since no terms for the empty subset are part of the sum in Eqn. 7), but allows for transitions where a new subset (of one vertex) is created. (Whenever a vertex is transferred to the empty subset from a subset with at least 2 vertices, so that no empty subset remains, a new empty subset is added to the partition.) Our implementation of SA (in Matlab) is described in pseudocode in Algorithm 1.

5.3. Experiment

Let $C$ be the connected components of $G$, with $w_{ij}$ equal to the number of times the rhyme $\{v_i, v_j\}$ occurs in the corpus. For each component $C_i \in C$, we found a partition $P_i = \{C_i^1, \ldots, C_i^{n_i}\}$ maximizing $Q$ by running Algorithm 1 thirty times, with $\beta = 0.01$, $\alpha = 1.01$, $\text{lastaccept}=10000$, and $\text{maxsteps}=200000$. Removing all edges between vertices in $C_i^j$ and $C_i^k$ (for $j, k \leq n_i, j \neq k$) induces a subgraph $G' \subset G$, with connected components $C' = \bigcup_j P_j$. Figs. 5–6 show examples of partitions found for some components of $G$, discussed further below.

Results. The algorithm was successful at increasing modularity by partitioning components of $G$, perhaps overmuch: for every component $C_i \in C$, a partition with higher modularity ($Q > 0$) than the trivial partition ($Q = 0$) was found. As a result, there are $|C'| = 257$ components in $G'$, compared with $|C| = 70$ in $G$. Our concern here is the extent to which these increases in modularity improve the quality of the rhyme graph. We take the “gold standard” to be a graph where there is a 1-1 correspondence between rhyme stems and components.

Does partitioning bring the rhyme graph closer to this gold standard? We first show some examples of the partitions found for individual components of $G$, then discuss some quantitative measures of how the quality (made more precise below) of $G'$ compares to that of $G$. 

14
Algorithm 1 Find a partition $\hat{P} : V \to \mathbb{N}$ of $G$ which maximizes modularity (Eqn. 7), using simulated annealing.

Input: Weighted, connected $G = (V, E, W)$
$\beta \in (0,1)$
$\alpha > 1$
thresh, maxsteps, reps $\in \mathbb{N}$

$Q_{\text{max}} \leftarrow 0$
$S_{\text{max}} \leftarrow \{V\}$
for $i = 1 \ldots \text{reps}$ do
    $S \leftarrow$ random initial partition of $V$ consisting of between 2 and $n - 1$ subsets.
    Add an empty subset to $S$.
    $Q \leftarrow Q(W, S)$ ### from Eqn. 7
    $t \leftarrow 0$, lastaccept $\leftarrow 0$
    while $(t - \text{lastaccept}) < \text{thresh}$ and $t < \text{maxsteps}$ do
        $t \leftarrow t + 1$
        Choose random $v \in V$.
        $s(v) \leftarrow$ subset of $S$ containing $v$.
        Choose subset $\sigma \in S \setminus s(v)$.
        $S' \leftarrow S$ with $v$ moved from $s(v)$ to $\sigma$. ### proposed transition $S \rightarrow S'$
        $Q' \leftarrow Q(W, S')$
        if $Q' > Q_{\text{max}}$ then
            $Q_{\text{max}} \leftarrow Q'$ ### Keep track of maximum $Q$ partition seen
            $S_{\text{max}} \leftarrow S'$
        end if
        $q \leftarrow \min\{1, e^{-\beta(Q - Q')}\}$.
        With probability $q$, accept. ### MCMC step
        if accept then
            $S \leftarrow S'$
            $Q \leftarrow Q'$
            lastaccept $\leftarrow t$
            If $S$ contains no empty subset, add one.
        end if
    end while
end for
return $\hat{P}$ corresponding to $S_{\text{max}}$. 

15
5.4. Examples

For many components, partitioning by modularity maximization yields the desired result: a component including several rhyme stems is broken up into smaller components, each corresponding to a single rhyme stem. We give two examples.

1. The component of $G$ shown in Fig. 5(a) corresponds to the three components of $G'$ shown in Fig. 5(b), with an increase in modularity of $\Delta Q = 0.38$. (Recall that $\Delta Q \in [0,1]$, by Eqn. 8.) The partition is also an improvement relative to the gold standard: three distinct rhyme stems ([$um$], [om], [aum]) are mixed in a single component in $G$, but correspond to distinct components in $G'$.

