MISHEARD ME ORONYMINATOR: USING ORONYMS TO VALIDATE THE CORRECTNESS OF FREQUENCY DICTIONARIES

A Thesis

Presented to

the Faculty of California Polytechnic State University

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

by

Jennifer "Jenee" Gayle Hughes

June 2012

COMMITTEE MEMBERSHIP

TITLE:	Misheard Me Oronyminator: Using Oronyms to Validate the Correctness of Frequency Dictionaries
AUTHOR:	Jennifer "Jenee" Gayle Hughes
DATE SUBMITTED:	June 2012
COMMITTEE CHAIR:	Zoë Wood, Ph.D.
COMMITTEE MEMBER:	Franz Kurfess, Ph.D.
COMMITTEE MEMBER:	John Clements, Ph.D.

Abstract

Misheard Me Oronyminator: Using Oronyms to Validate the Correctness of Frequency Dictionaries

Jennifer "Jenee" Gayle Hughes

In the field of speech recognition, an algorithm must learn to tell the difference between "a nice rock" and "a gneiss rock". These identical-sounding phrases are called oronyms. Word frequency dictionaries are often used by speech recognition systems to help resolve phonetic sequences with more than one possible orthographic phrase interpretation, by looking up which oronym of the root phonetic sequence contains the most common words.

Our paper demonstrates a technique used to validate word frequency dictionary values.We chose to use frequency values from the UNISYN dictionary, which uses tallies each word on a per-occurance basis in a propriatary text corpus.

In the first phase of our user study, we generated oronym strings for the phrase *"a nice cold hour"*, and had over a dozen people make 62 recordings of the most common oronyms for that phrase. In the second phase, we selected 15 of the phase one recordings, and had ∼74 different people transcribe each one, for a total of 953 transcriptions overall.

If the frequency dictionary values for our test phrases accurately reflected the real-world expectations of actual listeners, we would expect that the most commonly transcribed phrases in our user study would roughly correspond with our metric for the most likely oronym interpretation of the root phrase.

During the course of our study, we found that using per-occurance frequency values when computing our overall-phrase-frequency metric caused the end result

to be thrown off by excessively common words, such as “the”, “is”, and “a”. These super-common words had such high per-occurance tallies that they overpowered any effect that any regular word had on a frequency metric. However, when we tally on a document-count basis, instead of a by-occurance basis, we found that this effect was mitigated.

Contents

List of Tables

List of Figures

Chapter 1

Preliminary Vocabulary

Before we start, there are a few uncommon terms we will use fairly often in this paper. We have briefly defined them here.

1.1 Mondegreens

A mondegreen is a word or phrase resulting from a misinterpretation of a word or phrase that has been heard[11]. The word was coined by American author Sylvia Wright in her article, "The Death of Lady Mondegreen", published in a 1954 issue of Harper's Bazaar. In it, she describes the origin of the word:

> When I was a child, my mother used to read aloud to me from Percy's Reliques, and one of my favorite poems began, as I remember:
>
> > Ye Highlands and ye Lowlands,
> > Oh, where hae ye been?
> > They hae slain the Earl O' Moray,
> > And Lady *Mondegreen.*

The fourth line of the quote is actually "and laid him on the green"[31].

Additional commonly-cited mondegreens include[7][26][28]:

Gladly the Cross-Eyed Bear	Gladly the Cross I'd Bear
Scuse me while I kiss this guy	Scuse me while I kiss the sky
There's a bathroom on the right	There's a bad moon on the rise

1.2 Oronyms

Oronyms are phrases that may differ in meaning or spelling, but sound near-identical when spoken. They are similar to mondegreens, and the terms are often used interchangeably. The difference, however, lies in the context. The label "mondegreen" is used more often in regards to music lyrics, where pronunciation can be affected by the addition of music and tone to the phrase. Oronyms, on the other hand, refer to spoken words, not sung lyrics.[13]

Common oronyms include:

i scream ↔ ice cream

an ice cold hour ↔ a nice cold hour

grape ants ↔ gray pants

real eyes ↔ realize

1.3 Orthography

The word 'orthographic' comes from the Latin *orthographia*, meaning *correct writing*. Orthography itself is the part of language study concerned with letters and spelling. More specifically, its the standardized system of writing down words in a specific language, using a commonly-accepted set of letters according to accepted usage. [14]

The orthographic symbol set for a language is the commonly-accepted set of letters used to spell words in that language. In English, our orthographic symbol set is the Latin alphabet.

In this paper, "orthographic phrase", refers to a sequence of regularly-spelled words found in an English dictionary.

Example: "This is a orthographic phrase."

1.4 Corpus

The word corpus is Latin, and means *body*. In general, it's helpful to think of a text corpus as a "body of text" with some special constraints.

In linguistics, the term "corpora" refers to samples from various textual sources.

A "text corpus" refers to a large, structured body of text, consisting of those corpora (a.k.a. samples from various textual sources).

In order for a text corpus to be useful, it must be a representative subset of the larger language it wishes to represent.

To put together a general text corpus for the English language, one should pull from many sources: books, newspapers, movie scripts, magazines, academic literature, etc.

If any single genre is over-represented in the corpora (text samples), then the resultant text corpus can be biased, and not useful for general purposes. For example: if one pulls all their corpora (text samples) from Wikipedia, the resulting text corpus is likely to underrepresent most first-person and second-

person nouns and verbs, since those are verboten in Wikipedia articles.

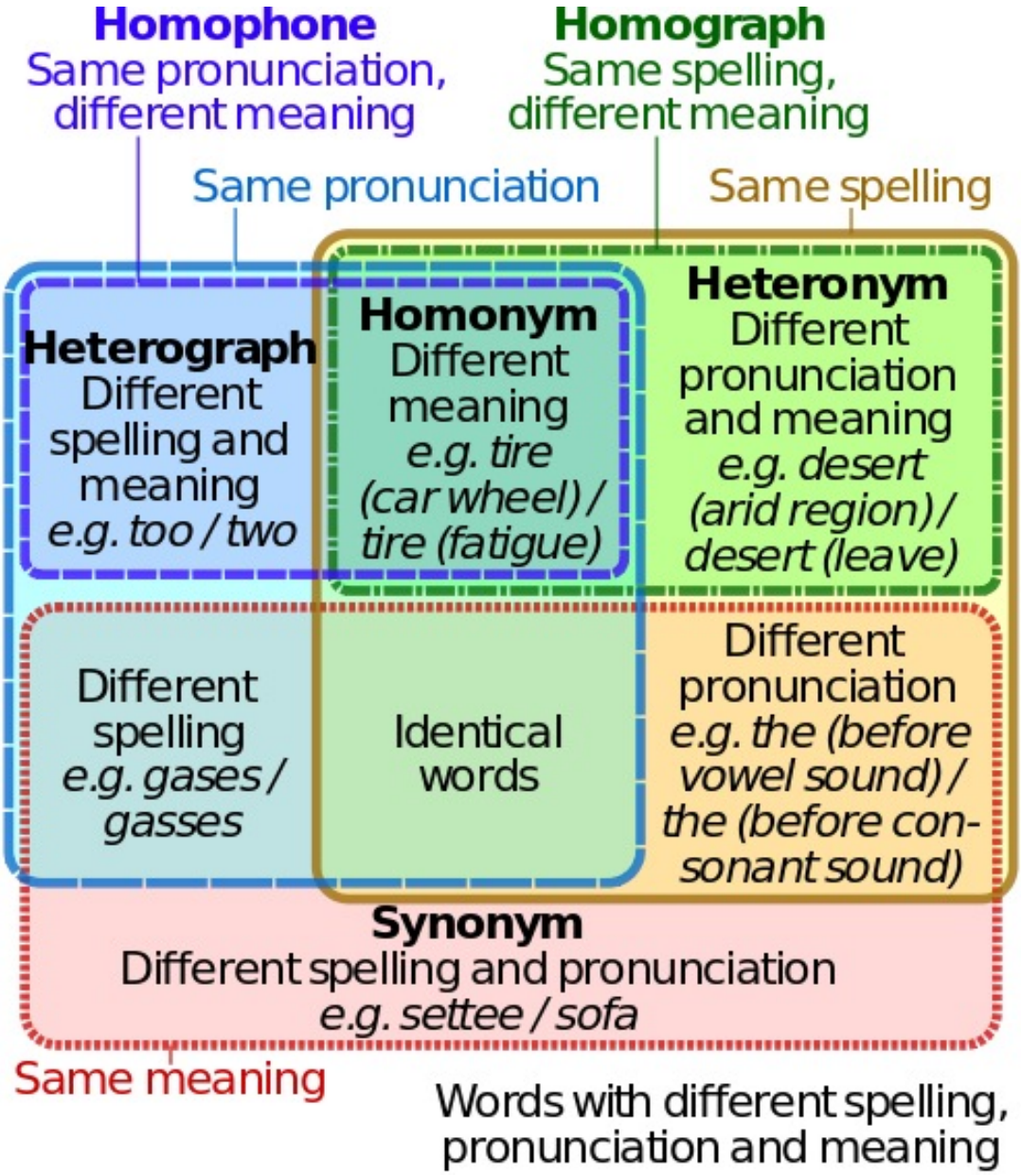
1.4.1 Uses of Text Corpora

A text corpus is generally used as the "control" set for comparing to data from linguistic experiments.

Given a good text corpus, word frequency can be generated simply by counting the number of occurrences of every word that appears in the corpus. This frequency data can be used by other applications, like Melody Matcher, to weight the possibility of resolving homophones[4], by calculating which words have a higher frequency count in the corpus. The higher the frequency, the are more common a word is, and the more likely it is to be heard.

In addition, given a text corpus, one can generate a dictionary of all words in the corpus. This dictionary can then be annotated with data such as part of speech, unique identifiers for homographs, and phonetic spelling.

1.5 Word Categorizations



1.5.1 Homographs

Homographs are orthographic words that are spelled identically. Typically, homographs are also pronounced differently, which makes them homographic het-

erophones. For example, there are two different pronunciations of "does" : the "multiple female deer" does (doze), and the "third-person singular present indicative form of 'do"' does (duhz). If the words are spelled identically AND pronounced identically, then then are homographic homophones, which are commonly known as homonyms.

1.5.2 Heterographs

Heterographs are orthographic words that are spelled differently. Typically, words are only called heterographs if they are also pronounced differently, making them heterographic homophones. The word "does" is a heterographic homophone with two different pronunciations: the "multiple female deer" does (doze), and the "third-person singular present indicative form of 'do"' does (duhz). (Most words in the English language are heterographic heterophones; that is, spelled and pronounced uniquely. Because this is the default state of a word set, we rarely describe such words in terms of homo/hetero phones/graphs. As such, the only time a word set is likely to be described as heterographic is if it is also a homophone.

1.5.3 Homophones

Homophones are orthographic words that are pronounced identically, but typically spelled differently. For example, the banes of every grammarian, "there", "their", and "they're", are heterographic homophones. Alternatively, "to", "too", and "two" are also heterographic homophones. If a homophone set is also homographic (that is, spelled identically as well as pronounced identically), then we refer to them as homonyms. As such, the only time the words in a set are likely

to be labelled "homophones" is if they are also heterographic.

1.5.4 Homonyms

Homonyms are orthographic words that are pronounced and spelled identically, but defined differently. For example, depending on context, the word "left" can mean left (the opposite of right), or left (the past tense of leave). Homonyms can also be referred to as homographic homophones.

1.6 Phonetics and Phonology

To discover oronyms for a phrase, we must first to translate the root orthographic phrase to a representation that allows us to unambiguously measure pronunciation. Phonology and phonetics are branches of linguistics that deal with pronunciation.

1.6.1 Phonetics

Phonetics is a branch of *descriptive* linguistics, and refers to the study of the actual, uttered sound of human speech. It deals with describing the physical phenomena of how these sounds are produced from the vocal tract, how they are transmitted once spoken, and how they are recieved by audiences. The building blocks of phonetics are *phones*, which represent atomic sounds.

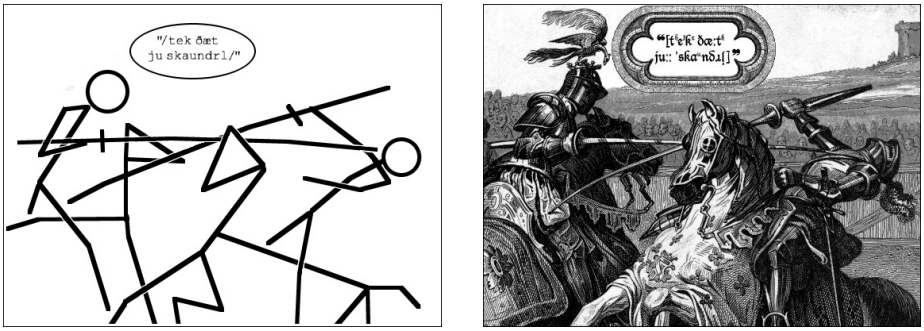
1.6.2 Phonology (aka phonemics)

Phonology is a branch of *theoretical* linguistics, and as such, is primarily concered with the abstract grammartical characterization of sounds. It describes the way that sounds function within a language and give meaning to words. The basis of phonological analysis is the grouping of sounds (*phones*) into distinct units within a languages. These distinct units are called *phonemes*.

These phonemes may contain different phones, depending on the accent of the speaker. For example, native speakers of General American English only generally recognize one 'L' sound phoneme. However, there are two different ways that that phoneme manifests itself: the 'l' in "male", and the 'l' in late. This difference is not noticable to a native speaker of American English, because that particular accent will parse any 'L' phone as the same 'L' phoneme.

1.6.3 Phonetics Vs Phonology

As we said previously, though the terms are sometimes used interchangably, the words 'phonemic' and 'phonetic' (and their corresponding sound building blocks, 'phone' and 'phoneme') indicate a different stages of sound parsing. *Phonemes* are idealized sounds; *phones* are the actual sounds that come out of a person's mouth. Figure 1.1 provides a final, illustrative metaphor of the difference.



(a) Phonology[8] (b) Phonetics[9]

Figure 1.1: The difference between phonetics and phonology

1.7 Phonemic/Phonetic Alphabets

As we stated in section 1.6.2, phonemes are the atomic building blocks of words. In a phonemic alphabet, every meaningful sound has its own "letter". The way that we interact with phonemes in a concrete way is by using phonetic alphabets and phonetic dictionaries.

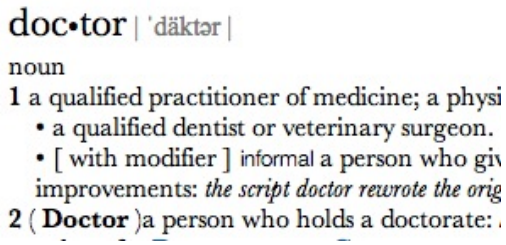

doc•tor | ˈdäktər |
noun
1 a qualified practitioner of medicine; a physi
• a qualified dentist or veterinary surgeon.
• [with modifier] informal a person who giv
improvements: *the script doctor rewrote the orig*
2 (**Doctor**)a person who holds a doctorate:

Figure 1.2: The characters to the right of the large bold word "doctor" are IPA symbols.

The most common phonetic alphabet is the IPA (International Phonetic Alphabet). It contains representations of every sound in every known language globally, and allows for cross-cultural pronunciation guidelines. As shown in figure 1.2, IPA representations of orthographic words are found in traditional dictionaries to aid pronunciation.

1.7.1 SAMPA

SAMPA (Speech Assessment Methods Phonetic Alphabet) is a computer-readable phonetic alphabet, based upon the symbols found in the more-standard-but-not-easily-computer-readable IPA (International Phonetic Alphabet). It uses "letters" consisting of 1-2 ASCII characters to represent each phoneme. The ASCII sequences corresponding to each of the SAMPA letters are designed so that any SAMPA sequence is deterministically parsible.

We chose to use SAMPA instead of IPA because its ASCII-compliance makes it easy to integrate into other systems.

See table **??** for a full table of each SAMPA phoneme, its description, and its sub-parts.

For some brief context, the SAMPA spelling of the name 'Jenee Hughes' is *dZEni hjuz*. 'Dr Zoe Wood' becomes *dAkt@`r zoui wUd*. 'Dr John Clements' becomes *dAkt@`r dZAn klEm@nts*. 'Dr Franz Kurfess' becomes *dAk@`r fr{nz k3`rfEs*.

Chapter 2

Introduction

Human brains are built to come to single conclusions about things that have more than one interpretation. That single conclusion is developed based upon your experiences, cutural immersion, and language familiarity [29]. When attempting to write English phrases that will be read aloud and heard by people different other linguistic biases, it is important to make your prose as deterministically understandable as possible. The first step towards this is understanding and identifying how many ways a particular textual phrase be misheard, and why.

2.1 You and me...and Leslie?

In the song *"Groovin' (on a Sunday Afternoon)"*, by the Young Rascals, there's a part in the bridge that many people hear as *"Life would be ecstasy, you an' me an' Leslie"*. In fact, the line is *"Life would be ecstasy, you and me endlessly"*. The confusion lies with the last three syllables of the phrase. The pronunciation of each version, if spoken normally, is as follows:

Orthographic:	and Les- lie	end- less- ly
SAMPA:	`@nd ‘‘lEs li`	`‘‘End l@s li`

In the song, the singer is doing what many singers are taught to do, to make it easier to sustain the singing of words that end with difficult-to-sing consonants: the unsingable consonant is displaced onto the front of the next word. In this case, the consonant “d” is not singable, so he displaces it onto the next syllable, when he can: “and ME” becomes “an dME”, and “end LESS” becomes “en dLESS”.

Basically, singers are *born* to ignore syllable boundries. So, our singer can effectively think of the sung phrase as:

YOU an dME en dLESS lee

This does not cause confusion for listeners, because they are used to hearing this technique used. This does mean, however, that lyric placement does not provide an accurate barometer to a listener of where a word actually ends.

In addition, the singer is fudging their vowels, like singers are taught to do. As a result, “and” and “end” sound almost indistinguishable. So, really, what listeners are hearing is this:

YOU en dME en dLESS lee

Now, the listener’s brain has to take this syllabic gobbledy-gook, and parse it into something useful. They’ve currently got this mess to deal with (represented in SAMPA syllables):

ju *En* ***dmi*** *En* ***dl@s*** *li*

They parse the first part just fine, because the emphases match:

you and **me** *En* ***dl@s*** *li*

But no one says endLESSly. People say ENDlessly. So, the listeners don't recognize it. They have to work with what they have. They already turned one "En d" into an "and", so they do it again:

you and **me** and ***l@s*** *li*

Now, they're just left with LESS lee. And that fits Leslie, a proper noun that fits in context and in emphasis placement. So, the final heard lyric is:

you and **me** and **Les-** lie

The misunderstanding can be traced back to improper analysis of the lyric's multiple aural interpretations. The songwriter probably didn't even think of that, and now he's stuck: a one-hit-wonder with a misunderstood song. We bet that in interview after interview, someone asks him who Leslie is. It's probably very frustrating — especially since he could have just moved the word an eight note later, and it would have been understood perfectly.

That's the sort of situation this program is going to help avoid.

2.2 Why it breaks down

There are two points at which the author's intendeded phrasing can be muddled : First, when the author's orthographic text becomes an orator's spoken (phonetic) interpretation, and second, when the orator's phonetic interpretation

is translated phonetically by an audience into a perceived orthographic phrase. Both of these interpretations must be made succesfully in order for the author's intended meaning to be conveyed.

The phrase "iced ink" undisputedly succeeds in the first translation, but fails on the second. Iced ink can only be pronouced one way, but it can be heard multiple ways–the most notable of which is "I stink", not "iced ink".

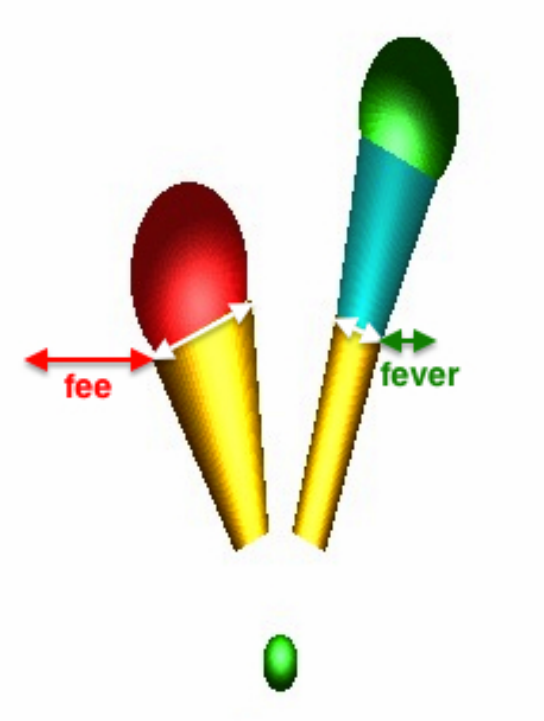
The phrase "a nice cold hour" can fail on both parts. First, the orator could have accidentally-capitalized the word Nice in their head, and made it sound like Nice, the city in France. An audience would likely hear this as "niece", and would be confused, at best. Even if the orator pronouces the phrase as the author intended, the audience could hear multiple orthographic phrases in the same phonetic sequence: "a nice cold hour", "an ice cold hour", or even "a nigh scold our".

A third, more rare and nefarious type of audience misunderstanding can be caused by parse-tree misdirection, where an audience member is absolutely sure they're hearing one phrase, only to get lost halfway through the lyric because they were interpreting a phonetic sequence in a way that resulted in an orthographic dead end. This happens due to the relative frequency of the possible words heard in the aurally-interpreted lyrics.

For example, when asked to sing along with the Adele song, Rolling in the Deep, people who were starting to sing enthusiastically dropped out around the line "reaching a fever pitch"[20]. Let us consider the phrase "fever pitch". This phrase has no exact oronyms, but it does have a potential dead end– a listener could hear the first syllable of the phrase as the word "fee", which has a frequency of 7265. That's more than double the frequency of the word "fever", which is

3095.

Looking at the oronym parse tree for the phrase "fever pitch" in figure 2.1, we can see that the branch for "fever" ends in a much smaller radius than the branch on the left for the word "fee". As you can see by the relative size of the end spheres of the branches, the word "fee" even outweighs the last word in the other branch as well (which is "pitch" with a frequency of 5104). Since the human brain is pre-disposed to parse more-familiar words, having that heavily-weighted dead-end branch is likely the cause of the casual listener not being able to memorize the lyrics.



[h]

Figure 2.1: Annotated Oronym Parse tree generated for the phrase "fever pitch"

Chapter 3

Implementation

We present a computer program which takes in a textual phrase in English, determines all oronyms for that phrase, and then visualizes those oronyms in tree form, with branch width scaled by word frequency metrics to indicate the likelihood of interpretation.

To accomplish this, the program has three major functional parts: a custom phonetic dictionary, a command-line oronym generator, and a OpenGL oronym-parse-tree visualization generator.

3.1 Customized Phonetic Dictionary

In order to discover oronyms for each phrase, we first needed to determine how each phrase is pronounced. Pronunciation can vary depending on the speaker's accent, so it was important for us to (1) chose an accent that we could easily replicate and (2) find a dictionary that supported that accent.

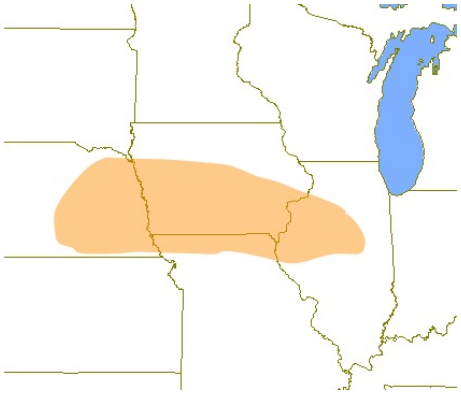


Figure 3.1: This is the geographic area whose accent most closely resembles the General American Accent [6]

3.1.1 Accent Choice

We decided to utilize a General American accent, due to its ubiquity in media and news sources. The General American accent, also known as the "Standard American English" dialect, is not spoken by most Americans, but is used as an "average accent". It most closely resembles the Midwestern accent used in the area in Figure 3.1, but is more commonly recognized as "the newscaster accent". Newscasters learn this accent for use on national TV, because it is the "least-accented" of the American accents[19].

