# Machine Recognition of Squiggles in SETI Signal Data

Travis Chen (travis14@stanford.edu), Kenny Smith (smithken@stanford.edu)
Jason Wang (jasonkwang@stanford.edu)

## I. INTRODUCTION

The singular question, are we alone?, has boggled scientists for centuries. The SETI Institute operates the Allen Telescope Array (ATA) to observe star systems for radio signals which may provide evidence of extraterrestrial intelligence. Signal events that maintain an intensity above an empirically-set threshold across all frequencies are converted into spectrogram images through a sliding-window Fast Fourier Transform (FFT) and set aside for human analysis.

A key problem is being able to pinpoint significant patterns in the signal stream that are not associated with known interferences, such as aircraft RFI and narrow-band zero-drift signals. SETI is currently constructing a pipeline to stratify known interferences within signal streams in real time. In the past few years, an unknown subset of signals, inelegantly referred to as squiggles, has become increasingly prevalent. Squiggles are broadly defined as modulating signals, hand-curated by scientists at NASA; their origin is unknown. In fact, the quest to understand squiggles is an open problem posed by SETI [1]. Our project will be centered on this signal subset.

Using spectrogram waterfall plots collected from the ATA SETI dataset, we hope to make an open-source contribution in two ways: 1) perform supervised learning to classify squiggles against nonsquiggles and 2) conduct unsupervised learning to identify potential squiggle subgroups and their characteristics.

## II. RELATED WORK

In the specific domain space of squiggle analysis, little work has been done. For the past decade, SETI has worked to optimize its real-time processing algorithm to identify notable signal events that warrant human intervention. SonATA is a open-source software system that currently includes a post-processing package to identify common pulsar and linear, carrier-wave signals.

There has been a significant amount of academic literature dedicated to the domain of classifying unknown signals streams and feature extraction from spectrogram images. Iversen et al. [2] constructs a combined artificial neural network classifier to classify various unknown radio signal formats. Due to our limited data set of only 883 unlabelled squiggle examples, we believe training a neural network to classify between squiggle and non-squiggle signals would lead to gross overfitting. Perovic et al. [3] provides a landscape overview of spectrogram analysis, spanning topological feature extraction to Kalman filtering to identify dynamic signals. We hope to use an approach unique to squiggles: conversion of squiggles into discrete time series data points.

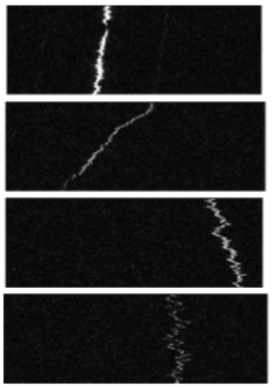

Fig. 1: Squiggle Spectrograms

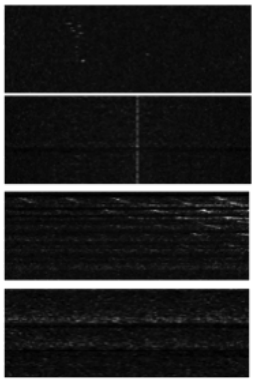

Fig. 2: Non-Squiggle Spectrograms

## III. DATASET

We used a dataset of 8490 spectrogram images provided to us by the SETI Institute. This includes a library of 883 hand-identified squiggles and 7607 nonsquiggles, attributed to known human interferences and random noise. Each spectrogram is comprised of a signal event that passed the SonATA pre-processing phase between the dates of 8/10/2014 and 11/2/2014. The PNG format is specified at 768x129 pixels, representing roughly 100MHz in bandwidth and 93 seconds, respectively (x-axis: frequency domain, y-axis: time domain).

## IV. METHODOLOGY AND PREPROCESSING

### A. Conversion of spectrograms into discrete timepoints

In exploring the spectrogram dataset, we arrived at the conclusion that each squiggle can be treated a discrete time series, by selecting one frequency from each time slice in a spectrogram. Our initial approach relied solely

on intensity, selecting the frequency with the maximum intensity from each timeslice. However, this approach failed to trace the squiggle accurately in the presence of strong background noise. Furthermore, many squiggles had gaps, sets of time slices over which the signal disappears almost entirely. To solve this issue of interpolation and to exclude outlying points, we sought the optimal path to minimize the following loss:

$$L(\mathbf{x}) = -\sum_{t=1}^{T}(\alpha I(t, x_t) + \beta I_n(t, x_t)) + \sum_{t=2}^{T}(1-\alpha-\beta)(x_t - x_{t-1})^2$$

where $I(x, y)$ gives the intensity at some discrete point $(x, y)$, and $I_n(t, x_t) = \sum_{\mathbf{o}\in\{-1,0,1\}^2\setminus\{(0,0)\}} I(t + o_1, x_t + o_2)$ represents the intensities of surrounding points. $\alpha$ and $\beta$ are the parameters of our loss function. We found that $\alpha = 0.5$ and $\beta = 0$ produced the best results. Hence, our final loss function was:

$$L(\mathbf{x}) = -\frac{1}{2}\sum_{t=1}^{T} I(t, x_t) + \frac{1}{2}\sum_{t=2}^{T}(x_t - x_{t-1})^2$$

The code was optimized in C++, with the ultimate goal of integration into a real-time analysis pipeline [4].