2. The component of $G$ (173 vertices) shown in Fig. 5(c) corresponds to the 10 components of $G'$ shown in Fig. 5(d), with an increase in modularity of $\Delta Q = 0.77$. The partition is a striking improvement relative to the gold standard. Let $C$ denote the original component of $G$ and $P$ the set of components of $G'$. $C$ contains 14 distinct rhyme stems ([$eil$], [em], [in], [en], [en], [en], [en], [en], [en], [en], [en], [en], [en], [en]). With small exceptions, each of the rhyme stems conflated in $C$ corresponds to a single component in $P$. Leaving aside the 2 rhyme stems corresponding to only one vertex each ([en], [en]), and 2 misclassified words (overygrown, again), the main errors are that $P$ splits [n] between two components, and puts [an] and [on] vertices in a single component.\footnote{We note that these are the (non-trivial) rhyme stems corresponding to the smallest numbers of vertices.}

Recall that modularity maximization found non-trivial partitions for all components of $G$, not just those known \textit{a priori} to contain several rhyme stems. Thus, for some components, partitioning can have a negative effect: a single positive component of $G$ corresponds to multiple positive components of $G'$, as in a third example:

3. The component of $G$ shown in Fig. 6(a) corresponds to the three components of $G'$ shown in Fig. 6(b), with an increase in modularity of $\Delta Q = 0.07$. The effect of this partition is negative relative to the gold standard: one rhyme stem corresponds to a single component in $G$, but two components in $G'$.

For nearly all components, partitioning has one of these two effects: a component containing several rhyme stems is broken up into several components corresponding to unique rhyme stems, or a component already corresponding to a unique rhyme stem is broken up. The first kind of partition brings the rhyme graph closer to the gold standard; the second takes the rhyme graph farther from the gold standard. Importantly, a third possible kind of partition – an improvement in modularity, but a negative effect relative to the gold standard – is rarely observed.\footnote{For example, in a component with $\text{MFRP} = \text{threshMFRP}$, many possible partitions would give one new bad component ($\text{MFRP} < \text{threshMFRP}$) and one new good component ($\text{MFRP} > \text{threshMFRP}$).} Intuitively, if the effects of the first kind of partition outweigh the effects of the second kind, we expect $G'$ to have higher overall quality than $G$. We now give quantitative evidence that this is the case.

5.5. Measuring the quality of $G'$ vs. $G$

There are many different ways the quality of a rhyme graph could be measured; we use several here. We first consider how close each of $G$ and $G'$ are to the gold standard, using three similarity measures from the literature for comparing clusterings of arbitrary sets. We then compare $G$ and $G'$ using several more intuitive measures for the particular case of rhyme graphs. For a given measure, the value for $G$ measures the quality of the rhyming corpus itself; comparing to the value for $G'$ measures how quality is improved by partitioning. By all measures, we find that $G'$ improves significantly on $G$.\footnote{We note that these are the (non-trivial) rhyme stems corresponding to the smallest numbers of vertices.}
Figure 5: Examples of a component of $G$ corresponding to (a,b) 2 post-partitioning components of $G'$. Dotted edges are not present in $G'$. Edge weights not shown.
5.5.1. General measures of similarity between clusterings

Consider a set $S = \{s_1, \ldots, s_N\}$ of $N$ data points, and let $U = \{U_1, \ldots, U_R\}$ and $V = \{V_1, \ldots, V_C\}$ be two partitions of $S$, into $R$ and $C$ clusters, respectively: each $s \in S$ is contained in exactly one of the $U_i$ and exactly one of the $V_j$. Let $u(s)$ and $v(s)$ be the indices of the clusters of $U$ and $V$ containing $s$: $s \in U_{u(s)}$, $s \in V_{v(s)}$. We consider two popular measures of similarity between partitions, the adjusted Rand index and the normalized mutual information, as well as a recently proposed measure, the adjusted mutual information.

In each case, we compare the similarity between $G$ and the gold standard to the similarity between $G'$ and the gold standard.

**Adjusted Rand Index.** Let $N_{11}$ be the number of pairs of points in $S$ which are in the same cluster in $U$ and the same cluster in $V$:

$$N_{11} = |\{s, t \in S : u(s) = u(t), \quad v(s) = v(t)\}|$$

Define $N_{00}$ as the number of pairs which are in different clusters in $U$ and different clusters in $V$, $N_{01}$ as the number of pairs in different clusters in $U$ but the same cluster in $V$, and $N_{10}$ as the number of pairs in the same cluster in $U$ but different clusters in $V$. $N_{00} + N_{11}$ is then the number of pairs on which $U$ and $V$ agree, and $N_{01} + N_{10}$ is the number of pairs on which they disagree. The Rand index [57] is the fraction of pairs on which $U$ and $V$ agree:

$$RI = \frac{N_{00} + N_{11}}{N_{00} + N_{11} + N_{01} + N_{10}}$$

However, the RI does not account for the fact that two random partitions will agree on many pairs. When comparing two clusterings, we would like to know whether they agree on more or fewer pairs than would be expected by chance.

