MISHEARD ME ORONYM TREE: USING ORONYM TREES 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 Oronym Tree: Using Oronym Trees 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 Oronym Tree: Using Oronym Trees 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. However, this approach is highly dependent upon the manner in which the frequency values in a word frequency dictionary are obtains.

Our paper demonstrates a technique used to validate word frequency dictionary values. We use oronym trees to compare phrase frequency values from dictionaries, to the frequency with which our human test subjects heard different variations of the root phrase. We chose to use frequency values from the UNISYN dictionary, which uses a simple word-occurance tally measurement.

Given any valid English phrase, herein referred to as the root phrase, our system will first generate all possible correct phonetic sequences for a General American accent. Then, it parses through these phonetic transcriptions depth-first, looking for valid orthographic words for each subsequent phonetic subsequence, generating full and partial phrases from these words. In the event that the entire phonetic sequence branch can be parsed into a valid orthographic phrase, we save this orthographic phrase as an oronym of the root phrase.

We also developed a visual representation of the oronym trees, to allow for visualizing phonetic dead-ends. In the event that a branch's phonetic "tail" is not orthographically interpretable, we visually "dead-end" the branch by drawing a red sphere. A particularly strong orthographic partial phrase before a phonetic dead-end can mislead a listener, causing them to lose track of the words in the rest of the phrase. In the event that the entire phonetic sequence can be parsed into a valid orthographic phrase, we indicate this successfully-found oronym with a green sphere.

Using the oronyms generated from our oronym tree, we then conducted a user study. Our multi-phase user study, incorporated over 851 data points from 208 test subjects. In it, we tested the validity of our oronym generation by having participants record themselves reading an oronym phrase. Then, a second set of subjects transcribed the recordings.

In the first phase, we generated oronym strings for the phrase *"a nice cold hour"*, and had over a dozen people make 72 recordings of the most common oronyms for that phrase. We then compared their pronunciations to the pronunciations we were expecting, and found that in 71 cases, the recorded phrase's phonemics matched our expectation. This indicates that, while not exhaustive, our pronunciation dictionary is a good match for actual American-English pronunciations. In the second phase, we selected 15 of the phase one recordings, and had 30 to 60 different people transcribe each one.

After the second phase, if the frequency metric calculated by summing the frequency dictionary values of the words in the phrase.

What we find is that our frequency metric was thrown off by excessively common words, such as "the", "is", and "a". These super-common words have

such high per-occurance tallies that it overpowered the effect that any regular word had on a frequency metric. However, when we tally on a per-document basis, instead of a by-occurance basis.

The best possible use case to show this is the case of "a nice cold hour", whose equally-common oronym is "an ice cold hour". The words "a" and "an" are identical in function, but "an" is only used in the case that the following word starts with a vowel sound. As there are more consonant sounds than vowel sounds, "an" is used far less often than "a" is. The UNISYN dictionary has a frequency value of 7,536,297 for "a", and of 794,169 for "an".

If these frequency dictionary values accurately reflect 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. In this case, we'd expect that the phrase "a nice cold hour" would be transcribed nearly ten times as often as "an ice cold hour".

In the event that this was not the case, we can conclude that the frequency dictionary values were obtained in a way that does not apply well to audience's auditory expectations.

I need help with statistics here. To facilitate comparison, we created two rankings for the actual result phrases: one list ranked by expected frequency, and one ranked by number of actual transcriptions by our test subjects. In our phase two results, we found that out of the 578 transcriptions acquired for 53 unique phrases, only 11 had less than a difference of 5 ranks between the actual and expected occurances. The top 10 unique results, accounting for 88.00% of total transcriptions, were on average 25 ranks more common than the frequency

metric ranking predicted they'd be, with over half of them more than 39 ranks higher (out of 53 total ranks). From this, we can conclude that the frequency dictionary that we used is flawed.

If these frequency dictionary values were correct for the phrase words, 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. In the event that this was not the case, we could conclude that the frequency dictionary values were in error for that phrase. To facilitate comparison, we created two rankings for the actual result phrases: one list ranked by expected frequency, and one ranked by number of actual transcriptions by our test subjects. In our phase two results, we found that out of the 578 transcriptions acquired for 53 unique phrases, only 11 had less than a difference of 5 ranks between the actual and expected occurances. The top 10 unique results, accounting for 88.00% of total transcriptions, were on average 25 ranks more common than the frequency metric ranking predicted they'd be, with over half of them more than 39 ranks higher (out of 53 total ranks). From this, we can conclude that the frequency dictionary that we used is flawed.

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"[26].

Additional commonly-cited mondegreens include[7][21][23]:

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.[**?**]

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, it's the standardized system of writing down words in a specific language, using a commonly-accepted set of letters according to accepted usage. [13]

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 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.

1.4.1 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.

1.4.2 Phonology (aka phonemics)

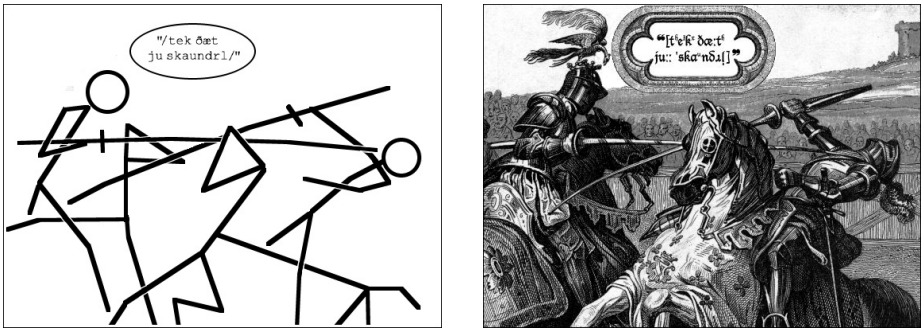
Phonology is a branch of *theoretical* linguistics, and as such, is primarily concered with the abstract grammartical characterization of sounds. It describes

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.

1.4.3 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

1.5 Phonemic/Phonetic Alphabets

As we stated in section 1.4.2, 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.

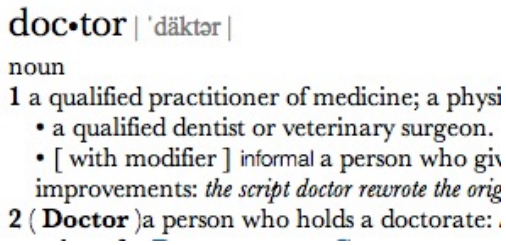

doc•tor | ˈdäktər |
noun
1 a qualified practitioner of medicine; a physi
• a qualified dentist or veterinary surgeon.
• [with modifier] informal a person who giv
improvements: *the script doctor rewrote the orig*
2 (**Doctor**)a person who holds a doctorate:

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 (Internation 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.

1.5.1 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 for the SAMPA letter 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 7 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. The way that you come to this end conclusion is dependent upon your experiences, cutural immersion, and language familiarity [24]. When attempting to write English phrases that will be read aloud and heard by people with other linguistic biases than you, it's 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 it. 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 singing fudging his vowels, like singers are taught to do, so “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 emphasis placement. 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".

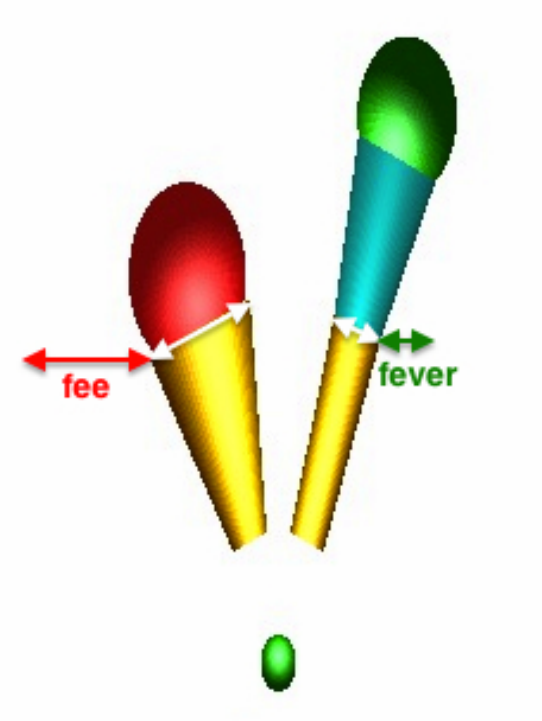
The phrase "a nice cold hour" can fail on both parts. First, the orator could have accidentally-capitalized the word Nice in their head, and made it sound like Nice, the city in France. An audience would likely hear this as "niece", and would be confused, at best. Even if the orator pronouces the phrase 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 though 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 lyrics heard.