The downside of using the General American accent is that, while it does give a good approximation of most American's speaking accents, it does not perfectly reflect a "singing accent". Singers tend to elongate syllables, changing emphasis placement in words, and vowels tend to be sung in a more "round" matter[22].For example, though the dictionary pronunciation of the word "baby" is `"be$bi` (bay-bee), in songs, you commonly hear the pronunciation `"be$be` (bay-bay). The `e` sound is easier to sing than the `i` (ee) sound, because the latter requires the the

```
ABBREVIATE  AH0 B R IY1 V IY0 EY2 T
```

Figure 3.2: Here is the CMU dictionary entry for the word "abbreviate"

singer move their mouth and vocal cord position further from neutral than the former does[16].

However, different singers will change pronunication based upon which vowels are easiest for them to sing, so using the General American accent still gives us a fairly good approximation[18].

3.1.2 Dictionary Options

We considered using three different phonetic dictionaries: the CMU dictionary, LC-STAR dictionary and UNISYN dictionary[10] [2] [17]. We started out by looking at the LC-STAR dictionary, but quickly decided that it wasnt going to be as useful to us, because the LC-Star project is relatively focused on Speech-to-Speech or Text-to-Speech tech. In addition, the dictionary is not well-maintained.

The CMU dictionary showed promise, but had a few shortcomings. It had a very simple way of encoding words: first the word, then the identifier number in parentheses (if needed), then a space, then a one-to-two char code for each sound in the word, with the numbers 0, 1, 2 appended to indicate emphasis (if needed), separated by spaces. An example of a CMU dictionary entry can be seen in Figure 3.2.

The problem that arose with this format, was that there was no explicit definition of where to hyphenate the word when splitting it up. This causes problems for words in song lyrics, where each note has its own syllable underneath

it, and each syllable might have many different sounds. In addition, it used non-standard symbols for its phonetic alphabet, which would complicate matters if, in the future, we chose to combine data from other dictionaries with our existing dictionary.

However, unlike some other dictionaries we considered, the CMU dictionary (1) was actively maintained, (2) included proper nouns, which are often found in lyrics, but not in dictionaries, and (3) was ridiculously easy to read.

With the downsides and benefits in mind, the CMU dictionary could not be used in isolation, especially if we wanted to incorporate contextual data from other sources.

The UNISYN dictionary is used primarily to phonetically translate words into multiple accents. It has its own formated dictionary, with a bunch of wild-cards representing different phones. UNISYN provides some semi-functioning perl scripts that allow you to specify a dialect youd like to use (For example, a Californian would say "cooking" differently than someone from the Deep South, and both would say it differently than someone from London. However, they are all speaking English. The UNISYN dictionary facilitates this translation).

It had all the information we needed, and then some. However, it was case-insensitve, meaning that it didn't make it easy to differenciate pronunciations for some words. For example, the word "nice" is pronounced differently from the city "Nice", but they were both stored as "nice" in the orthography of UNISYN. The CMU dictionary did keep track of capitalization. The obvious conclusion, then, was to grab the capitalizations from the CMU dictionary and put them in the UNISYN dictionary, aligning them by pronunciation and part of speech.

However, we ran into a setback, mentioned in the very first article we found

Example:

```
transfer : 2 : VB/VBP : tr{ns"f3‘r : tr{nsf3‘r : {trans==fer}
                                : 7184
```

Figure 3.3: Here is an example an entry in our custom phonetic dictionary, using the word "transfer"

references to both dictionaries in: the dictionaries were inconsistent[27]. They didnt always put stresses in the same place, nor did they always have the same pronunciation. Because of this, it was difficult to match words, especially words that were homographic heteronyms[1]. Because of this, we decided to use the UNISYN dictionary exclusively.

3.1.3 Custom dictionary fields

Here is the format for the fields in an entry in our custom phonetic dictionary, after we were done with fixing the UNISYN output:

<ortho> : <uniqueID> : <partOfSpeech> : <SAMPAspelling> : <SAMPAnoEmph> : <extendedOrtho> : <freq>

<ortho> is the regular, orthographic spelling of the word.

<uniqueID> is a number (and optional string) used to differentiate homographs[2].

<partOfSpeech> is used to identify the specific part of speech for the word.

[1] Homographic heteronyms are words that are spelled identically by pronouced differently, such as "Do you know what a buck *does* to *does*?"

[2] A homograph shares the same written form as another word but has a different meaning; For example, a farmer would **sow** *(verb)* seeds in a field, but could also raise a **sow** *(noun)* for bacon.

<SAMPAspelling> is the breakdown of the word, phonetically. It uses the SAMPA alphabet, and separators to show where breaks in the word are, and how they're emphasized. If a separator is `$`, the subsequent phones (until the next separator) are not emphasized. If it's `%`, then they are pronounced using secondary emphasis. If it's `"`, then they are given the primary emphasis in the word.

<SAMPAnoEmph> is the same as *<SAMPASpelling>*, but with all emphasis separator characters stripped out. We chose to add this field so that we could more easily look up phonetic sequence matches.

<extendedOrtho> allows for stemming analysis of words, for possible use in future work.

<freq> is the frequency at which the word occurs in language, according to UNISYN. The frequency count is "taken from a composite of a number of on-line sources of word-frequency. It includes frequencies from the British National Corpus and Maptask, and frequencies derived from Time articles and online texts such as Gutenberg. They were weighted to give more importance to sources of spoken speech, and also to increase the numeric frequency of smaller corpuses"[25].

An example of a entry in our custom phonetic dictionary can be seen in **Figure 3.3**.

3.1.4 Transferring the dictionary to a sqlite database

Because there are several hundred thousand entries in our phonetic dictionary, it was necessary to have a database, rather than store them all in-program in a multi-dimensional array. We decided to use a SQLite database for this purpose.

To turn the colon-delimited dictionary file into a SQLite database, we decided to use a program called the SQLite Database Browser, an open source, public domain, freeware visual tool to create, design, and edit SQLite3.x database files. We specifically used version 2.0b1 of the program, which was built with version 3.6.18 of the SQLite engine[15].

3.2 Oronym Generation

3.2.1 Step 1: Find all phonemic variations of an orthographic phrase

First, our program takes an orthographic phrase to find oronyms for (Figure 3.4).

‘a nice cold hour’

Figure 3.4: A valid orthographic phrase

We then tokenize this phrase into its component words, using whitespaces as a delimiter (Figure 3.5).

‘a’, ‘nice’, ‘cold’, ‘hour’

Figure 3.5: The root orthographic phrase, tokenized

For each word in the phrase, we query our phonetic dictionary for all possible SAMPA pronunciations (Figure 3.6). .

Now that we have the pronunciation of each of the words in the form of SAMPA strings, we can list all the possible phonetic permutations of the original

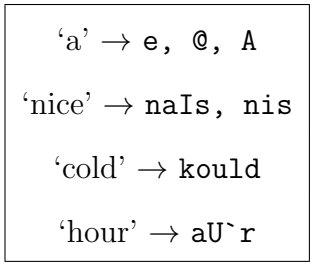


Figure 3.6: In this and all subsequent diagrams, a 'string in quotes' indicates an orthographic word or phrase, and a `monospaced string` indicates that it is a SAMPA word or phrase.

phrase(Figure 3.7). .

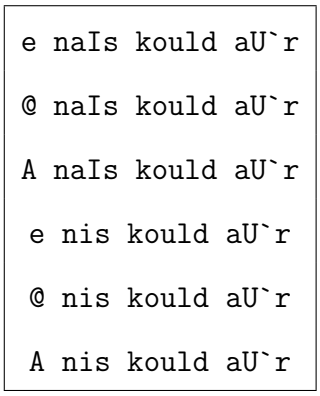


Figure 3.7: Phonetic permutations of the ortho phrase "a nice cold hour"

The pseudocode for this process can be reviewed in figure 3.8.

3.2.2 Step 2: Finding all Orthographic phrases for a Phonemic Sequence

Then, for each phonemic phrase, we want to figure out all valid orthogrpahic interpretations. For this, we have to go back to our phonetic dictionary.

The ideal way to think about searching for words in a phonetic sequence is by picturing the phoenetic sequence in a tree form, similar to the tree pictured

```
findAllPhoneSeqsForOrthoPhrase( orthoPhrase ) {
  allFullPhrasePhoneSeqs = empty list of list of phones
  orthoWords = split orthoPhrase on spaces

  origNumFullPhrases = 0
  for( orthoWord in orthoWords with index i ) {
    nextWordSampaPhoneSeqs = possible phone seqs following orthoWord

    if ( orthoWord is the first word in orthoPhrase ) {
      for( phoneSubSeq in nextWordSampaPhoneSeqs ) {
        append phoneSubSeq to allFullPhrasePhoneSeqs[i]
      }
    } else {
      origNumFullPhrases = allFullPhrasePhoneSeqs.size()
      if theres more than one vector <phone> in nextWordSampaPhoneSeqs
        then we need to create duplicates of all existing allFullPhrasePhoneSeqs
    }

    for( m = 0 to allPhrasePhoneSeqs.size() ) {
      phraseToAppendIndex = m / origNumFullPhrases
      phoneSeqToAppend = nextWordSampaPhoneSeqs[phraseToAppendIndex]
      append phoneSeqToAppend to allFullPhrasePhoneSeqs[m]
    }
  }

  return allFullPhrasePhoneSeqs
}
```

Figure 3.8: Algorithm to get all phonetic sequences for an orthographic phrase.

in abbreviated form in Figure 3.9. For example, if I had a phonetic tree with the entire dictionary in it, each phonetic tree node would have at least 45 child nodes: one for each phone. A node might also have "word" nodes, if the phones along the path to that node construct a valid orthographic word:

When there are multiple orthographic interpretations at a single phonetic node, the most likely interpretation can be determined by checking the frequency of use for each word. For example, the sequence `n aI s` is much more likely to be "nice" than "gneiss". Figure 3.9 shows a visual representation of traversing an entire dictionary's phonetic tree for nodes along the paths for the SAMPA sequences `aI s` and `n aI s`.

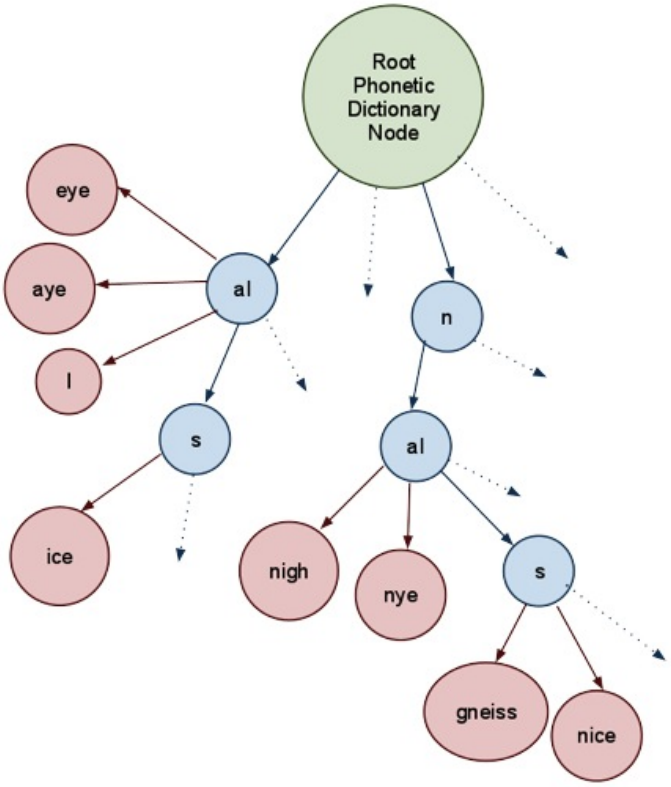


Figure 3.9: Word tree

We can use this dictionary tree method to discover all valid orthographic

interpretations for any phonetic sequence of our root orthographic phrase, as shown in figure 3.10 for the phrase:

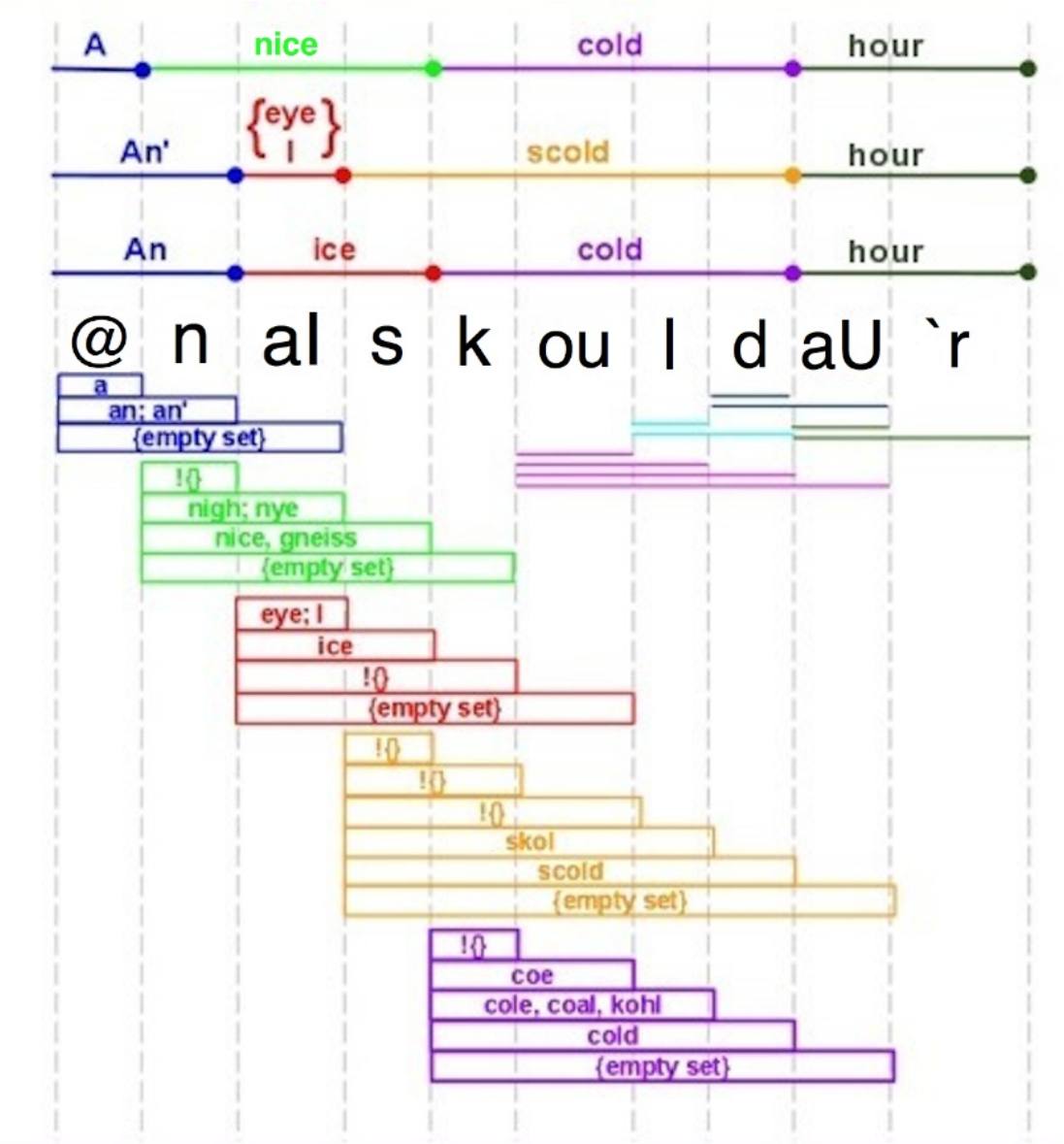


Figure 3.10: Phoneme to ortho graph of "a nice cold hour"

Once we have grabbed all the orthographic interpretations for each phonetic sequence, we combine them all into an orthographic oronym phrase list. This process may leave us with some redundant oronyms, so we de-duplicate that list.

After this, we have a list of all unique and valid oronyms for the original root

```
discoverOronymsForPhrase( origOrthoPhrase, includeDeadends ) {
   orthoMisheardAsPhrases = empty list
   allPhoneSeqsOfOrigPhrase = origOrthoPhrase.findAllPhoneSeqs()

   for( curPhoneSeqWithEmph in allPhoneSeqsOfOrigPhrase ) {
      // Remove emphasis marking for easier lookups
      curPhoneSeq = curPhoneSeqWithEmph.stripEmphasis()

      altOrthoPhrases = findOrthoStrsForPhoneSeq( curPhoneSeq )

      for( altOrthoPhrase in altOrthoPhrases ) {
         // Ensure it contains valid ortho text in all cases, and if
         // includeDeadends=false, contains no deadEndDelims so we only add
         // fully valid strings
         if ( ( includeDeadends == true &&
                altOrthoPhrase != deadEndDelim1 &&
                altOrthoPhrase != deadEndDelim2 ) ||
              ( altOrthoPhrase.contains( deadEndDelim1 ) == false &&
                altOrthoPhrase.contains( deadEndDelim2 ) == false ) ) {
            append altOrthoPhrase to orthoMisheardAsPhrases
         }
      }
   }

   orthoMisheardAsPhrases.removeDuplicates()

   return orthoMisheardAsPhrases
}
```

Figure 3.11: Algorithm to get all oronyms for an orthographic phrase.

phrase.

In the case of "a nice cold hour", this returns 290 oronyms, as seen in the first column of figure A.1.

The pseudocode for this process can be reviewed in figure 3.11

3.2.3 Word Frequency Evaluation

Next, we want to evaluate all our oronyms based on how common each oronym's component words are. For example, "a nice cold hour" is much more likely to be heard "a gneiss cold hour," even though both are phonetically identical.

To do this, we tokenize each oronym phrase into its component words, once again delimiting by non-newline whitespaces.

Then, we query our phonetic dictionary with each word to get that word's frequency value. We store each word's value separately. When we have retrieved the frequencies for all the words in a phrase, we then sum up all the frequencies to give a combined frequency of the entire phrase.

You can see these frequency counts for each word in all oronyms of the phrase "a nice cold hour" in table A.1.

3.3 Visual Representation

We created two different oronym visualizations. The first, oronym trees, were chosen for their ability to show the phonetic dead ends that may happen during oronym generation. Our particular oronym tree visualization is written in C++ using OpenGL, which allows for future integration into another codebase.

The second visualization shows all valid oronyms of a root phrase using what is known as a sunburst diagram. These sunburst diagrams were created using the ProtoVis library, and use javascript data files with html wrappers. The javascript data files were generated using the same C++ code that the first visualization

used, so both visualizations show similar data. However, the oronym sunburst diagrams more easily exhibit the weighting of the different oronym paths with their frequency dictionary values.

3.3.1 Oronym Tree Visualization

We go about building the visual representation of the oronym parse tree in much the same way that we build the textual list of oronyms, with one important caveat: our oronym parse trees may contain oronym fragments. To deal with these, we've got to keep track of all our abandoned sub-phrases. For example, for the root oronym phrase "iced ink" (`aI s t I N k`), a listener may hear and interpret up to "I sting"(`aI s t I N`), and then be confused when the last phone, `k`, comes along.

Our algorithm for building the tree diagram is recursive, called from a parent function that draws the tree's 'seed' sphere. This parent function engages in a depth-first traversal of the oronym tree, and is documented in figure 3.23

We start in the parent function by getting all the oronyms of our orthographic phrase, using the process outlined in sections 3.2.1 and 3.2.2. However, instead of ignoring any incomplete orthographic interpretation

aye sting xxx

aye stink ___SUCCESS!___

ay sting xxx

ay stink ___SUCCESS!___

eye sting xxx

eye stink ___SUCCESS!___

i sting xxx

i stink ___SUCCESS!___

ice ting xxx

ice xxx

iced ink ___SUCCESS!___

Figure 3.12: All partial and complete oronyms for the phrase "iced ink"

of a phonetic sequence, as we do in section 3.2.2, we add the incomplete phrases to the list of oronyms, keeping track of them by appending 'xxx' or 'fff' to the end of each incomplete oronym string. For example, as shown in figure 3.12. the phrase "iced ink" may only have has five complete oronyms, but it has six additional oronym fragments, making for 11 possible interpretations.

Then, we tokenize our phrases by whitespace, and look up the frequency of each word, as shown for "iced ink" in figure 3.13. We will later scale our branches' radii using the maximum and minimum word frequency values found during this run. In this case, the maximum is 9,937,877 for the word "I", and the minimum is 124 for the word "ting".

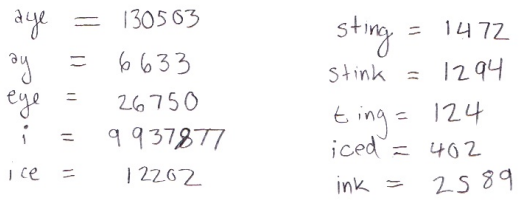


Figure 3.13: Frequency values for all unique words in "iced ink" oronyms

Once we have all partial and complete oronyms, plus the maximum and minimum word frequency values found in all those phrases, we pass them into our recursive function, along with the radius of the seed sphere. That radius will be the beginning radius of each root-level branch, as shown in figure 3.14

Inside our recursive function, we pull the first word out of each orthographic phrase, and create a set of unique first words, as seen in figure 3.15.

We then go through this set of unique first words iteratively.

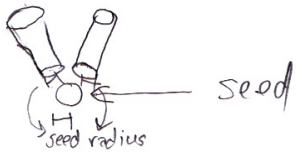


Figure 3.14: Seed Sphere vs branch radius comparison

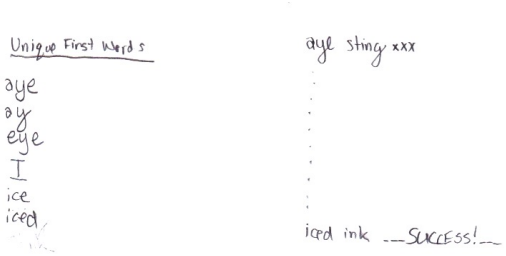


Figure 3.15: All uniqueirst words of "iced ink" oronyms

For each word, we look up frequency in the phonetic dictionary. Then, we use the maximum and minimum frequencies that we found in our parent function, plus constants for maximum and minimum radius size, to scale that frequency into a usable radius size.

Then, we check the contents of the word string.

If the word is "xxx" or "fff", then it's not a word at all—just an indication of the dead end of an oronym fragment. In this case, as seen in figure 3.16, we draw a red sphere with the radius of the branch's ancestor, using the radius parameter passed into our recursive function for 'lastRadius'.

If the word is "___SUCCESS!___", that indicates a full oronym has been successfully found, and is terminating at that point. This time, we draw a green sphere using the *lastRadius* parameter for size, as seen in figure 3.17 for the phrase "I stink".

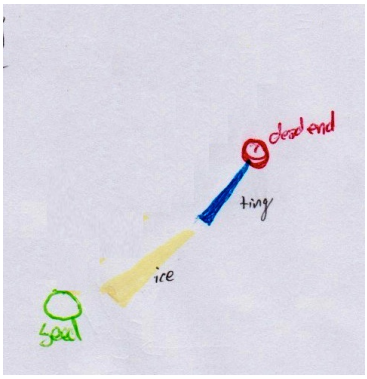


Figure 3.16: Dead end sphere for oronym fragment "ice ting"

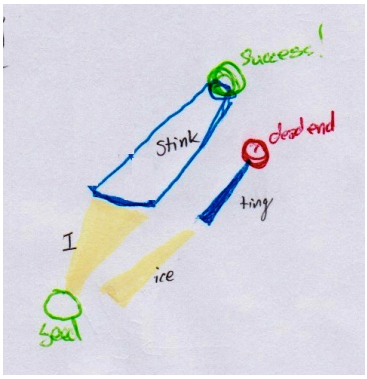


Figure 3.17: Success indicator sphrere for complete oronym "I stink"

If the word isn't "xxx", "fff", or "___SUCCESS!___", it is a valid orthographic word, and we draw a cylinder "branch" representing that word. The cylinder's bottom radius is equal to *lastRadius*, and the top radius is equal to the scaled radius that we derived using the word's frequency. An example of this branch radius scaling is show in figure 3.18.

After we draw the cylinder, we then go through the full list of phrases, and compile a list of all phrases that start with the word we just drew the cylinder for, as in figure 3.19. Then, we remove the first word from each of those phrases, deduplicating the resulting list of "tail" phrases, which is shown in figure 3.20.

