Mining Massive Archives of Mice Sounds with Symbolized Representations

Jesin Zakaria[1] Sarah Rotschafer[2] Abdullah Mueen[1]
[1]Department of Computer Science and Engineering
University of California Riverside
{jzaka001, mueen}@cs.ucr.edu

Khaleel Razak[2] Eamonn Keogh[1]
[2]Department of Psychology
University of California Riverside
eamonn@cs.ucr.edu

ABSTRACT

Many animals produce long sequences of vocalizations best described as “songs.” In some animals, such as crickets and frogs, these songs are relatively simple and repetitive chirps or trills. However, animals as diverse as whales, bats, birds and even the humble mice considered here produce intricate and complex songs. These songs are worthy of study in their own right. For example, the study of bird songs has helped to cast light on various questions in the nature vs. nurture debate. However, there is a particular reason why the study of mice songs can benefit mankind. The house mouse (*Mus musculus*) has long been an important model organism in biology and medicine, and it is by far the most commonly used genetically altered laboratory mammal to address human diseases. While there has been significant recent efforts to analyze mice songs, advances in sensor technology have created a situation where our ability to collect data far outstrips our ability to analyze it. In this work we argue that the time is ripe for archives of mice songs to fall into the purview of data mining. We show a novel technique for mining mice vocalizations directly in the visual (spectrogram) space that practitioners currently use. Working in this space allows us to bring an arsenal of data mining tools to bear on this important domain, including similarity search, classification, motif discovery and contrast set mining.

Keywords

Similarity, Classification, Clustering, Mice Vocalization, Human Disease

1 INTRODUCTION

The house mouse (*Mus musculus*) is one of the most important model organisms in biology and medicine because of genetic engineering tools available to model human diseases. Basic and translational research on diseases as diverse as diabetes, obesity, Alzheimer’s, autism, and cancer has benefited from several genetic lines of mice that recapitulate at least some of the characteristics of human diseases [12][28] [29][33]. Mice offer significant advantages for scientific research because of their remarkable genetic similarity to humans, ease of handling, and fast reproduction rate. Thus, the mouse has been the vertebrate species of choice for scientific research. For example, in 2009, approximately 83% of scientific procedures on animals involved the use of mice or other rodents [19].

Recently, there has been an increased interest in the ultrasonic vocalizations produced by mice. Mice produce stereotyped vocalizations during behaviors such as mating, aggression, and mother-pup interactions. As shown in the snippet in Figure 1, most of these vocalizations are inaudible to humans, as they occur in the ultrasonic frequency range (30-110 kHz) [12]. The importance of these vocalizations lies in the fact that they provide an important social biomarker for communication behaviors. Also of practical importance is the fact that mice do not have to be trained to produce these calls, and they produce a rich repertoire of stereotyped calls that are known or suspected to be correlated with various behaviors. These calls can be used to probe communication dysfunctions, a hallmark of several human diseases such as autism, fragile X syndrome, and specific language impairments [29].

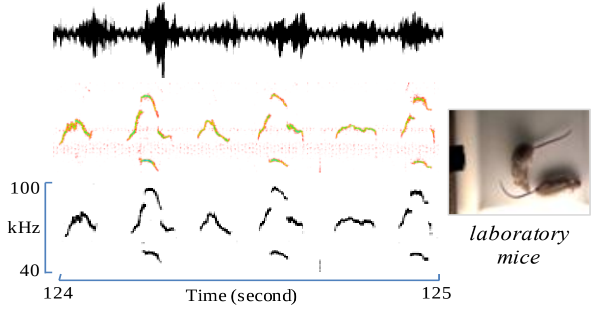


Figure 1: *top*) A waveform of a sound sequence produced by a lab mouse, *middle*) A spectrogram of the sound, *bottom*) An idealized version of the spectrogram

Recent studies have explored vocalizations in *Knock-Out* (KO) mouse models. A knockout mouse is a genetically engineered mouse in which an existing gene has been inactivated, or “knocked out,” by replacing it or disrupting it with an artificial piece of DNA. The loss of gene activity often causes changes in a mouse’s phenotype[1], which includes appearance, behavior, and other observable physical and biochemical characteristics. Note that *vocalizations* are examples of a phenotype.

Shu et al. (2005) showed that mice with a mutation in the *Foxp2* gene produce fewer vocalizations compared to wild type mice [29]. The investigators were also able to determine the altered brain structures correlated with

[1] A phenotype is an organism’s observable characteristics or traits, such as its morphology, developmental or physiological properties, and critically for this paper, products of *behavior* such as vocalizations.

reduced vocal production in the knockout mice. Foxp2 is implicated in speech production in humans. Shu et al. showed that this gene's function is conserved across mice and humans and that mice vocalizations can serve an important function in understanding *human* speech production. Wohr et al. [33] showed a similar drop in the ultrasonic calling rate in a mouse model of autism.

The increasing interest in mouse vocal behavior reveals the importance of developing analytical tools to study different properties of calls. Most current studies only quantify the most basic properties of calls such as the calling rate, call duration, and dynamic range of frequencies. However, to serve as a model for more complex vocalizations such as human speech, we need to know if there are higher-order properties in mouse calls. In addition, we also need to know if there are correlations between properties of calls and specific aspects of behavior.

In spite of the importance of the problem and the massive archives of mice audio data produced in many labs around the world, data mining has had very little impact on this field. Most published results are based on relatively simple statistical tests on small *hand-annotated* datasets [4][21][26][27][28]. In this work we attempt to repair this omission. We show a novel, highly robust technique to extract the most fundamental elements of mice vocalizations, "syllables," from large, potentially noisy audio archives. Having extracted these syllables, we further show that we can bring a wealth of data mining tools to bear on this domain, finding motifs, rules, and regularities that were hereto unknown.

The overarching motivation of our work is to *simplify* and therefore *accelerate* vocalization research. For example, the most commonly used commercial tool to find (but not classify) mouse vocalizations is Avisoft[2]. After adjusting some basic spectrogram parameters a dialogue box appears that invites the user to set the parameters in more than 50 check boxes, drop down boxes, text boxes, etc. [25]. In fairness, this superb tool can gather statistics that we are not considering here. However, it is clear that setting so many parameters does not invite the fast interactive exploration of the data that we can support, and moreover, Avisoft cannot classify syllables based on their similarity.

The rest of this paper is organized as follows. In Section 2 we give the intuition as to why we believe that symbolizing the sound files is the key step that will allow us to efficiently and effectively mine the data of interest. Section 3 presents a detailed discussion of background and related work. In Section 4 we describe novel algorithms for symbolizing mice vocalizations; this is both a contribution of this work and a necessary step for higher-level data mining algorithms we introduce in Section 5.

2 THE INTUITION BEHIND SYMBOLIZING THE SPECTROGRAMS

We argue that data mining is desperately needed in this domain, because human time and skill are currently the major bottleneck. Consider a recent study that attempts to quantify the universality of certain structural song properties. In order to do this study, the data had to be painstakingly coded by hand: "*...two persons, who were not informed about the 'aim of our study' independently did the following analyses. They* ***visually compared*** *spectrographic displays of the recordings and counted the number of..*" [4] (our emphasis). There are at least three problems with this approach. First, there is the obvious financial cost of human effort; second there is the difficulty of subjectivity when multiple humans code the data; and finally, if in the iterative process of research it is decided that a different coding scheme should be used, the entire process must be repeated. In fairness, most researchers do everything they can to mitigate the subjectivity problem. For example, [21] notes that they made sure that the person doing the hand-coding of syllables was "*blind to the age and gender of the interacting mice*." Nevertheless, removing humans from this step can only help improve and accelerate research.

There is a wealth of literature on techniques for analyzing sounds in the *audio* space. Such techniques differentiate syllables by audio features such as energy and frequency. However, we argue that to produce an accurate and usable tool for the community we should work directly in the *visual* domain. There are at least three reasons for this:

- The original audio domain is high dimensional, as the data is typically recorded at 250 kHz. While this data *could* be reduced in the original audio space, we shall show that the data can have both its dimensionality and cardinality greatly reduced in the visual space with little loss of information.
- The visual domain both allows us to directly see what matching invariances are needed (warping, uniform scaling, etc.) and allows us to achieve them efficiently.
- Ultimately, the community analyzes and communicates findings in the visual space [4][12][26][29], and our goal is to support these researchers' work in their native space.