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"[16]. 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 textual phrases in English, determines all oronyms for that phrase and then visualizes them with associated information 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 speakers 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.

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

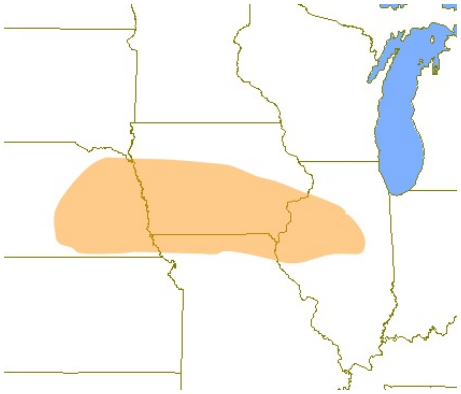


Figure 3.1: This is the geographic area whose accent most closely resembles the General American Accent [6]

American English" dialect, is not spoken by the majority of people in America, but is used as an "average accent". It most closely resembles the Midwestern accent using in the area in Figure 3.1 and more commonly recognized as "the newscaster accent". Newscasters learn this accent for national TV, because it is the "least-accented" of the American accents.

3.1.1 Dictionary Options

We considered using three different phonetic dictionaries: the CMU dictionary, LC-STAR dictionary and UNISYN dictionary[10] [2] [15]. 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

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

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

parens (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. The benefits of the CMU dictionary over some other dictionaries were that (1) it was actively maintained, (2) it included proper nouns, which are often found in lyrics, but not in dictionaries, and (3) it was ridiculously easy to read.

The downsides were that (1) it included no part of speech data or hyphenation data, and (2) it used non-standard symbols for its phonetic alphabet. With the downsides and benefits in mind, the CMU dictionary could not be used in isolation, especially if we someday want to attempt generation of original lyrics (which part of speech data would be vital for).

The UNISYN dictionary is used primarily to phonetically translate words into multiple accents. It has its own formated dictionary, with a bunch of wildcards in it. They also provide 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 references to both dictionaries in: the dictionaries were inconsistent[22]. 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 (same writing, different sounds, like "Do you know what a buck *does* to *does*?"). Because of this, we decided to use the UNISYN dictionary exclusively.

3.1.2 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 spelling of the word

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

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”

<**partOfSpeech**> is used to identify the specific part of speech

<**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 theyre emphasized. If a separator is ’ $ ’, the following phones (until the next separator) are not emphasized. If it’s ’ % ’, then they are the secondary emphasis. If it’s ’ “ ’, then they are the primary emphasis.

<**SAMPAnoEmph**> is the same as *¡SAMPASpelling¿*, but with all emphasis 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”[20].

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

3.1.3 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[14].

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.

'a nice cold hour'

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

'a', 'nice', 'cold', 'hour'

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

'a' → `e`, @, `A`
'nice' → `naIs`, `nis`
'cold' → `kould`
'hour' → `aU`r`

Figure 3.4: 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.

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 phrase.

e naIs kould aU`r
@ naIs kould aU`r
A naIs kould aU`r
e nis kould aU`r
@ nis kould aU`r
A nis kould aU`r

Figure 3.5:

The pseudocode for this process can be reviewed in figure 3.6.

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.

```
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 3.6: Algorithm to get all phonetic sequences for an orthographic phrase.

The ideal way to think about searching for words in a phonetic sequence is by picturing the phoenetic sequence in a tree form. 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 3.7 shows a visual representation of traversing an entire dictionary's phonetic tree for nodes along the paths for the SAMPA sequences 'aIs' and 'nice'.

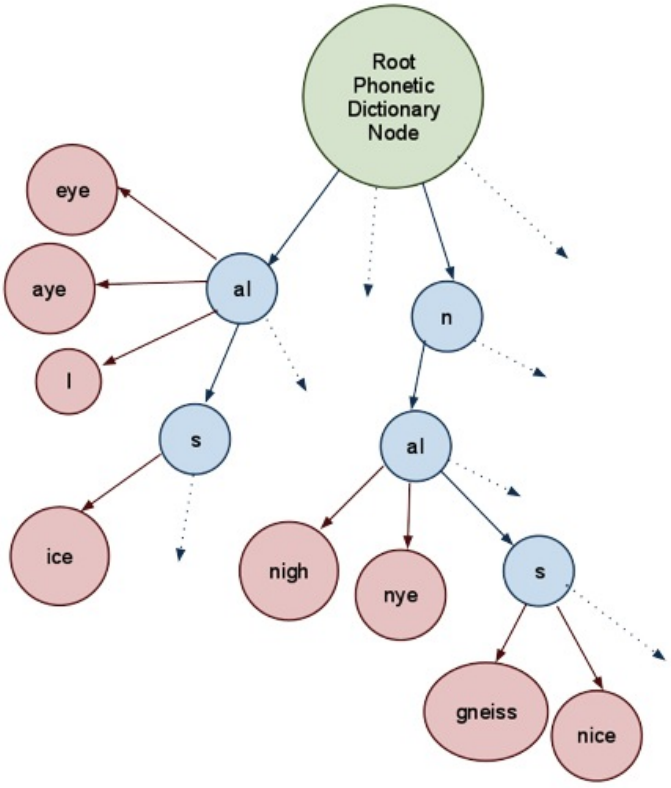


Figure 3.7:

We can use this dictionary tree method to discover valid orthographic interpretations for each phonetic sequence. Using the dictionary tree method above, we can orthographically interpret each phonetic transcription of our root orthographic phrase, as shown in figure 3.8:

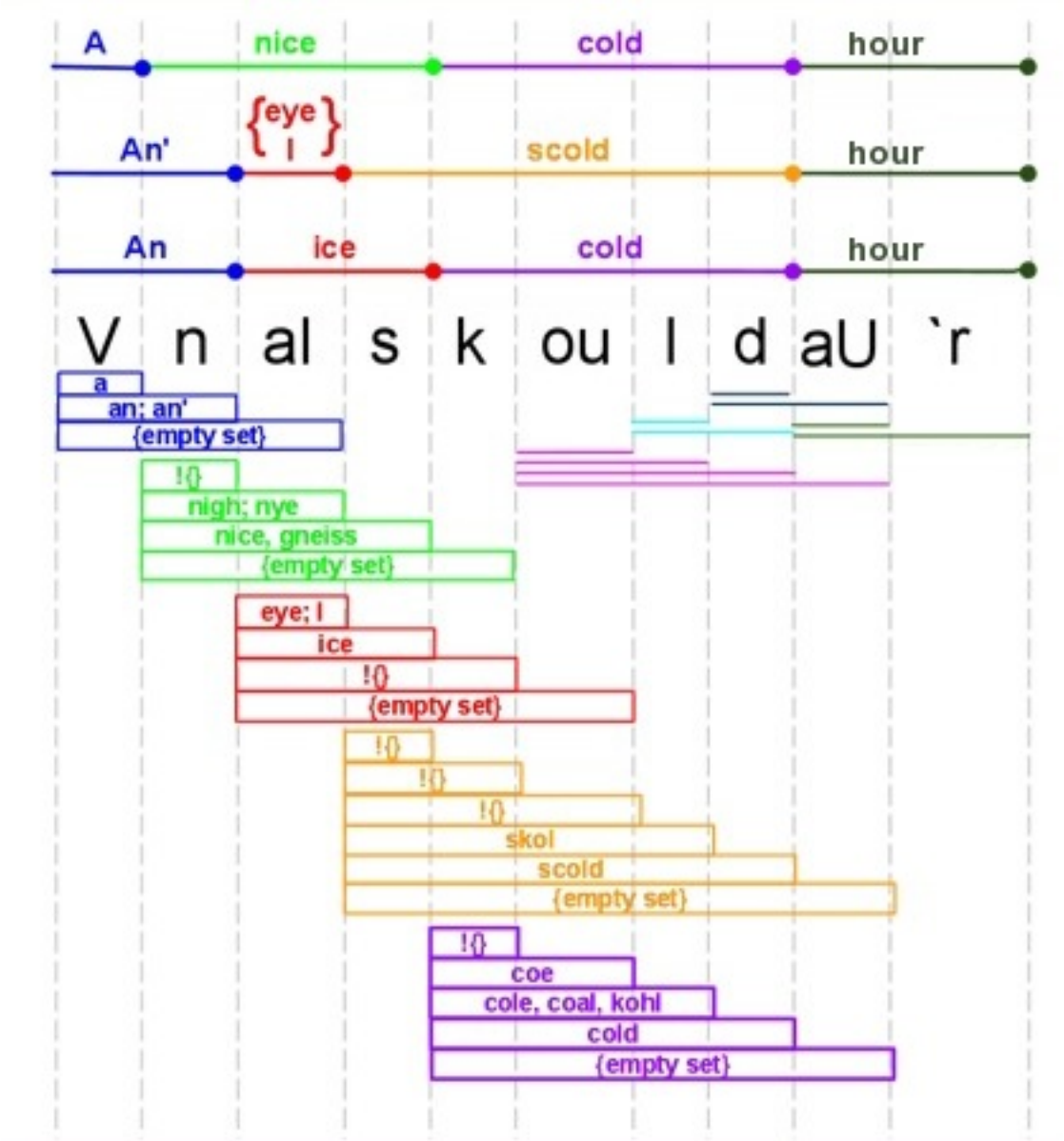


Figure 3.8:

Once we have grabbed all the orthographic interpretations for each phonetic sequence, we combine them all into a orthographic oronym phrase list. This