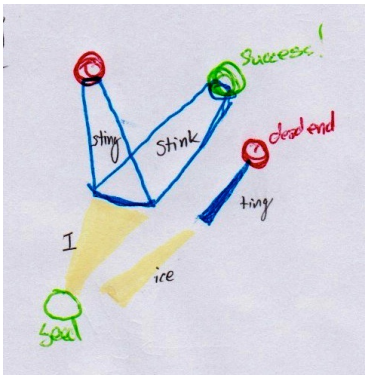


Figure 3.18: Scaled branch radii showing frequency difference

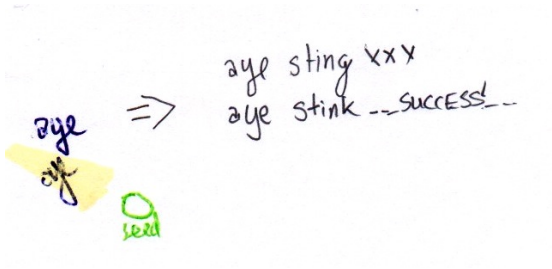


Figure 3.19: All oronym phrases of "iced ink" starting with "aye"

Then, we change our material color (so that different levels of branches will be different colors), and make a recursive call to our current function, passing as parameters the scaled radius and the list of tail phrases.

After this recursive call, we change our color material back to whatever it was before the call, and then continue on to the next unique first word in our set, which, in this case, is "ay."

Once we have looped through all our unique first words, we know we're done drawing that set of branches, and we return.

This gives us the oronym parse tree seen in figure 3.21. As shown in figure 3.22 (the annotated version of figure 3.21) each branch on the tree represents a single orthographic word.

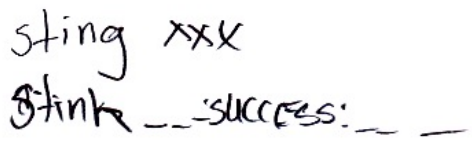


Figure 3.20: Tail phrases for oronyms of "iced ink" that begin with "aye"

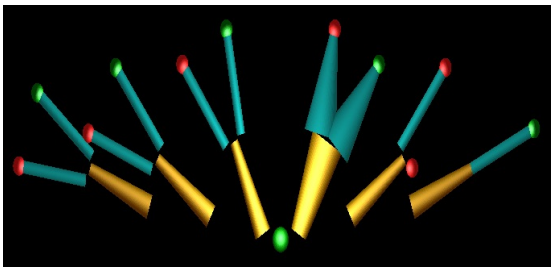


Figure 3.21: Oronym tree for the phrase "iced ink"

3.3.2 Oronym Sunburst Visualization

For our second visualization type, we chose to use sunburst diagrams.

Sunburst diagram generation

To generate these sunburst diagrams, we modified our existing C++ program to output data in the Protovis.js format, which is seen in figure 3.25.

The labels before the colons are displayed on the diagram in their respective segment, as seen in figure 3.26.

Some minor adjustments were made to the C++ output. The Protovis document format does not allow for non-alphanumeric characters to appear in labels, so words like "ice-cold" or "it's" caused errors. We found it necessary to remove all non-alphanumeric characters so that the sunbursts would generate success-

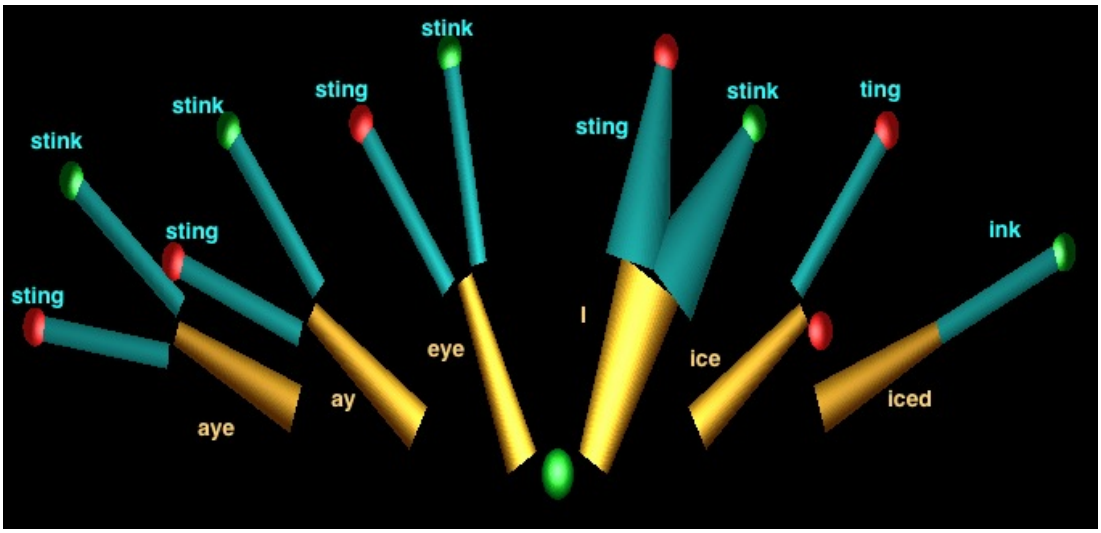


Figure 3.22: Annotated oronym tree for the phrase "iced ink"

fully.

One of the main benefits of the Protovis data format is that, once the relevant data has been formatted correctly, many different types of graphs can be trivially generated. For our data format, we can generateboth sunburst and icicle graph views. We chose to use sunburst graphs, as results from an informal user poll indicated a preference for sunbursts over icicles.

Reading a sunburst diagram

To read a sunburst diagram, start at the very center, which in our case is labelled "root". Then, pick any one segment from the first ring surrounding the root. The word contained in this segment will be the first word in the oronym phrase.

Next, look at all the outer arc segments that directly touch the first segment picked. Every word in those adjoining arc segments is a valid subsequent word for the oronym phrase starting with the word in the first, inner arc segment.

```
buildAndDrawFullTree( orthoPhrase ) {
   fullPhrases = orthoPhrase.discoverOronyms()
   (maxWordFreq, minWordFreq) = fullPhrases.getMaxAndMin()

   // Draw the tree's seed.
   glPushMatrix()
   {
      glTranslated(0.0, -1.0 * DEFAULT_BRANCH_LEN, 0.0)
      materials(GreenShiny)
      drawSphere(DEFAULT_RADIUS)
      materials(allMaterials.at( mat % allMaterials.size() ) )

      drawBranchesAtFork ( fullPhrases, DEFAULT_RADIUS )
   }
   glPopMatrix()
}
```

Figure 3.23: Given an orthographic phrase, this function prepares to draw the tree

Continue this process, using one segment per level, until you reach a segment that has no subsequent outer segments. At this point, you will have compiled a full oronym phrase. The size of the final outer segment, relative to size of the rest of the segments in its particular ring, shows you the relative commonness of the phrase whose path ends at that segment.

So, for example, interpreting the sunburst diagram in figure 3.26 would result in the following faux-oronym phrases: "child1 child1A child1Ai", "child1 child1A child1Aii", "child1 child1B child1Bi", "child2 child2A child2Ai", and finally, "child2 child2A child2Aii".

Example Sunburst Diagrams

We generated several types of sunburst diagrams, using both artificially-balanced path weights and using the frequency values for each path derived from

```
drawBranchesAtFork( fullPhrases, lastRadius) {
   if( fullPhrases.size() == 0 ) {
      return
   }

   // Use a set to ensure no duplicates.
   firstWords = empty set

   for( phrase in fullPhrases ) {
      if( phrase.size() > 0 ) {
         firstWords.insert( phrase.firstWord() )
      }
   }

   // Calculate positioning variables for the spread of branches for firstWords.
   for ( curFirstWord in firstWords ) {
      firstWordFreq = curFirstWord.frequency()
      newAdditiveRadius = firstWordFreq.scaleToRadius()

      glPushMatrix()
      {
         // Translate and rotate into place
         if( curFirstWord == deadEndDelim1 || curFirstWord == deadEndDelim2 ) {
            // Draw a red sphere at the end of the last branch
         } else if ( curFirstWord == successDelim ) {
            // Draw a green sphere at the end of the last branch
         } else {
            // Draw a branch
            drawBranch( radiansToDegrees(tiltAngle), curXOffset, curYOffset,
                        newAdditiveRadius, lastRadius )

            // Find all phrases in fullPhrases that start with that firstWord
            tailsVect = fullPhrases.findAllWithPrefix(curFirstWord)

            // Change the colors for each branch level

            // Pass those phrases to drawBranchesAtFork
            drawBranchesAtFork( tailsVect, newAdditiveRadius, curXOffset, curYOffset )

            // Change the colors back to ensure consistency for each branch level
         }
      }
      glPopMatrix()
   }
}
```


Figure 3.24: This is the function that facilitates the in-time drawing of the tree as we parse though our oronyms possibilities

```
var root = {
        Child1: {
                Child1A: {
                        Child1Ai: actualCount,
                        Child1Aii: actualCount
                },
                Child1B: {
                        Child1Bi: actualCount
                }
        },
        Child2: {
                Child2A: {
                        Child2Ai: actualCount,
                        Child2Aii: actualCount
                }
        }
};
```

Figure 3.25: Protovis sunburst data format: This example sunburst data file, once the *actualCount* occurances were replaced with actual values, would generate a sunburst diagram

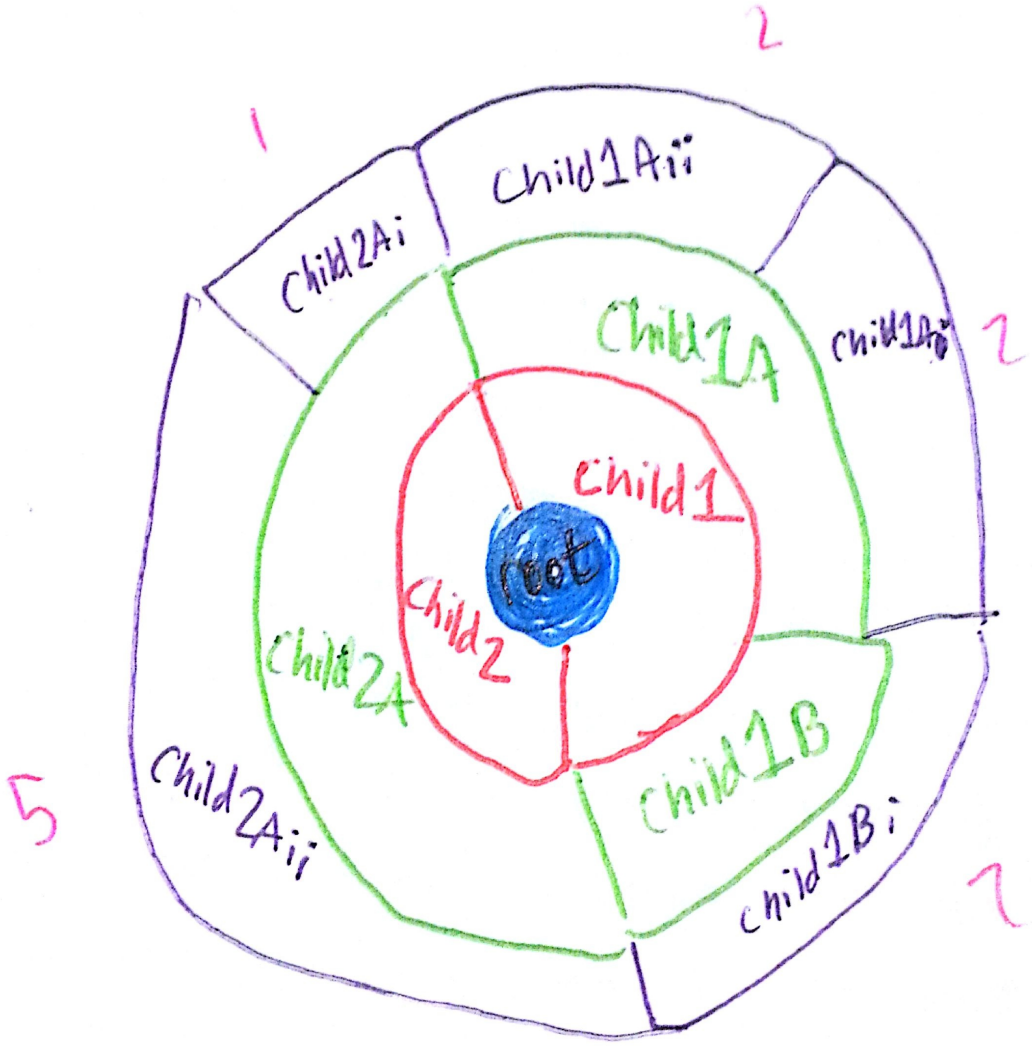


Figure 3.26: This example sunburst diagram is what would be generated by the example data file in figure 3.25

our UNISYN dictionary. A clearer view of how we used sunburst diagrams can be provided with some concrete examples.

Consider the two sunburst diagrams for the oronyms of the phrase "iced ink", shown in figures 3.27 and 3.28. The phrase "iced ink" has five different oronyms: "iced ink", "ay stink", "aye stink", "eye stink", and "I stink". The five different outer segments (which are easier to see on the equal-weighted sunburst diagram in figure 3.27) represent the end word of each of those oronyms. The sunburst diagram that uses the frequency metric (shown in figure 3.28) shows that people are overwhelmingly more likely to hear "I stink" than any other possible oronym.

For a more complicated example, take the sunburst diagrams for oronyms of the phrase "why that's insane". The diagram shown in figure 3.29 shows the seven possible oronyms that can result from phonetic interpretation of that phrase, and even though one phrase is one word shorter than all the rest, the weights of the paths remain equal. The weighted diagram, shown in figure 3.30, shows that the UNISYN-predicted frequency of each phrase's occurance.

Lastly, consider the sunburst diagrams for the phrase we chose to focus on in our user study: "an ice cold hour". As seen in figure 3.31, the equally-weighted sunburst diagram shows all possible oronym paths. When compared to the sunburst diagram in figure 3.32 that uses the UNISYN-derived frequency metric, we can see that some paths, such as those that begin with the word "a", are much more likely to be heard than those that begin, for example, with the word "n".

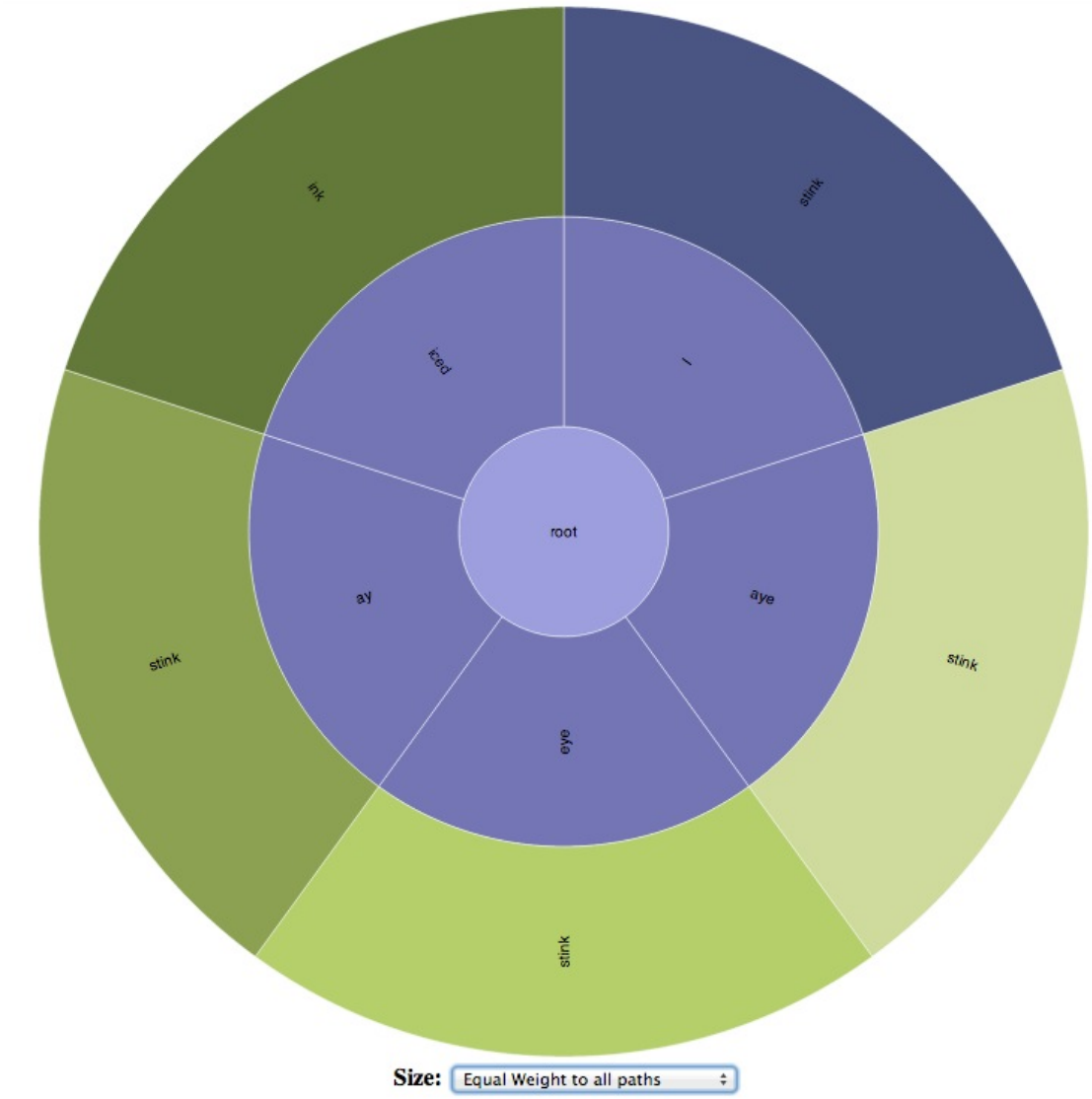


Figure 3.27: Equally-Weighted Sunburst Diagram for the oronyms of "iced ink"

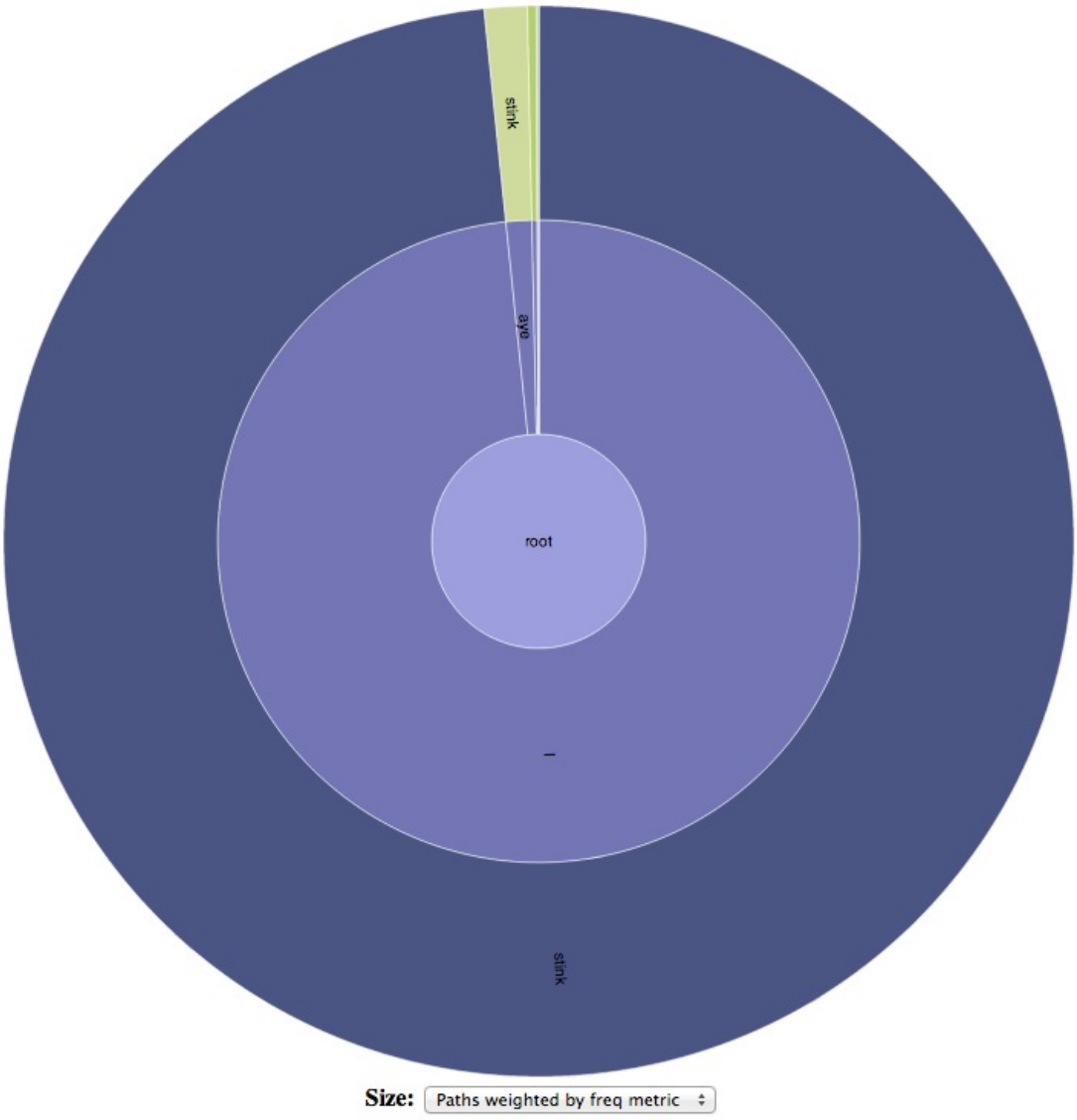


Figure 3.28: Sunburst Diagram for the oronyms of "iced ink" weighted by UNISYN freq metric

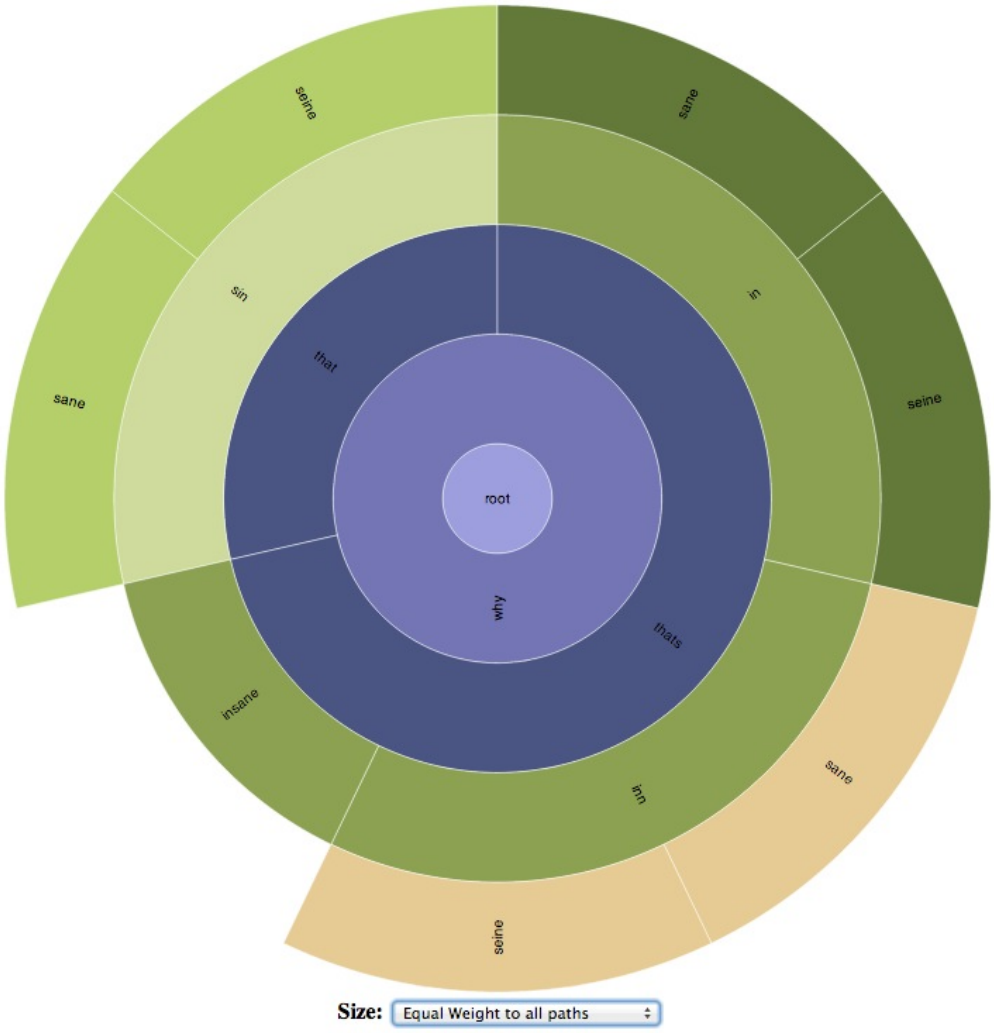


Figure 3.29: Equally-Weighted Sunburst Diagram for the oronyms of "why that's insane"