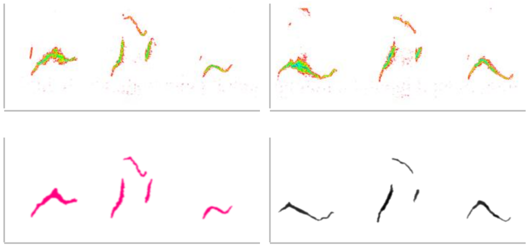


Figure 2: *top*) Two 0.5 second spectrogram representations of fragments of the vocal output of a male mouse. *bottom*) Idealized (by human intervention) versions of the above

[2] The name *Avi*soft belies the tool's origin in *bird* song processing. However, it is also used for mice, bats etc.

In Figure 2.*top*, we see two examples of snippets of mouse songs represented as *thresholded* spectrograms. Note that while the unprocessed spectrogram previously shown in Figure 1 is "dense" (that is to say, every pixel has some non-zero value), in practice pixels with a value beyond a certain threshold range are deleted for visual clarity.

In Figure 2.*bottom,* we show idealized versions of the original snippets, which have been cleaned with careful human intervention. While the reader will immediately recognize the similarity of the two snippets, this is not computationally trivial to discover directly in the image space. For example, in Figure 3 we show the two fragments aligned so as to maximize their overlap.

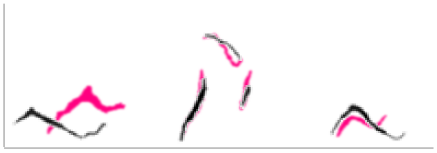


Figure 3: The two fragments of data shown in Figure 2.*bottom* aligned to produce the maximum overlap. (Best viewed in color)

In spite of this optimal alignment, *less* than 25% of the pixels from both images overlap. This means that distance measures that rely on a pixel to pixel alignment, such as sum-of-squared-difference [20], generalized Hough transform [36], Hausdorff distance, geometric hashing, etc., are doomed to failure if we attempt to match *long* sequences. There is an apparently obvious solution to this problem, using some kind of image warping or earth-movers distance that would allow invariance to the minor differences in shape and timing we observed. However, there are two reasons why this is undesirable. First, such measures typically have several parameters to constrain the allowed distortions, because if left completely unconstrained all discrimination ability is lost. Setting these parameters is non-trivial and opens the possibility of over-fitting. Second, these distance measures typically have a time complexity that is at least quadratic in the number of black pixels[3] [30]. This would not be a problem if we were clustering a handful of such patterns. However, for the data mining tasks we need to support, we may have to do *millions* of such calculations.

Figure 4 gives the intuition behind our solution. If we can symbolize the syllables, we can answer similarity queries with efficient string processing algorithms. In this case, the complex image matching problem can be reduced to finding the (possibly weighted) string edit distance between **AXQXP** and **AXCXP**.

This is an attractive idea because there are off-the-shelf tools for query by content, classification, motif discovery, and contrast set mining for strings. Moreover, because the symbolization step vastly decreases the numerosity, cardinality, and dimensionality of the data, we can expect drastic speed-ups for our mining tasks.

Note that working with *syllables* rather than *phrases* does not completely eliminate the difficulties of matching images in bitmap space. However, as we shall show in Section 4, matching at the short syllable level is significantly easier than matching at the longer phrase level, because small differences in timing has less time to accumulate differences that must be accounted for. Furthermore, as we hinted at in this example, we can compensate for some of the inevitable errors in symbolization by achieving robustness in the string processing algorithms. For example, we can allow wildcards or weight the *substitution* operator of string edit distance to an appropriately small value for easily confused syllables.

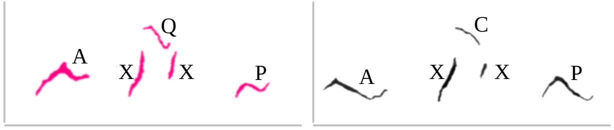


Figure 4: The data shown in Figure 2 augmented by labeled syllables

In a sense, our motivation to treat the vocalizations as symbolic text for the purposes of indexing and mining are obvious. For example, many smart phones allow you to search the web for *spoken* queries, such as the utterance "*Clint Eastwood bio*." The system is not searching for *sound files* that are similar to this snippet. Rather, the sound file is processed to a discrete string, possibly with errors, such as GLINT EASTWOOD BIO, and the robustness of the search occurs in the search engine. In this example, Google asks "**Did you mean**: *Clint Eastwood bio*".

This is *exactly* our idea. We will do our best to correctly symbolize the mouse utterances, and achieve robust data mining and indexing in the *discrete* representation.

3 RELATED AND BACKGROUND WORK

We begin by defining the relevant notation and definitions used in this work.

Mice are highly vocal animals, producing complex calls:

> ***Definition 1***: A *mouse call* is the sound uttered by a mouse for the purpose of auditory communication. It is a continuous sequence $\boldsymbol{D} = (D_1, D_2, \ldots, D_{T1})$ of *T1* real valued data points; *T1* represents the entire calling bout of the call.

While the calls are recorded as audio, for reasons that will become apparent, they are almost always processed in the visual space, by conversion to spectrograms:

> ***Definition 2:*** A sound *spectrogram* is a time varying spectral representation of an audio signal. The relative intensity of a sound at any particular time and frequency is indicated by the color of the spectrogram at that point.

We can consider a *spectrogram* as a two-dimensional matrix of real values, where the horizontal dimension corresponds to time, reading from left to right, and the vertical dimension corresponds to frequency (or pitch). Note that we are deliberately ignoring *intensity* information, which can be ambiguous due to variation in

[3] The most general case of elastic image matching is NP-Complete [14].

the distance between the mouse and the microphone. Figure 1 shows the waveform and spectrogram of a mouse call. We chose the time/frequency parameters in a way that make the vocalization's frequency contours clearly visible in the spectrogram. In Appendix B we present more detail on how we produced spectrograms.

As noted before, we plan to index and mine mouse vocalizations in the symbolic space. In order to do this, we must first examine the spectrogram to extract *syllables*:

Definition 3: A *syllable* is a discrete atomic unit of sound separated by silence from other sound units. It consists of continuous marking on a spectrogram.

Figure 5 shows a spectrogram that contains seven *syllables*. For clarity, we encircled each syllable with a gray line. Each syllable is approximately 50 to 100 milliseconds long.

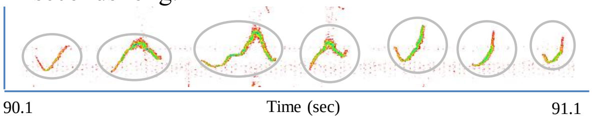


Figure 5: A snippet spectrogram that has seven *syllables*

The reader will appreciate that the syllables shown in Figure 5 appear to fall into discrete classes; for example, the last three are very similar to each other, and very different to the ones that precede them. Thus, we believe that we can meaningfully speak of a syllable's *type*:

Definition 4: A *syllable type* is a category of syllables observed regularly in a mouse vocalization, distinct from other syllable types. We will also refer to the *syllable type* as the *class* of syllables.

How many types of syllables are there? There is no universal agreement, with various researchers claiming from four [12] to ten [28]. In many cases the syllables are unambiguous; most of the discrepancy comes from simple syllables which can vary in a linear way. For example, there is a syllable that is an upward frequency modulated sweep rather like a forward slash ('/'). This slash can appear at various angles, from nearly horizontal to nearly vertical. Some researchers consider this as a single syllable and others further discretize it into two or more "angle" classes [11][28]. Clearly it would not be fruitful for us to try to "solve" this issue here. More importantly, it is not necessary. For example, if we search (using Google) for "*Jörg Sørensen*" we get almost the same result as if we search for "*Jorg Sorensen*". Similarly, we can push the robustness of syllable mapping to the higher-level indexing and mining algorithms.

Nevertheless, in order to make progress we need some initial tentatively labeled data. The two authors who are domain familiar (S.R. and K.R.) hand annotated a subset of the data to provide us *ground truth* data:

Definition 5: A ground truth (G) dataset is a set of annotated syllables that have been classified by expert human intervention. Each class in the ground truth may be represented by one or multiple exemplars.