```
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 3.9: Algorithm to get all oronyms for an orthographic phrase.

process may leave us with some redundant oronyms, so we de-duplicate that list.

This process gives us a list of all unique and valid oronyms for the original root phrase.

In the case of "a nice cold hour", this returns 290 oronyms, as seen in the first column of figure .2.

The pseudocode for this process can be reviewed in figure 3.9

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 using non-newline whitespaces as a delimiter.

Then, we query our phonetic dictionary with each word for the 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 add all the frequencies up to give a combined-frequency of the entire phrase.

You can see these frequency counts for the phrase "a nice cold hour" in figure .2.

3.3 Visual Representation

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 difference: our oronym parse trees may contain oronym fragments. To deal with these we've got to keep track of all our abandoned sub-phrases.

Our algorithm for doing this is recursive, called from a parent function that draws the tree's 'seed' sphere. This parent function is documented in figure 3.10

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

any incomplete orthographic interpretation of a phonetic sequence, as we do in section 3.2.2, we add them to the list of oronyms, keeping track of them by appending 'xxx' or 'fff' to the end of the incomplete oronym string. Then, we tokenize our phrases by whitespace, and look up the frequency of each word, keeping track of only the maximum and minumum values. We will later scale our branches' radiuses using these values.

Once we have all the partial and complete oronyms and the max and min word frequency values for them, we pass them into our recursive function, along with the radius of the seed sphere. That radius will be the beginning radius of each first-level branch.

Inside our recursive function, we pull the first word out of every orthographic phrases we were passed, and create a set of unique first words.

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

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

Then, we check the contents of the word.

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

If the word is "___SUCCESS!___", that is also not a real word. It indicates that 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.

If the word is neither of these, then it must be a real word. We then 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 got from the word's frequency.

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. Then, we remove the first word from each of those phrases, deduplicating the resulting list of "tail" phrases.

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 phrase.

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.

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 3.12. As shown in figure 3.13 (the annotated version of figure 3.12) each branch on the tree represents a single orthographic word.

