Discovering Excitatory Networks from Discrete Event Streams with Applications to Neuronal Spike Train Analysis

Debprakash Patnaik
Department of Computer Science
Virginia Tech, VA 24061, USA
Email: patnaik@vt.edu

Srivatsan Laxman
Microsoft Research
Bangalore 560080, India
Email: slaxman@microsoft.com

Naren Ramakrishnan
Department of Computer Science
Virginia Tech, VA 24061, USA
Email: naren@vt.edu

***Abstract*—Mining temporal network models from discrete event streams is an important problem with applications in computational neuroscience, physical plant diagnostics, and human-computer interaction modeling. We focus in this paper on temporal models representable as excitatory networks where all connections are stimulative, rather than inhibitive. Through this emphasis on excitatory networks, we show how they can be learned by creating bridges to frequent episode mining. Specifically, we show that frequent episodes help identify nodes with high mutual information relationships and which can be summarized into a dynamic Bayesian network (DBN). To demonstrate the practical feasibility of our approach, we show how excitatory networks can be inferred from both mathematical models of spiking neurons as well as real neuroscience datasets.**

***Keywords*-Frequent Episodes; Dynamic Bayesian Network; Computational Neuroscience; Spike train analysis; Temporal Data Mining**

I. Introduction

Discrete event streams are prevalent in many applications, such as neuronal spike train analysis, physical plants, and human-computer interaction modeling. In all these domains, we are given occurrences of events of interest over a time course and the goal is to identify trends and behaviors that serve discriminatory or descriptive purposes.

A Multi-Electrode Array (MEA) records spiking action potentials from an ensemble of neurons which after various pre-processing steps, yields a spike train dataset providing real-time, dynamic, perspectives into brain function (see Fig. 1). Identifying sequences (e.g., cascades) of firing neurons, determining their characteristic delays, and reconstructing the functional connectivity of neuronal circuits are key problems of interest. This provides critical insights into the cellular activity recorded in the neuronal tissue.

Similar motivations arise in other domains as well. In physical plants the discrete event stream denotes diagnostic and prognostic codes from stations in an assembly line and the goal is to uncover temporal connections between codes emitted from different stations. In human-computer interaction modeling, the event stream denotes actions taken by users over a period of time and the goal is to capture aspects such as user intent and interaction strategy by understanding causative chains of connections between actions.

Beyond uncovering structural patterns from discrete events, we seek to go further, and actually uncover a generative temporal process model for the data. In particular, our aim is to infer dynamic Bayesian networks (DBNs) which encode conditional independencies as well as temporal influences and which are also interpretable patterns in their own right. We focus exclusively on excitatory networks where the connections are stimulative rather than inhibitory in nature (e.g., 'event A stimulates the occurrence of event B 5ms later which goes on to stimulate event C 3ms beyond.') This constitutes a large class of networks with relevance in multiple domains, including neuroscience.

Our main contributions are three-fold:

1) **New model class of excitatory networks**: Learning Bayesian networks (dynamic or not) is a hard problem and to obtain theoretical guarantees we typically have to place restrictions on network structure, e.g., assume the BN has a tree structure as done in the Chow-Liu algorithm. Our focus on excitatory networks places restrictions on the nature of the conditional probability tables (CPT) instead of network structure and we show how this leads to a tractable formulation.
2) **New methods for learning DBNs**: We demonstrate that DBNs can be learnt by creating bridges to frequent episode mining literature. In particular, the focus on excitatory networks allows us to relate frequent episodes to parent sets for nodes with high mutual information. This enables us to predominantly apply fast algorithms for episode mining, while relating them to probabilistic notions suitable for characterizing DBNs.
3) **New applications to spike train analysis**: We demonstrate a successful application of our methodologies to analyzing neuronal spike train data, both from mathematical models of spiking neurons and from real cortical tissue.

The paper is organized as follows. Sec. II gives a brief overview of DBNs and Sec. III presents our formalism for modeling event streams using DBNs. Sec. IV defines excitatory networks and develops the theoretical basis for efficiently learning such networks. Sec. V introduces fixed-delay episodes and relates frequencies of such episodes with marginal probabilities of a DBN. Our learning algorithm is presented in Sec. VI, experimental results in Sec. VII and conclusions in Sec. VIII.