We defer details about our ground truth dataset to Section 4.3.

Assuming we can extract and classify all the syllables in a given spectrogram, we are in a position to represent the data with a string of discrete symbols, where each symbol corresponds to a class label of a syllable present in the original space. As a result, instead of data mining in the original audio or image (spectrogram) space, we can work in the more efficient and compact string space. Moreover, we can take advantage of algorithms and data structures that are only defined for discrete data, such as hashing, Markov models, string edit distance, suffix trees [34], etc.

Finally, as we have already hinted at above, it is useful to do some preprocessing of a spectrogram prior to using our extraction algorithm. We call this step the *idealization* of a spectrogram. As the exact method of idealization is not critical to our work, we defer details to Appendix B. Figure 6 illustrates the result of idealization on a typical spectrogram.

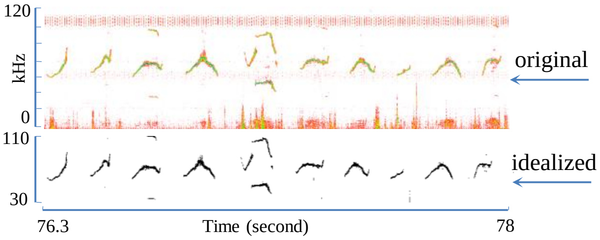


Figure 6: *top*) Original spectrogram, *bottom*) Idealized spectrogram (after thresholding and binarization)

3.1 A Brief Review of GHT

Having defined syllables, we are almost in a position to discuss how we can extract and classify them. Our basic idea is to scan the spectrograms for connected sets of pixels, which we refer to as candidate syllables, and compare these candidate syllables to all of the items in our ground truth dataset *G*. If the candidate is sufficiently similar to a labeled example, it is symbolized with that example's class label.

This opens the question of *which* distance measure to use to consider if a set of pixels is "sufficiently similar". We desire a measure that is fast, robust to the inevitable noise left even after idealization, and at least somewhat invariant to the significant intraclass variability we observe. After careful consideration and provisional tests of dozens of possibilities, we converged on a distance measure based on the Generalized Hough Transform [2].

The Hough Transform [13] was introduced as a tool for finding well-defined geometric shapes (lines, curves, rectangles, etc.) in images [8]. Ballard et al. generalized the idea and introduced the *Generalized Hough Transform* to detect arbitrary shapes in images [2]. The computation time of Ballard's method is relatively expensive. It takes quadratic time, $O(n_b^2)$, to calculate the distance between a pair of windows. Here, n_b is the number of black pixels in the window. However, Zhu et al. [36] augmented GHT in a way that reduces the amortized time for a *single* comparison significantly. Zhu et al. achieve speed-up by creating a computationally cheap tight lower bound to the GHT. Moreover, they present modifications to the classic definition that allow the measure to be symmetric and obey the triangular inequality, two properties that are highly desirable because they allow various algorithms to be used that exploit (or at least expect) these properties. We refer the interested reader to [36] for more details on GHT.

We claim that the GHT measure is ideal for this domain. However, this is difficult to show objectively because of the paucity of ground truth data. Indeed, our work is partly motivated by the lack of such objectively labeled data in this domain. Currently we have just a few hundred hand-labeled items in our ground truth dataset (cf. Section 4.3). However, we can demonstrate our claim with large-scale objective classification experiments on a very similar problem. As we show in Figure 7, handwritten Farsi digits are a surprisingly good proxy for mouse syllables.

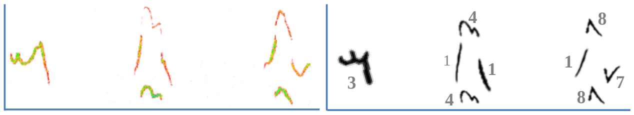


Figure 7: *left*) A real spectrogram of a mouse vocalization can be approximated by samples of handwritten Farsi digits (*right*). Some Farsi digits were rotated or transposed to enhance the similarity

We obtained a dataset of Farsi digits comprised of a 60,000/20,000 train/test split [9][24] and tested the GHT measure by using it to perform one-nearest neighbor classification. There is only one parameter to set, the *resolution* at which we view the data, and following the suggestion of [36] we simply choose 20×20 pixels.

Using GHT we obtained an error rate of 4.54%. This is very competitive with *all* published results on this dataset that we are aware of. For example, Ebrahimpour et al. test on ten percent of this dataset, reporting a 4.70% error rate for a Mixture of RBF Experts approach [9]. Razavi et al. test over twelve combinations of parameters for several neural-network based approaches, reporting a best error rate of 9.81% [24]. Borji et al. performed extensive empirical tests on this dataset, testing five different algorithms, each with four parameter choices. Of the *best* twenty reported error rates, the mean was 8.69% [5]. It is important to note that all these methods were optimized for this task, with 4 to 8 parameters being set. In contrast, we set just one parameter before seeing any test (or even *training*) data. In summary, these experiments strongly suggest that the GHT method is at least sensible for this domain, an idea which is confirmed by the results in Section 4.3 and elsewhere in this work.

4 SYLLABLE EXTRACTION/CLASSIFICATION

While the focus of our work is on the possibilities of analyzing and mining massive archives of data once it has been symbolized, the symbolizing step itself is currently unsolved. That is to say, there are currently no tools for *automatic* classification of mice vocalizations. Dr. Maria Luisa Scattoni, a domain expert and the editor of a recent special issue of the Journal *Genes, Brain and Behavior* devoted to mouse vocalizations [26], confirmed to us that she is unaware of *any* classification tools [27].

It is important to recognize that not all the connected sets of pixels in a spectrogram are syllables. Even when the data collection is conducted with the greatest of care, the data is still replete with non-mouse vocalization sounds, such as the mice interacting with the feeding apparatus, miscellaneous sounds from the lab (doors slamming, human speech, etc.), and electronic noise in the recording equipment. Thus, we treat symbolization as a two-step process. In the next section we consider the task of candidate syllable *extraction,* and then given this set of tentative syllables, we consider the syllable *classification* problem in Section 4.2. Note that this means our classification algorithm must be able to assign objects to a special "*non-mouse-utterance*" class when necessary.

4.1 Extracting Candidate Syllables

We use the algorithm in Table 1 to extract all the candidate syllables from the spectrogram of a mouse vocalization.

Table 1: Extract candidate syllables

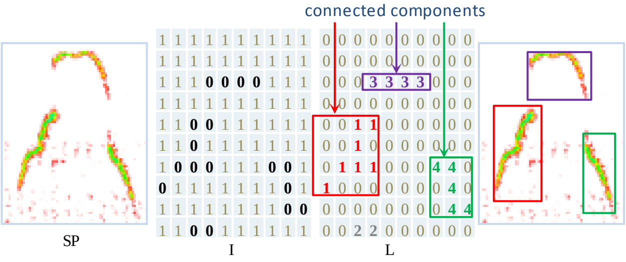
```
Algorithm 1 ExtractCandidateSyllables(SP)
Require: spectrogram of a mouse vocalization
Ensure: set of candidate syllables
1:  $I \leftarrow$ idealized spectrogram
2:  $L \leftarrow$ set of connected components in $I$
3:  $R \leftarrow$ row index of connected points
4:  $C \leftarrow$ column index of connected points
5:  $V \leftarrow$ value of connected points // value ranges from 1 to $|L|$
6:  $[A\ B] \leftarrow$ sort($V$, 'ascend') // $A$ has values of $V$ sorted and $B$ has the index
7:  $S \leftarrow []$       // set of candidate syllables in $SP$, initially empty
8:  $c_1 \leftarrow d_{min}$, $c_2 \leftarrow d_{max}$ // min and max duration of a syllable
9:  $j \leftarrow 1$, $k \leftarrow 1$
10: for $i \leftarrow 1$ to $|L|$ do {every connected component $l_i$ in $L$}
11:    $n \leftarrow 1$
12:    while $A(k)=i$ do
13:       $RW_{l_i}(n) \leftarrow R(B(k))$ // $RW_{l_i}$ contains row indices of $l_i$
14:       $CL_{l_i}(n) \leftarrow C(B(k))$ // $CL_{l_i}$ contains column indices of $l_i$
15:       $n \leftarrow n + 1$
16:       $k \leftarrow k + 1$
17:    $m \leftarrow L(\min(RW_{l_i}):\max(RW_{l_i}), \min(CL_{l_i}):\max(CL_{l_i}))==i$
                                  //minimum bounding rectangle (MBR) of $l_i$
18:    $[r\ c] \leftarrow$ size of $m$
19:    if $|c| < c_1$ or $|c| > c_2$
20:       continue                         // filter out noise
21:    else
22:       $S_j \leftarrow m$
23:       add $S_j$ to $S$
24:       $T1_j \leftarrow \min(CL_{l_i})$            // start time of $S_j$
25:       $T2_j \leftarrow \max(CL_{l_i})$            // end time of $S_j$
26:       $j \leftarrow j + 1$
27: return $S$, $T1$, $T2$ // candidate syllables in $SP$ with start/end times
```