At the end of this process, we have generated a tree like the one in figure 3.12. Each branch represents a word, as can be seen in figure 3.13.

```
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 3.10: Given an orthographic phrase, this function prepares to draw the tree

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

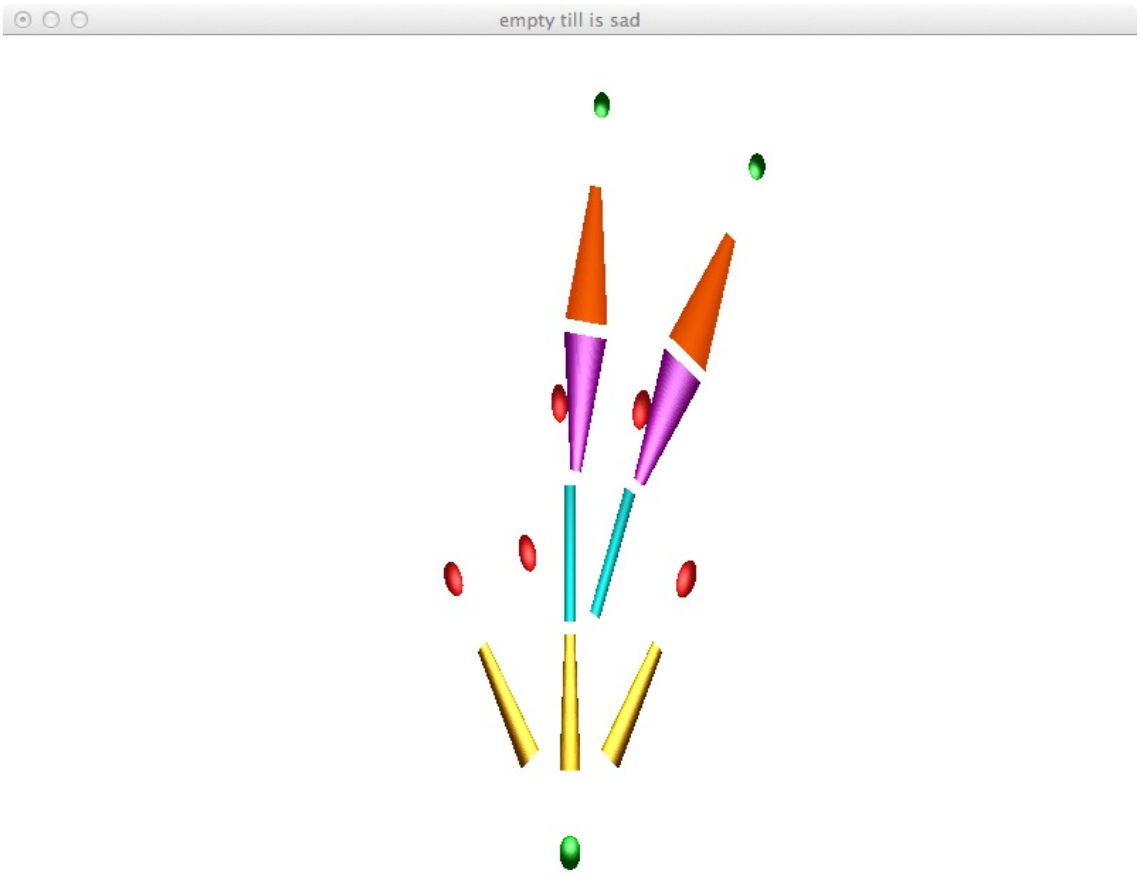


Figure 3.12: This is the parse tree for the phrase "empty till is sad"

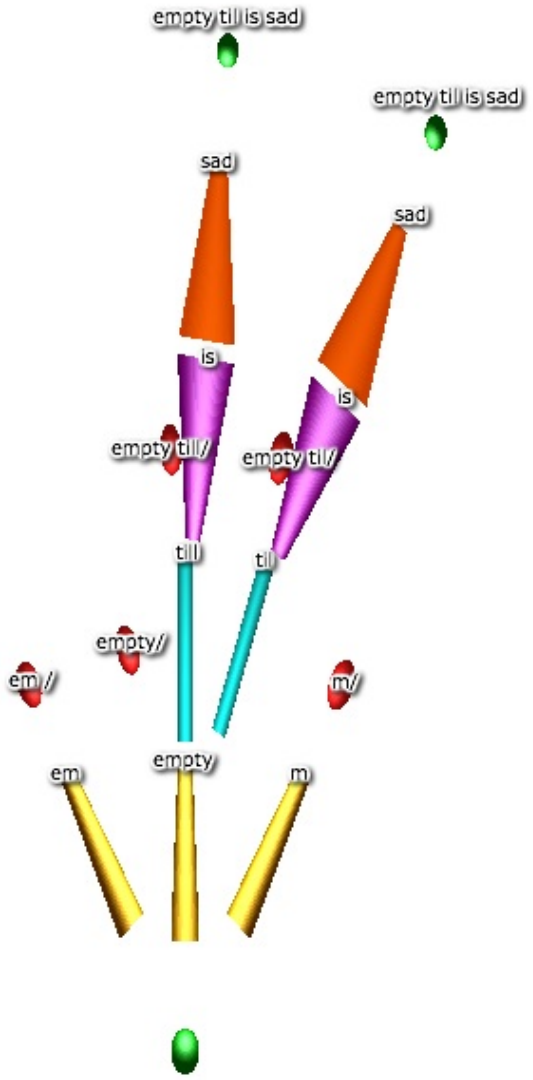


Figure 3.13: This is the annotated parse tree for the phrase "empty till is sad"

Chapter 4

User Study

4.1 Structure

We created a multi-wave user study to examine the effectiveness of different parts of our program.

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

In the second phase, we took 15 recordings of oronyms from phase one, and gathered 30 to 60 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 transcriptions of the recorded oronym phrases to the calculated oronyms for 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 [25].

4.3 Methodology

4.3.1 First Phase: Recitation

In this wave 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". To keep track of the 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 4.1.

Table 4.1: Here are the phrases we recorded, how many times they were recorded, and the identifiers we used for each phrase

orthoPhrase	numRecordings	phraseID
a nice cold our	3	A.17.51 a nice cold our
an ice cold our	2	A.17.135 an ice cold our
a nye scold our	2	A.17.69 a nye scold our
ah nye scold our	2	A.18.109 ah nye scold our
an eye scold our	2	A.18.125 an eye scold our
on aye scold our	2	A.18.267 on aye scold our
a nigh scold our	2	A.18.65 a nigh scold our
a nye skol dower	2	A.18.71 a nye skol dower
an aye skol dower	2	A.19.119 an aye skol dower
an eye skol dower	2	A.19.127 an eye skol dower
an ice coal dower	2	A.19.133 an ice coal dower
eh nice coal dower	2	A.20.159 eh nice coal dower
ah nice coal dower	2	A.20.89 ah nice coal dower
fourth wry to	2	B.15.19 fourth wry to
fourth wry too	2	B.16.20 fourth wry too
forth right ooh	2	B.17.1 forth right ooh
fourth rite ooh	2	B.17.13 fourth rite ooh
forth wright ooh	2	B.18.6 forth wright ooh
on i scold our	1	A.16.279 on i scold our
an i scold hour	1	A.17.128 an i scold hour
an i skol dower	1	A.17.131 an i skol dower

Continued on next page

Table 4.1 – continued from previous page

orthoPhrase	numRecordings	phraseID
an ice-cold our	1	A.17.141 an ice-cold our
on i scold hour	1	A.17.278 on i scold hour
on i skol dower	1	A.17.281 on i skol dower
an aye scold our	1	A.18.117 an aye scold our
an ice cold hour	1	A.18.134 an ice cold hour
an ice-cold hour	1	A.18.140 an ice-cold hour
eh nye scold our	1	A.18.179 eh nye scold our
on eye scold our	1	A.18.275 on eye scold our
on ice cold hour	1	A.18.284 on ice cold hour
on ice-cold hour	1	A.18.290 on ice-cold hour
a nye scold hour	1	A.18.68 a nye scold hour
ah nice cold our	1	A.18.91 ah nice cold our
ah nigh scold our	1	A.19.105 ah nigh scold our
ah nye scold hour	1	A.19.108 ah nye scold hour
ah nye skol dower	1	A.19.111 ah nye skol dower
an aye scold hour	1	A.19.116 an aye scold hour
an ice kohl dower	1	A.19.139 an ice kohl dower
eh nice cold hour	1	A.19.160 eh nice cold hour
eh nigh scold our	1	A.19.175 eh nigh scold our
eh nye skol dower	1	A.19.181 eh nye skol dower
on aye skol dower	1	A.19.269 on aye skol dower
on eye scold hour	1	A.19.274 on eye scold hour

Continued on next page

Table 4.1 – continued from previous page

orthoPhrase	numRecordings	phraseID
on ice coal dower	1	A.19.283 on ice coal dower
on ice kohl dower	1	A.19.289 on ice kohl dower
a nice coal dower	1	A.19.49 a nice coal dower
a nigh scold hour	1	A.19.64 a nigh scold hour
ah nice cold hour	1	A.19.90 ah nice cold hour
eh nice cole dower	1	A.20.163 eh nice cole dower
eh nigh scold hour	1	A.20.174 eh nigh scold hour
eh nigh skol dower	1	A.20.177 eh nigh skol dower
ah nice cole dower	1	A.20.93 ah nice cole dower
ah nice kohl dower	1	A.20.95 ah nice kohl dower
forth wry two	1	B.15.10 forth wry two
forth rye two	1	B.15.5 forth rye two
forth write ooh	1	B.17.7 forth write ooh
