Automatic Spike Sorting: Visualizing Massive Data Sets

David Brody

Department of Computer Science, Stanford University, Stanford, California

Introduction

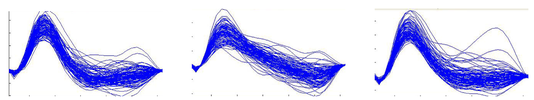
This project is aimed to create software to help neuroscientists analize their electrophysiological data. Specifically, this software was designed to help automate the currently tedious task of spike sorting within the Baccus Lab. The software takes the output of an automatic spike sorter and allows the scientist to explore and modify the data as well as save and load their sessions. Finally, it allows the user to export their work for further analysis using other tools.

Problem and Data Description

Learning how the data is collected and what it represents is crutial to ensure the software address the challenge correctly. The data is collected during a wet experiment in which the scientist disects a salamander and places a retina on a microelectrode array. The many ganglion cells of the retina are then stimulated with visual stimulus causing them to spike. All the spikes from a single cell have a similar shape however each cell may have a different shape. Three example spike shapes may look like:

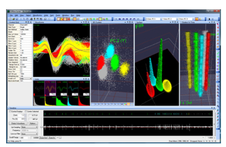
Each electrode on the array constantly records the voltage at its point at a rate of 10KHz. However, since the recording is extracellular it picks up the voltage traces of many neighboring cells. A sample recording from an electrode may look like the following:

This data is then fed into an automatic sorter which pulls the spikes out of the data stream and into clusters which may be cells. Taking each cluster and overlaying each spikeshape gives the following examples of what the dataset includes:

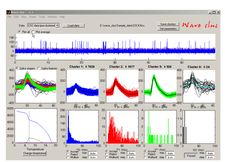


The scientists want to make sure that certain properties of each cluster are correct before they accept it as a potential cell to further analize. The tool presented here aims to help them explore and refind the data.

Other Current Solutions



- Limited To 3 Electrodes (max)
+ Great 3d Visualizations
+ Real Time Interactive
- Not Matlab



- Limited To 5 clusters (max)
- Not Real Time Interactive
+ Matlab

Univariate Class Label Potentials

Features

- Person node degree - $0, 1, 2, \ldots$
- Average outgoing email length - discretized into 31 intervals
- Ratio of outgoing emails - $\frac{\#sent}{\#sent+\#recieved}$ discretized in 4 bins by 0.25
- 1936 subject title word indicators - binary random variables

Clustering

- Trained using various amounts of clusters 7,...,15
- Compared Log Likelihood and BIC

Feature Strengths

Word	Rank	Count	Score[1]
testimony	1	12	1.0499
sac	2	5	0.9934
website	8	19	0.8036
...			
status	1933	79	0.00017
investments	1934	5	0.00017
2	1935	132	0.00015
crude	1936	8	0.00015

[1] Score is variance of resulting CPD

Univariate Link Activity Potentials

Overview

Allows us to incorporate structure of social graph of the training data by assigning a $\gamma_{ij}(Y_{ij})$

Method

Define $\gamma_{ij}(Y_{ij})$ for $1 \leq i \leq j \leq n$ by:

- $\gamma_{ij}(0) = 1$
- $\gamma_{ij}(1) = 1 + score(i, j)$

Various $score(i, j)$ definitions:

- Common Neighbors Score
- Jaccard coefficient score
- Zero Score (Uniform)

Triangle Template Potentials

Overview

- Use training data as instantiation of unrolled network to observe Y_{ij} then complete data using $C_i = c_i^* = \arg\max_{c_i} \phi_i(c_i)$ for every i.
- Gradient ascent for parameters of (c_i, c_j, y) intractable since it requires inference over the joint assignment to all variables in $\mathcal{X}$.

inference

Method 1

- Used hard assignment to $\phi_i(C_i)$ since potentials usually assign high probability to a single class.
- Gradient of (conditional) likelihood becomes more tractable:

$$\frac{\partial}{\partial w_{c_1,c_2,y}} \frac{1}{M} \ell(\theta : \mathcal{D}) = \frac{1}{M} M[c_1, c_2, y] - P(c_1, c_2, y)$$

- No inference or gradient ascent by setting to 0.

Method 2

- Network distribution is represented as

$$P(\mathcal{X}) = \frac{1}{Z} \left(\prod_i \phi_i(C_i) \right) \left(\prod_{i<j} \psi(C_i, C_j, Y_{ij}) \right) \left(\prod_{i<j} \gamma_{ij}(Y_{ij}) \right)$$

- Structure creates correspondence between $\phi(C_i, C_j, Y_{ij})$ and $P(Y_{ij}|C_i, C_j)$
- Learn θ as entires of CPD for $P(Y|C_i, C_j)$

Model Querying

- Run loopy belief propigation to obtain marginal distributions for $P(C_i)$ and $P(Y_{ij})$ for all i and j.
- Structure is similar to Bethe construction
- Graph is family preserving and satifies running intersection property.

Experimental Results

We focussed our comparison on two specific predictors, common neighbors and Jaccardâs coefficient.

Prediction Method	Accuracy
Common neighbors	0.1407
Jaccard coefficient	0.1759
Random guessing	0.0194
Full inference method 1; Common Neighbor link potentials	0.1407
Full Inference method 1; Jaccard link potentials	0.1759
Full Inference method 1; Trivial link potentials	0.0151
Full inference method 2; Common Neighbor link potentials	0.1206
Full inference method 2; Jaccard Score link potentials	0.0402
Full inference method 2; Trivial link potentials	0.0201
Full conditional inference; Common Neighbor link potentials	0.1206
Full conditional inference; Jaccard link potentials	**0.1859**
Full conditional inference; Trivial link potentials	0.0251

High score of method 1 with nontrivial link potentials accuracy led to believe triangle template factors were contributing little useful information.

Tested link prediction with triangle template potentials alone with and without link potentials:

Prediction Method	Accuracy
Triangle template potential trial 1; Trivial link potential	0.0201
Triangle template potential trial 1; Common Neighbor link potential	0.0754
Triangle template potential trial 1; Trivial link potential, 3 classes	0.0302
Triangle template potential trial 1; Common Neighbor link potential, 3 classes	0.0905
Triangle template potential trial 2; Trivial link potential	0.0201
Triangle template potential trial 2; Common Neighbor link potential	0.0754

Conclusion

Conclusions

- Achieved result close to graph-structure based link predictors
- Experiments show that success of prediction comes from having accurate univariate link potential
- Model preserves information
- Adding classifications of people does not significantly add to predictive power
  - May be because Email is poor indicator of useful class partitioning

Learnings

- Make sure features chosen have suffient mutual informtion with variables of interest

Future Work

- Try with features of email that may be more indicitive of future communications
- Try in domain that has more direct features of who a person may communicate with
  - Facebook information would allow for application to social network structure as well as indicitive features of each person including location, school, interests, etc.
- Test model in scenario where limited amounts of test data are observable