Instead of extracting candidate syllables from the original spectrogram (*SP*) we use idealized version (*I*) of *SP*, as it produces fewer false negatives to be checked. *SP* is *idealized* (as in Figure 6) using the method described in Appendix B. In line 2, we convert the matrix *I* into a set of connected components, L. L has the same size as I, but it has the connected pixels marked with number 1 to $|L|$. The set of candidate syllables in SP is initialized with an empty set in line 7.

As noted in the previous section, a syllable is a contiguous set of pixels in a spectrogram; we can thus consider it as a set of connected points in I. The **for** loop in lines 10-26 is used to search for a connected component l_i in I. In order to make the search time linear to the number of candidate syllables, in lines 3-5 while creating L (a set of connected components), we save the row and column indices and also the values of all the connected points in arrays R, C and V, respectively. In line 6 we sort the array V in ascending order and save indices in B. In the **while** loop in lines 12-16, we use the indices in B to find the row and column indices of a connected component l_i in I. We use the minimum and maximum values of the row and column indices to extract the MBR (minimum bounding rectangle) of l_i.

Recall that not all of the connected components are candidate syllables. The idealized spectrogram is still replete with non-mouse vocalization sounds. To speed up the classification algorithm presented in Table 2, we filter out those noises. In the **if** block of lines 19-20 we check the duration of a connected component l_i and include those l_i in S which are within the range of thresholds c_1 and c_2. Since the minimum and maximum duration of a syllable can vary slightly across different mice, the values of c_1 and c_2 should be set after manual inspection of a fraction of the data. In our experiments, we set the values to 10 and 300, respectively. In lines 24-25, we save the start time and end time of a syllable, as they are used for subsequent analysis. Figure 8 visually describes our method. Our algorithm runs faster than real time, and thus does not warrant further optimizations for speed.

In Figure 8, we present a snippet spectrogram SP, matrices corresponding to the idealized version of the spectrogram I and connected components L. For brevity, original matrices for I and L are resized to 10x10. Finally we mark the MBRs of the candidate syllables in the snippet spectrogram.



1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	0	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1	1	1
1	1	0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	0	0	1
0	1	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	0	0
1	1	0	0	1	1	1	1	1	1

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	3	3	3	3	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	1	1	1	0	0	0	4	4	0
1	0	0	0	0	0	0	0	4	0
0	0	0	0	0	0	0	0	4	4
0	0	2	2	0	0	0	0	0	0

Figure 8: *from left to right*) A snippet of a spectrogram, the resized matrix corresponding to an idealized spectrogram *I*, the resized matrix corresponding to the set of connected components *L*, and the MBRs of the candidate syllables

4.2 Classifying Candidate Syllables

After running the algorithm in Table 1 we will have a set of *candidate* syllables and will be in a position to classify them. For this purpose, we need a set of annotated syllables, which we call *Ground Truth* (G), and a set of thresholds for each class of syllables. For the moment, assume that we have these; in Section 4.3 we present a detailed explanation of how the ground truth is created.

The reader may wonder why we need a threshold; can't we simply assign the candidate syllable to the class of its nearest neighbor? The answer is no, because a large fraction of the candidate syllables will inevitably be noise, and it is the thresholds that allow us to reject them.

Given a set of candidate syllables S and ground truth syllables (G) with their matching thresholds (τ), the algorithm shown in Table 2 classifies some syllables in S, and rejects all others.

Table 2: Syllable classification algorithm

Algorithm 2 *ClassifyCandidateSyllables*(S, G, T)

Require: candidate syllables, ground truth, set of thresholds
Ensure: set of labeled syllables

```
1:  // $S = \{S_1, S_2, \ldots S_n\}$ is set of candidate syllables,
2:  // $G = \{G_1, G_2, \ldots G_m\}$ is ground truth and
3:  // $\tau = \{ \tau_1, \tau_2, \ldots \tau_{11}\}$ is set of thresholds
4:  // normalize all the syllables in $S$ and $G$ to equal size
5:  // initialize all syllables' class $\{c_{S1}, c_{S2}, \ldots\}$ to 0 or not classified
6:  for $i \leftarrow 1$ to $n$ do        // $|S| = n$
7:      NNdist = inf            // initially set the NN distance to infinity
8:      for $j \leftarrow 1$ to $m$ do    // $|G| = m$
9:          dist ← dist_GHT($S_i$, $G_j$)  //calculate GHT between $S_i$ and $G_j$
10:         if dist < NNdist
11:             NNdist ← dist   // update nearest neighbor distance
12:             $NN \leftarrow j$          // update nearest neighbor ($NN$)
13:     if NNdist $\leq \tau(C_{NN})$     // $C_{NN}$ is the class label of $G_{NN}$
14:         $c_{Si} \leftarrow C_{NN}$
15: return $\{c_{s1}, c_{s2}, \ldots c_{sn}\}$ // class labels of all candidate syllables
```

In order to classify a candidate syllable we look for its nearest neighbor in G in the **for** loop of lines 8-12. In the **if** block of lines 13-14, we assign the class label of the nearest neighbor to a candidate syllable only if the distance between a candidate syllable and its nearest neighbor from G is less than the threshold of the nearest neighbor's class.

4.3 Ground Truth Editing

The algorithm in the previous section requires a ground truth dataset augmented with thresholds. There appears to be no way to obtain this, other than asking domain experts to annotate some data. Fortunately, they only have to spend one or two hours labeling this data. Moreover, they are very motivated to do so, because once our extraction/classification system works, it can save weeks or months of tedious manual labor on future work (assuming that the initial annotations generalize and our tool is accurate, assumptions we explicitly test below) [26][27].

However, the human annotation of data is a non-trivial step. We found that even when we asked two experts from the same lab to label data (co-authors S.R. and K.R.) they disagreed on the labels of many instances. Moreover, each expert wanted to place some individual exemplars into two or more classes.

There are two different reasons why an expert might want to place individual exemplars into two or more classes:

- There might simply be some very subtle class distinctions. For example, the task of hand labeling animals as {alligator, crocodile, elephant} would probably have indecisive people assign some crocodilians[4] to two classes.
- There might be logically overlapping classes (in spite of our best efforts to avoid this). For example, if we had classes {mammal, carnivore, bird}, we would clearly have some animals that belong in two of those classes.

Our initial results suggest that *both* problems occur in this domain. Below we discuss our efforts to mitigate this.

The domain experts provided us with an initial tentative set of sixteen syllable classes, as shown in Figure 9.

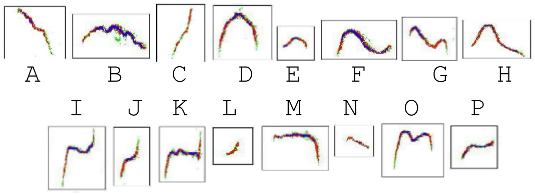


Figure 9: Sixteen syllables provided by domain experts

These sixteen syllable classes were based on both their significant experience in collecting mice vocalizations data and an extensive survey of the literature [26][29].

As our starting point, we extracted candidate syllables from the first ten minutes of a 32-minute-long recording ('03171102CTCT'). We asked the domain experts to classify the data into these sixteen classes (or the special class: non-syllable).