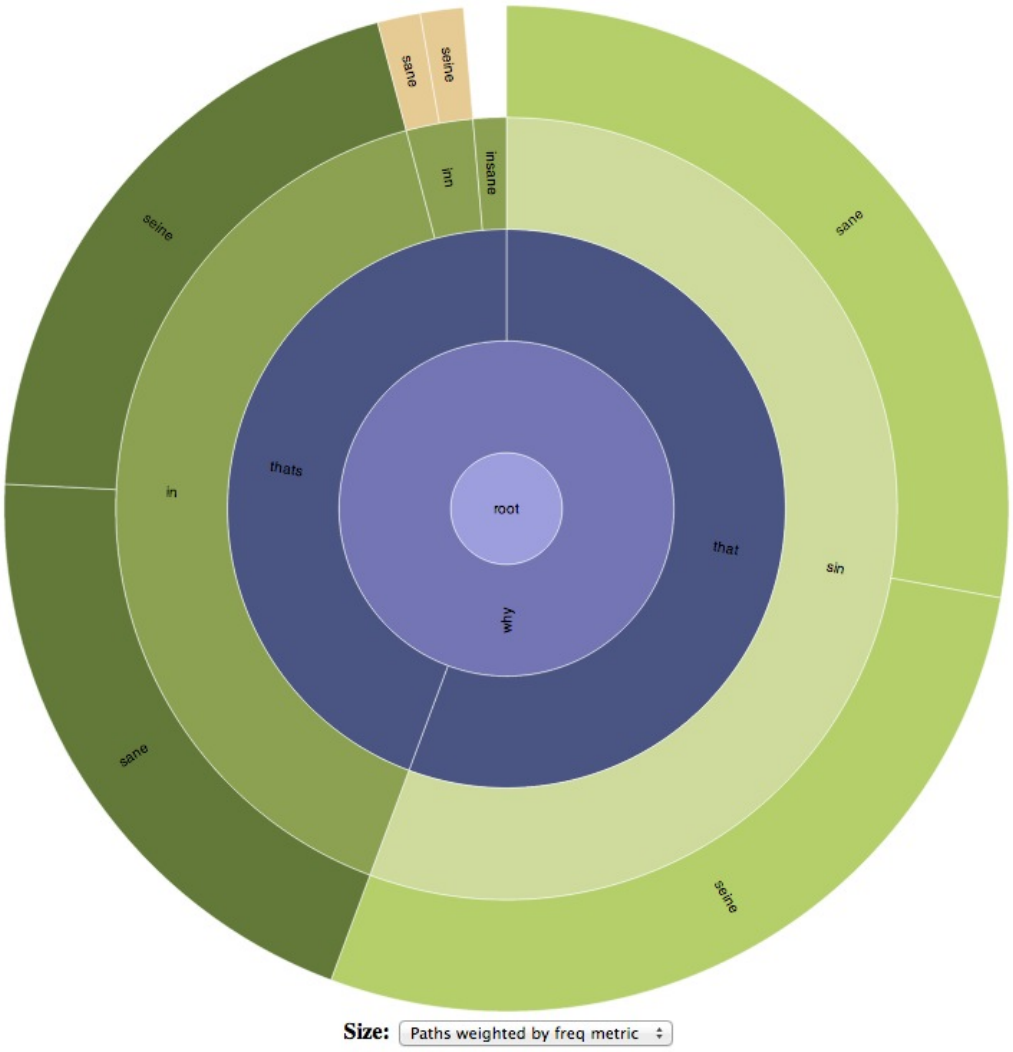


Figure 3.30: Sunburst Diagram for the oronyms of "why that's insane" weighted by UNISYN freq metric

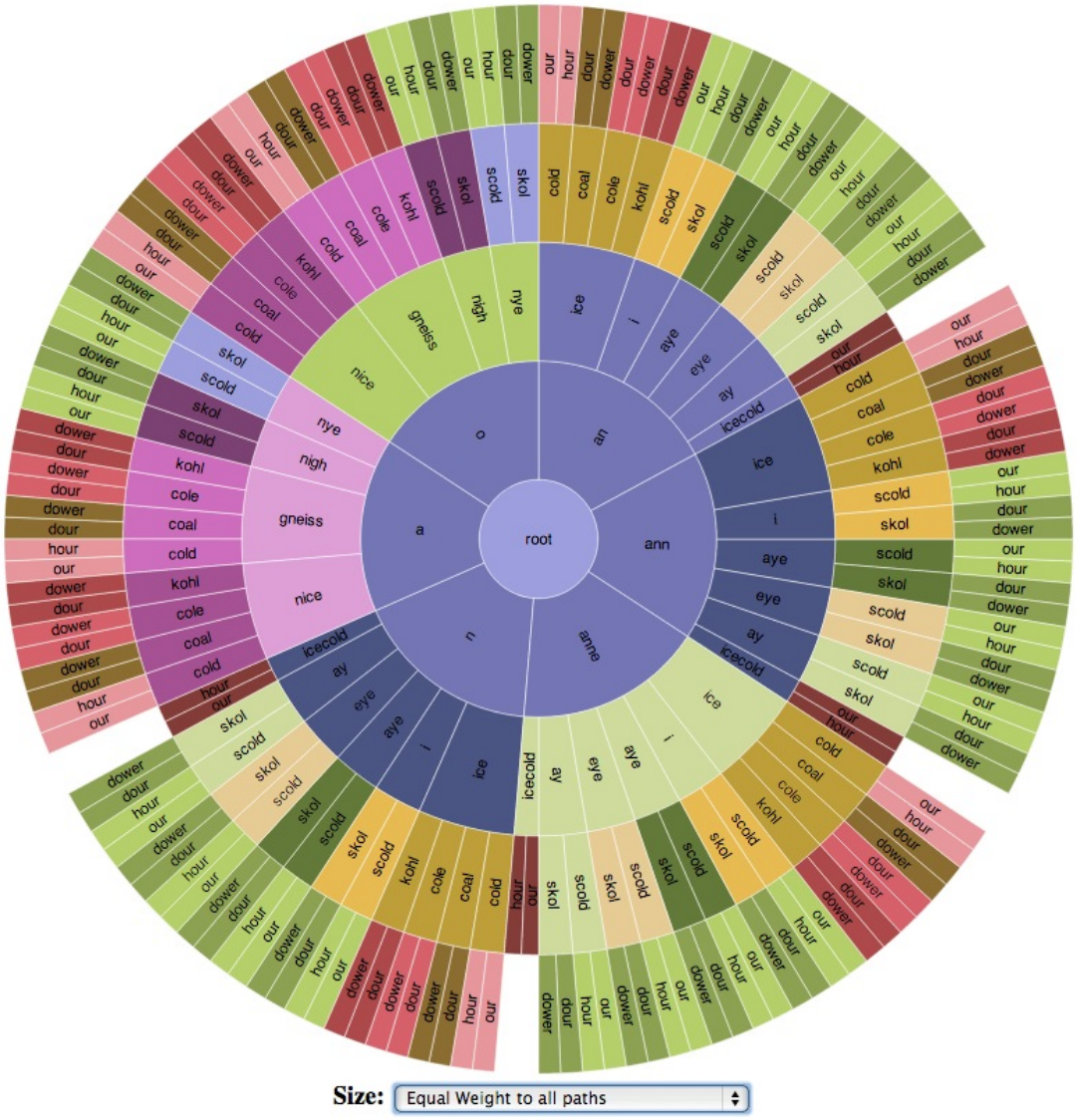


Figure 3.31: Equally-Weighted Sunburst Diagram for the oronyms of "an ice cold hour"

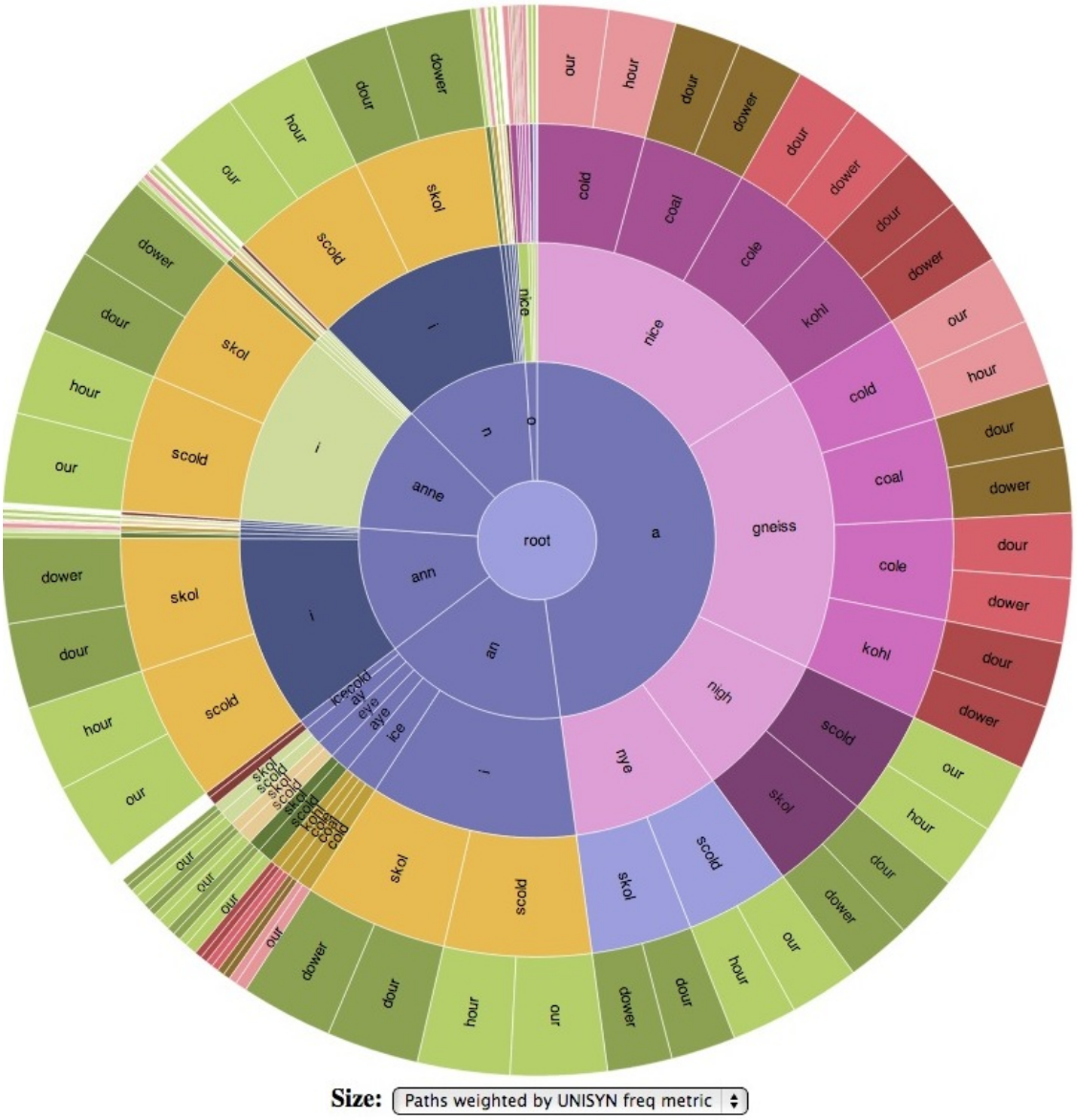


Figure 3.32: Sunburst Diagram for the oronyms of "an ice cold hour" weighted by UNISYN freq metric

Chapter 4

User Study

4.1 Structure

We created a multi-phase user study to examine the correctness of our method of word frequency metric generation, which uses the UNISYN dictionary.

First, we generated oronyms for the phrases "a nice cold hour" and "fourth rye to". Using the selection criteria outlined in section 4.3.1, we narrowed down our recording options to 48 out of the 290 oronyms generated for "a nice cold hour", and 10 out of the 39 oronyms generated for "fourth rye to".

In the first phase, we had a dozen people record over 72 recordings of 58 different phrases. This phase served two purposes: one, to see if our phonemic transcriptions were valid, and two, to gather recordings for the second phase.

In the second phase, we took 15 recordings of oronyms from phase one, and gathered ∼74 transcriptions for each recording, resulting in a total of 851 transcriptions. These transcriptions were provided by 208 unique users (127 from the United States). We then compared the observed frequency of transcriptions

of the recorded oronym phrases to the calculated frequency metric for oronyms of the original root phrase.

4.2 User Sampling Population

We drew our test subjects from a pool of Amazon Mechanical Turk workers (hired for $ 0.02 to $ 0.10 per task) and, for part of phase 1, volunteers from Reddit.com [4] [5].

Amazon Mechanical Turk is an online crowdsourcing service where requesters can hire workers to complete Human Intelligence Tasks, or HITs. The efficacy of using Mechanical Turk for user studies has been widely studied in academia, and specifically proven in the linguistic community [30].

Due to the online nature of Mechanical Turk, we we able to gather an international pool of volunteers. As you can see in figure 4.1, the bulk of our responses came from the United States and India.

4.3 Methodology

4.3.1 First Phase: Recitation

In this phase of the user study, we used a combination of a dozen Mechanical Turk workers (hired for $ 0.10 per task) to record 72 different phrases. These phrases were oronyms of one of two phrases: phrase A, “a nice cold hour” or phrase B, “fourth rye to”.

Given that “a nice cold hour” has 290 oronyms, and “fourth rye to” has 39 ,

it was necessary to narrow down the number of oronyms recorded in phase one of our user study.

To select which oronyms we would submit to Mechanical Turk for workers to record, we decided on the following selection criteria:

- We discarded oronyms that included words with frequency values of less than 30. Asking a general audience to pronounce uncommon words would likely result in a high rate of unusable recordings. In addition, including uncommon words wouldnt be testing our algorithm—it would be testing the vocabulary of the user study subjects, which is outside the scope of this project.

- We discarded all oronym phrases that contained words that capitalize to proper nouns, if that capitalization leadto alternative pronunciations. For example, "nice" maps to the phonetic sequence `n aI s`, but "Nice" maps to the phonetic sequence `n i s` (as in "niece") .

- We discarded oronym phrases that included implicit punctuation. For example, the phrase "Anne I scold our", has an implicit comma between Anne and I. We did this to avoid halting or "dramatic" recording of phrases.

- We discarded oronyms with any words whose pronunciation is position specific. For example, the word " 'n' " is pronounced `@ n` when found in "Rock 'n' Roll, but when on its own, is pronounced `E n`, which doesn't map back to the original root phrase "an ice cold hour". We also removed some phrases involving the word " o' ". When pronounced `@`, as it is in "top o' the morning", it maps back to the original root phrase, but when found outside of that phrase, as in a last name like O'Donnell, it is pronounced `oU` , and doesn't map back.

- We only chose oronyms for which all pronunciations of the child oronym phrases were also found in the pronunciations of the root oronym phrase. For example, a root phrase that begins with "a" would have an child oronym phrase that begins with "et", using French pronunciation which drops the trailing `t` sound. However, "et" can also be pronounced with the `t`, using an American accent, as in the phrase "et al". Since our root phrase doesnt include that `t` sound, any child oronym phrases that begin with "et" have at least one pronunciation that does not match the root phrase's pronunciation. So, we discard all child oronyms that begin with "et".

At the end of this process, we were left with 48 out of the 290 oronyms generated for "a nice cold hour", and 10 out of the 39 oronyms generated for "fourth rye to".

To keep track of the chosen phrases, we assigned each phrase an phraseID, built off of the phrase letter, phrase length, and phrase text. We gave Mechanical Turk workers three minutes to record each phrase and email it to us with the phrase identifier in the subject of the email. The number of recordings per phrase, along with their identifiers, can be seen in table **??**.

We then interpreted the phonetics of each of the recordings in SAMPA by ear. In a stunning example of a use case for our project, we discovered that we had unintentionally included some phrases for recording which were not deterministically phonetically parsible, meaning that our oronyms had multiple pronunciations, not all of which mapped back to the original phrase. While we had caught and removed the phrases that began with "et", our algorithm mapped the orthographic word "a" to the phoneme `A` (as in "a capella" or "f*a*ther". That `A` phoneme can be combined with the subsequent `n` phoneme from the word "nice"

to create the SAMPA sequence `A n`, which corresponds with the orthographic word "on". Since that doesn't map back to our original root phrase, we were forced to discard all transcriptions that ended up with that pronunciation. That being said, the remaining pronuciations fit with our model, and we found no unexpected phonetic anomalies when comparing our recordings to the expected SAMPA pronunications of each phrase.

4.3.2 Recording Sample Pool

We had originally intended to use all the phase one recordings in phase two, but had to discard all but 15 of the recordings for various reasons, the most common being that the recording was too loud and we wanted to spare our user's ears. Occasionally, the person recording left excessive amounts of space between words that overly-segmented the phrase, interrupting the natural flow of the phonetic sequence and rendering it unusable for our purposes. The recordings for the "fourth rye to" oronyms were all unusable for phase two, because our users tended to insert exclamation points any time they said "ooh" or "too", overloading their microphones or over-segmenting the phrase.

All 15 recordings we used were oronyms for the phrase "a nice cold hour", and were recorded by one man from the midwest, whose accent which made him the best approximation we could get for a General American accent. All other recordings gathered in phase one did not have appropriate accents, and were summarily discarded from further data collection.

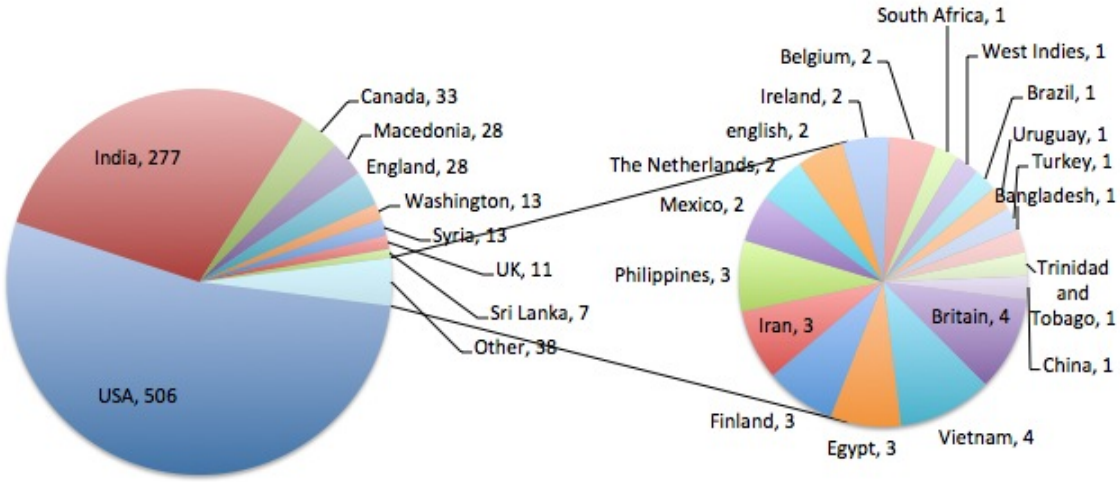


Figure 4.1: Our user study primarily polled people from the United States and India, as can be seen by the number of responses originating from each country.

4.3.3 Second Wave: Transcription

We hired 208 unique Mechanical Turk workers to transcribe our oronym recordings for $ 0.02 to $ 0.03 per transcription. Each of the 15 recordings was transcribed ∼74 times, resulting in a total of 851 transcriptions. These transcriptions were provided by 208 unique users (127 from the United States). In addition to transcribing the recording, in each task the worker was asked what country they were from. We did this to help differentiate native American English speakers from non-native speakers.

Response By Country	Num Responses
USA	506
India	277
Canada	33
England	28
Macedonia	28
Washington	13
Syria	13
UK	11
Sri Lanka	7
Britain	4
Vietnam	4
Egypt	3
Finland	3
Iran	3
Phillipines	3
Other	18

Table 4.1: Here's a table with the number of responses per country

Chapter 5

Results

5.1 Phase One Results

In this phase, we recorded a dozen users reciting any of the 48 oronyms of the phrase "an ice cold hour", or any of the 10 oronyms for the phrase "fourth rye to". Out of 72 recordings, only the recordings of the oronyms of "fourth rye to" were found to diverge from our excepted phonetic patterns, likely due to poor microphone quality not being able to pick up the aspriated *'f'* sound at the beginning of the phrase[24]. All other oronyms were found to be within reasonable tolerance levels, with 15 recordings from one particular speaker found to be a close enough match to the general American accent to use his recordings in phase two.

5.2 Phase Two Results

We gathered 953 transcriptions for our 15 recorded phrases, with each recording garnering ∼74 transcriptions each. Worldwide, the top four most-frequently transcribed phrases made up for 70% of total transcriptions. The top transcribed phrase worldwide was "an ice cold hour", with 352 transcriptions, followed by "a nice cold hour", with 217 transcriptions. Following that, "a nice gold hour" had 63 transcriptions, and "in ice cold hour" had 38 transcriptions. The breakdown of these top four can be seen in figure 5.1 and table 5.1.

All of the worldwide top transcribed phrases were predicted by our oronym-generator, except for "a nice gold hour". This is a known limitation of our project, though, because we chose to focus on exact phonetic matches. The cold/gold mishearing is a product of phoneme voiced/voiceless pair swapping, which we cover in-depth in section 6.2. It is outside the current scope of our project.

5.2.1 Transcribed oronyms' observed frequency vs expected frequency

Though the most commonly transcribed phrases were predicted by our method of oronym generation, figure 5.3 shows an unexpected distribution of the number of times each phrase was transcribed versus the frequency metric that we calculated. We hypothesized that a simple summation of the UNISYN-provided word frequencies for each word in a phrase would produce a meaningful indicator of whether a phrase's likelyhood to be heard.

Unfortunately, that proved not to be the case. In figure 5.2, we see the sun-

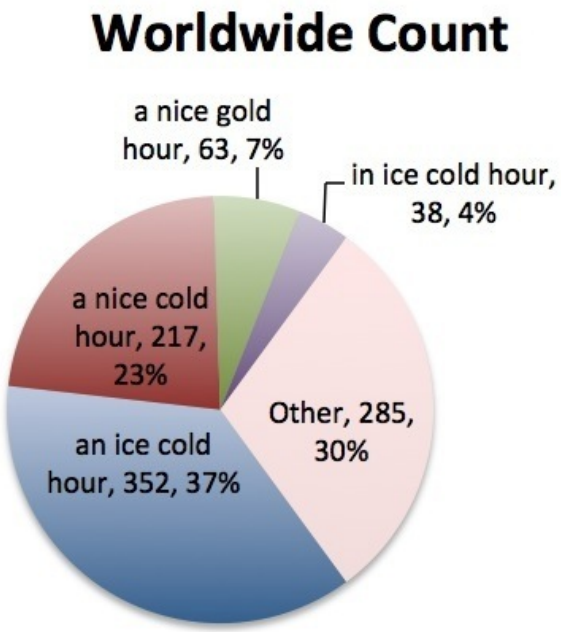


Figure 5.1: Our top two transcriptions were "a nice cold hour" and "an ice cold hour"

predicted freq	phrase transcribed	total answers
931028	an ice cold hour	352
7851662	a nice cold hour	217
0	a nice gold hour	63
5503158	in ice cold hour	38
0	an ice gold hour	18
859307	an eye scold hour	13

Table 5.1: In this table, we list all oronyms that were transcribed more than five times. Out of this list, all but the two containing the word "gold" were predicted by our oronym algorithm. However, we expected that any voiced/voiceless phoneme substitutions, like "cold"/"gold" would be missed by our algorithm.