### *B. Spectrogram Analysis*

We extracted two features from the data prior to conducting any time series analysis.

1) Loss - We found that the final value of the loss function resulting from our discretization algorithm over the resultant squiggle proved to be a reliable measure of overall intensity and coherence of the signals in our data set.
2) Width - Letting $I$ = the max intensity of a given time slice, and $i$ be the corresponding index, we estimated signal width of all 129 time slices based on the number of indices in $[i - 10, i + 10]$ with intensities $\geq \frac{1}{2}I$. We then took the mean of signal widths falling between the 10th and 90th percentiles.

### *C. Time series Analysis*

Each spectrogram image has the same frequency units but captures a different window of bandwidth. Unfortunately, the frequency ranges of each plot were unavailable to us, so we conducted our analysis in a manner agnostic to absolute frequencies. Before any time series analysis, we modified each time series to have mean 0 by subtracting the original mean. After this normalization, we extracted the following features:

1) Variance - We took the overall mean squared error (MSE) of the time series around the mean.
2) Modulation -
   $x_t = \beta_1 t + \beta_0 + w_t$, $w_t \sim N(0, \sigma^2)$ i.i.d $\forall t$
   We first fit a linear model (above) to the data using simple linear regression. We then estimated $\sigma^2$ by taking the MSE of the model.
3) ARIMA(1, 1, 1) parameters -
   $(1 - \phi B)\nabla x_t = (1 + \theta B)w_t + \mu$
   $x_t = (1 + \phi)x_{t-1} - \phi x_{t-2} + w_t + \theta w_{t-1} + \mu$
   $w_t \sim N(0, \sigma^2)$ i.i.d $\forall t$
   Using the `arima` function in R, we first tried out various models in the ARIMA class of models before settling on ARIMA(1, 1, 1), which consistently provided the lowest Akaike Information Criterion (AIC) when fitted to squiggle time series. We fitted ARIMA(1, 1, 1) models to each squiggle time series, and extracted estimates for the parameters $\phi$, $\theta$, $\mu$, and $\sigma^2$, and incorporated these estimates into our analysis.
4) Hurst Exponent -
   $\mathrm{E}\left[\frac{R(n)}{S(n)}\right] = Cn^H$
   We estimated the Hurst Exponent, a measure of the long term memory of the system, using the numpy `polyfit` function on the lag vector we obtained.
5) Fast Fourier Transform -
   $X_k = \sum_{t=0}^{T} x_t e^{\frac{2\pi i k t}{128}}, k = 1, \ldots, 63$
   We applied a Fast Fourier Transform (FFT) to the squiggle time series to extract the component frequencies in the signal. We sampled the FFT output at 63 uniformly-spaced frequencies from 0 to $\pi$. We ignored the FFT at $k = 0$ which corresponds roughly to the mean of the signal. Our resultant features were the absolute values of the FFT at these points.

## V. CLASSIFICATION

To ensure an unbiased classification, the full dataset was split into 90% training and 10% test. Using the training set, we applied 10-fold cross validation to tune model parameters. Performance metrics were then scored by applying the fitted classifier to the 10% held-out validation set. We used two performance metrics: 1) ACC: accuracy defined as 1 - misclassification error and 2) AUC: area under the Receiver Operating Characteristic (ROC) curve. The ROC curve measure the true positive rate against the false positive rate at various threshold settings. A steep slope in the beginning

indicates good predictive performance - increasing the threshold increases the true positive rate while introducing few false positives. When the curve flattens, we observe the introduction of false positives as the threshold increases. Note that an AUC value of 0.5 refers to a random classifier that stratifies positive and negative samples arbitrarily. In our models, we denote a positive and negative label as nonsquiggle and squiggle, respectively.