The experts did not find any example of class O, and the overall agreement on other classes was poor. Examining the confusion matrix, we discovered that most of the confusion was concentrated on a handful of classes. For example, D and E, and G and H were frequently confused. In order to reduce this ambiguity, we merged the frequently confused classes and deleted a few classes (O and P) (c.f. Figure 11). Thus, the number of classes reduced to ten with a total of 260 labeled syllables. Using those 260 instances we ran our syllable extraction and classification algorithm on the entire trace. The classification result was then validated by a domain expert (S.R.). She reassigned many instances, discarded a few dubious examples and labeled some instances from the non-syllable as a new class, **k**. Finally, we were left with a total of 692 labeled syllables of eleven classes.

To see how well our GHT measure agreed with the domain experts we used it to conduct leave-one-out 1-Nearest Neighbor classification of the 692 labeled syllables. We obtained an accuracy of 83.82%. While this is a reasonable accuracy and approaching the inter-expert agreement, we attempted to improve on this with *data editing* [31][22][32]. Data editing (also known as *numerosity reduction* or *condensing*) is the technique of judiciously removing instances from the training set in order to improve generalization accuracy (and, as a fortunate side effect, reduce the time and space requirements for classification).

While there are many data editing techniques available, we opted for a simple variant of forward search [31]. We first ensured our datasets had one member of each class by choosing the most typical instance from each class. Here *most typical* means the instance that had the minimum sum of distance to all other members of the same class. We call this set *C*.

We then began an iterative search for an instance we could add to *C* that would improve (or make the minimal decrease in) the leave-one-out classification accuracy of *C*. Since there are many tying instances (especially in the early stages of this search) we break ties by choosing the instance that has the minimal distance to its nearest neighbor (of the correct class). Figure 10 shows the progress of the accuracy of leave-one-out as we add more instance to *C* (bold/red line).

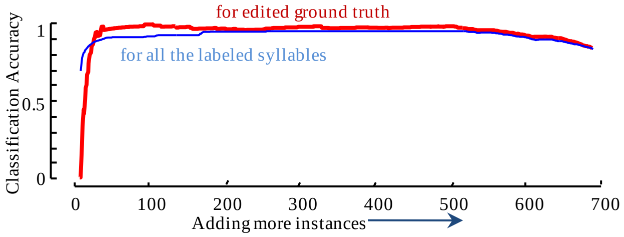


Figure 10: Thick/red curve represents the accuracy of classifying syllables of edited ground truth. Thin/blue curve represents the accuracy of classifying 692 labeled syllables using edited ground truth

We can see that the accuracy quickly climbs to a maximum of 99.07% when there are just 108 syllables in the edited ground truth, and thereafter holds steady for a while before beginning to decline.

It is well understood that greedy search strategies for data editing run a risk of over fitting, or at least producing optimistic results [31][22][32]. As a sanity check we tested to see how well various-sized training sets *C* would do if we evaluated them on the *entire* 692 instances. This is shown in Figure 10 with the fine/blue line. These results also suggest that a smaller set of instances is better than using all instances and that our search produced only slightly optimistic results. Based on this, we use the set $|C| = 108$ as the ground truth for the remainder of this work. In Figure 11 we present the eleven classes.

At this point we have a small set of robust exemplars for our eleven classes. We still need to set the thresholds. We do this by simply computing the GHT distances between every annotated syllable to its nearest neighbor from the same class. Then the mean plus two standard deviations is chosen as the threshold distance for that class. We can best judge the correctness of the threshold values by examining the high accuracy achieved in Figure 10.

[4] *Crocodilians* is the order that includes the alligator, caiman, crocodile, and gharial families.

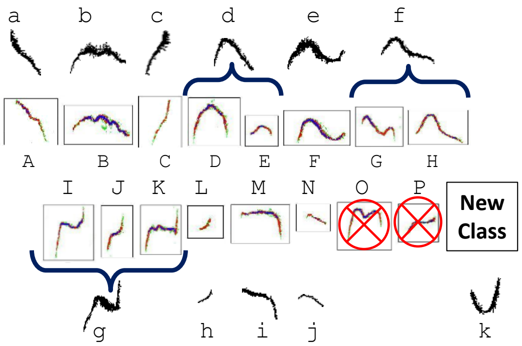


Figure 11: Ambiguity reduction of the original set of syllable classes. Representative examples from the reduced set of eleven classes are labeled as small letters

5 DATA MINING MICE VOCALIZATIONS

We are finally in a position to discuss data mining algorithms for large collections of mouse vocalizations. Note that while in every case the algorithms operate on the discrete symbols, we report and visualize the answers in the original spectrogram space, since this is the medium that the domain experts are most comfortable working with and it is visually intuitive.

5.1 Clustering Mouse Vocalizations

We begin with a simple sanity check to confirm that the automatic extracted syllables can produce subjectively intuitive and meaningful results, and that a direct application of a proposed image processing method cannot [9][30]. In Figure 12 we show a clustering of eight snippets of mouse vocalization spectrograms using the string edit distance on the extracted syllables.

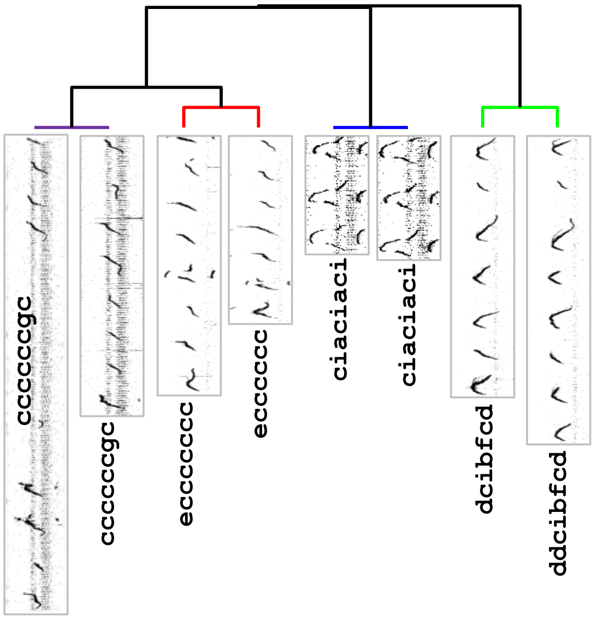


Figure 12: A clustering of eight snippets of mouse vocalization spectrograms using the string edit distance on the extracted syllables (spectrograms are rotated 90 degrees for visual clarity)

This figure illustrates an obvious invariance achieved by working in the symbolic syllable space; the method is invariant to the length of the patterns in the original space. The most logical way to achieve this for correlation-based methods is to compare two sequences of different lengths by sliding the shorter one across the longer one and recording the minimum value. Figure 13 shows the result of doing this. In the next section, we will see that it is possible to find similar regions automatically.

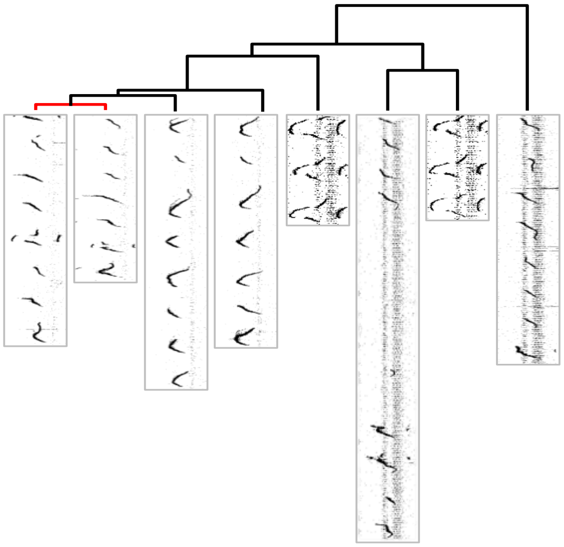


Figure 13: A clustering of the same eight snippets of mouse vocalization shown in Figure 12 using the correlation method. The result appears near random

5.2 Query by Content in Mouse Vocalizations

In addition to clustering, we can also search for any specific query in a mouse vocalization. There are two ways we can do this. First, we can simply "type in" queries based on experience with data. For example, we have noticed that long runs of **c** are often observed (c.f. Figure 12); we could ask similarly if long runs of **e** are observed, by querying the string **eeeeee**, etc.