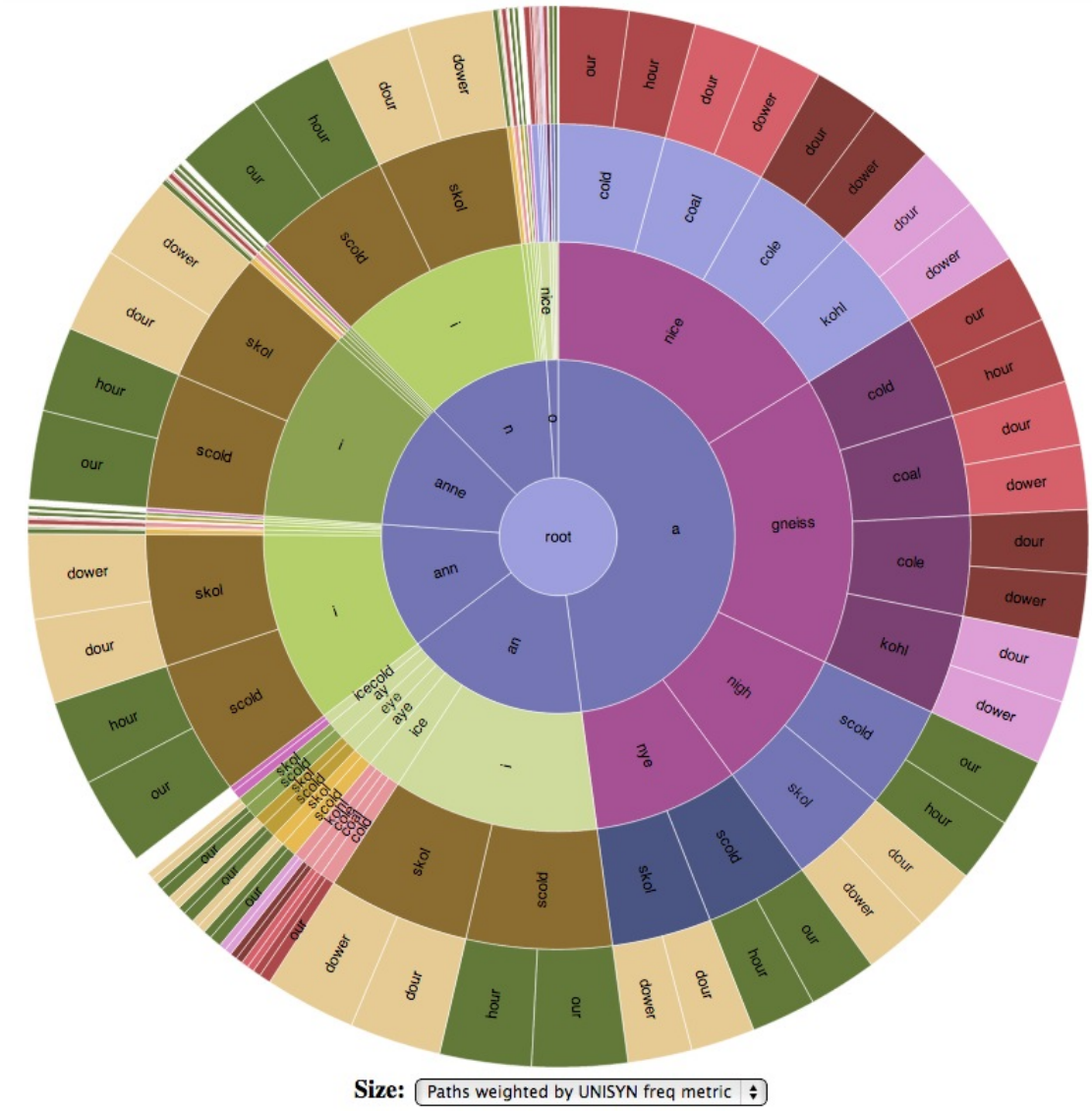


Figure 5.2: Sunburst Chart for A Nice Cold Hour using UNISYN metrics for comparison to observed frequency sunburst

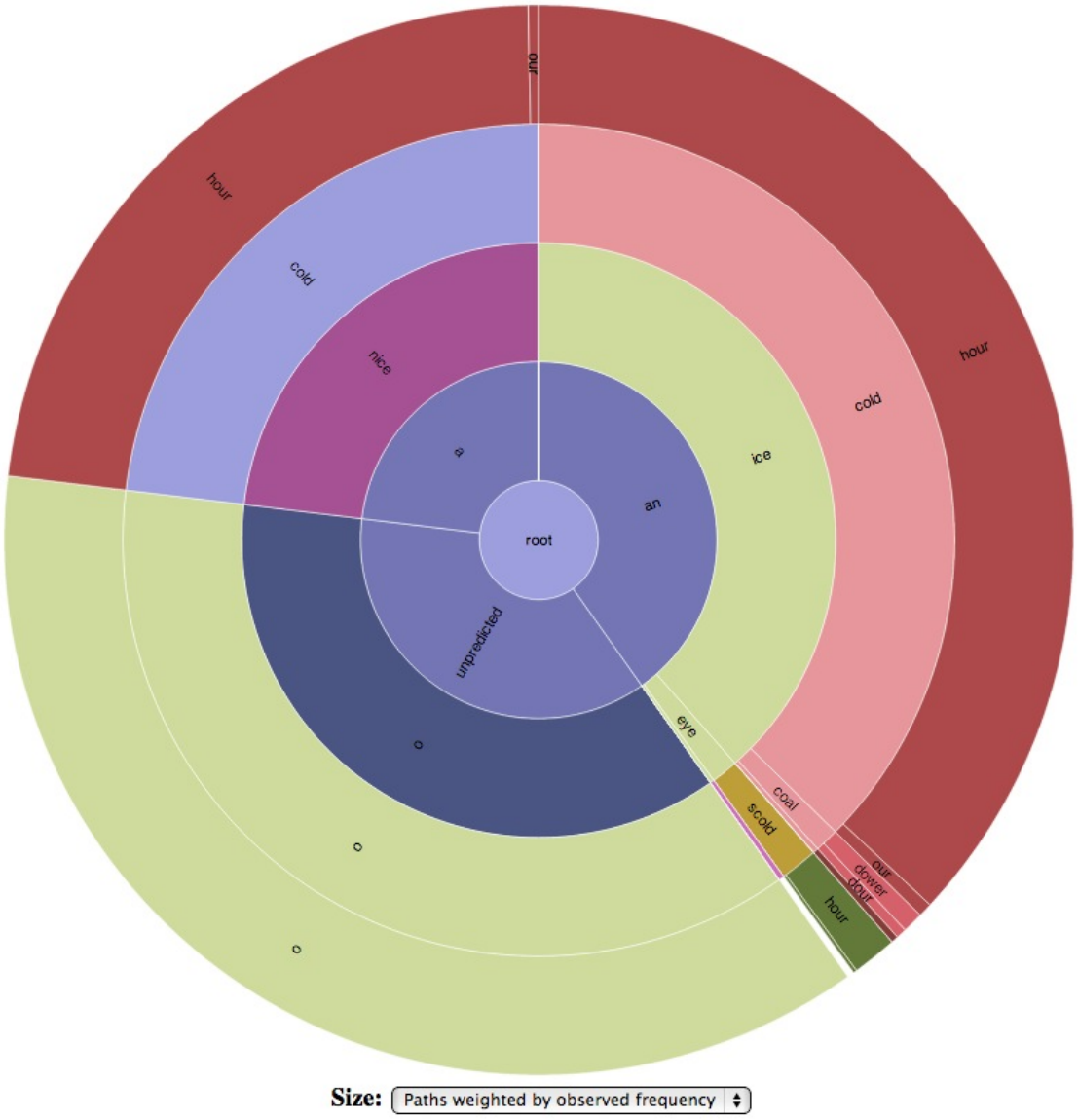


Figure 5.3: Sunburst Chart for A Nice Cold Hour using observed frequencies

burst diagram for the expected distribution of transcribed phrases based upon the UNISYN frequency metric that we calculated. In figure 5.3, we see the sunburst diagram for the transcriptions we observed, with a special slice representing all the transcriptions we did not predict. The unpredicted slice is nearly as large as the slice for "an ice cold hour", which was observed the most often out of all expected transcriptions.

5.2.2 Statistical measurement of expected versus observed transcription frequency

A statistical analysis of the observed dataset versus the expected dataset, using a one-proportion z test, further proves that the calculated UNISYN frequency values were not a good predictor of observed transcription. Using the top two observed transcriptions as our sample population, we take a look at the phrases "a nice cold hour" and "an ice cold hour" . The phrase "a nice cold hour" has a calculated UNISYN freq metric of 7851662 , and had 125 actual transcriptions observed among people living in the United States. The phrase "an ice cold hour" has a calculated UNISYN freq metric of 931028 , and had 191 actual transcriptions observed among people living in the United States. Therefore, the expected population is 8782690 , and the observed population is 316 .

Given those population, the expected population proportion for "a nice cold hour" would be 7851662 ÷ 8782690 , or 0.8934 .

In our user study, we found that 125 people transcribed "a nice cold hour" , and 191 people transcribed "an ice cold hour" , for a ratio of 0.65 to 1, where "a nice cold hour" accounts for 39.56% (p = 0.3956) of the combined count.

Given the observed population proportion of 0.3956 and the expected popu-

lation proportion 0.8934 , we did a one-proportion z test with an α of 0.01 . The z value returned was 18.0971 , meaning that the observed population proportion was 18.0971 standard deviations away from the expected population proportion. When we used this z value to compute a p value, we got a value so low that we were unable find a calculator with enough decimal places to show it without rounding it to zero.

In short, the per-occurance frequency metric predictions derived from UNISYN don't even remotely match the observed data.

The below is here mostly for my edification: (I'll delete it when my edits are all done.)

Givens for Phrase (1) ("a nice cold hour") :

Calculated metric: 7851662

Actual count: x = 125

Givens for Phrase (2) ("an ice cold hour") :

Calculated metric: 931028

Actual count: x = 191

α = significance Level = 0.01

Calculated sum: 7851662 + 931028 = 8782690

Actual sum: 125 + 191 = 316

p = population proportion of "a nice cold hour" occurrences

p = 7851662 ÷ 8782690 = 0.0000

H_o : p = 0.0000 H_a : p $\neq$ 0.0000

Actual: 125 ÷ 316 =

1-proportion z-test

z = 18.0971 std deviations away from expected.

If pvalue ¡ α, reject H_o

pvalue $\approx$ 0 ¡ 0.01

So, reject H_o

5.2.3 Observations on Transcription Count per Recording for each transcribed phrase

The Transcription Count by Recording graphs show how many occurances of a certain transcription were produced from each recording. Each graph represents one transcription, and has bars for each recording, where each bar shows how many times the transcription was observed for that particular recording. The graph also compares the observed incidences of those transcriptions with the expected UNISYN frequency metric. The X axis lists the transcribed phrase. The right Y axis corresponds to the smaller, multi-colored bars. Each bar represents the number of times that a transcribed phrase was observed for that particular recording. The left X axis corresponds to the large blue bar behind the smaller bars. The blue bar represents the calculated UNISYN frequency metric for the transcription.

When you compare the bars from the two y axes, some interesting patterns appear.

An ice cold hour

When looking at figure 5.4, we notice that all but 12 out of the 362 transcriptions of "an ice cold hour" come from recordings of phrases that similarly begin with "an". This suggests the existence of some un-measured value related to pronunciation that makes the theoretically identical phonetic sequences of "an ice cold hour" and "a nice cold hour" be heard as functionally different. Also, note that in figure 5.4, the blue bar representing the UNISYN frequency prediction for the transcribed phrase underpredicts the number of transcriptions from recordings that begin with "an", while doing a fairly good job of predicting transcription incidence for recordings that begin with "a".

A nice cold hour

In figure 5.5, we see that, while most of the transcriptions of "a nice cold hour" came from recordings of phrases that begin with 'a', a not-inconsiderable number came from the recordings for the phrases "an ice cold our" and "an ice-cold our". When taken in regards to the conclusions we drew from figure 5.4 in section 5.2.3, we can conclude that, while listeners appear not to be able to hear an 'a' as an 'an', listeners can, under certain circumstances, hear an 'an' as an 'a'. The blue prediction bar shows that the expected incidence over all recordings was much higher than the observed incidence, and only came close to being correct for the recorded phrases "a nice cold hour" and "a nice coal dower".

In ice cold hour

This chart in figure 5.6 is for the third most common transcription, "in ice cold hour". Transcriptions of this phrase are fairly evenly distributed among

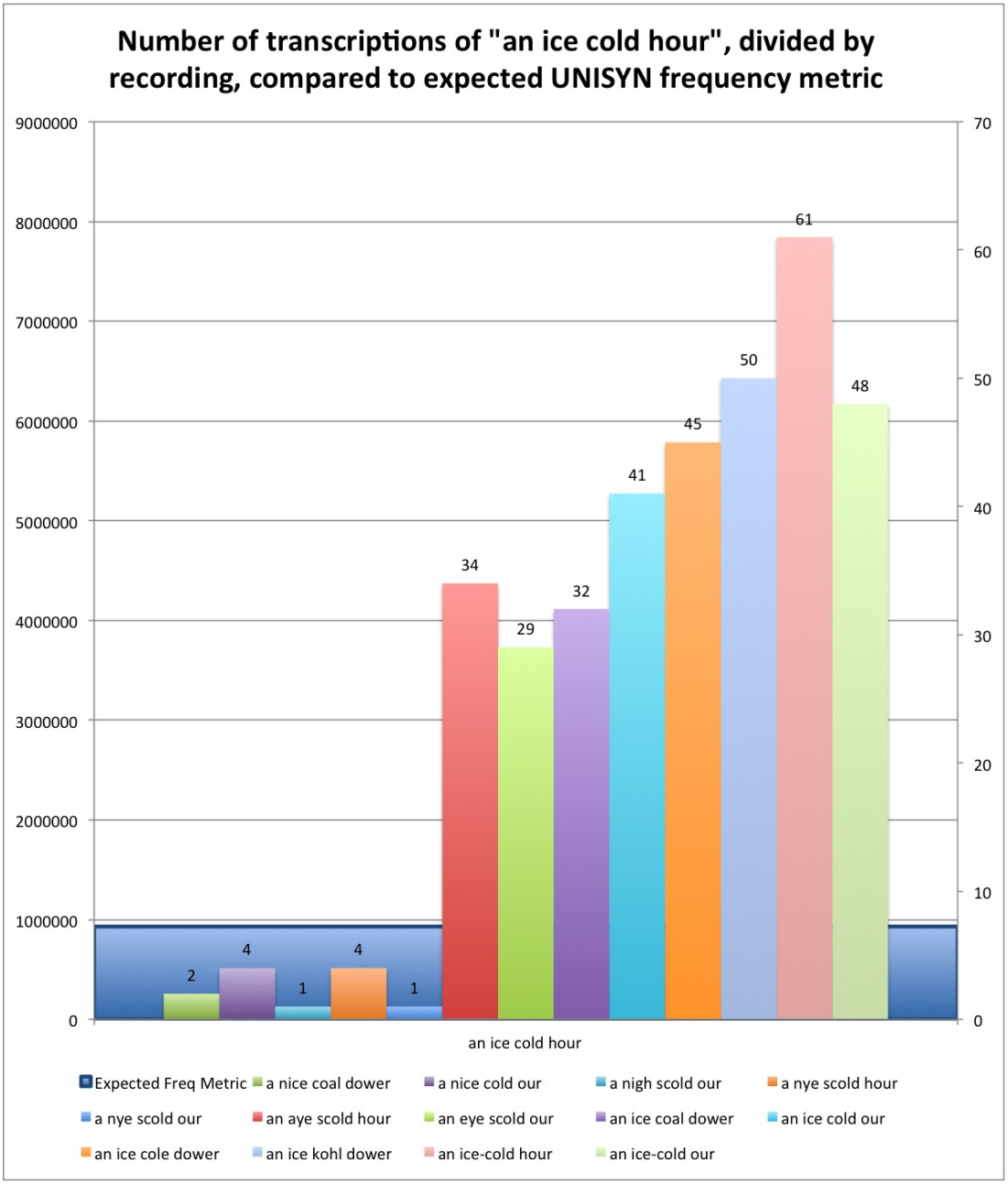


Figure 5.4: This graph represents all transcriptions of the phrase "an ice cold hour", divided into columns based on what recording were transcribed as "an ice cold hour". The large blue bar in the background shows the predicted frequency metric for the phrase in question.

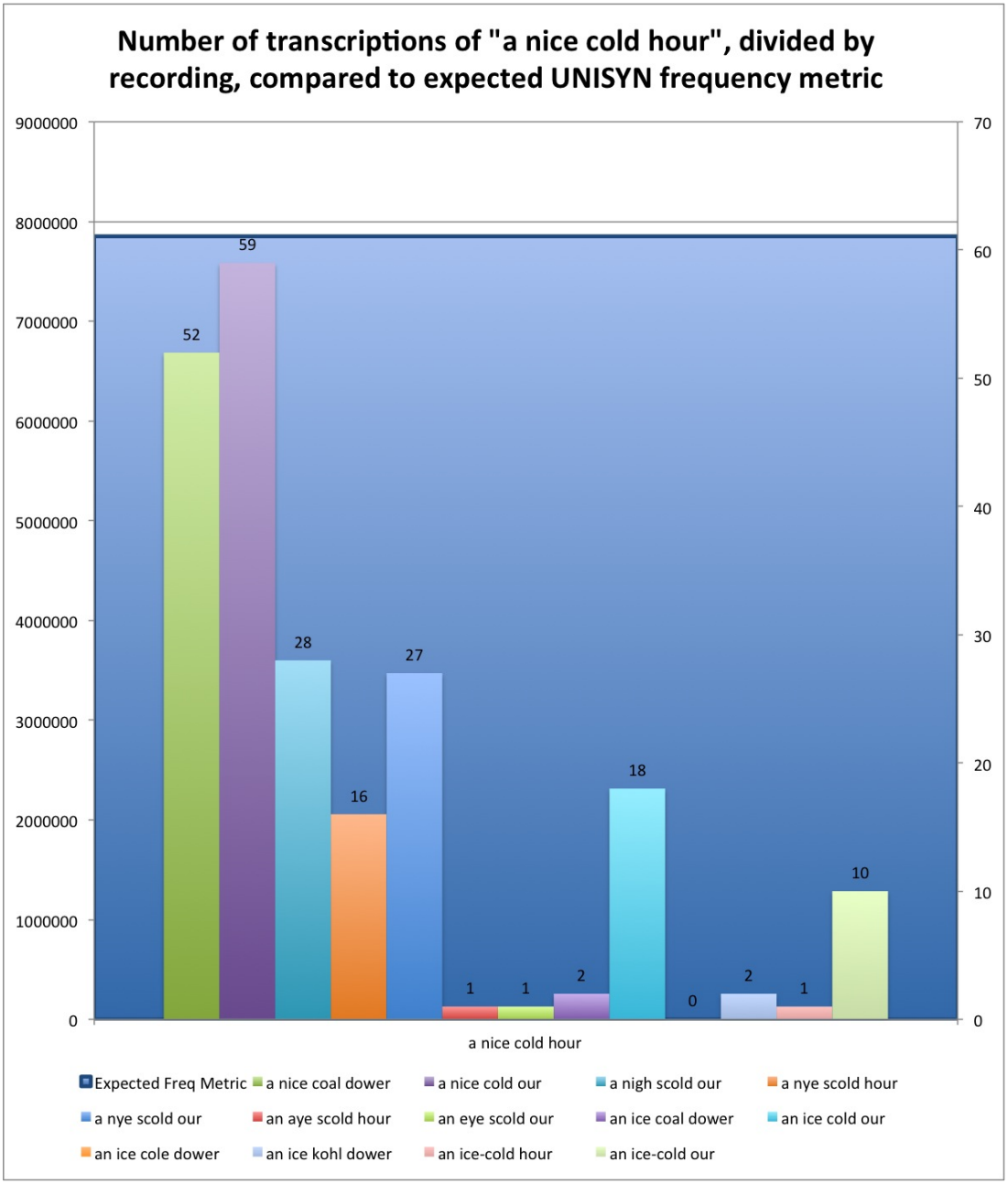


Figure 5.5: This graph represents all transcriptions of the phrase "a nice cold hour", divided into columns based on what recording were transcribed as "a nice cold hour". The large blue bar in the background shows the predicted frequency metric for the phrase in question.

the recordings, though there is a spike of transcriptions on "an ice-cold hour". Additionally, similar to earlier observations, the transcriptions occur a lot less frequently than the UNISYN frequency metric bar suggests they should.

A nice gold hour

The chart in figure 5.7 for the transcriptions of "a nice gold hour" shows that the source recordings of those transcriptions are very specific: only the recordings of "a nye scold our", "a nye scold hour", and "a nigh scold our" produce the 'g'/'c' substitution. In fact, all of our transcriptions that involved the word "gold" arose from these recordings. This suggests a relationship between the phonemes `s k` and `g`, and warrants further investigation, as suggested later in section 6.2. As shown by the lack of a background blue bar in this diagram, this transcription was not predicted by our oronym generation algorithm, due to the aforementioned phoneme swapping.

An ice cold dower

Figure 5.8 shows the transcriptions for the phrase "an ice cold dower". This phrase features repeated-phoneme auto-deletion or auto-insertion. Repeated-phoneme auto-deletion or -insertion occurs when two identical and adjacent phonemes are blurred into one sound. This can result in the listener putting two phonemes where only one exists (as in this case), or putting one phoneme where two exist. This phenomenon, while known to us, is outside the scope of the current project, and suggested for future work. Due to this limit in scope, this transcription was not predicted by our oronym generation algorithm, as shown by the lack of a background blue frequency prediction bar in figure 5.8.

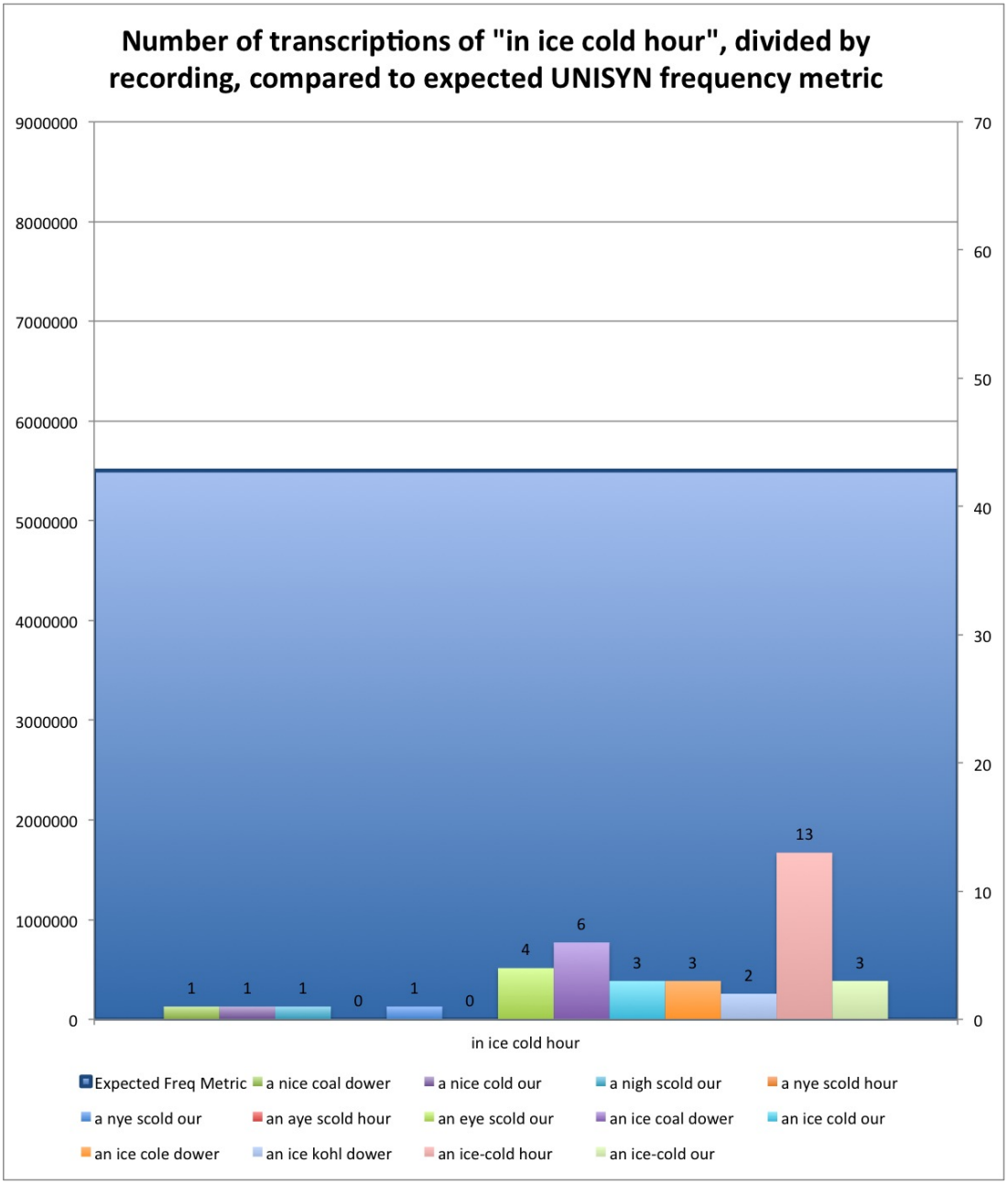


Figure 5.6: This graph represents all transcriptions of the phrase "in ice cold hour", divided into columns based on what recording were transcribed as "in ice cold hour". The large blue bar in the background shows the predicted frequency metric for the phrase in question.