### A. Baseline Model

We first applied unregularized, L1-regularized (lasso), and L2-regularized (ridge) logistic regression using the normalized 129 time series points. We achieved a baseline accuracy of 87.5% on the test set, with all three models classifying the full dataset as nonsquiggle. And in fact, the true negative rate is 100% and the false positive rate is 12.5% for all three models. We can note the AUC of 0.5, denoting a random classifier. Since our dataset is unbalanced, with 10.4% squiggle and 89.6% nonsquiggle, our accuracy begins relatively high despite using a model that deems all input as nonsquiggle. Improvements in ACC and AUC metrics are in relation to this baseline.

Baseline Results

| Logistic Model | Train ACC | Train AUC | Test ACC | Test AUC |
|---|---|---|---|---|
| Unregularized | 0.898 | 0.597 | 0.875 | 0.504 |
| Lasso (L1) | 0.898 | 0.5 | 0.875 | 0.5 |
| Ridge (L2) | 0.898 | 0.5 | 0.875 | 0.5 |

### B. Intermediate Model

We improved our baseline model by transforming our features space from the 129 time series points to the 63 FFT frequency samples. We then applied unregularized, L1-regularized (lasso), and L2-regularized (ridge) logistic regression once more, achieving a significant improvement in both ACC and AUC. Unregularized logistic regression produced the greatest AUC while ridge regression produced the greatest ACC.

Intermediate Results

| Logistic Model | Train ACC | Train AUC | Test ACC | Test AUC |
|---|---|---|---|---|
| Unregularized | 0.952 | 0.959 | 0.955 | 0.967 |
| Lasso (L1) | 0.951 | 0.985 | 0.953 | 0.963 |
| Ridge (L2) | 0.952 | 0.960 | 0.959 | 0.962 |

### C. Final Model

In our final model, we incorporated all 72 features, comprised of the 63 FFT frequency samples, the 4 parameters from the ARIMA(1,1,1) model, the variance, modulation, and hurst exponent of the 129-slice time series, and signal width extracted from the spectrogram, and loss from the dynamic programming algorithm. We applied 4 families of classifiers: 1) Logistic Regression using L1, L2, and no regularization; 2) Support Vector Machines (SVM) using linear, radial, polynomial, and sigmoid kernels; 3) Tree-Based Methods including boosting, bagging, and random forests; and 4) K-Nearest Neighbors (KNN). Using 10-folds cross validation on the training set, we performed hyperparameter optimization on each classifier. We identified the optimal parameters: shrinkage parameter $\lambda$ in lasso and ridge logistic regression; kernel parameters and soft-margin parameter $C$ for SVMs; number of trees in tree-based methods; number of neighbors considered $k$ in KNN. We chose the parameter(s) based on cross-validation misclassification rate using the one standard error rule, favoring simpler models to reduce the potential for overfitting. For multi-parameter classifiers, we performed a grid search.

In aggregate, we achieved our highest ACC of 99.2% using the boosting tree-based method. However, nearly all tree-based methods resulted in relatively low AUC metrics. Thus, accounting for our unbalanced dataset, we believe unregularized logistic regression is the optimal classifier, boasting a test AUC of 99.7%. In fact, one of the key properties of AUC is that is it invariant to class skew, meaning that an unbalanced dataset of 90% positive labels will result in the same ROC curve as a dataset of 50% positive, 50% negative labels.

Final Results

| Logistic Model | Train ACC | Train AUC | Test ACC | Test AUC |
|---|---|---|---|---|
| Unregularized Logistic | 0.992 | 0.997 | 0.988 | 0.997 |
| Lasso Logistic (L1) | 0.991 | 0.996 | 0.985 | 0.996 |
| Ridge Logistic (L2) | 0.985 | 0.992 | 0.974 | 0.974 |
| SVM Linear | 0.992 | 0.995 | 0.989 | 0.979 |
| SVM Radial | 0.992 | 0.987 | 0.988 | 0.981 |
| SVM Polynomial | 0.999 | 0.999 | 0.982 | 0.987 |
| SVM Sigmoid | 0.983 | 0.985 | 0.954 | 0.983 |
| Boosting (19 Iterations) | 1.00 | 1.00 | 0.992 | 0.963 |
| Bagging (16 trees) | 1.00 | 0.996 | 0.990 | 0.957 |
| Random Forests (17 trees) | 1.00 | 0.998 | 0.991 | 0.963 |
| 4-NN | 0.990 | N/A | 0.987 | N/A |

### D. Feature Significance

We identified the features that had notable predictive power in the logistic family of classifiers. For unregularized logistic regression, we noted 7 variables with a p-value less than 0.001: loss, hurst exponent, signal width, modulation, and the $\phi$, $\mu$, $\sigma^2$ parameters from the ARIMA(1,1,1) model. For L1-regularized logistic regression, we noted that 4 variables boasted nonzero coefficients: loss, signal width, modulation, and ARIMA(1,1,1) $\sigma^2$. The p-values are representative of