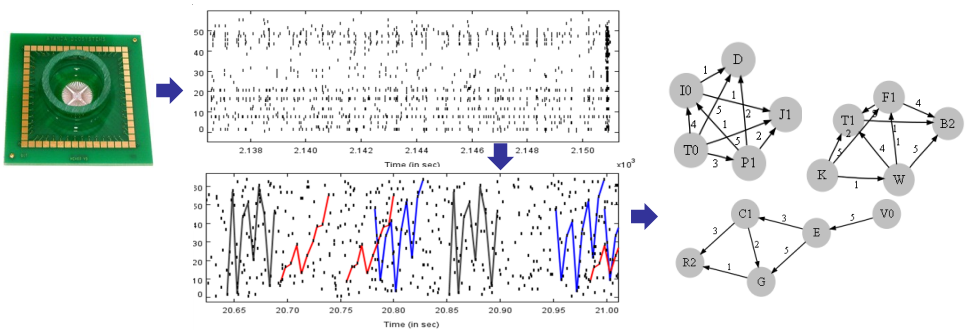


Figure 1. A multi-electrode array (MEA; left) produces a spiking event stream of action potentials (middle top). Mining cascaded firings (middle bottom) in the event stream helps uncover excitatory circuits (right) in the data.

II. Bayesian Networks: Static and Dynamic

Formal mathematical notions are presented in the next section, but here we wish to provide some background context to past research in Bayesian networks (BNs). As is well known, BNs use directed acyclic graphs to encode probabilistic notions of conditional independence, such as that a node is conditionally independent of its non-descendants given its parents (for more details, see [1]). The earliest known work for learning BNs is the Chow-Liu algorithm [2]. It showed that, if we restricted the structure of the BN to a tree, then the optimal BN can be computed using a minimum spanning tree algorithm. It also established the tractability of BN inference for this class of graphs.

More recent work, by Williamson [3], generalizes the Chow-Liu algorithm to show how (discrete) distributions can be approximated using the same general ingredients as the Chow-Liu approach, namely mutual information quantities between random variables. Meila [4] presents an accelerated algorithm that is targeted toward sparse datasets of high dimensionality. The approximation thread for general BN inference is perhaps best exemplified by Friedman's sparse candidate algorithm [5] that presents various greedy approaches to learn (suboptimal) BNs.

DBNs are a relatively newer development and best examples of them can be found in specific state space and dynamic modeling contexts, such as HMMs. In contrast to their static counterparts, exact and efficient inference for general classes of DBNs has not been studied well.

III. Modeling Event Streams using DBNs

Consider a finite alphabet, $\mathcal{E} = \{A_1, \ldots, A_M\}$, of event-types (or symbols). Let $s = \langle (E_1, \tau_1), (E_2, \tau_2), \ldots, (E_n, \tau_n) \rangle$ denote a data stream of n events over $\mathcal{E}$. Each $E_i, i = 1, \ldots, n$, is a symbol from $\mathcal{E}$. Each $\tau_i, i = 1, \ldots, n$, takes values from the set of positive integers. The events in s are ordered according to their times of occurrence, $\tau_{i+1} \geq \tau_i, i = 1, \ldots, (n-1)$. The time of occurrence of the last event in s, is denoted by $\tau_n = T$. We model the data stream, s, as a realization of a discrete-time random process $\mathbf{X}(t), t = 1, \ldots, T$; $\mathbf{X}(t) = [X_1(t) X_2(t) \cdots X_M(t)]'$, where $X_j(t)$ is an indicator variable for the occurrence of event type, $A_j \in \mathcal{E}$, at time t. Thus, for $j = 1, \ldots, M$ and $t = 1, \ldots, T$, we will have $X_j(t) = 1$ if $(A_j, t) \in s$, and $X_j(t) = 0$ otherwise. Each $X_j(t)$ is referred to as the *event-indicator random variable* for event-type, A_j, at time t.

Example 1: The following is an example event sequence of $n = 7$ events over an alphabet, $\mathcal{E} = \{A, B, C, \ldots, Z\}$, of $M = 26$ event-types:

$$\langle (A, 2), (B, 3), (D, 3), (B, 5), (C, 9), (A, 10), (D, 12) \rangle \quad (1)$$

The maximum time tick is given by $T = 12$. Each $\mathbf{X}(t)$, $t = 1, \ldots, 12$, is a vector of $M = 26$ indicator random variables. Since there are no events at time $t = 0$ in the example sequence (1), we have $\mathbf{X}(1) = \mathbf{0}$. At time $t = 2$, we will have $\mathbf{X}(2) = [1000 \cdots 0]'$. Similarly, $\mathbf{X}(3) = [0101 \cdots 0]'$, and so on.