fourth right ooh	1	B.18.12 fourth right ooh
fourth wright ooh	1	B.19.17 fourth wright ooh

We then transcribed the phonetics of each of the recording 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 recordings were not deterministically phonetically parsible, meaning that our oronyms had multiple pronunciations, not all of which mapped back to the original phrase. For example, the orthographic word "a" can be interpreted as the phoneme 'A', and that 'A' phoneme can be combined with the subsequent 'n' phoneme from the word "nice" to create the

SAMPA sequence ‘An’. That being said, this fit with our model, and we found no unexpected anomalies when comparing our transcriptions to the expected SAMPA spellings of each phrase.

4.3.2 Recording Sample Pool

We had originally intended to use all the phase one recordings in phase two, but eventually 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, or the person recording left excessive amounts of space between words that overly-segmented the phrase. 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 with remarkably smooth diction from the midwest, which made him the best approximation we could get for a General American accent.

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 30 to 60 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

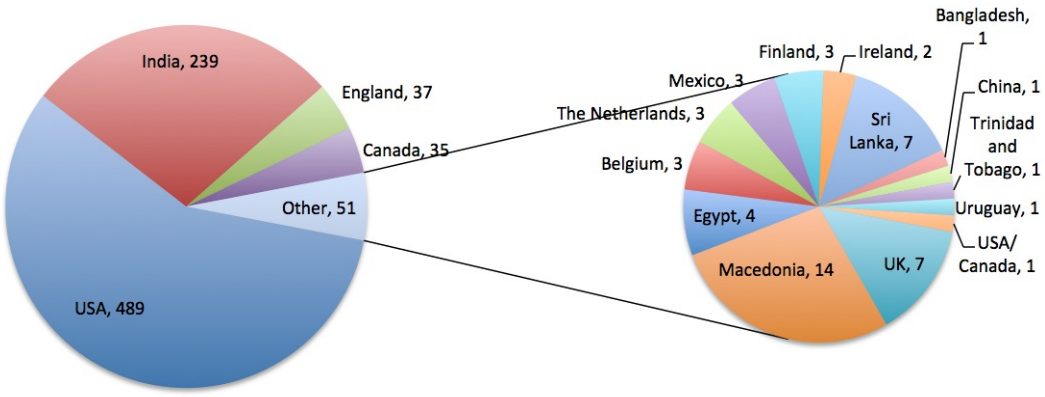


Figure 4.1: Our user study primarily polled people from the United States and India, as can be seen by the number of responses originating from each country.

English speakers from non-native speakers.

Response By Country	Num Responses
USA	489
India	239
England	37
Canada	35
UK	7
Macedonia	14
Egypt	4
Belgium	3
The Netherlands	3
Mexico	3
Finland	3
Ireland	2
Sri Lanka	7
Bangladesh	1
China	1
Trinidad and Tobago	1
Uruguay	1
USA/Canada	1

Table 4.2: 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 56 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[19].

5.2 Phase Two Results

Our top five transcribed oronyms, as seen in table 5.1, were "an ice cold hour", "a nice cold hour", "a nice gold hour", "on ice cold hour", and "in ice cold hour". All of these were predicted by our oronym-generator, except for "a nice gold hour". This is a known limitation of MisheardMe Oronym Tree, though,

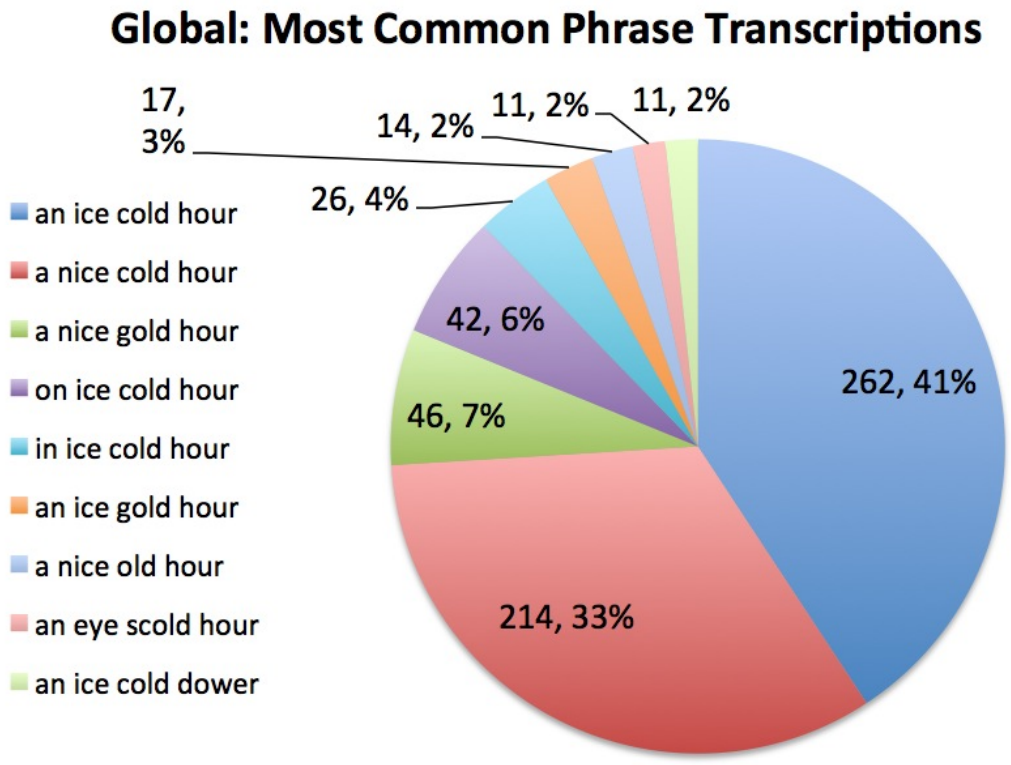


Figure 5.1: Our top two transcriptions were "a nice cold hour" and "an ice cold hour"

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 6.3. It is outside the current scope of our project.

5.2.1 Transcription oronyms' actual frequency vs calculated frequency

Though the most commonly transcribed phrases were found by our oronym generation, figure 5.3 shows an unexpected distribution of the number of times each phrase was recorded versus the frequency metric that we calculated. We

predicted freq	phrase transcribed	total answers
931028	an ice cold hour	262
7851662	a nice cold hour	214
0	a nice gold hour	46
2911102	on ice cold hour	42
5503158	in ice cold hour	26
0	an ice gold hour	17
8013781	a nice old hour	14
892949	an ice cold dower	11
859307	an eye scold hour	11

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

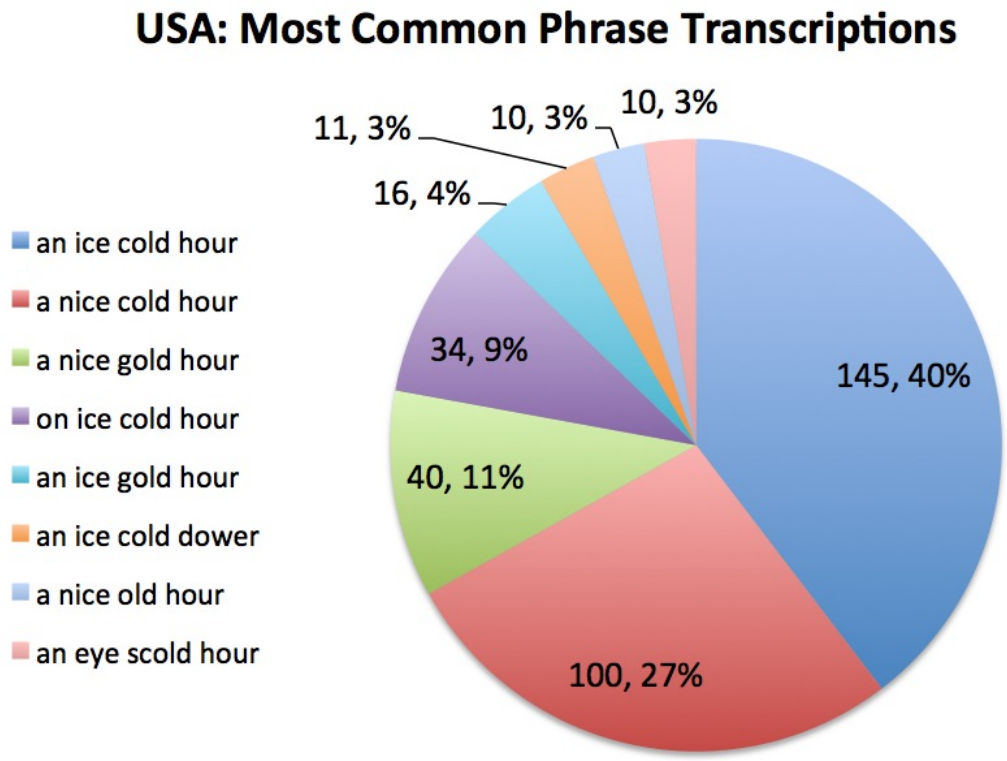


Figure 5.2: Though the breakdown is a bit different than the global transcription breakdown, you can still see the clear trend of "a nice cold hour" and "an ice cold hour" being the most common. There is a slightly larger gap between these two phrase, we hypothesize, because the American transcribers are familiar with what words normally are in proximity to others.

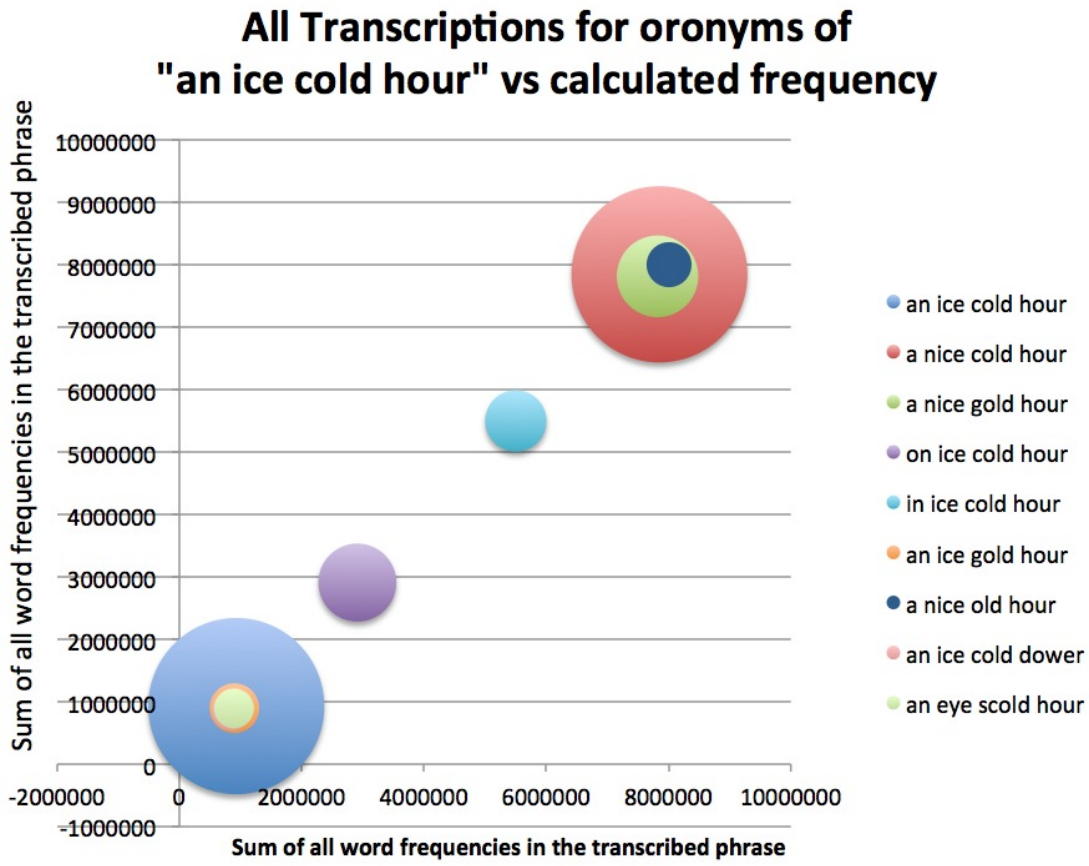


Figure 5.3: Bubble Chart of All Transcribed Phrases mapped against their predicted frequency

hypothesized that a simple summation of the UNISYN-provided word frequency for each word in a phrase would give a semi-meaningful indicator of whether a phrase's likelyhood to be heard.

Unfortunately, that proved not to be the case. In figure 5.4, we see a 2d block version of our 3d oronym parse tree, as it looks when only considering the transcriptions that were actually entered. If our frequency metric was valid, we would see that figure 5.5 would resemble figure 5.4. Past the first branch, they only vaguely resember eachother, showing that our frequency metric could use some improvement.