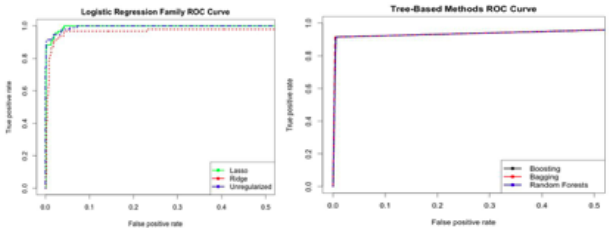


Fig. 3: The ROC curve of the logistic family smooths out at the top bend, resulting in a higher AUC metric. The ROC curve of tree-based methods is less smooth.

each features significance as a predictor for the response; it should be noted, however, that correlated features tend to reduce each others significance. As verified by an Analysis of Variance (ANOVA) test, we claim that ARIMA(1,1,1) $\sigma^2$ and modulation are the two most significant features, reducing the residual variance of our logistic model the most. We can further verify these results using forward or backward stepwise regression, choosing features iteratively based on adjusted R-squared or Akaike Information Criterion (AIC).

| Source/Model | Feature | Logistic (Lasso) | Logistic (Unregularized + Ridge) | SVM | Tree-Based |
|---|---|---|---|---|---|
| Image | Signal Width | • | • | | • |
| Discretization Algorithm | Loss | • | • | | |
| Linear Model | $\hat{\sigma}^2$ | • | • | • | • |
| White Noise Process | $\hat{\sigma}^2$ | | | | |
| ARIMA (1, 1, 1) Process | $\hat{\mu}$ | | | | |
| | $\hat{\sigma}^2$ | • | • | • | • |
| | $\hat{\phi}$ | | • | • | |
| | $\hat{\theta}$ | | • | • | |
| Long Memory Process | $\hat{H}$ | | | | |
| FFT | $X(W_{128}^n), n=1,\ldots,63$ | | • | | |

Fig. 4: Significance Features from the Different Models

## VI. CLUSTERING

### A. Methodology

Before clustering, we normalized all features to have mean 0 and variance 1, then performed dimensionality reduction using principal component analysis (PCA), projecting our 72 features into 5 principal component vectors, capturing $> 98\%$ of the variance. We hypothesized that the squiggles likely followed a continuous spectrum rather than occupying distinct subgroups. Thus, we sampled a variety of distance metrics including Euclidean, Manhattan, and Canberra distance, as well as various clustering algorithms, namely k-means and several hierarchical algorithms including single, average, complete, McQuitty, centroid, median, and Ward linkage clustering. We also applied divisive clustering, which yielded unfavorable results and tended to place outlier points in their own clusters.

For each method, we plotted the average silhouette score over all clusters while varying k from 1 to 15. We observed that 4 clusters produced favorable silhouette scores for the methods which utilized the Euclidean distance metric.

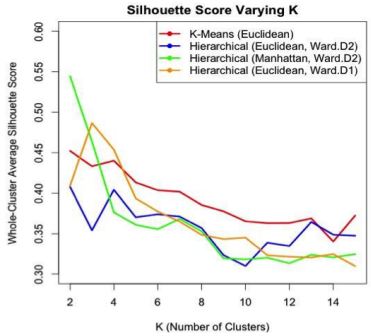


Fig. 5: Silhouette Scores from Different Methods

The question naturally arises whether the four clusters uncovered by each of the algorithms are in fact the same four clusters. To determine this, we mapped the four clusters to each other so as to maximize the proportion of points that are in the same cluster across all three methods. We have graphed the corresponding clusterings below.

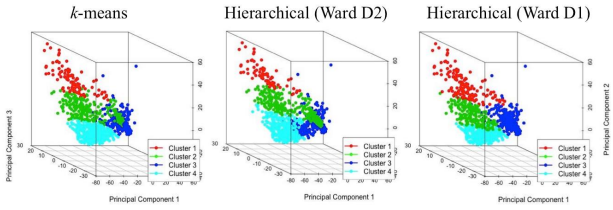


Fig. 6: Principal Components Visualizations for Different Clustering Schemes

With the mapping given above, the proportion of points that appear in the same cluster in all three instances is calculated to be 536/833, or 0.6070215. We performed the same set of tests 100,000 times on randomized Gaussian $\mathcal{N}(0, I)$ noise. The average value of the above proportion was 0.320, and only 98/100,000 test runs produced a proportion higher than 0.607, signifying a p-value of around 0.001 with the null hypothesis that the data was generated from Gaussian random noise. The high level of concordance between the clusters generated signifies that the results we found are robust to the exact method of clustering. Below are examples of squiggles sampled randomly from each of the four clusters (of the points that were assigned unanimously to a cluster).