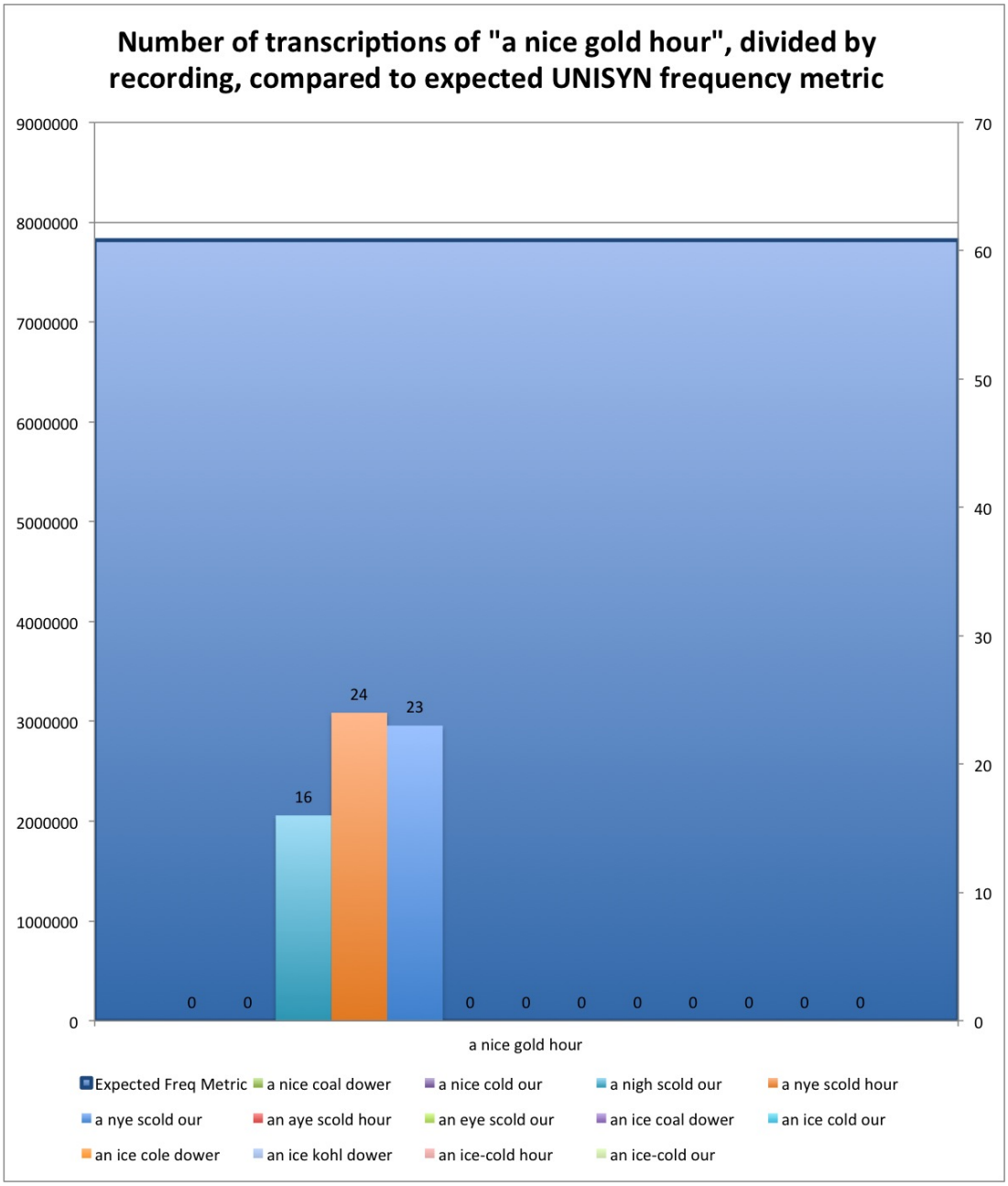


Figure 5.7: This graph represents all transcriptions of the phrase "a nice gold hour", divided into columns based on what recording were transcribed as "a nice gold hour". The large blue bar in the background shows the predicted frequency metric for the phrase in question.

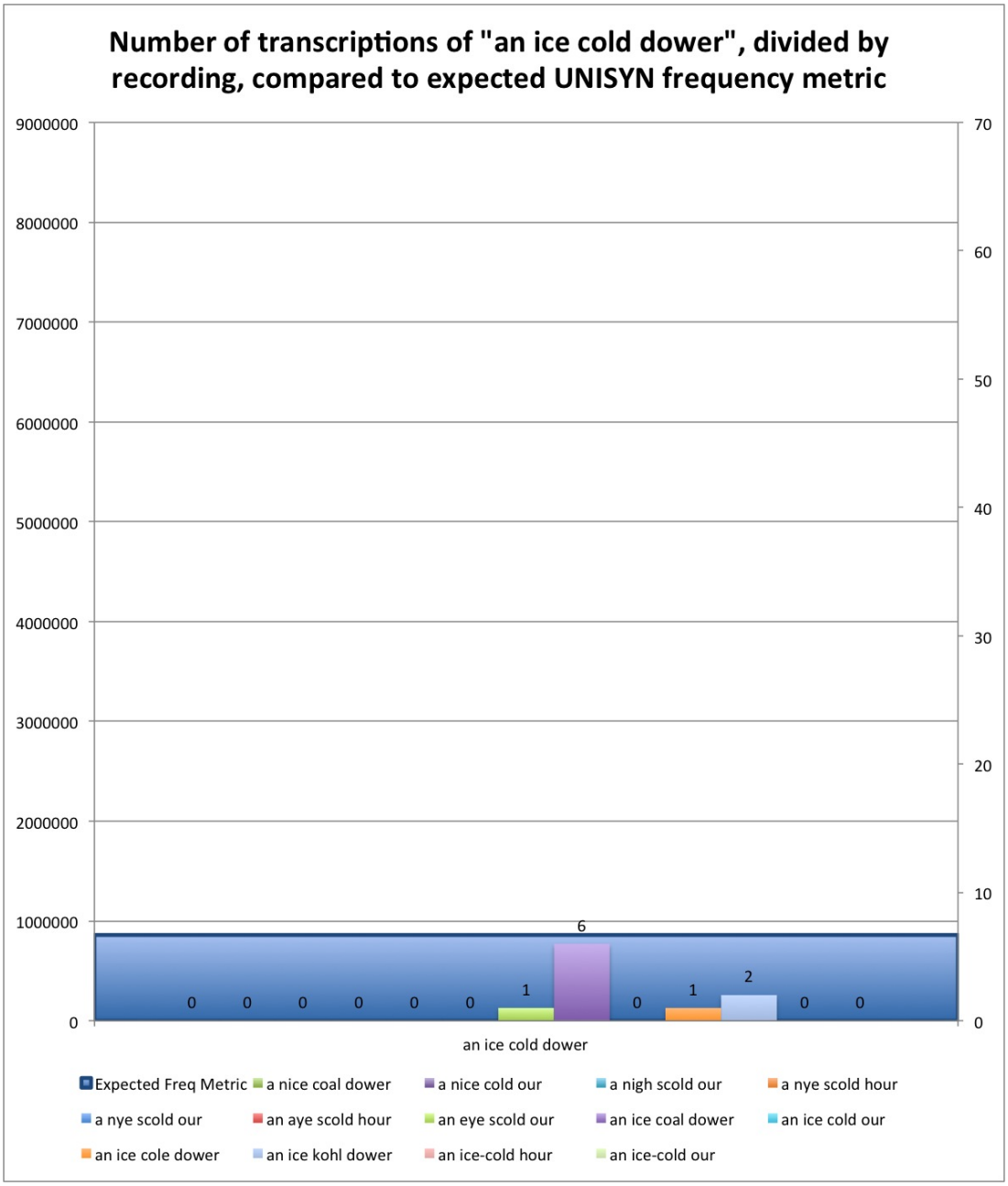


Figure 5.8: This graph represents all transcriptions of the phrase "a nice cold dower", divided into columns based on what recording were transcribed as "a nice cold dower". The large blue bar in the background shows the predicted frequency metric for the phrase in question.

An eye scold hour

The chart for "an eye scold hour", shown in figure 5.9, is primarily interesting in that it appears more or less deterministically interpretable. The only recordings that resulted in this phrase were those for "an eye scold hour" or "an aye scold hour". We hypothesize that this has something to do with word emphases, and suggest investigating this for future work. Interestingly enough, the predicted frequency value (scaled) was fairly close to the actual occurance for that phrase.

An ice coal dower

The chart for "an ice coal dower" in figure 5.10 is notable in that all its transcriptions came from recordings that began in "an" and ended in "dower", like the phrase itself does. As in 5.9, we suggest that the near-deterministic interpretation has something to do with emphases, and suggest investigating this for future work. The UNISYN frequency metric once again predicted higher expected incidences than were actually observed for this transcription.

5.2.4 Transcription Breakdown By Country

When comparing transcriptions from countries where English is the dominant language (as shown in figure 5.11) to those from countries where it is not (as shown in figure 5.12), we can make some interesting observations.

1. The most common transcription for both is "an ice cold hour", with 36 % native-speaker transcriptions (218) and 24 % non-native (134).

2. The second most common transcription is also the same for both ("a nice cold hour"), but it accounts for a larger percentage of the non-native pie (

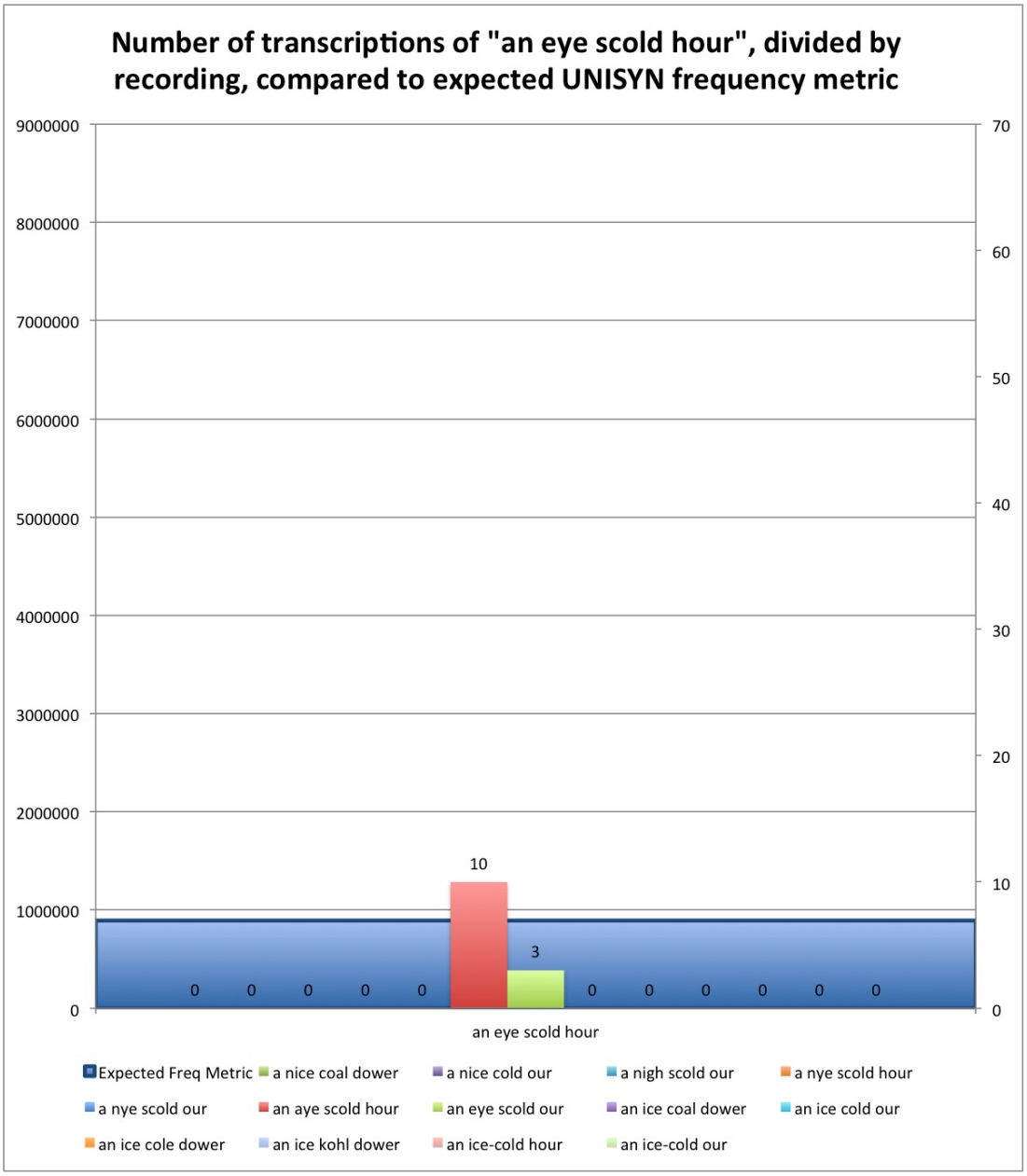


Figure 5.9: This graph represents all transcriptions of the phrase "an eye scold hour", divided into columns based on what recording were transcribed as "an eye scold hour". The large blue bar in the background shows the predicted frequency metric for the phrase in question.

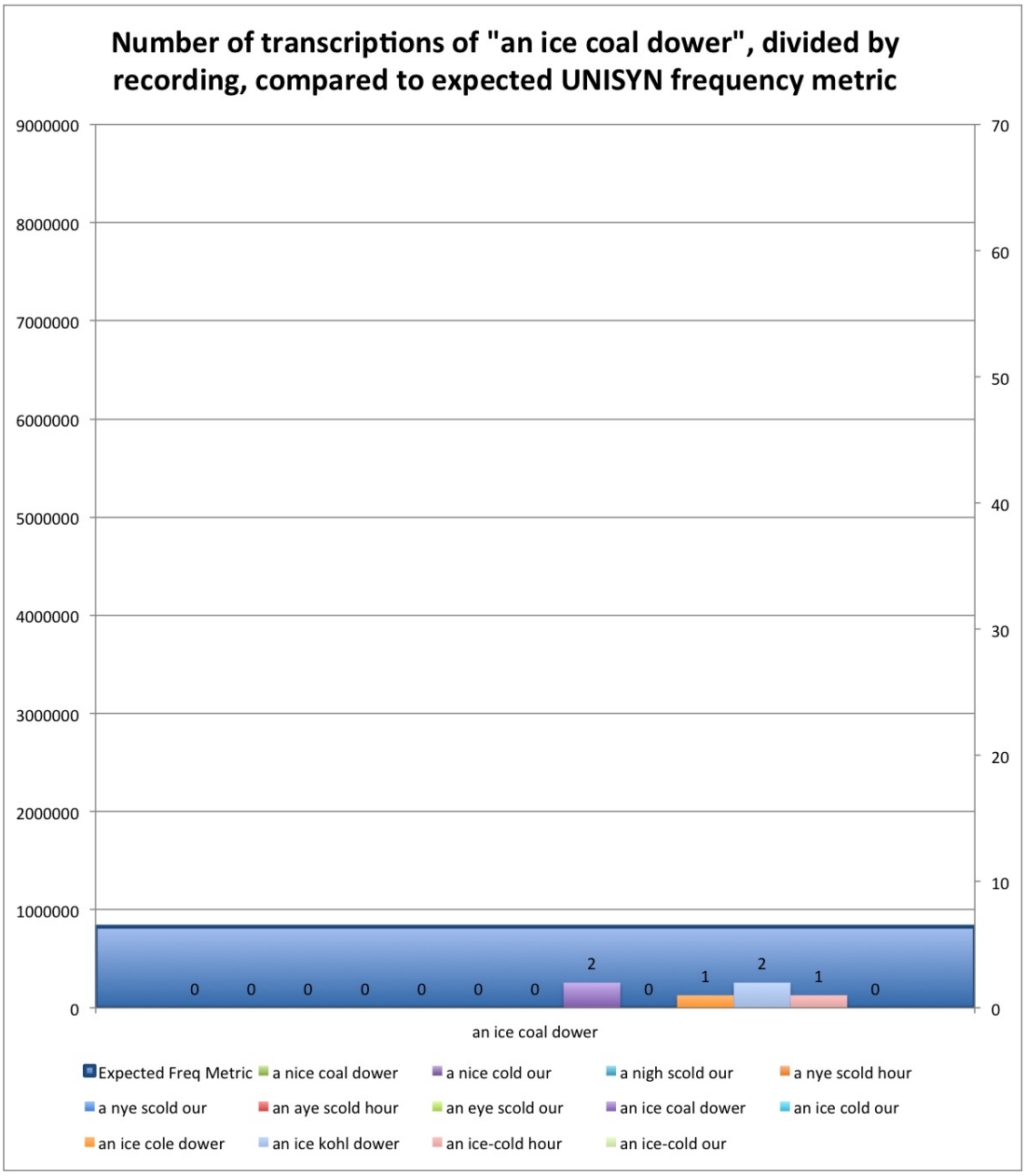


Figure 5.10: This graph represents all transcriptions of the phrase "an ice coal dower", divided into columns based on what recording were transcribed as "an ice coal dower". The large blue bar in the background shows the predicted frequency metric for the phrase in question.

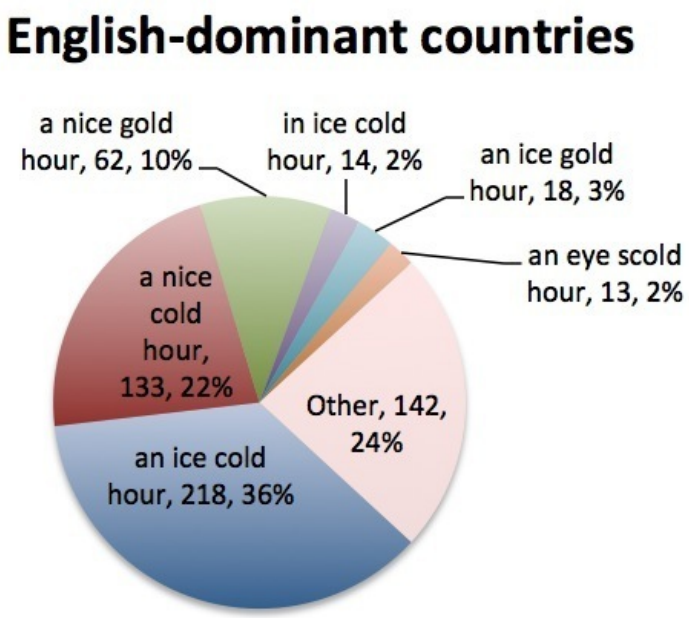


Figure 5.11: Pie Chart of transcriptions from countries that are primarily English-speaking.

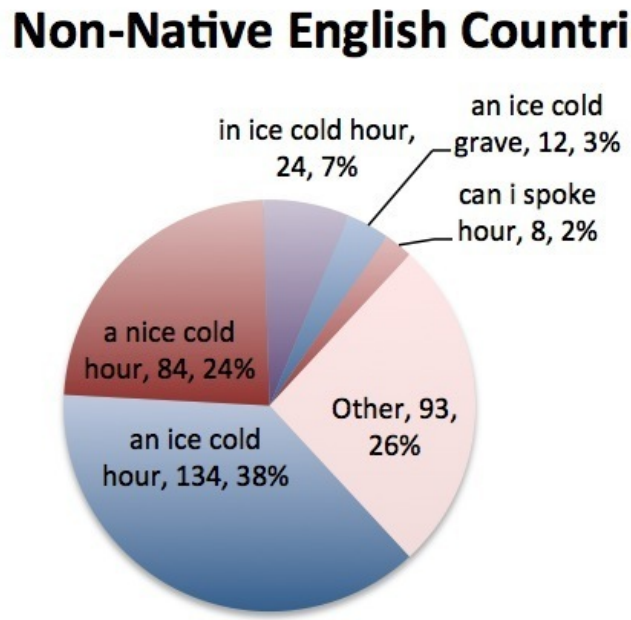


Figure 5.12: Pie Chart of transcriptions from countries that are Non-native English speakers.

24 % compared to the native 22 %).

3. The third most common transcription differs for native and non-native speakers.

   Native speakers transcribed “a nice gold hour” 62 times, accounting for 10 % of all native transcriptions. In comparison, only 1 non-native speaker transcribed that phrase, for a measley 0.2 % of total non-native transcriptions.

   This brings up an interesting data point—the third most popular transcription for native speakers barely shows up at all for non-native transcribers. There may be something about common phoneme substitution that native speakers pick up on that non-natives do not; specifically, a cold/gold merger.

   The third most common transcription for non-native speakers is “in ice cold hour”, which was transcribed 24 times, and makes up 7 % of non-native transcriptions. This phrase was the fifth most common transcription for native speakers, with 14 transcriptions making up 2 % of total transcriptions.

4. The fourth most common transcription for native speakers was “an ice gold hour”, getting 3 % of the total with 18 transcriptions (by 14 unique transcribers). This phrase was not transcribed by any non-native speakers. This exhibits the same phone/phoneme substitution that we saw with “a nice gold hour”.

   The fourth most common phrase for non-native speakers, “an ice cold grave”, was only transcribed by one unique worker, and as such, is not going to be taken into serious consideration. The fifth most common phrase,

"can I spoke hour", was also only transcribed by one unique worker, and so also cannot be taken into serious consideration.

Chapter 6

Future Work

6.1 Deficiencies in The Oronyminator

In some cases, our expected, UNISYN-derived phrase-frequency metric did not accurately line up with the observed transcription frequencies from our user studies. We believe that there are several possible reasons for this.

6.1.1 Frequency Validity

Our frequency source data from UNISYN ended up being less than statisfactory, due to several factors, deliniated below.

Corpus Composition deficiencies

The lack of quality phonemic frequency data is a known defeciency in our source dictionary, UNISYN. According to the authors of the UNISYN lexicon documentation:

> It should be noted that the frequency field, as it was obtained from simple word lists, is *not particularly reliable* (emphasis mine).[25]

The UNISYN frequency count is based upon a large but not exhaustive corpus of text. It has some particularly glaring deficiencies in the medical arena. We find this frustrating, because knowledge about common medical mondegreens could be used to prevent mistakes in patient's treatment plans[21]. Also, it meant that the word "colitis" wasn't in our dictionary, and we therefore couldn't use the example "the girl with colitis goes by"/"the girl with kaleidescope eyes".

Homograph Differentiation

Additionally, our source frequency data cannot and does not distinguish between words that may be homographs (that is, words that sound different but are spelled the same). This makes our program improperly weight some phrases over others.

For example, take the words for the animals "bucks" and "does". "Bucks" has a frequency of 1133, and "does" has a frequency of 508386. For comparison, "deer" has a frequency of 1896. You can see the relative scale of these in figure 6.1. It seems highly unlikely that the male and female labels for a species would be as or more common than the actual name of the species, given that we don't see this for sheep (sheep at 13572 , ewe at 186 ,and ram at 681) or horses (horse at 27559 , mare at 1055 , and stallion at 644). What is much more likely is that "bucks" is getting extra hits through its meaning as a slang synonym for dollars (dollars at 8927), and "does" is getting most of its frequency count for the 3rd person present tense of the verb "to do". That seems very likely, given that the frequency for the singular "doe" is only 1077.

Figure 6.1: Bubble Chart comparison of Frequency for deer, does, and bucks

Future work on our project would benefit from using a dictionary with some way of distinguishing homographs when counting frequency.