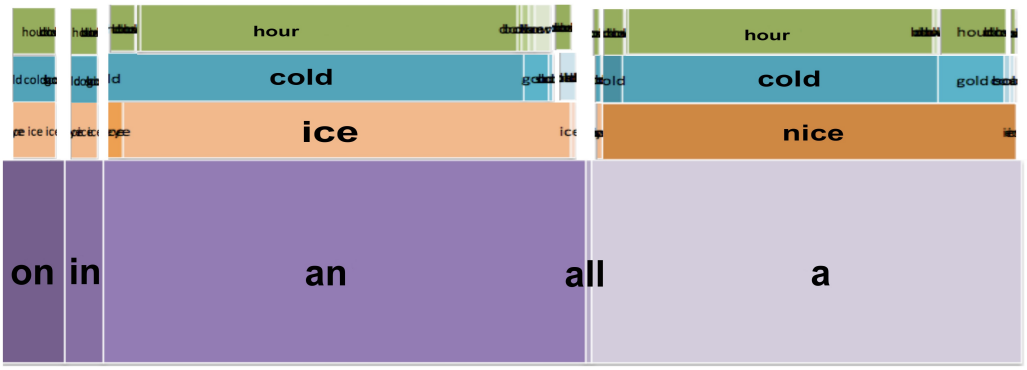


Figure 5.4: 2d block version of our 3d oronym parse tree, containing all the transcribed oronyms from mechanical turk. Instead of branches with varying radiuses, we have blocks that are scaled by the number of times that word occurs after the word block it is on top of.

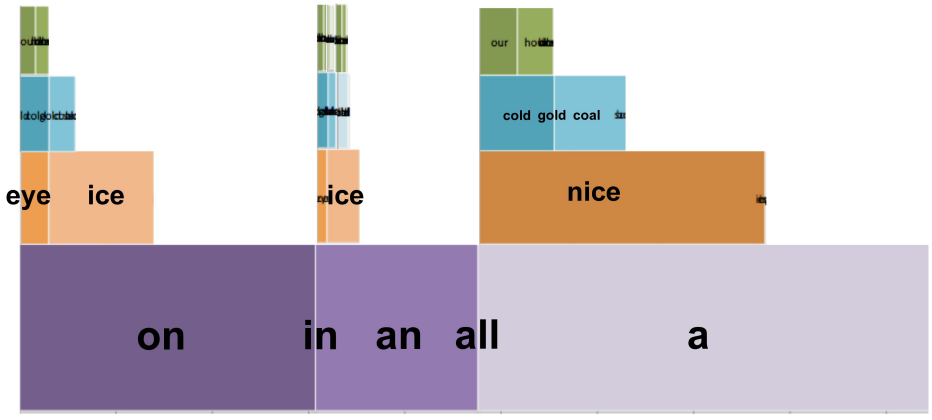


Figure 5.5: If our predictive frequency metric were completely correct, then this would be a solid block. Each filled-in part is a transcription that was actually typed by a person. All the empty spaces represent oronyms that our frequency metric incorrectly predicted would be likely to be transcribed.

Chapter 6

Future Work

6.1 Direct Improvements To Misheard Me Oronym ParseTree

Our oronyms trees display all the phonetically-matched oronyms that our users came up with. Unfortunately it also displayed a few that no human in their right mind would think of, and incorrectly weighted some others.

6.2 Places for improvement

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

6.2.1 Frequency Validity

Our frequency source data ended up being less than statisfactory. The lack of phonemic frequency data is a known defeciency in our source dictionary, UNISYN. According to the authors of the UNISYN lexicon documentation:

> Unfortunately there is currently no method for distinguishing between homographs by frequency. Furthermore, it should be noted that the frequency field, as it was obtained from simple word lists, is not particularly reliable.

[20] The UNISYN frequency count is based upon a large but not exhausting 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[17]. 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".

Also, the fact that our program cannot distinguish between words that may be homographs (that is, words that sound different but are spelled the same) makes it 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 reference, "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 more common than the actual name of the species, given that we don't see this for sheep (sheep , 13572 , ewe , 186 , ram , 681) or horses (horse , 27559 , mare , 1055 , stallion , 644). What is much more likely is that "bucks" is getting extra hits through its meaning as a slang synonym for dollars (dollars, 8927), and "does" is getting most of its frequency count for the 3rd person present tense of the verb "to do". That seems

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

very likely, given that the frequency for the singular "doe" is only 1077.

In the future, we'd like to find a dictionary with some way of distinguishing homographs when counting frequency, and 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 COCAE 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 overcomes 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 familiar. In the UNISYN dictionary, we only have contextless frequency counts. In the COCAE frequency dictionary, they keep two types of counts: one for how many times the word has been found, and one for how many documents is 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[18].

6.2.2 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

[1]

Though we are happy with our findings, we believe that we could create even better likelihood metrics with the integration of n-grams, and would suggest this for future work.

6.3 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 it's 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 **??**.

6.4 Melody Matcher master project

MisheardMe Oronym Tree is a part of the 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. This is significant, because very little research has been done specifically on the quantifiable measurement of English-language lyric intelligibility, other than our project.