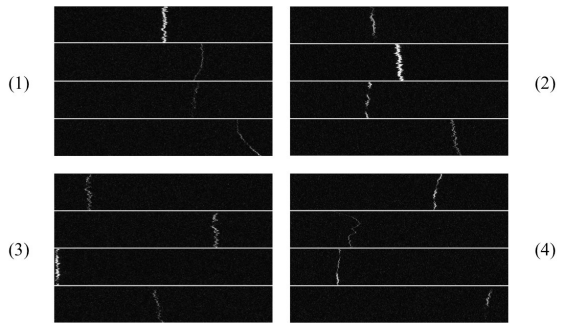


Fig. 7: The presence of 4 clusters may imply the existence of 4 distinct sources, which may be further investigated by the SETI team.

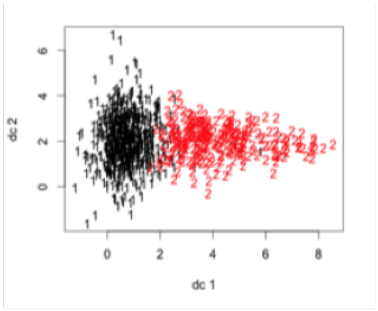


Fig. 8: The results of K=2, Ward Hierarchical Clustering using Euclidean Distance on the PCA squiggle set. We then projected the clusters onto the first two linear discriminant functions.

### *B. Chi-Squared Test Insights*

With potential subgroups identified, we assessed the characteristics of each cluster using known temporal and polarization characteristics. The analysis of K=2, Ward D2 Hierarchical clustering under Euclidean distance proved most promising. We ran a series of independent chi squared tests to assess the dependency of temporal (month, 4-hour timeframe) and polarization variables against membership in the clusters above. The table lists characteristics that were shown to exhibit the most significant dependency on cluster membership. In particular, we can note that nearly all members of the red cluster, corresponding to squiggles with large variance in frequency, were produced/detected between 12pm and 4pm. This dependency on the 24-hour Earth cycle could imply that the source is terrestrial, rather than external. Once denser clusters of squiggles are identified, we hope to replicate similar chi-squared analyses.

Chi-Squared Test Significant Results

| Characteristic | p-value |
|---|---|
| August | 5.75E-03 |
| 4 AM - 8 AM | 2.31E-06 |
| 8 AM - 12 PM | 5.79E-06 |
| **12 PM - 4 PM** | **7.32E-16** |
| L-Polarization | 1.24E-03 |
| R-Polarization | 3.14E-05 |

## VII. CONCLUSIONS & FUTURE WORK

As discussed, our model is agnostic to absolute frequencies. We consider this to be a severe drawback, especially in clustering, as unique signals from the same source are likely to have similar frequencies.

Although we were able to capture a wide range of modulation speeds, our ability to do so was hampered by our discretization algorithm, which assigned a single value per time point. At higher modulation speeds, the signal simply appears to be a jagged wideband signal. Simulation and image processing could be used to turn modulation speed into a more tangible parameter, instead of relying on the Fast Fourier Transform. Prior to pushing our code to the SETIQuest repository, we will also perform model selection to trim our predictors to only the most indicative to decrease both computation time and overfitting for real-time classification.

If squiggles do in fact come from a wide array of sources, the logical next step would be to continue uncovering dense clusters of squiggles that come from the same source on a predictable basis. SETI recently opened access to 400,000 unknown spectrogram signals; we can apply our existing classifier to curate more squiggles, thus providing a larger dataset to conduct unsupervised learning.

## VIII. ACKNOWLEDGMENTS

Our work on this project was also counted toward CS 341: Project in Mining Massive Datasets. Our advisor was Professor Jeffrey Ullman and our additional team member for CS 341 was Frank Fan.

## REFERENCES

[1] `http://setiquest.org/wiki/index.php/Enhancement_of_Algorithm_to_Detect_Pulse_signals`. 2011.

[2] Classification of Communication Signals and Detection of Unknown Formats Using Artificial Neural Networks: Alexander Iversen, Nicholas K. Taylor and Keith E. Brown. 2006.

[3] Automatic Recognition of Features in Spectrograms Based on some Image Analysis Methods: Aleksandar Perovi , Zoran orevi, Mira Paskota , Aleksandar Takai , Aleksandar Jovanovi. 2013.

[4] `https://github.com/jwang198/SETI_TimeSeries`.