Hubert and Arabie [58] proposed adjusting the Rand index for chance, as follows. Assume that $N$ (the number of points) is fixed, and consider all pairs of partitions such that the number and size of clusters in the first and second partitions are the same as in $U$ and $V$, respectively. Let $Expected$ and $Max$ be the expected and maximum value of $RI$ over all such partitions. The **adjusted Rand index (ARI)** is then:

$$ARI = \frac{RI - Expected}{Max - Expected}$$

The ARI is 0 when $U$ and $V$ agree on no more pairs than expected by chance, and is bounded above by 1.

In the case of rhyme graphs, $S$ is the set of nodes (words), and $U$ is the partitioning corresponding to the gold standard: each $U_i$ corresponds to a single rhyme stem, and two words $w_1, w_2 \in S$ have $u(w_1) = u(w_2)$ if

---

18 $Max$ and $Expected$ can be written down exactly in terms of entries of the contingency table for $U$ and $V$. We do not do so here, for brevity.


their rhyme stems are identical. \( V \) is the partition corresponding to the rhyme graph \( G \) (before partitioning) or \( G' \) (after partitioning): each \( V_j \) corresponds to a component, and \( v(w_1) = v(w_2) \) if \( w_1 \) and \( w_2 \) are in the same component. For both long and short RS, we compute the ARI for each of \( G \) and \( G' \) relative to the gold standard; these are shown in Table 6. For both types of rhyme stem, ARI > 0 before partitioning, and partitioning substantially increases ARI: \( G \) is closer to the gold standard than would be expected by chance, but \( G' \) is much closer to the gold standard than \( G \).

Normalized Mutual Information, Adjusted Mutual Information. To define the mutual information of two partitions of a set, we must first state them in terms of random variables. Let \( P(i) \) denote the probability that a point \( s \in S \) has \( u(s) = i \), \( P'(j) \) the probability that \( v(s) = j \), and \( P''(i, j) \) the probability that \( u(s) = i \) and \( v(s) = j \):

\[
P(i) = \frac{|U_i|}{N}, \quad P'(j) = \frac{|V_j|}{N}, \quad P''(i, j) = \frac{|U_i \cap V_j|}{N}
\]

The entropy and mutual information (MI) of partitions \( U \) and \( V \) are then defined via these random variables:

\[
H(U) = -\sum_{i=1}^{R} P(i) \log P(i), \quad H(V) = -\sum_{j=1}^{S} P'(j) \log P'(j)
\]

\[
I(U, V) = \sum_{i=1}^{R} \sum_{j=1}^{S} P''(i, j) \log \left( \frac{P''(i, j)}{P(i)P'(j)} \right)
\]

MI is a symmetric measure of how predictive two random variables are of each other. In the case of clustering, MI quantifies how similarly two partitions cluster the same set of data points. MI is non-negative, and is upper bounded by \( \min\{H(U), H(V)\} \). Based on these observations, Strehl and Ghosh [59] proposed the normalized mutual information (NMI) as a measure for comparing partitions:

\[
NMI(U, V) = \frac{I(U, V)}{\sqrt{H(U)H(V)}}
\]

NMI is bounded by 0 and 1. Unlike ARI, NMI is not adjusted for chance. A version of MI adjusted for chance, the adjusted mutual information (AMI), has recently been proposed by Vinh et al. [60]. Adjustment for chance is done similarly as for ARI, and we do not give details here. Like ARI, AMI is bounded above by 1, and is 0 when \( U \) and \( V \) agree on no more pairs than expected by chance.

For our case of rhyme graphs, we consider both NMI and AMI. \( U \) and \( V \) are defined as above (for ARI). For both long and short rhyme stems, we compare \( G \) and \( G' \) by computing NMI and AMI for each, relative to the gold standard; these are shown in Table 6. For both types of rhyme stem, partitioning increases NMI. AMI shows similar patterns to ARI: \( G \) is closer to the gold standard than would be expected by chance, but \( G' \) is significantly closer to the gold standard than \( G \).