Melody Matcher aims to replicate the human ability to identify lyrics in a song that are easily misheard. We started on this project, thinking that there would be carefully-specified research on how lyrics match melodies, mathematically. As it turned out, there was very little objective literature on the subject. Because of the lack of objective information of the subject, we had to develop our method

from scratch. As we progressed through our work, we went from thinking that understandability depended only on emphasis-matching, to realizing that syllable length played a huge part as well, to realizing that there are many other musical, harmonic, and linguistic factors.

6.4.1 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.

Melody Matcher analyzes the intelligibility of song lyrics by investigating several 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)
  - Phone sequences with multiple syntactically-correct interpretations

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 Oronym Tree, a computer program which takes in textual phrases in English, determines all oronyms for that phrase and then visualizes them with associated information to indicate the likelihood of interpretation. We have demonstrated all three major functional parts: our custom phonetic dictionary, our command-line oronym generator, and our OpenGL oronym-parse-tree visualization generator. Our custom phonetic dictionary has some inconsistencies in word frenquency, due to the source dictionary for it's word frequency values not being generated from a well-sampled corpus. However, it 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 parse tree visualization had two goals: one, visually represent the likelihood of each oronym interpretation, visualized using wider branches for more common phraseds; 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 visualization can successfully do both of those things.

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

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

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

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

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

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

[19] 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.

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

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

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

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

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

[25] 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.

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

SAMPA	Example	Manner of Articulation/ Type	Voiced/ Voiceless	Starts as	Ends as	General type	Weight
p	pen, spin, tip	plosive	voiceless	block	block	Consonant	8
b	but, web	plosive	voiced	block	block	Consonant	7
t	two, sting, bet	plosive	voiceless	block	block	Consonant	8
d	do, odd	plosive	voiced	block	block	Consonant	7
tS	chair, nature, teach	affricate	voiceless	block	continuous frication	Consonant	6
dZ	gin, joy, edge	affricate	voiced	block	continuous frication	Consonant	5
k	cat, kill, skin, queen, thick	plosive	voiceless	block	block	Consonant	8
g	go, get, beg	plosive	voiced	block	block	Consonant	7
f	fool, enough, leaf	fricative	voiceless	continuous frication	continuous frication	Consonant	4
v	voice, have, of	fricative	voiced	continuous frication	continuous frication	Consonant	3
T	thing, breath	fricative	voiceless	continuous frication	continuous frication	Consonant	4
D	this, breathe	fricative	voiced	continuous frication	continuous frication	Consonant	3
s	see, city, pass	fricative	voiceless	continuous frication	continuous frication	Consonant	4
z	zoo, rose	fricative	voiced	continuous frication	continuous frication	Consonant	3
S	she, sure, emotion, leash	fricative	voiceless	continuous frication	continuous frication	Consonant	4
Z	pleasure, beige	fricative	voiced	continuous frication	continuous frication	Consonant	3

SAMPA	Example	Manner of Articulation/ Type	Voiced/ Voiceless	Starts as	Ends as	General type	Weight
h	ham	fricative	voiceless	continuous frication	continuous frication	Consonant	4
m	man, ham	nasal	voiced	redirect	redirect	Consonant	1