Frequency Dictionary Tallying Methods

In the future, we'd also like to use word frequency values from a dictionary that takes a larger, more-diverse dataset into its frequency count, such as the frequency lists from the Corpus of Contemporary American English[3]. The COCA corpus is entirely focused on word frequency, and as such, does not contain any phonetic data. However, it contains several different ways of determining frequency of words that overcome some of the shortcomings we ran into trying to compare the semantically-identical words 'a' and 'an'. 'A' is found much for frequently than 'an', but both are just as common. In the UNISYN dictionary, we only have contextless frequency counts. In the COCA frequency dictionary, they keep two types of counts: one for how many times the word has been found total, and one for how many documents the word has been found in. This way, even

though ‘a’ is found almost seven times as often than ‘a’ overall, we know that they’re equally-familiar words, because they are both found in approximately 160k corpus entries[23].

As a proof of concept, we created a sunburst diagram for “an ice cold hour” using COCA by-document frequency values (shown in figure 6.2), to compare it against the sunburst diagram created using UNISYN frequency values(shown in figure 6.3). The COCA-based sunburst, while still inaccurate, at least has a ratio of phrases beginning with “a” to “an” that is closer to the actual observed ratio (which can be seen back in figure 5.3).

A statistical analysisof the observed dataset frequencies versus a COCA-derived frequency dataset, using a one-proportion z test, further proves that the COCA frequency values are a better match for the observed data than the UNISYN-derived frequencies. Again using the top two observed transcriptions as our sample population, we take a look at the phrases “a nice cold hour” and “an ice cold hour” . The phrase “a nice cold hour” has a calculated COCA freq metric of 247719 , and had 125 actual transcriptions observed among people living in the United States. The “an ice cold hour” has a calculated COCA freq metric of 227405 , and had 191 actual transcriptions observed among people living in the United States. Therefore, the expected population is 475124 , and the observed population is 316 .

Given those population, the expected population proportion for “a nice cold hour” would be 247719 ÷ 475124 , or 0.4793 .

In our user study, we found that 125 people transcribed “a nice cold hour” , and 191 people transcribed “an ice cold hour” , for a ratio of 0.65 to 1, where “a nice cold hour” accounts for 39.56% (p = 0.3956) of the combined count.

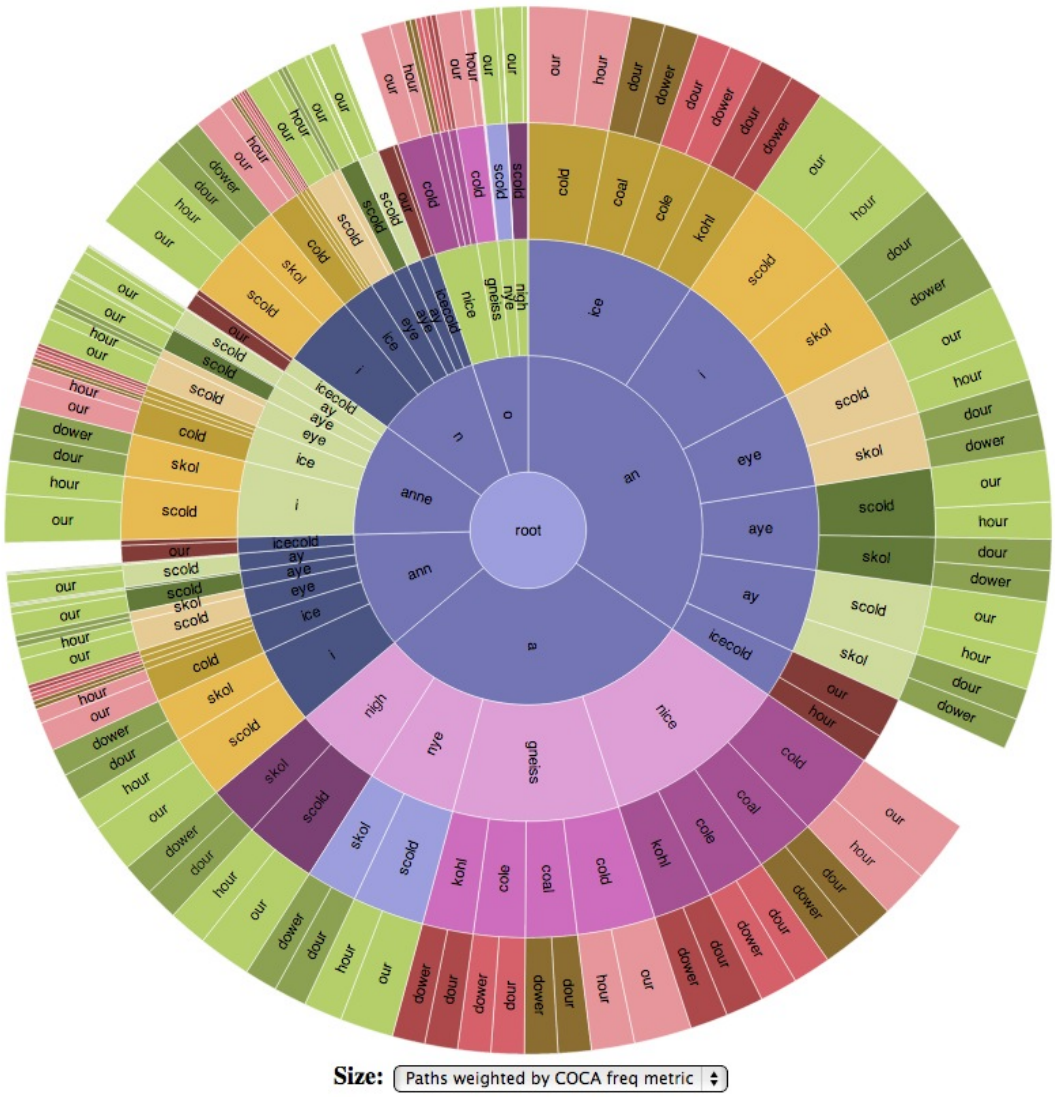


Figure 6.2: Sunburst diagram for "an ice cold hour" using COCA by-document freq metric

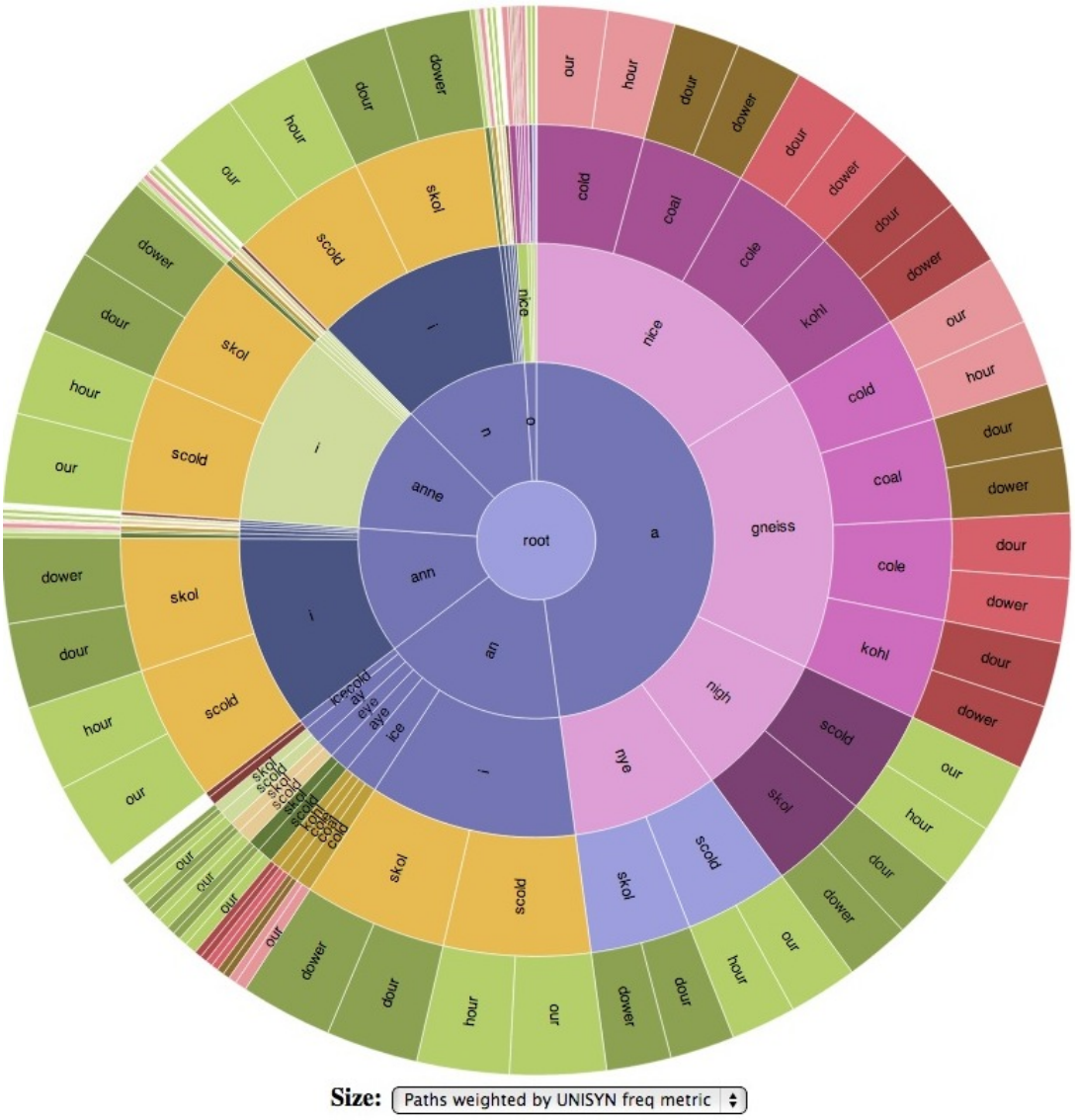


Figure 6.3: Sunburst diagram for "an ice cold hour" using UNISYN freq metric

Given the observed population proportion of 0.3956 and the expected population proportion 0.4793 , we did a one-proportion z test with an α of 0.01 . The z value returned was 3.0428 , meaning that the observed population proportion was 3.0428 standard deviations away from the expected population proportion. When we used this z value to compute a p value, we were left with a pvalue of 0.0023 , which is greater than our α of 0.01 . Therefore, there is approximately a 0.01% chance that the observed data could match the COCA predictions, but it's still not incredibly likely. It's significantly more likely than the UNISYN predictions being correct, however.

6.1.2 Higher-order frequency data

Right now, our program only takes into account the frequency of standalone words, without taking their context into consideration. In the future, we'd like to integrate n-grams into our program. N-grams are a probabilistic model of predicting the next item that will follow in a sequence, based upon frequencies of how often those N items occur in sequence in a corpus of text[12]. A word-level 4-gram, for example, would be a series of four words. Here are some 4-gram phrases, along with counts of how often they occur, from the Google Ngram corpus:

serve as the informational 41
serve as the infrastructure 500
serve as the initial 5331
serve as the initiating 125
serve as the initiation 63
serve as the initiator 81
serve as the injector 56
serve as the inlet 41
serve as the inner 87
serve as the input 1323

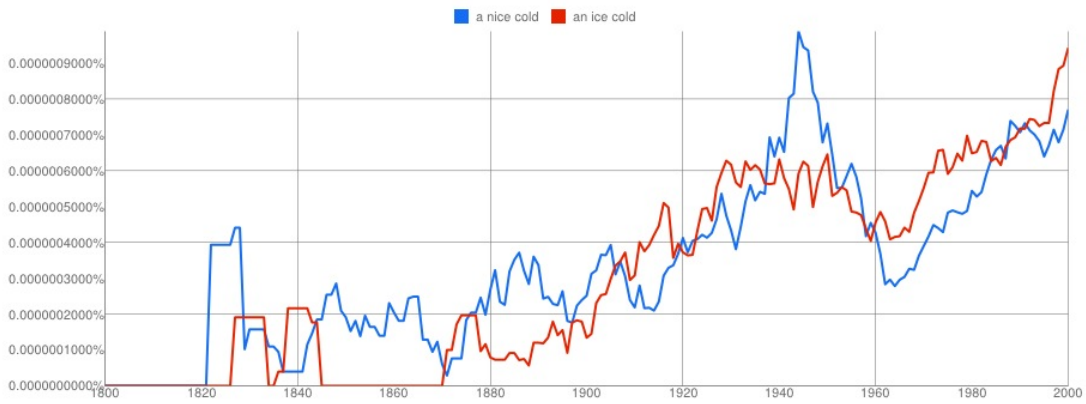


Figure 6.4: Historical N-gram data comparing the three-grams "a nice cold" and "an ice cold"

[1]

To give an example of what that might look like, we looked up historical n-gram occurence percentage of n-grams contained in oronyms of our main test phrase "an ice cold hour". As seen in figure 6.4, comparing the 3-grams "a nice cold" and "an ice cold" results in a fairly even split, though the latter phrase is slightly more likely to occur in modern-day settings. When we look at the 2-grams of that 3-gram, we see more interesting trends. In figure 6.5, we see that "a nice" is consistently more frequently found in text than "an ice" is. However, when we compare the 2-grams "ice cold" and "nice cold", as we do in figure 6.6, we see that the phrase "ice cold" is leaps and bounds more likely to be encountered in everyday language.

Though we are happy with our findings, we believe that we could create even better likelihood metrics with the integration of several different orders of n-grams, and would suggest this for future work. However, if the final purpose of the oronyminator ends up being in the song lyric domain, everyday-usage

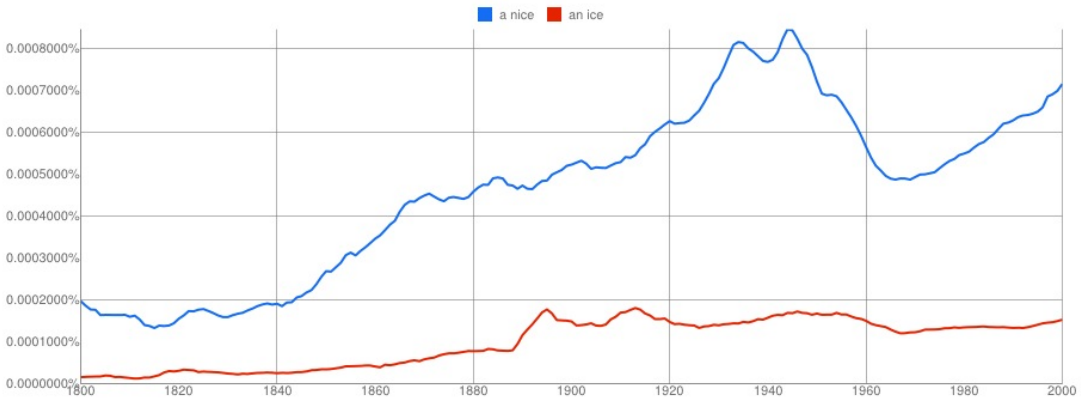


Figure 6.5: Historical N-gram data comparing the two-grams "a nice" and "an ice"

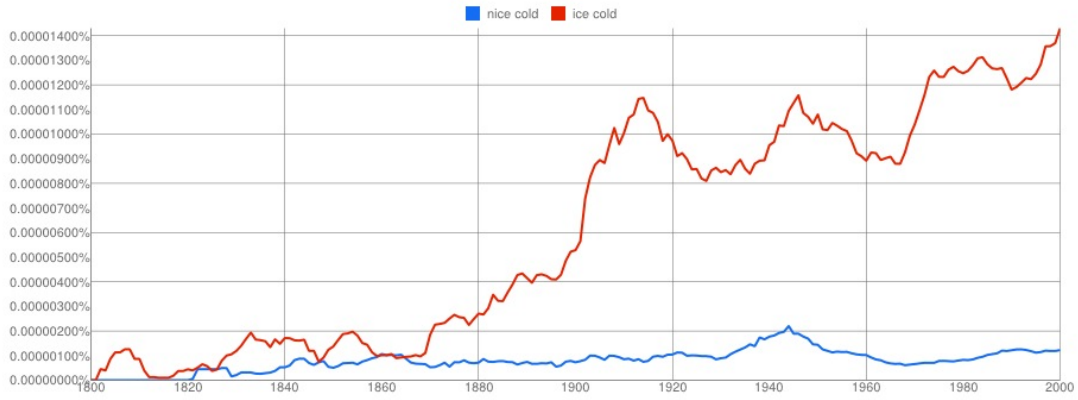


Figure 6.6: Historical N-gram data comparing the two-grams "nice cold" and "ice cold"

syntactical predictability may not be particularly relevant, due to the differences in vocabulary and grammar found in song lyrics versus regular prose.

6.2 Phoneme swapping

Often when speaking, humans substitute easier-to-say phones for more time-intensive phones. One of the main ways that this substitution occurs is through voiced/voiceless pairs. To voice a phone means to cause the vocal chords to vibrate. Voiced phones are singable, whereas voiceless phones are not. Voiceless phones are like a hiss, and simply direct streams of escaping air. Most consonant phonemes are part of a voice/voiceless pair, such as 't' and 'd' (The word "pretty", when spoken quickly, often uses a 'd' sound instead of a 't' sound, because the phoneme for 'd' is easier to say). Phones are paired when the only differences between their pronuciation is the voicing, aka, when their manner of articulation (i.e. their manner of directing air during the sound), mouth end position, and mouth start position are the same (To view all phones in the SAMPA alphabet, along with enough information to determine whether they are pairs, see table **??**). In our project, we came across an example of phoneme swapping in the "cold"/"gold" transcriptions, which we went over in figure 5.7 and section 5.2.3. For future work, we suggest looking into phoneme swap pairings, and integrating the findings into the existing algorithm.

6.3 Melody Matcher master project

MisheardMe Oronyminator is a part of the greater Melody Matcher suite. Melody Matcher is a semi-automated music composition support program. It

analyzes English lyrics along with a melody, and alerts the composer of the locations in the song where the lyrics are not deterministically understandable. Basically, it's grammar- and spell-check for songs.

Melody Matcher aims to replicate the human ability to identify lyrics in a song that are easily misheard.

6.3.1 Target Audience and Goals

This program is to be used as a compositional aid by anyone who wants to write songs and make them sound good, technically. It should allow the song writer to focus on more subjective criteria of what makes a song "good", because it will make the structural rules of lyric composition immediately apparent.

Our hope for this project is that it will be useful to burgeoning songwriters, who have the creative spark to make wonderfully poetic lyrics, but lack the "ear" to match their lyrics successfully to music. It should be particularly helpful to songwriters who place a high emphasis on understandability of lyrics (such as parody song writers, or lyricists for musical theater).

Additionally, Melody Matcher will be useful for songwriters for whom English is a second language. While they may be a master lyricist in their native language, writing lyrics in English can be a particular challenge, since so much of lyric-writing is dependent upon knowing the cadence of the language you're writing lyrics in, and since English has no easily-discernible rules for emphasis placement in words.

While MisheardMe Oronyminator only takes into account phonetics and fre-

quencies, Melody Matcher analyzes the intelligibility of song lyrics by investigating several additional root causes:

- Lyric/Music emphasis mismatch, due to:
  - Note intervals
  - Phrase emphases
  - Word emphases
- Word "cramming", due to:
  - Syllable lengths that exceed that of note length
  - Mouth movement delta time intervals
- Word misidentification, due to:
  - Altered pronunciation of words
  - Phone similarity
    - Voicing (voiced vs. voiceless)
    - Beginning/end mouth positions
    - Type (Plosive, Fricative, affricate, nasal, lateral, approximant, semivowel)

The fully-implemented Melody Matcher program will eventually take into account all of these causes of unintelligibility.

Chapter 7

Conclusion

In this paper, we have demonstrated MisheardMe Oronyminator, a computer program which takes in textual phrases in English, determines all oronyms for that phrase and then visualizes them using associated frequency information to indicate the likelihood of interpretation. We have demonstrated all four major functional parts: our custom phonetic dictionary, our command-line oronym generator, our OpenGL oronym-parse-tree visualization generator, and our Protovis sunburst diagrams. Our custom phonetic dictionary has some inconsistencies in word frenquency, due to the UNISYN source dictionary's frequency values not being generated from a well-sampled corpus. However, our program has no major structural flaws, and can be succesfully used for phrase with words with frequencies on the same order of magnitude. Our command-line oronym generator successfully generates all oronyms that are exact phonetic matches for an orthographic phrase. The user studies that we did supported our generated phrases, if not our frequency metrics. Our oronym visualizations had two goals: one, visually represent the likelihood of each oronym interpretation, visualized by scaling branches or arcs by phrase frequency values; and two, to exhibit orthographic

phrases that may not have any exact oronyms, but have many dead-end, partial oronyms that could cause ambiguity. Our visualizations successfully accomplish both of those goals.

Appendix A

Implementation Details

Appendix A

Oronym Tables

Table A.1: All Oronyms for 'A Nice Cold Hour' with frequency values

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
on i scold our	13185760	on	2774243	i	9937877	scold	217	our	473423
on i scold hour	12784150	on	2774243	i	9937877	scold	217	hour	71813
on i skol dour	12712244	on	2774243	i	9937877	skol	5	dour	119
on i skol dower	12712217	on	2774243	i	9937877	skol	5	dower	92
an i scold our	11205686	an	794169	i	9937877	scold	217	our	473423
an i scold hour	10804076	an	794169	i	9937877	scold	217	hour	71813
an i skol dour	10732170	an	794169	i	9937877	skol	5	dour	119
an i skol dower	10732143	an	794169	i	9937877	skol	5	dower	92
'n' i scold our	10411517	'n'	0	i	9937877	scold	217	our	473423
'n' i scold hour	10009907	'n'	0	i	9937877	scold	217	hour	71813
'n' i skol dour	9938001	'n'	0	i	9937877	skol	5	dour	119
'n' i skol dower	9937974	'n'	0	i	9937877	skol	5	dower	92

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
a nice cold our	8253272	a	7536297	nice	190708	cold	52844	our	473423
a niece cold our	8064257	a	7536297	niece	1693	cold	52844	our	473423
a gneiss cold our	8062585	a	7536297	gneiss	21	cold	52844	our	473423
a ne scold our	8017040	a	7536297	ne	7103	scold	217	our	473423
a knee scold our	8016076	a	7536297	knee	6139	scold	217	our	473423
a nigh scold our	8011331	a	7536297	nigh	1394	scold	217	our	473423
a nye scold our	8009974	a	7536297	nye	37	scold	217	our	473423
a nice cold hour	7851662	a	7536297	nice	190708	cold	52844	hour	71813
a nice coal dour	7747572	a	7536297	nice	190708	coal	20448	dour	119
a nice coal dower	7747545	a	7536297	nice	190708	coal	20448	dower	92
a nice cole dour	7729197	a	7536297	nice	190708	cole	2073	dour	119
a nice cole dower	7729170	a	7536297	nice	190708	cole	2073	dower	92
a nice kohl dour	7728036	a	7536297	nice	190708	kohl	912	dour	119
a nice kohl dower	7728009	a	7536297	nice	190708	kohl	912	dower	92
a niece cold hour	7662647	a	7536297	niece	1693	cold	52844	hour	71813
a gneiss cold hour	7660975	a	7536297	gneiss	21	cold	52844	hour	71813
a ne scold hour	7615430	a	7536297	ne	7103	scold	217	hour	71813
a knee scold hour	7614466	a	7536297	knee	6139	scold	217	hour	71813
a nigh scold hour	7609721	a	7536297	nigh	1394	scold	217	hour	71813
a nye scold hour	7608364	a	7536297	nye	37	scold	217	hour	71813
a niece coal dour	7558557	a	7536297	niece	1693	coal	20448	dour	119
a niece coal dower	7558530	a	7536297	niece	1693	coal	20448	dower	92