5.5.2. Intuitive measures of rhyme graph quality

The general similarity measures just considered tell us that \( G' \) is closer to the gold standard than \( G \), but not how it is closer. To better understand how \( G \) compares with \( G' \), we consider several more intuitive measures of quality for the case of rhyme graphs.

Component MFRP. Recall that above (Sec. 4.2), we divided components of \( G \) into positive and negative classes, based on whether MFRP was above or below a threshold value. One measure of a graph’s quality is how large the positive class is: the percentage of components with MFRP > threshMFRP. If we wish to weight components by their sizes, we could consider the percentage of vertices, edges (adjacent vertices) or rhymes (weights on adjacent vertices) lying in components with MFRP > threshMFRP.

Rows 4–7 of Table 6 give these four measures for \( G \) and \( G' \), for short and long rhyme stems. \( G' \) improves substantially (21–46%) on \( G \) in each case. Although \( G \) had low scores to begin with, the dramatic increases
Table 6: Measures of rhyme graph quality for $G$ and $G'$, short and long rhyme stems.

<table>
<thead>
<tr>
<th>Quality measure</th>
<th>Short RS</th>
<th>Long RS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adjusted Rand Index vs. gold standard</td>
<td>0.248</td>
<td>0.188</td>
</tr>
<tr>
<td>Normalized Mutual Information</td>
<td>0.571</td>
<td>0.556</td>
</tr>
<tr>
<td>Adjusted Mutual Information</td>
<td>0.584</td>
<td>0.491</td>
</tr>
<tr>
<td>% CCs with MFRP $&gt; 0.85$</td>
<td>55.7</td>
<td>47.1</td>
</tr>
<tr>
<td>% vertices</td>
<td>25.1</td>
<td>26.2</td>
</tr>
<tr>
<td>% neurons in CCs with MFRP $&gt; 0.85$</td>
<td>22.8</td>
<td>18.4</td>
</tr>
<tr>
<td>% rhymes</td>
<td>24.0</td>
<td>18.4</td>
</tr>
<tr>
<td>% edges with identical RS (all CCs)</td>
<td>81.4</td>
<td>76.3</td>
</tr>
<tr>
<td>% rhymes</td>
<td>87.3</td>
<td>84.3</td>
</tr>
</tbody>
</table>

Figure 7: Histogram of most frequent rhyme percentage (MFRP) for components of the unpartitioned (left) and partitioned (right) graphs, for short (top) and long (bottom) rhyme stems.

seen confirm the intuition that partitioning was largely successful in decreasing the number of components, especially large components, containing several rhyme stems.

For a more direct look at the effect of partitioning on MFRP, we can consider how its distribution (across all components) changed from $G$ to $G'$. Fig. 7 shows histograms of MFRP for components of $G$ and $G'$, for short and long rhyme stems. It is visually clear that the distributions for $G'$ (partitioned) are much more concentrated near 1 (same rhyme stem for all vertices of a component) than the distributions for $G$ (unpartitioned). In particular, components with $G' < 0.5$, like the example discussed above in Fig. 5(c), are almost completely gone.

Percentage identical rhyme stems. Instead of measuring the size of positive components in $G$ and $G'$, we could consider a more basic quantity, without reference to components: the type and token percentage of full rhymes. Rows 8–9 of Table 6 show the percentage of edges between vertices with identical rhyme stems, and the analogous percentage of rhymes (again for $G$, $G'$, and short and long rhyme stems). $G'$ again improves on $G$ in all cases. Although the gains are much more modest (2–4%) than for whole components (above), they are important; they indicate that partitioning removed many more edges corresponding to half rhymes than corresponding to full rhymes, supporting the intuition that many components of $G$ are
Component cuttability. In the classification task above, when the type of rhyme stem was fixed, we found that the distinction between positive and negative components could be boiled down to a single spectral feature, representing the quality of the best bipartition of a component: its “cuttability”.

For short rhyme stems, this feature was cut lower bound \( \lambda_{11} \); for long rhyme stems it was subset perim/area bound. As a measure of how the cuttability of components was changed by partitioning, we can look at the distribution of these features (across components) for \( G \) and \( G' \), shown as histograms in Figs. 8–9.

Recall that lower \( \lambda_{11} \) and higher subset perim/area bound correspond to higher cuttability. \( \lambda_{11} \) has mean 0.22 for components of \( G \) and mean 0.93 for components of \( G' \); further, the distribution of \( \lambda_{11} \) is much less concentrated near 0 in \( G' \) than in \( G \). subset perim/area bound has mean 1.23 for components of \( G \) and mean 2.06 for components of \( G' \); also, its distribution is skewed right for \( G \) (skewness=0.84) and skewed left for \( G' \) (skewness=-0.18). Overall, the distributions of \( \lambda_{11} \) and subset perim/area bound for \( G' \) versus \( G \) reflect that components of \( G' \) are much less cuttable.