n	no, tin	nasal	voiced	redirect	redirect	Consonant	1
N	singer, ring	nasal	voiced	redirect	redirect	Consonant	1
l	left, bell	lateral	voiced	continuous	continuous	Consonant	0
r	run, very	approximant	voiced	continuous	continuous	Consonant	0
w	we	semivowel	voiced	continuous	end	Consonant	2
j	yes	semivowel	voiced	continuous	end	Consonant	2
W	what (Scottish)	approximant	voiceless	continuous	end	Consonant	3
x	loch (Scottish)	fricative	voiceless	continuous frication	continuous frication	Consonant	4
A	father, not, law	short				Vowel	0.25
I	city	short				Vowel	0.25
E	bed	short				Vowel	0.25
3‘	bird, winner	short				Vowel	0.25
‘r	bird, winner	short				Vowel	0.25
{	lad, cat, ran	short				Vowel	0.25
u	soon, through	short				Vowel	0.25
@	about	short				Vowel	0.25
jU	use, pupil	diphthong		syllabic consonant semivowel	short	Diphthong	0.5
ju	use, pupil	diphthong		syllabic consonant semivowel	short	Diphthong	0.5
i	see	long				Vowel	0.75

SAMPA	Example	Manner of Articulation/ Type	Voiced/ Voiceless	Starts as	Ends as	General type	Weight
V	run, enough	short				Vowel	0.25
U	put	long				Vowel	0.75
e	day	long				Vowel	0.75
O	or, shore	long				Vowel	0.75
a	DNE in GenAm	long				Vowel	0.75
aI	my, height	diphthong		long	short	Diphthong	1
OI	boy	diphthong		long	short	Diphthong	1
oU	boat	diphthong		short	long	Diphthong	1
ou	boat	diphthong		short	long	Diphthong	1
aU	now	diphthong		long	long	Diphthong	1.5
=	ridden	syllabic consonant semivowel				Vowel	0.25

Table .1: Here are the metrics that we use to determine the weights of the phonemes

.1 Individual Recording/Transcription Breakdowns

These are recording-by-recording transcription breakdowns for phase one of our user study. Each bubble charts represent the most common transcriptions for the recorded phrase listed at the top. The size of the bubble and the position along the x axis is indicative of the number of times that phrase was transcribed for this particular recording. The position along the y axis shows how often this phrase was transcribed over all the recordings.

Note that, for the purposes of clarifying these graphs, we did not chart any anomalous transcriptions–that is, transcriptions with only one occurance in this recording, or transcriptions with less than 5 percent of the recording's transcriptions with no other occurances over the entire set of recordings. Doing so did not give us useful visual data, because the bubbles stacked and obscured eachother. A full table of the transcriptions per recorded phrase can be provided upon request.

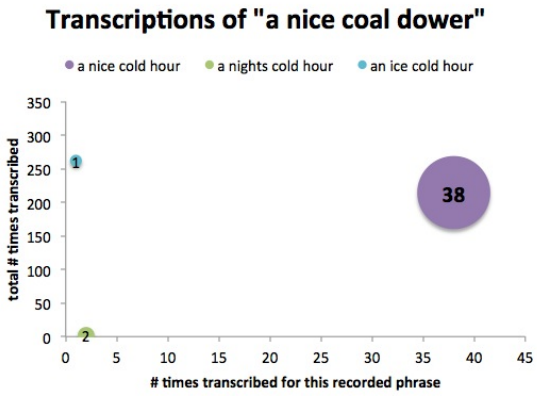


Figure .1:

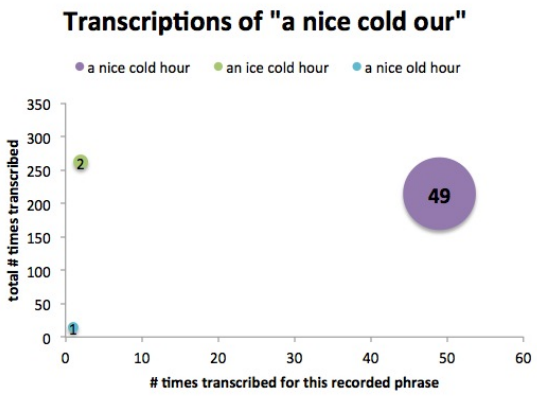


Figure .2:

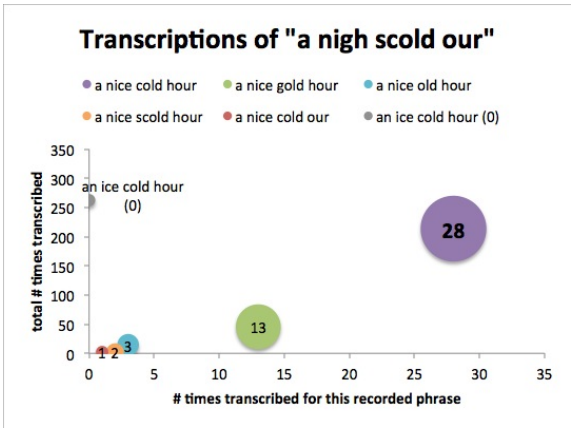


Figure .3:

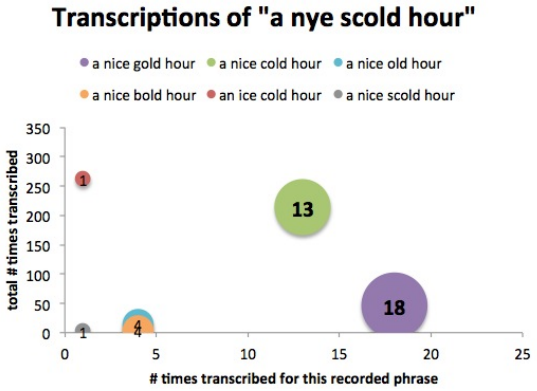


Figure .4:

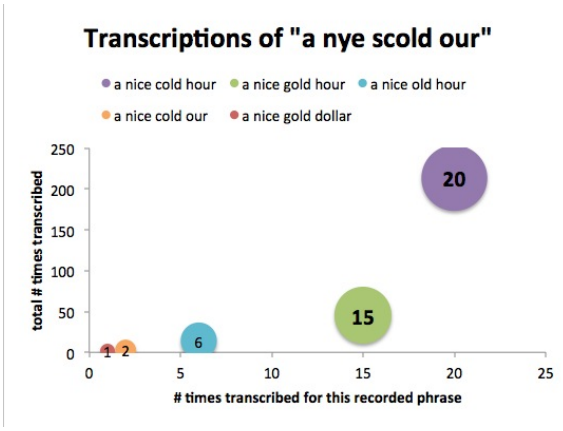


Figure .5:

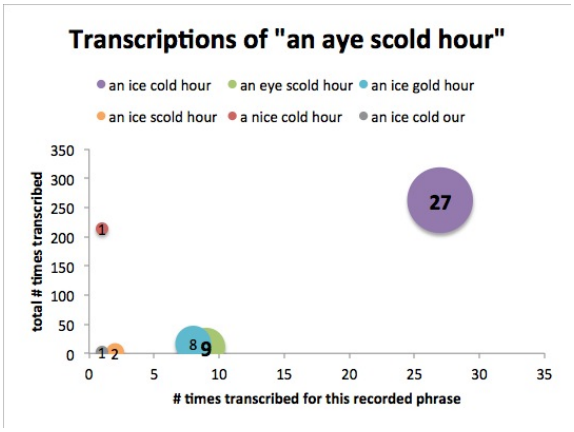


Figure .6:

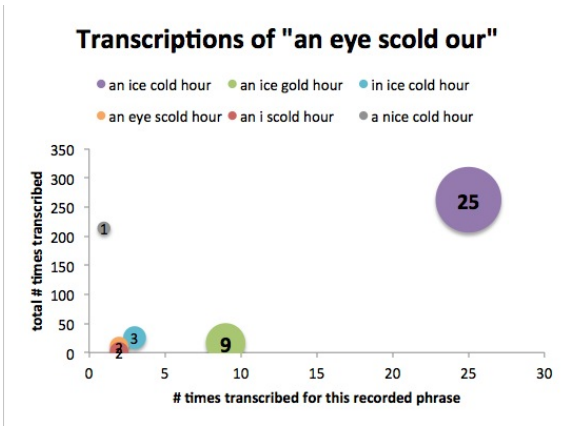


Figure .7:

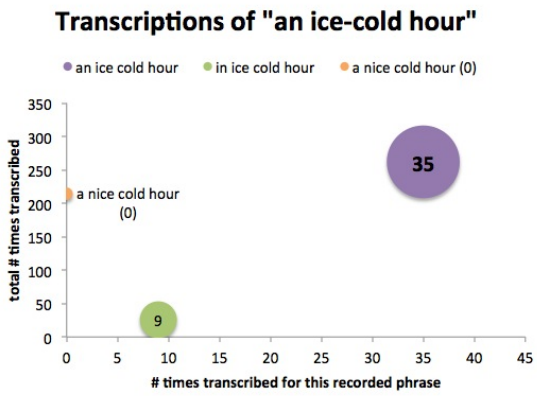


Figure .8:

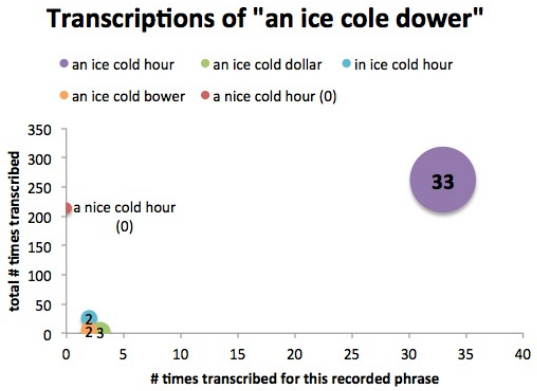


Figure .9:

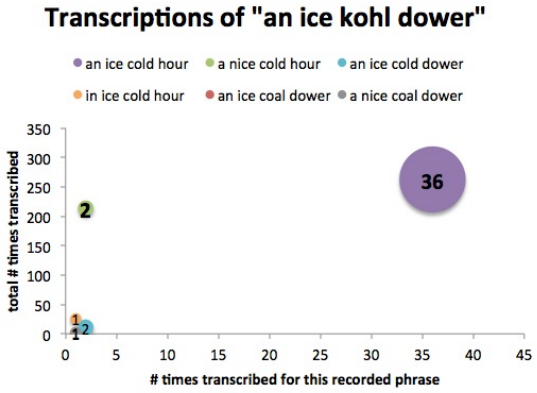


Figure .10:

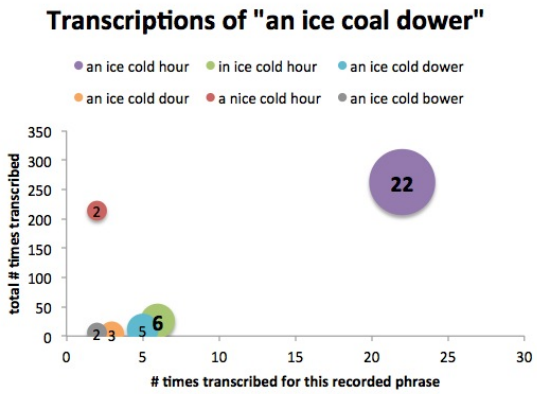


Figure .11:

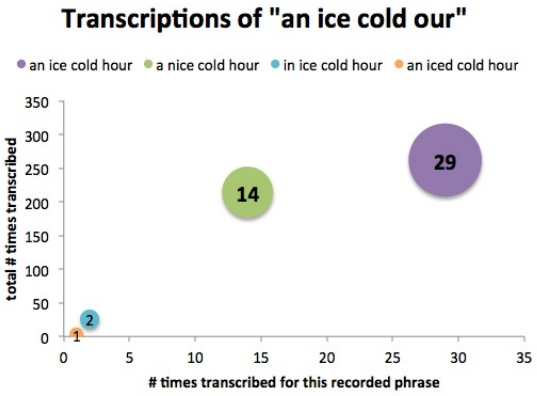


Figure .12:

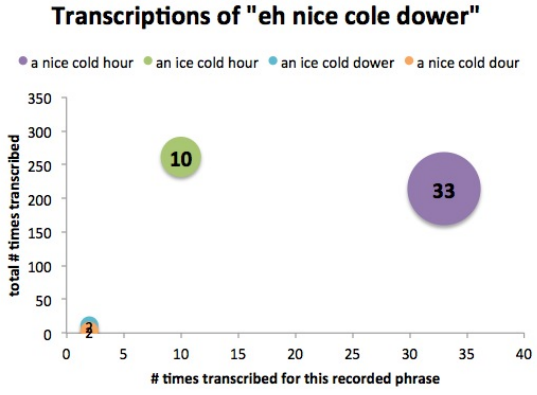


Figure .13:

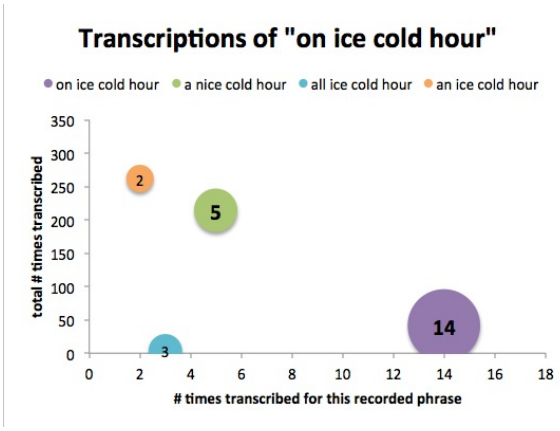


Figure .14: This phonetic sequence deterministically parses to the word "on". Unsurprisingly, in all recording with the word "on", it was nearly always heard and trascribed as "on"

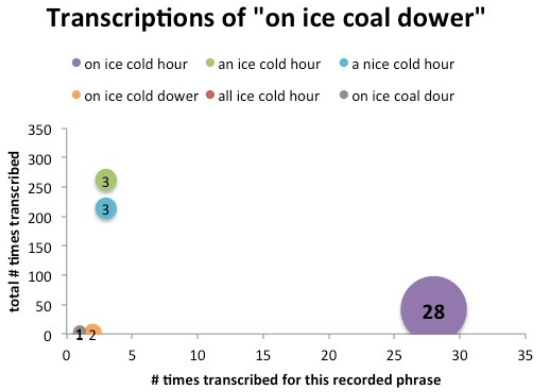


Figure .15: This phonetic sequence deterministically parses to the word "on". Unsurprisingly, in all recording with the word "on", it was nearly always heard and trascribed as "on"

Table .2: 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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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
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

Continued on next page

Table .2 – continued from previous page

phrase	total freq	word1	freq1	word2	freq2	word3	freq3	word4	freq4
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