Second, given either a sound file or a query high-quality image (including a screen dump from a paper), we can automatically label the syllables using the algorithms in Table 1 and Table 2, to produce a symbolic query. In Figure 14 we have done exactly this with a figure taken from [11].

Note that while irrelevant aspects of the image presentation are different (the published work is significantly cleaner and the syllables are "finer", perhaps due to superior data collection/cleaning), our algorithm is invariant to this and manages to find truly similar subsequences.

In Figure 15 we present another example of query-by-content and include the four best matches from two different types of mice (*control* (*CT*) and *Fmr1* KO, see Appendix A for more details on the mice). The query image is a screen dump from [12].

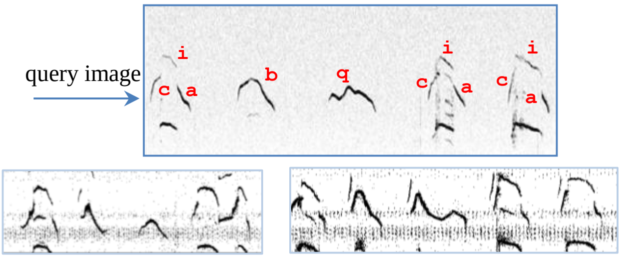


Figure 14: ***top*****) A query image from [11], The syllable labels have been added by our algorithm to produce the query** `ciabqciacia`**,** ***bottom*****) the two best matches found in our dataset; corresponding symbolic strings are** `ciafqcicia` **and** `ciqbqcaacja`**, with edit distance 2 and 3, respectively**

We have omitted until now a discussion of *how* we efficiently answer queries. While we plan to scale our work to a size that will eventually require an inverted index or similar text-indexing technique, our dataset currently only contains on the order of tens of thousands of syllables, and thus allows for a sub-second brute force search. The fact that we can search data corresponding to many hours of audio data in few seconds is a vindication of our decision to data mine mice vocalizations in the symbolic space.

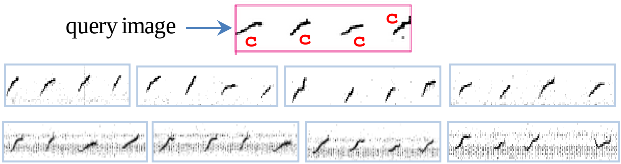


Figure 15: ***top*****) The query image from [12] was transcribed to** `cccc`**. Similar patterns are found in** ***CT*** **(*****first row*****) and** ***KO*** **(*****second row*****) mouse vocalizations in our collection**

5.3 Motif Discovery in Mouse Vocalizations

In Section 3 we noted that working in the symbolic space allows us to adapt ideas from bioinformatics to our domain. One example of a useful idea we can borrow from the world of string processing/bioinformatics is the concept of *motif* [7]. DNA sequence motifs are short, recurring patterns in DNA that are presumed to have a biological function. Motif discovery has proved to be a fundamental tool in bioinformatics, because it enables dozens of higher level algorithms and analyses, including defining genetic regulatory networks and deciphering the regulatory program of individual genes. To the best of our knowledge, no one has considered computational motif discovery for mouse vocalizations[5]. To redress this, we begin by defining a *motif* for our domain:

> ***Definition 6:*** A motif is a pair of non-overlapping syllable sequences which are similar. In particular, a *t*-motif is a motif pair that is no more than *t* distances apart under some distance function such as string edit distance.

Figure 16 presents an example of motif we discovered in our data that are 1-edit distance apart.

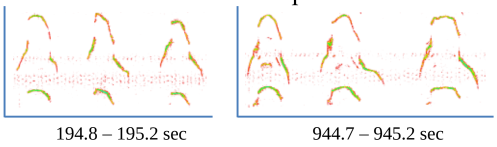


Figure 16: A motif that occurred in two different time intervals of a vocalization. The ***left*** **and** ***right*** **one correspond to the symbolic strings** `ciaciacia` **and** `ciacjacia`

As mice can produce harmonic sounds, we sometimes find multiple syllables in the same time stamp, as in the example of Figure 16; in such cases we classify the syllable with a higher frequency and ignore the syllable with a lower frequency.

Given our definition, how can we find motifs in a large dataset? The bioinformatics literature is replete with suggested algorithms. However, as with the query-by-content example in the previous section, our problem is much easier in scale because of our decision to work in the symbolic space. A typical half-hour recording of mouse vocalizations may have as many as 4,000 syllables, a large number, but clearly not approaching genome-sized data. Thus, we content ourselves with a brute force algorithm for now.

As shown in Table 3, to find all *t*-motifs we simply do a brute force search over all possible pairs of substrings, at increasing lengths starting from length *t* +1, until no more motifs are discovered. The algorithm reports all *t*-motifs sorted longest first.

Table 3: Motif discovery algorithm

```
Algorithm 3 MotifDiscovery(SP, S, t)
Require: a spectrogram, a string consisting of class labels of all syllables extracted from the spectrogram and edit distance
Ensure: set of motifs
//$ts = \{ts_1, ts_2, \dots ts_n\}$, start time of all syllables in $S$
//$te = \{te_1, te_2, \dots te_n\}$, end time of all syllables in $S$, $|S|=n$
$l \leftarrow t+1$   // length of motifs initially set to $t$+1, t is edit distance
$\sigma_l \leftarrow \{\sigma_{l1}, \sigma_{l2}, \dots \sigma_{l|\sigma l|}\}$ // set of strings of length $l$ that occur at least twice in $S$
while true
  $l \leftarrow l + 1$
  $\sigma_l \leftarrow$ [], $MTF_l \leftarrow$ []   // set of motifs of length $l$
  for $i \leftarrow 1$ to $|\sigma_{l-1}|$ do // for each repeated string of length $l$-$1$
    for ç ← a to k do  // search for all combinations
      $st \leftarrow$ add ç to $\sigma_{l-1(i)}$
      $cnt \leftarrow$ find number of occurrence of $st$ in $S$
      if $cnt > 1$
        add $st$ to $\sigma_l$
        for $j \leftarrow 1$ to $cnt$
          $ts_j \leftarrow$ start time of first syllable in $st_j$
          $te_j \leftarrow$ end time of last syllable in $st_j$
          $sp_j \leftarrow$ part of $SP$ from $ts_j$ to $te_j$
        add set of $sp_j$ to $MTF_l$
    if $\sigma_l$ is empty, break
return $MTF$ //set of motifs of all possible lengths
```

The algorithm requires a spectrogram *SP* and a string *S*, which consists of class labels of the syllables in *SP*. We

[5] There are published examples of repeated patterns found in mice vocalizations; however, all were discovered by manual inspection.

start our algorithm with a set of substrings of length *l* that occur at least twice in *S*. *l* is initially set equal to *t*+1, where *t* is the allowed edit distance. In the **while** loop of lines 5-19, we increase the length of the substring until we no longer find repeated substrings. In the nested **for** loops in lines 8-19, we add a syllable type to each of the substrings in σ_{l-1} and search for it in *S*. Each repeated substring of length *l* corresponds to a motif of length *l*. By repeated substrings we mean all those strings which are no more than *t*-edit distance apart from the query string.

In order to report the corresponding motifs from the original spectral space, we use the start time and end time of the first and last syllable of a substring to extract the motif from *SP*. The **if** block of lines 12-18 is used for this purpose.

5.3.1 *Assessing Motif Significance*

The task of finding all motifs is computationally tractable, but it leaves us with the more challenging problem of assessing their significance. Our motif discovery algorithm always reports the longest motifs in our data, but do they represent some *meaningful* conserved vocal behavior, or might they have been produced by chance, by a mouse randomly babbling?

The task of assessing motif significance in DNA is still an area of active research, and our task is arguably more difficult, given our larger alphabet and the inherent uncertainty of the syllables' true labels. Thus, while we do not claim to have the final word on motif significance here, for completeness we will show a tentative idea that gives plausible and intuitive results.

In order to assess the significance of motifs of length *l*, we calculate the *z-score* of each substring of length *l* [1][10]. In Figure 17.*top*, we present the distribution of *z*-scores of all of the substrings of length nine from a *KO* mouse recording that has more than four thousand syllables in it. The edit distance is set equal to one. We show two sample motifs from the spectral space that have *z*-scores approximately equal to two and three respectively.