5.6. Summary

After partitioning \( G \) via modularity maximization to give \( G' \), we found that by several measures, \( G' \) is closer than \( G \) to the gold standard, where there is a 1-1 correspondence between rhyme stems and components. This improvement is much more marked at the level of components (percentage positive components) than at the level of individual rhymes (percentage full rhymes).
Figure 10: Example of a “merger”: a component from $G_{1600}$ classified (using short rhyme stems) as negative ($\lambda_{11} < 0.12$), with $\text{MFRP} > \text{mfrpThresh}$ according to modern pronunciation (a) but $\text{MFRP} < \text{mfrpThresh}$ according to 1600 pronunciations (b). Edge weights not shown.

6. Discussion

6.1. Future work

The methods used here are initial attempts to make use of rhyme graphs, and should be refined in future work. Allowing multiple pronunciations for a single spelling and increasing the size of the corpus should increase the quality of the data; relaxing the sanitizing steps (considering only components above a certain threshold size, excluding hapaxes) would test the robustness of our results. Different measures of components quality besides $\text{MFRP}$ should be explored, as should the effect of replacing the binary classification task with a regression task (where $\text{MFRP}$ is predicted from features).

Mergers and splits. Improvements are necessary for our long-term goal, to use the connection shown here between pronunciation and rhyme graph structure for historical inference. Suppose we had a near-perfect classifier for the “goodness” of rhyme graph components. This classifier could be applied to the graph of a historical rhyming corpus, say from poetry written around 1600. Using positive/negative labels from current pronunciations, we expect the classifier to make many more “errors” than for a graph corresponding to present-day data; these errors would indicate components where one of two types of pronunciation change has occurred.

1. Merger: Positively labeled component classified as negative; corresponds to words whose rhyme stems were different in 1600, but are the same today.

2. Split: Negatively-labeled component classified as positive; corresponds to words whose rhyme stems were the same in 1600, but are different today.

The trouble is that for the moment, we do not have a highly-accurate classifier. Even with 90% accuracy, we cannot distinguish a priori between vanilla classifier errors and errors which indicate pronunciation changes.

Nonetheless, we are encouraged by some preliminary work in this direction. We constructed a corpus of poetry written around 1600, of similar size to $G$, whose graph we denote as $G_{1600}$; classified its components using the most predictive feature ($\lambda_{11}$) of $G$ (for short rhyme stems); and used rhyme stems from CELEX. It is indeed the case that the classifier makes many more “errors” than on $G$, and some of these errors correspond to independently-known mergers and splits.

Fig. 11 shows an example split, a component of words ending (orthographically) in -ence. This suffix corresponds to two rhyme stems ([ms] and [ns]) in today’s pronunciation (Fig. 10(a)), but a single short
Figure 11: Example of a “split”: a component from $G_{1600}$ classified (using short rhyme stems) as positive ($\lambda_{11} > 0.12$), with MFRP < mfrpThresh according to modern pronunciation (a) but MFRP > mfrpThresh according to 1600 pronunciations (b). Edge weights not shown.

rhyme stem ([ens]) in 1600 (Fig. 10(b)). Fig. 11 shows an example merger, a component of words with two rhyme stems ([ail], [aill]) in 1600, which have merged to a single RS ([eil]) today. Both examples reflect larger sound changes in English: unstressed vowels have often reduced (to [ə] or [i]) word-finally, and the vowels pronounced in Early Modern English as [a] and [ai] merged to [eil] by 1800 [2].

Other languages and poetic traditions. All methods used in this paper would be straightforward to extend to rhyming corpora from other languages, provided that a pronunciation dictionary exists, and that the definition of the rhyme stem is changed appropriately. Indeed, it is important to check in future work whether the salient aspects of the English rhyme graphs considered here hold for other languages. If rhyme graphs do not show some sort of similar structure cross-linguistically, they cannot be used for pronunciation reconstruction in the most interesting cases, where the historical pronunciation of a language is unknown.