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
a gneiss coal dour	7556885	a	7536297	gneiss	21	coal	20448	dour	119
a gneiss coal dower	7556858	a	7536297	gneiss	21	coal	20448	dower	92
a ne skol dour	7543524	a	7536297	ne	7103	skol	5	dour	119
a ne skol dower	7543497	a	7536297	ne	7103	skol	5	dower	92
a knee skol dour	7542560	a	7536297	knee	6139	skol	5	dour	119
a knee skol dower	7542533	a	7536297	knee	6139	skol	5	dower	92
a niece cole dour	7540182	a	7536297	niece	1693	cole	2073	dour	119
a niece cole dower	7540155	a	7536297	niece	1693	cole	2073	dower	92
a niece kohl dour	7539021	a	7536297	niece	1693	kohl	912	dour	119
a niece kohl dower	7538994	a	7536297	niece	1693	kohl	912	dower	92
a gneiss cole dour	7538510	a	7536297	gneiss	21	cole	2073	dour	119
a gneiss cole dower	7538483	a	7536297	gneiss	21	cole	2073	dower	92
a nigh skol dour	7537815	a	7536297	nigh	1394	skol	5	dour	119
a nigh skol dower	7537788	a	7536297	nigh	1394	skol	5	dower	92
a gneiss kohl dour	7537349	a	7536297	gneiss	21	kohl	912	dour	119
a gneiss kohl dower	7537322	a	7536297	gneiss	21	kohl	912	dower	92
a nye skol dour	7536458	a	7536297	nye	37	skol	5	dour	119
a nye skol dower	7536431	a	7536297	nye	37	skol	5	dower	92
on aye scold our	3378386	on	2774243	aye	130503	scold	217	our	473423
on e scold our	3356846	on	2774243	e	108963	scold	217	our	473423
on ice cold our	3312712	on	2774243	ice	12202	cold	52844	our	473423
on eye scold our	3274633	on	2774243	eye	26750	scold	217	our	473423

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
on ay scold our	3254516	on	2774243	ay	6633	scold	217	our	473423
on ice-cold our	3247715	on	2774243	ice-cold	49	our	473423		
on aye scold hour	2976776	on	2774243	aye	130503	scold	217	hour	71813
on e scold hour	2955236	on	2774243	e	108963	scold	217	hour	71813
on ice cold hour	2911102	on	2774243	ice	12202	cold	52844	hour	71813
on aye skol dour	2904870	on	2774243	aye	130503	skol	5	dour	119
on aye skol dower	2904843	on	2774243	aye	130503	skol	5	dower	92
on e skol dour	2883330	on	2774243	e	108963	skol	5	dour	119
on e skol dower	2883303	on	2774243	e	108963	skol	5	dower	92
on eye scold hour	2873023	on	2774243	eye	26750	scold	217	hour	71813
on ay scold hour	2852906	on	2774243	ay	6633	scold	217	hour	71813
on ice-cold hour	2846105	on	2774243	ice-cold	49	hour	71813		
on ice coal dour	2807012	on	2774243	ice	12202	coal	20448	dour	119
on ice coal dower	2806985	on	2774243	ice	12202	coal	20448	dower	92
on eye skol dour	2801117	on	2774243	eye	26750	skol	5	dour	119
on eye skol dower	2801090	on	2774243	eye	26750	skol	5	dower	92
on ice cole dour	2788637	on	2774243	ice	12202	cole	2073	dour	119
on ice cole dower	2788610	on	2774243	ice	12202	cole	2073	dower	92
on ice kohl dour	2787476	on	2774243	ice	12202	kohl	912	dour	119
on ice kohl dower	2787449	on	2774243	ice	12202	kohl	912	dower	92
on ay skol dour	2781000	on	2774243	ay	6633	skol	5	dour	119
on ay skol dower	2780973	on	2774243	ay	6633	skol	5	dower	92

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
an aye scold our	1398312	an	794169	aye	130503	scold	217	our	473423
an e scold our	1376772	an	794169	e	108963	scold	217	our	473423
an ice cold our	1332638	an	794169	ice	12202	cold	52844	our	473423
an eye scold our	1294559	an	794169	eye	26750	scold	217	our	473423
an ay scold our	1274442	an	794169	ay	6633	scold	217	our	473423
an ice-cold our	1267641	an	794169	ice-cold	49	our	473423		
an aye scold hour	996702	an	794169	aye	130503	scold	217	hour	71813
an e scold hour	975162	an	794169	e	108963	scold	217	hour	71813
ah nice cold our	946271	ah	229296	nice	190708	cold	52844	our	473423
an ice cold hour	931028	an	794169	ice	12202	cold	52844	hour	71813
an aye skol dour	924796	an	794169	aye	130503	skol	5	dour	119
an aye skol dower	924769	an	794169	aye	130503	skol	5	dower	92
an e skol dour	903256	an	794169	e	108963	skol	5	dour	119
an e skol dower	903229	an	794169	e	108963	skol	5	dower	92
an eye scold hour	892949	an	794169	eye	26750	scold	217	hour	71813
an ay scold hour	872832	an	794169	ay	6633	scold	217	hour	71813
an ice-cold hour	866031	an	794169	ice-cold	49	hour	71813		
an ice coal dour	826938	an	794169	ice	12202	coal	20448	dour	119
an ice coal dower	826911	an	794169	ice	12202	coal	20448	dower	92
an eye skol dour	821043	an	794169	eye	26750	skol	5	dour	119
an eye skol dower	821016	an	794169	eye	26750	skol	5	dower	92
an ice cole dour	808563	an	794169	ice	12202	cole	2073	dour	119

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
an ice cole dower	808536	an	794169	ice	12202	cole	2073	dower	92
an ice kohl dour	807402	an	794169	ice	12202	kohl	912	dour	119
an ice kohl dower	807375	an	794169	ice	12202	kohl	912	dower	92
an ay skol dour	800926	an	794169	ay	6633	skol	5	dour	119
an ay skol dower	800899	an	794169	ay	6633	skol	5	dower	92
eh nice cold our	783938	eh	66963	nice	190708	cold	52844	our	473423
ah niece cold our	757256	ah	229296	niece	1693	cold	52844	our	473423
ah gneiss cold our	755584	ah	229296	gneiss	21	cold	52844	our	473423
et nice cold our	723706	et	6731	nice	190708	cold	52844	our	473423
o’ nice cold our	717438	o’	463	nice	190708	cold	52844	our	473423
ah ne scold our	710039	ah	229296	ne	7103	scold	217	our	473423
ah knee scold our	709075	ah	229296	knee	6139	scold	217	our	473423
ah nigh scold our	704330	ah	229296	nigh	1394	scold	217	our	473423
ah nye scold our	702973	ah	229296	nye	37	scold	217	our	473423
’n’ aye scold our	604143	’n’	0	aye	130503	scold	217	our	473423
eh niece cold our	594923	eh	66963	niece	1693	cold	52844	our	473423
eh gneiss cold our	593251	eh	66963	gneiss	21	cold	52844	our	473423
’n’ e scold our	582603	’n’	0	e	108963	scold	217	our	473423
eh ne scold our	547706	eh	66963	ne	7103	scold	217	our	473423
eh knee scold our	546742	eh	66963	knee	6139	scold	217	our	473423
ah nice cold hour	544661	ah	229296	nice	190708	cold	52844	hour	71813
eh nigh scold our	541997	eh	66963	nigh	1394	scold	217	our	473423

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
eh nye scold our	540640	eh	66963	nye	37	scold	217	our	473423
'n' ice cold our	538469	'n'	0	ice	12202	cold	52844	our	473423
et niece cold our	534691	et	6731	niece	1693	cold	52844	our	473423
et gneiss cold our	533019	et	6731	gneiss	21	cold	52844	our	473423
o' niece cold our	528423	o'	463	niece	1693	cold	52844	our	473423
o' gneiss cold our	526751	o'	463	gneiss	21	cold	52844	our	473423
'n' eye scold our	500390	'n'	0	eye	26750	scold	217	our	473423
et ne scold our	487474	et	6731	ne	7103	scold	217	our	473423
et knee scold our	486510	et	6731	knee	6139	scold	217	our	473423
et nigh scold our	481765	et	6731	nigh	1394	scold	217	our	473423
o' ne scold our	481206	o'	463	ne	7103	scold	217	our	473423
et nye scold our	480408	et	6731	nye	37	scold	217	our	473423
'n' ay scold our	480273	'n'	0	ay	6633	scold	217	our	473423
o' knee scold our	480242	o'	463	knee	6139	scold	217	our	473423
o' nigh scold our	475497	o'	463	nigh	1394	scold	217	our	473423
o' nye scold our	474140	o'	463	nye	37	scold	217	our	473423
'n' ice-cold our	473472	'n'	0	ice-cold	49	our	473423		
ah nice coal dour	440571	ah	229296	nice	190708	coal	20448	dour	119
ah nice coal dower	440544	ah	229296	nice	190708	coal	20448	dower	92
ah nice cole dour	422196	ah	229296	nice	190708	cole	2073	dour	119
ah nice cole dower	422169	ah	229296	nice	190708	cole	2073	dower	92
ah nice kohl dour	421035	ah	229296	nice	190708	kohl	912	dour	119

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
ah nice kohl dower	421008	ah	229296	nice	190708	kohl	912	dower	92
eh nice cold hour	382328	eh	66963	nice	190708	cold	52844	hour	71813
ah niece cold hour	355646	ah	229296	niece	1693	cold	52844	hour	71813
ah gneiss cold hour	353974	ah	229296	gneiss	21	cold	52844	hour	71813
et nice cold hour	322096	et	6731	nice	190708	cold	52844	hour	71813
o' nice cold hour	315828	o'	463	nice	190708	cold	52844	hour	71813
ah ne scold hour	308429	ah	229296	ne	7103	scold	217	hour	71813
ah knee scold hour	307465	ah	229296	knee	6139	scold	217	hour	71813
ah nigh scold hour	302720	ah	229296	nigh	1394	scold	217	hour	71813
ah nye scold hour	301363	ah	229296	nye	37	scold	217	hour	71813
eh nice coal dour	278238	eh	66963	nice	190708	coal	20448	dour	119
eh nice coal dower	278211	eh	66963	nice	190708	coal	20448	dower	92
eh nice cole dour	259863	eh	66963	nice	190708	cole	2073	dour	119
eh nice cole dower	259836	eh	66963	nice	190708	cole	2073	dower	92
eh nice kohl dour	258702	eh	66963	nice	190708	kohl	912	dour	119
eh nice kohl dower	258675	eh	66963	nice	190708	kohl	912	dower	92
ah niece coal dour	251556	ah	229296	niece	1693	coal	20448	dour	119
ah niece coal dower	251529	ah	229296	niece	1693	coal	20448	dower	92
ah gneiss coal dour	249884	ah	229296	gneiss	21	coal	20448	dour	119
ah gneiss coal dower	249857	ah	229296	gneiss	21	coal	20448	dower	92
ah ne skol dour	236523	ah	229296	ne	7103	skol	5	dour	119
ah ne skol dower	236496	ah	229296	ne	7103	skol	5	dower	92

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
ah knee skol dour	235559	ah	229296	knee	6139	skol	5	dour	119
ah knee skol dower	235532	ah	229296	knee	6139	skol	5	dower	92
ah niece cole dour	233181	ah	229296	niece	1693	cole	2073	dour	119
ah niece cole dower	233154	ah	229296	niece	1693	cole	2073	dower	92
ah niece kohl dour	232020	ah	229296	niece	1693	kohl	912	dour	119
ah niece kohl dower	231993	ah	229296	niece	1693	kohl	912	dower	92
ah gneiss cole dour	231509	ah	229296	gneiss	21	cole	2073	dour	119
ah gneiss cole dower	231482	ah	229296	gneiss	21	cole	2073	dower	92
ah nigh skol dour	230814	ah	229296	nigh	1394	skol	5	dour	119
ah nigh skol dower	230787	ah	229296	nigh	1394	skol	5	dower	92
ah gneiss kohl dour	230348	ah	229296	gneiss	21	kohl	912	dour	119
ah gneiss kohl dower	230321	ah	229296	gneiss	21	kohl	912	dower	92
ah nye skol dour	229457	ah	229296	nye	37	skol	5	dour	119
ah nye skol dower	229430	ah	229296	nye	37	skol	5	dower	92
et nice coal dour	218006	et	6731	nice	190708	coal	20448	dour	119
et nice coal dower	217979	et	6731	nice	190708	coal	20448	dower	92
o’ nice coal dour	211738	o’	463	nice	190708	coal	20448	dour	119
o’ nice coal dower	211711	o’	463	nice	190708	coal	20448	dower	92
’n’ aye scold hour	202533	’n’	0	aye	130503	scold	217	hour	71813
et nice cole dour	199631	et	6731	nice	190708	cole	2073	dour	119
et nice cole dower	199604	et	6731	nice	190708	cole	2073	dower	92
et nice kohl dour	198470	et	6731	nice	190708	kohl	912	dour	119

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
et nice kohl dower	198443	et	6731	nice	190708	kohl	912	dower	92
o' nice cole dour	193363	o'	463	nice	190708	cole	2073	dour	119
o' nice cole dower	193336	o'	463	nice	190708	cole	2073	dower	92
eh niece cold hour	193313	eh	66963	niece	1693	cold	52844	hour	71813
o' nice kohl dour	192202	o'	463	nice	190708	kohl	912	dour	119
o' nice kohl dower	192175	o'	463	nice	190708	kohl	912	dower	92
eh gneiss cold hour	191641	eh	66963	gneiss	21	cold	52844	hour	71813
'n' e scold hour	180993	'n'	0	e	108963	scold	217	hour	71813
eh ne scold hour	146096	eh	66963	ne	7103	scold	217	hour	71813
eh knee scold hour	145132	eh	66963	knee	6139	scold	217	hour	71813
eh nigh scold hour	140387	eh	66963	nigh	1394	scold	217	hour	71813
eh nye scold hour	139030	eh	66963	nye	37	scold	217	hour	71813
'n' ice cold hour	136859	'n'	0	ice	12202	cold	52844	hour	71813
et niece cold hour	133081	et	6731	niece	1693	cold	52844	hour	71813
et gneiss cold hour	131409	et	6731	gneiss	21	cold	52844	hour	71813
'n' aye skol dour	130627	'n'	0	aye	130503	skol	5	dour	119
'n' aye skol dower	130600	'n'	0	aye	130503	skol	5	dower	92
o' niece cold hour	126813	o'	463	niece	1693	cold	52844	hour	71813
o' gneiss cold hour	125141	o'	463	gneiss	21	cold	52844	hour	71813
'n' e skol dour	109087	'n'	0	e	108963	skol	5	dour	119
'n' e skol dower	109060	'n'	0	e	108963	skol	5	dower	92
'n' eye scold hour	98780	'n'	0	eye	26750	scold	217	hour	71813

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
eh niece coal dour	89223	eh	66963	niece	1693	coal	20448	dour	119
eh niece coal dower	89196	eh	66963	niece	1693	coal	20448	dower	92
eh gneiss coal dour	87551	eh	66963	gneiss	21	coal	20448	dour	119
eh gneiss coal dower	87524	eh	66963	gneiss	21	coal	20448	dower	92
et ne scold hour	85864	et	6731	ne	7103	scold	217	hour	71813
et knee scold hour	84900	et	6731	knee	6139	scold	217	hour	71813
et nigh scold hour	80155	et	6731	nigh	1394	scold	217	hour	71813
o' ne scold hour	79596	o'	463	ne	7103	scold	217	hour	71813
et nye scold hour	78798	et	6731	nye	37	scold	217	hour	71813
'n' ay scold hour	78663	'n'	0	ay	6633	scold	217	hour	71813
o' knee scold hour	78632	o'	463	knee	6139	scold	217	hour	71813
eh ne skol dour	74190	eh	66963	ne	7103	skol	5	dour	119
eh ne skol dower	74163	eh	66963	ne	7103	skol	5	dower	92
o' nigh scold hour	73887	o'	463	nigh	1394	scold	217	hour	71813
eh knee skol dour	73226	eh	66963	knee	6139	skol	5	dour	119
eh knee skol dower	73199	eh	66963	knee	6139	skol	5	dower	92
o' nye scold hour	72530	o'	463	nye	37	scold	217	hour	71813
'n' ice-cold hour	71862	'n'	0	ice-cold	49	hour	71813		
eh niece cole dour	70848	eh	66963	niece	1693	cole	2073	dour	119
eh niece cole dower	70821	eh	66963	niece	1693	cole	2073	dower	92
eh niece kohl dour	69687	eh	66963	niece	1693	kohl	912	dour	119
eh niece kohl dower	69660	eh	66963	niece	1693	kohl	912	dower	92

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
eh gneiss cole dour	69176	eh	66963	gneiss	21	cole	2073	dour	119
eh gneiss cole dower	69149	eh	66963	gneiss	21	cole	2073	dower	92
eh nigh skol dour	68481	eh	66963	nigh	1394	skol	5	dour	119
eh nigh skol dower	68454	eh	66963	nigh	1394	skol	5	dower	92
eh gneiss kohl dour	68015	eh	66963	gneiss	21	kohl	912	dour	119
eh gneiss kohl dower	67988	eh	66963	gneiss	21	kohl	912	dower	92
eh nye skol dour	67124	eh	66963	nye	37	skol	5	dour	119
eh nye skol dower	67097	eh	66963	nye	37	skol	5	dower	92
'n' ice coal dour	32769	'n'	0	ice	12202	coal	20448	dour	119
'n' ice coal dower	32742	'n'	0	ice	12202	coal	20448	dower	92
et niece coal dour	28991	et	6731	niece	1693	coal	20448	dour	119
et niece coal dower	28964	et	6731	niece	1693	coal	20448	dower	92
et gneiss coal dour	27319	et	6731	gneiss	21	coal	20448	dour	119
et gneiss coal dower	27292	et	6731	gneiss	21	coal	20448	dower	92
'n' eye skol dour	26874	'n'	0	eye	26750	skol	5	dour	119
'n' eye skol dower	26847	'n'	0	eye	26750	skol	5	dower	92
o' niece coal dour	22723	o'	463	niece	1693	coal	20448	dour	119
o' niece coal dower	22696	o'	463	niece	1693	coal	20448	dower	92
o' gneiss coal dour	21051	o'	463	gneiss	21	coal	20448	dour	119
o' gneiss coal dower	21024	o'	463	gneiss	21	coal	20448	dower	92
'n' ice cole dour	14394	'n'	0	ice	12202	cole	2073	dour	119
'n' ice cole dower	14367	'n'	0	ice	12202	cole	2073	dower	92

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
et ne skol dour	13958	et	6731	ne	7103	skol	5	dour	119
et ne skol dower	13931	et	6731	ne	7103	skol	5	dower	92
'n' ice kohl dour	13233	'n'	0	ice	12202	kohl	912	dour	119
'n' ice kohl dower	13206	'n'	0	ice	12202	kohl	912	dower	92
et knee skol dour	12994	et	6731	knee	6139	skol	5	dour	119
et knee skol dower	12967	et	6731	knee	6139	skol	5	dower	92
et niece cole dour	10616	et	6731	niece	1693	cole	2073	dour	119
et niece cole dower	10589	et	6731	niece	1693	cole	2073	dower	92
et niece kohl dour	9455	et	6731	niece	1693	kohl	912	dour	119
et niece kohl dower	9428	et	6731	niece	1693	kohl	912	dower	92
et gneiss cole dour	8944	et	6731	gneiss	21	cole	2073	dour	119
et gneiss cole dower	8917	et	6731	gneiss	21	cole	2073	dower	92
et nigh skol dour	8249	et	6731	nigh	1394	skol	5	dour	119
et nigh skol dower	8222	et	6731	nigh	1394	skol	5	dower	92
et gneiss kohl dour	7783	et	6731	gneiss	21	kohl	912	dour	119
et gneiss kohl dower	7756	et	6731	gneiss	21	kohl	912	dower	92
o' ne skol dour	7690	o'	463	ne	7103	skol	5	dour	119
o' ne skol dower	7663	o'	463	ne	7103	skol	5	dower	92
et nye skol dour	6892	et	6731	nye	37	skol	5	dour	119
et nye skol dower	6865	et	6731	nye	37	skol	5	dower	92
'n' ay skol dour	6757	'n'	0	ay	6633	skol	5	dour	119
'n' ay skol dower	6730	'n'	0	ay	6633	skol	5	dower	92

Continued on next page

Table A.1 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
o' knee skol dour	6726	o'	463	knee	6139	skol	5	dour	119
o' knee skol dower	6699	o'	463	knee	6139	skol	5	dower	92
o' niece cole dour	4348	o'	463	niece	1693	cole	2073	dour	119
o' niece cole dower	4321	o'	463	niece	1693	cole	2073	dower	92
o' niece kohl dour	3187	o'	463	niece	1693	kohl	912	dour	119
o' niece kohl dower	3160	o'	463	niece	1693	kohl	912	dower	92
o' gneiss cole dour	2676	o'	463	gneiss	21	cole	2073	dour	119
o' gneiss cole dower	2649	o'	463	gneiss	21	cole	2073	dower	92
o' nigh skol dour	1981	o'	463	nigh	1394	skol	5	dour	119
o' nigh skol dower	1954	o'	463	nigh	1394	skol	5	dower	92
o' gneiss kohl dour	1515	o'	463	gneiss	21	kohl	912	dour	119
o' gneiss kohl dower	1488	o'	463	gneiss	21	kohl	912	dower	92
o' nye skol dour	624	o'	463	nye	37	skol	5	dour	119
o' nye skol dower	597	o'	463	nye	37	skol	5	dower	92

Bibliography

[1] All our n-gram are belong to you | research blog. http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html.

[2] The CMU pronouncing dictionary. http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

[3] Corpus-based word frequency lists, collocates, and n-grams. http://www.wordfrequency.info/comparison.asp.

[4] Dear R/Assistance, i'm about to finish my master's thesis, but i need your help! (tasks are online; i'm in san luis obispo, CA). : Assistance. http://www.reddit.com/r/Assistance/comments/ubty1/dear_rassistance_im_about_to_finish_m

[5] Dear RecordThis: i'm finishing up my masters thesis, and i need your help! : recordthis. http://www.reddit.com/r/recordthis/comments/ubt9f/dear_recordthis_im_finishing_up_my_ma

[6] File:General american.png - wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/File:General_American.png.

[7] Keep thou my way, hymnlyrics.org. http://www.hymnlyrics.org/newlyrics_k/keep_thou_my_way

[8] knights_emic.gif (GIF image, 552 407 pixels) - scaled (0%).

[9] knights_phonetic.jpg (JPEG image, 552 407 pixels) - scaled (0%).

[10] LCStar project web – schedule. http://www.lc-star.com/schedule.htm.

[11] Mondegreen | define mondegreen at dictionary.com. http://dictionary.reference.com/browse/mondegreen?s=t.

[12] N-grams: corpus based (COCA, COHA, spanish, portuguese). http://www.ngrams.info/.

[13] oronym - definition and meaning. http://www.wordnik.com/words/oronym.

[14] Orthography | define orthography at dictionary.com. http://dictionary.reference.com/browse/orthography.

[15] SQLite database browser. http://sqlitebrowser.sourceforge.net/.

[16] Understanding how to sing the vowels - a technical description for classical singers. http://ezinearticles.com/?Understanding-How-to-Sing-the-Vowels—A-Technical-Description-For-Classical-Singers&id=1671860.

[17] Unisyn lexicon. http://www.cstr.ed.ac.uk/projects/unisyn/.

[18] Why standard american english really is "no accent". http://boards.straightdope.com/sdmb/archive/index.php/t-619668.html.

[19] 'At the tone' it will be jane barbe, america's answer to big ben. *People Magazine*, Aug. 1976.

[20] (1) "Rolling in the deep" cover, front porch band, Jan. 2012.

[21] J. Aronson. When i use a word words misheard: Medical mondegreens. *QJM*, 102(4):301–302, Apr. 2009.

[22] D. Crystal. DCblog: on singing accents, Nov. 2009.

[23] M. Davies. Word frequency data from the corpus of contemporary american english (COCA)., 2011.

[24] G. W. Elko, J. Meyer, S. Backer, and J. Peissig. Electronic pop protection for microphones. In *Applications of Signal Processing to Audio and Acoustics, 2007 IEEE Workshop on*, pages 46 –49, Oct. 2007.

[25] S. Fitt. Documentation and user guide to UNISYN lexicon and post-lexical rules. *Center for Speech Technology Research, University of Edinburgh, Tech. Rep*, 2000.

[26] J. Hendrix. Purple haze, June 1967.

[27] T. Polyakova and A. Bonafonte. Fusion of dictionaries in voice creation and speech synthesis task. In *Proc. of SPECOM*, 2007.

[28] C. C. Revival. Bad moon rising, Apr. 1969.

[29] G. P. Smith. Music and mondegreens: extracting meaning from noise. *ELT Journal*, 57(2):113121, 2003.

[30] J. Sprouse. A validation of amazon mechanical turk for the collection of acceptability judgments in linguistic theory. *Behavior Research Methods*, 43(1):155–167, 2011.

[31] S. Wright. The death of lady mondegreen. *Harpers Magazine*, 209(1254):4851, 1954.