Note that our motif ranking algorithm does allow significant redundancy. For example, most of the eleven motifs with a z-score of about three are variants on *strings consisting mostly of* **c**. There exist several techniques in the bioinformatics literature for mitigating this problem [1], but for brevity we omit the discussion.

The *z*-score is a well-known technique in bioinformatics for assessing motif significance. It is a standard quantitative measure of over-representativeness (and, sometimes, *under*-representativeness) of an existing pattern. There are several ways of calculating *z*-score depending on the assumptions made about the domain [1][10]. For our purposes, we calculate the *z*-score of a substring simply by subtracting the expected number of occurrences of the substring from its observed number of occurrences. If a string *S* has *n* symbols, then there are in total $(n - l + 1)$ substrings of length *l*. The *z*-score of the i^{th} substring is computed as follows:

$$z_i = f_i - e_i$$

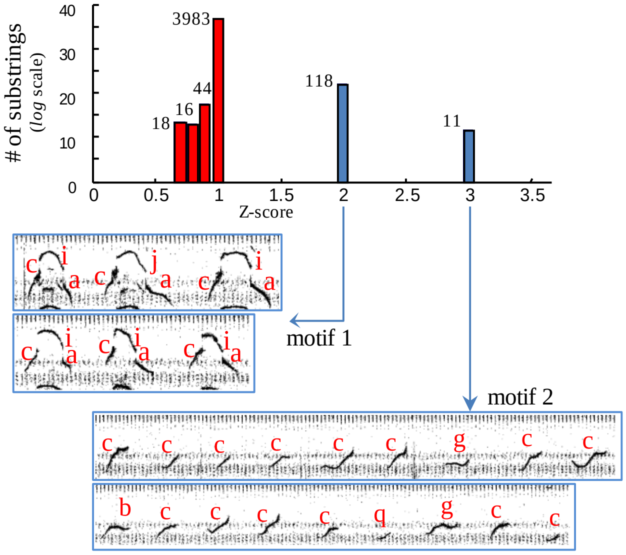


Figure 17: *top*) Distribution of *z*-scores, *bottom*) two sets of motifs from spectral space with a z-score of approximately two and three, respectively

In the above formula, f_i is the observed frequency of the substring s_i in *S*, and e_i is the expected number of occurrences of s_i in *S*. The expected number of occurrences e_i of a substring is computed as follows:

$$e_i = (n - l + 1) \times (P1_i \times P2_i \times \ldots \times Pl_i)$$

Here, $P1_i, \ldots\ Pl_i$ are the probabilities of occurrences of each symbol of s_i in *S*. For example, given the first symbol of s_i, if **a** and **a** appears *m* times in *S*, $P1_i$ would be m/n. While computing the number of occurrences of a substring s_i, we consider all the substrings that are no more than *t*-edit distance apart from s_i.

5.4 Contrast Sets for Mouse Vocalizations

A fundamental task in investigative data analysis is to determine the differences between two or more groups. The *groups* in question may be natural, such as male vs. female, or induced by the experimenter, such as the genetic manipulations inherent in knockout vs. control mice.

The task of determining the differences between groups from a data mining perspective has been formalized as *contrast set mining*, as elucidated by Bay and Pazzani [3]. There are a plethora of algorithms and heuristics in the literature for contrast set mining. We refer the interested reader to [6] and the references therein for an overview of the growing literature on this application.

In order to determine the discriminative patterns between knockout and control mice, we have converged on an adaptation of *information gain* [16][23].

Intuitively, we treat each recording session as a class-labeled (i.e. knockout/control) object and consider the information gain of all substrings of length *l* as a criterion to separate the two classes. We hope to find substrings that always (or very frequently) occur in one class, but not (or very rarely) in the other.

Let S be the set of all of the substrings of length l that occur in any of the mice vocalizations. The information gain IG_i for a substring s_i is calculated as follows:

$$IG_i = E(S) - E(S|s_i)$$

$E(S)$ is the entropy of the mouse class and $E(S|s_i)$, is the entropy of the mouse class given the substring s_i. If k of the substrings in S belongs to the knockout, c of the substrings belong to the control and n is the total number of substrings in S, then the probability of a mouse being KO and CT are computed as,

$$P_{KO}(mouse\ class = KO) \ = k/n, \text{ and}$$
$$P_{CT}(mouse\ class = CT) \ = c/n$$

We compute the entropy $E(S)$ as follows:

$$E(S) = -P_{KO} \times \log_2 P_{KO} - P_{CT} \times \log_2 P_{CT}$$

Given that n_i is the total number of substrings in S that are no more than t-edit distance apart from s_i in S, k_i is the total number of substrings that belong to KO and c_i is the total number of substrings that belong to CT, we can compute the entropy of s_i as:

$$E(s_i) = -(k_i/n_i) \times \log_2(k_i/n_i) - (c_i/n_i) \times \log_2(c_i/n_i)$$

The entropy of the mouse class given a substring s_i is computed as:

$$E(S|s_i) = (n_i/n) \times E(s_i)$$

In Figure 18 we present two sets of syllable sequences that are significant in KO and CT mice, respectively. For brevity we include only few examples from each set.

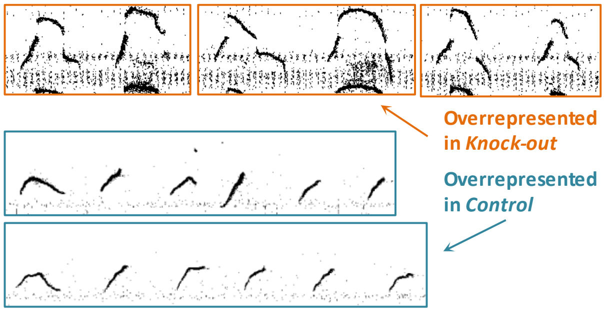


Figure 18: Examples of contrast set phrases. *top*) Three examples of a phrase `ciacia` **that is overrepresented in KO, appearing 24 times in KO but never in CT. *bottom*) Two examples of a phrase** `dccccc` **that appears 39 times in CT and just twice in KO**

We conducted an extensive review of the literature to see if these patterns were previously known to the community. The **`ciacia`** phrase that we found is overrepresented in KO is essentially identical to the `hdu`[6] phrase (or rather, a repeat of this phrase, `hduhdu`) described by Holy and Guo [12]. They found this pattern to be rare, comprising less than 2% of the vocalizations made by a test set of 45 CT males. While this is suggestive, it is not clear that our abundance of this phrase is due to the knockout condition. Grimsley et al. [11] found that a combination of flat (less than 6 kHz of modulation) and upward FM syllables comprised the most common motif in adult mice vocalizations. This pattern is consistent with the overrepresented phrase in CT, **`dccccc`**. As before, we must be careful to temper any claim that the knockout gene *caused* the paucity of this phrase in KO. However, this experiment shows the utility of our algorithm in creating avenues for further "wet" experiments.

It is important to note that the *individual* syllables cannot be used as contrast sets for different types of mice. If we calculate the frequency distributions of different types of syllables in KO and CT mice vocalizations, we find an almost identical distribution for both (full distributions are at [35]). Syllable **`c`** has the highest frequency, whereas syllable **`k`** is very rare. This is rather like the situation with the natural languages English and French. Based on just *letter* frequencies, it would be essentially impossible to tell the original language of a text, but the presence of a *phrase* such as *`bonjour`* or *`fait accompli`* would be a strong clue as to the language we are dealing with.

6 CONCLUSION

Many of the questions relating to the nature vs. nurture debate for mouse vocalizations are still open. As we were conducting this research, PLoS ONE took the unusual step of publishing two papers on mouse vocalizations that *explicitly* contradicted each other [11][15]. We believe that an at least partial solution to reduce such uncertainty is to simply examine *much* larger datasets. Moreover, NIH has just announced a $110 million project to create 5,000 strains of KO mice in the next five years, a project that will surely produce tens of terabytes of data. As we have discussed earlier, the main bottleneck in analyzing mice vocalizations thus far has been human effort. We hope that the ideas contained herein will help mitigate this. With this in mind, we have made all our code and data publicly available at [35].

ACKNOWLEDGEMENT

Funded by NSF grants 0803410 and 0808770.