The methods used in this paper are also applicable to data from other poetic traditions. Rhyming in Modern English poetry requires that pairs of words be similar in a particular way near their endings. Different poetic traditions require that sets of words be similar, but define similarity very differently. In alliterative verse, pairs of words must begin with the same phonemes; this is the dominant structuring device in most verse written in Old English (such as Beowulf) and Old Norse (the ancestor of modern-day Icelandic) [61, 62]. In Welsh poetry, different types of cynghanedd (“harmony”) require various complex patterns of consonantal correspondence and rhyming among words within individual lines [63, 64]. In principle, the methods used in this paper could be extended to data from any poetic tradition, like alliterative verse or cynghanedd, where some sort of similarity between the pronunciation of sets of words is implied by poetic form.

Full rhyme to half rhyme ratio. For the poetic corpus considered here – English rhyming verse written around 1900 – we found that the rhyme graph largely consists of full-rhyming clusters, connected by half-rhyming edges. A natural extension would be to check how robust this finding is for rhyming corpora where the ratio of half rhymes to full rhymes is greater. In general, verse from various genres and dates will differ in how common half rhymes are relative to full rhymes. For example, half rhymes seem to be more frequent in (contemporary, English-language) song lyrics than in rhyming poetry: in Katz’ English hip-hop corpus

---

19See Footnote 2.
[12], 56% of rhymes have identical long rhyme stems, compared to 84% in our corpus. Half rhymes also may be more common in translations of rhyming verse into English, where faithfulness to the rhyme scheme may require that the translator use more half rhymes.

6.2. Summary

In Sec. 2–3, we introduced a corpus of rhymes from recent poetry, and explored its rhyme graph, $G$. We found most components of $G$ either consist of a densely connected set of vertices (with edges corresponding to full rhymes), or several such sets, with few edges between sets (corresponding to half rhymes); relatively few components correspond to spelling rhymes or poetic pronunciation conventions. In other words, graph structure for the most part transparently reflects actual pronunciation. This is not a trivial fact – it could have been the case that half rhymes occur frequently enough to obscure the distinction between half and full rhymes, or that spelling rhymes or poetic pronunciation conventions are widespread. That structure reflects pronunciation in poetry means it is (in principle) possible to “read off" pronunciation from structure, as discussed above.

In Sec. 4, we found that spectral features are much more predictive of component “goodness" than non-spectral features. Though it possible that a different set of non-spectral features would perform better, it is striking that for both short and long rhyme stems, no non-spectral feature is more predictive than any spectral feature. We tentatively conclude that a component’s eigenvalue spectrum is more predictive of its “goodness" (i.e. class label) than the non-spectral measures often used in network research. Overall, we confirmed the intuition that component goodness corresponds, for the most part, to whether a good partition exists.

In Sec. 5, we found that applying modularity-based partitioning to components of $G$, resulting in a new graph $G'$, significantly improves the quality of the data, especially when seen from the perspective of components, rather than individual rhymes. For the short RS case, for example, 79% of components in $G'$ are positive, corresponding to 72% of words, compared to 56%/25% for $G$. For the long-term goal of using rhyme graphs for pronunciation reconstruction, this is our most important finding; by partitioning components, we go from 50/50 positive/negative components to 80/20. Where a random component of $G$ contains many half rhymes at chance, a random component of $G'$ probably does not.

Overall, we have shown three cases where it is possible and profitable to understand groups of rhymes in terms of their corresponding rhyme graphs. We can roughly summarize our findings by three correspondences between a given group of rhymes $R$, corresponding to a connected component $G(R)$ of rhyme graph $G$:

<table>
<thead>
<tr>
<th>Group of rhymes</th>
<th>Component of rhyme graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Most rhymes in $R$ are full, fewer are half.</td>
<td>$G(R)$ has community structure.</td>
</tr>
<tr>
<td>$R$ contains half-rhymes.</td>
<td>$G(R)$ has a good partition.</td>
</tr>
<tr>
<td>Which groups of rhymes in $R$ are definitely full?</td>
<td>What is the best partition of $G(R)$?</td>
</tr>
</tbody>
</table>

We thus add to the recent body of work illustrating that in very different settings (e.g. [65, 66, 67, 68, 69, 70, 71, 72]; [73] gives a bibliography), considering linguistic data as graphs (or networks) gives new insights into how language is structured and used. Specifically, like [74, 75, 76], we found a strong and striking association between graph spectra and linguistic properties.

Acknowledgments

We thank Max Bane, Joshua Grochow, Ross Girshick, Sravana Reddy, and Alan Yu for insightful discussion and suggestions, as well as three anonymous reviewers for helpful comments. This work was supported in part by a Department of Education GAANN Fellowship.

References

San Francisco, 2005.