REFERENCES

[1] A. Apostolico, M. E. Bock, S. Lonardi, *Monotony of Surprise and Large-Scale Quest for Unusual Words*, Journal of Computatinal Biology, vol(10):283–311(2003).

[2] D. H. Ballard, *Generalizing the Hough transform to detect arbitrary shapes*, Patt. Recognition, 13(2): 111-22 (1981).

[3] S. D. Bay, M. J. Pazzani, *Detecting change in categorical data: mining contrast sets*, KDD '99, pp. 302–306.

[4] H. Bhattacharya, et al., *Universal features in the singing of birds uncovered by comparative research*, Our Nature, 6: 1–14, (2008).

[5] A. Borji, M. Hamidi, F. Mahmoudi, *Robust handwritten character recognition with features inspired by visual ventral stream*, Neural Processing, 28(2): 97–111, (2008).

[6] Contrast Data Mining: Methods and Applications, *http://www.cs.wright.edu/~gdong/ICDMtutorial.ppt*

[6] Note that the syllable labels used by various groups are arbitrary; thus, **`c`** ≈ `h`, **`i`** ≈ `d`, **`a`** ≈ `u`.

[7] P. D'Haeseleer: *What are DNA sequence motifs?*, Nat Biotechnol, 24:423-425, (2006).

[8] R. O. Duda, P. E. Hart, *Use of the Hough transform to detect lines and curves in pictures*, Comm. ACM 15: 11–15, (1972).

[9] R. Ebrahimpour, A. Esmkhani, S. Faridi, *Farsi handwritten digit recognition based on mixture of RBF experts*, IEICE Electronics, 7(14): 1014–19, (2010).

[10] P. G. Ferreira, P. J. Azevedo, *Evaluating deterministic motif significance measures in protein databases*, Algorithms for Molecular Biology, 2:16 (2007).

[11] J. M. S. Grimsley, et al., *Development of Social Vocalizations in Mice*. PLoS ONE 6(3): e17460 (2011).

[12] T. E. Holy, Z. Guo, *Ultrasonic songs of male mice*, PLoS Biol 3(12): e386, (2005).

[13] P. V. C. Hough, *Method and means for recognizing complex patterns*, U.S. Patent 3069654, (1962).

[14] D. Keysers, W. Unger 2003. *Elastic image matching is NP-complete*. Pattern Recogn. Lett. 24, 1-3, 445–453.

[15] T. Kikusui, et al. *Cross fostering experiments suggest that mice songs are innate*, PLoS One 6:e17721 (2011).

[16] R. L. Mantaras, *A Distance-Based Attribute Selection Measure for Decision Tree Induction*, ML, 6, 81–92, (1991).

[17] T. E. McGill, *Sexual behavior in three inbred strains of mice*, *Behaviour* 19: 341-350, (1962).

[18] D. W. Mosig, D. A. Dewsbury, *Studies of the Copulatory Behavior of House Mice (Mus musculus)*, Behavioral Biology 16: 463-473, (1976).

[19] National Centre for the Replacement, Refinement and Reduction of Animals in Research, *www.nc3rs.org.uk/*

[20] C. F. Olson, *Maximum-likelihood image matching*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(6): 853–857, (2002).

[21] J. B. Panksepp, et al., *Affiliative behavior, ultrasonic communication and social reward are influenced by genetic variation in adolescent mice*, PLoS ONE 4:e351 (2007).

[22] E. Pekalska, *Prototype selection for dissimilarity-based classifiers*, Pattern Recognition, 39(2): 189-208, (2006).

[23] J. R. Quinlan, *Induction of Decision Trees*, ML 1:81-106, 1986.

[24] S. M. Razavi, M. Taghipour, E. Kabir, *Improvement in Performance of Neural Network for Persian Handwritten Digits Recognition Using FCM Clustering*, Applied Sciences, 9(8): 898–906, (2010).

[25] URL (2011). Retrieved on September 2nd 2011 *www.avisoft.com/images/tutorial_rat_apm.gif*

[26] M. L. Scattoni, (Editor) Special interest section on mouse ultrasonic vocalizations. Genes, Brain and Behavior. Volume 10, Issue 1, pages 1–3, Feb 2011.

[27] M. L. Scattoni, (Personal Communication) Sep 2nd 2011.

[28] M. L. Scattoni, S. U. Gandhy, L. Ricceri, J. N. Crawley, *Unusual Repertoire of Vocalizations in the BTBR T+tf/J Mouse Model of Autism*, PLoS ONE 3: e3067, (2008).

[29] W. Shu, et al., *Altered ultrasonic vocalization in mice with a disruption in the Foxp2 gene*, Proc Natl Acad Sci U S A. 102(27): 9643–9648, (2005).

[30] S. Uchida, H. Sakoe. *A survey of elastic matching techniques for handwritten character recognition*, IEICE Trans' on Information and Systems, 1781-90 (2005).

[31] K. Ueno, X. Xi, E. J. Keogh, D. Lee, *Anytime Classification Using the Nearest Neighbor Algorithm with Applications to Stream Mining*, ICDM 2006: 623–632.

[32] D. R. Wilson, T. R. Martinez, *Reduction techniques for instance-based learning algorithms*, Machine Learning, 38: 257-286, Kluwer Academic Publishers, (2000).

[33] M. Wöhr, et al. *Communication impairments in mice lacking Shank1: reduced levels of ultrasonic vocalizations and scent marking behavior*, PLoS One, 6(6):e20631, (2011).

[34] R. Yan, P. C. Boutros, I. Jurisica, *A tree-based approach for motif discovery and sequence classification*, Bioinformatics 27(15): 2054-2061 (2011).

[35] J. Zakaria, Mouse Vocalization Mining Webpage, www.cs.ucr.edu/~jzaka001/mouse.html

[36] Q. Zhu, X. Wang, E. Keogh, S.H. Lee, *Augmenting the Generalized Hough Transform to Enable the Mining of Petroglyphs*, KDD 2009, pp. 1057–1066 (2009).

APPENDIX A: DESCRIPTION OF THE MICE

Our Knock-Out (KO) mice are *Fmr1* KO. Our mice were obtained from Jackson Laboratories and housed in an accredited vivarium with 12 hour light/dark cycles. *Fmr1* KO mouse is a valid model of the Fragile X Syndrome [28]. During mating trials control male mice were paired with control females, and *Fmr1* KO male mice were paired with *Fmr1* KO females. All mice used were virgins between 60 and 90 days [17][18]. During mating trials, mice were placed in a 28.8 x 21.6 x 28.8 cm enclosure. Ultrasonic vocalizations were recorded using a full spectrum Petterssen D1000x bat detector (250 kHz sampling rate) 5cm above the enclosure.

APPENDIX B: SPECTROGRAM DETAILS

Our algorithms are *very* robust and largely invariant to the exact details of how the spectrograms are created. Nevertheless, for completeness we give details here. We used a Matlab function to create the spectrogram:

```
[Y,F,T,P]=spectrogram(S,512,256,512,FS,'yaxis');
C = -10*log10(P);
```

Here, *S* is the audio signal and *FS* is the sampling frequency rate of the signal. The size of the hamming window is equal to 512 bits, the amount of overlap is 50% of the hamming window (i.e., 256 bits), and *NFFT* (number of frequency points used to calculate the discrete Fourier transforms) is set equal to 512. *F* and *T* are two vectors of frequencies and times at which the spectrogram is computed. *Y* is a matrix, each element of which represents an estimate of the time-localized frequency content of the signal *S*. The number of rows and columns in *Y* are equal to the size of *F* and *T*, respectively. *P* is a matrix that is equal in size to *Y* and represents the power spectral density of each segment of *Y*. The value of *P* is very low, so we take the negative log of *P* and multiply it by 10. This is the matrix (or bitmap) *C* we use to extract the syllables.

While recording a mouse vocalization, miscellaneous noise from the lab is captured in addition to the signal of interest. In order to exclude irrelevant data we set the upper and lower limit of the frequency band equal to 30 kHz and 110 kHz, respectively. However, some noise are still in the frequency band of the ultrasonic vocalizations. While binarizing the spectrogram, we replace values that are within 35 to 85 with 0 or a black pixel and the rest with 1 or a white pixel. These actions are more or less standard practice in the community.