A DBN [6] is a DAG with nodes representing random variables and arcs representing conditional dependency relationships. We model the random process $\mathbf{X}(t)$ (or equivalently, the event stream s), as the output of a DBN. Each event-indicator, $X_j(t)$, $t = 1, \ldots, T$ and $j = 1, \ldots M$, corresponds to a node in the network, and is assigned a set of parents, which is denoted as $\pi(X_j(t))$ (or simply $\pi_j(t)$). A parent-child relationship is represented by an arc (from parent to child) in the DAG. In a DBN, nodes are conditionally independent of their non-descendants given their parents. The joint probability distribution of $\mathbf{X}(t)$ under the DBN model, can be factorized as a product of $P[X_j(t) \,|\, \pi_j(t)]$ for various j, t. In this paper we restrict the class of DBNs using the following two constraints:

- A1 *[Time-bounded causality]* For user-defined parameter, $W > 0$, the set, $\pi_j(t)$, of parents for the node, $X_j(t)$, is a subset of event-indicators out of the W-length history at time-tick, t, i.e. $\pi_j(t) \subseteq \{X_k(\tau) : 1 \leq k \leq M,\ (t - W) \leq \tau < t\}$.
- A2 *[Translation invariance]* If $\pi_j(t) = \{X_{j_1}(t_1), \ldots, X_{j_\ell}(t_\ell)\}$ is an ℓ-size parent set of $X_j(t)$ for some $t > W$, then for any other $X_j(t')$, $t' > W$,

its parent set, $\pi_j(t')$, is simply a time-shifted version of $\pi_j(t)$, and is given by $\pi_j(t') = \{X_{j_1}(t_1 + \delta), \ldots, X_{j_\ell}(t_\ell + \delta)\}$, where $\delta = (t' - t)$.

While **A1** limits the range-of-influence of a random variable, $X_k(\tau)$, to variables within (a user-defined) W time-ticks of τ, **A2** is a structural constraint that allows parent-child relationships to depend only on relative (rather than absolute) time-stamps of random variables. Further, we also assume that the underlying data generation model is stationary, so that joint-statistics can be estimated using frequency counts of suitably defined temporal patterns in the data.

A3 *[Stationarity]* For every set of event-indicators, $X_{j_1}(t_1), \ldots, X_{j_\ell}(t_\ell)$, and for every time-shift δ, we have $P[X_{j_1}(t_1), \ldots, X_{j_\ell}(t_\ell)] = P[X_{j_1}(t_1+\delta), \ldots, X_{j_\ell}(t_\ell + \delta)]$.

Learning network structure involves learning the map, $\pi_j(t)$, for each $X_j(t)$, $j = 1, \ldots, M$ and $t > W$. Let $I[X_j(t)\ ;\ \pi_j(t)]$ denotes the *mutual information* between $X_j(t)$ and its parents, $\pi_j(t)$. DBN structure learning can be posed as a problem of approximating the data distribution, $P[\cdot]$, by a DBN distribution, $Q[\cdot]$. Let $D_{KL}(P||Q)$ denote the KL divergence between $P[\cdot]$ and $Q[\cdot]$. Using **A1**, **A2** and **A3**, and following the lines of [2], [3], it is possible to show that for parent sets with sufficiently high mutual information to $X_j(t)$, $D_{KL}(P||Q)$ will be concomitantly lower [7]. In other words, a good DBN-based approximation of the underlying stochastics (i.e. one with small KL divergence) can be achieved by picking, for each node, a parent-set whose corresponding mutual information exceeds a user-defined threshold.

However, picking such sets with high mutual information (while yields a good approximation) falls short of unearthing useful dependencies among the random variables. This is because mutual information is non-decreasing as more random variables are added to a parent-set (leading to a fully-connected network always being optimal). For a parent-set to be interesting, it should not only exhibit sufficient correlation (or mutual information) with the corresponding child-node, but should also successfully encode the conditional independencies among random variables in the system. This can be done by checking if, *conditioned* on a candidate parent-set, the mutual information between the corresponding child-node and all its non-descendants is always close to zero. (We provide more details later in Sec. VI-C).

IV. Excitatory networks

The structure-learning approach described in Sec. III is applicable to any general DBN that satisfies **A1** and **A2**. In this paper, we focus on a further specialized class of networks, called *excitatory networks*, where only certain kinds of conditional dependencies among nodes are permitted. In general, each event-type has some baseline propensity in the data which is small and *less than 0.5* (This corresponds to a sparse-data assumption). A collection, Π, of random variables is said to have an *excitatory* influence on an event-type, $A \in \mathcal{E}$, if occurrence of events corresponding to the variables in Π, increases the propensity of A to *greater than 0.5*. We define an excitatory network as one in which nodes can only exert excitatory influences on one another. For example, in an excitatory network it is possible that "if B, C and D occur (say) 2 time-ticks apart, the probability of A increases." By contrast, in excitatory networks, it is not possible to model relationships like "when A does not occur, the probability of B occurring 3 time-ticks later increases." Similarly, excitatory networks cannot model inhibitory relationships like "when A occurs, the probability of B occurring 3 time-ticks later decreases." Excitatory networks are natural in neuroscience, where one is interested in unearthing conditional dependency relationships among neuron spiking patterns. Several regions in the brain are known to exhibit predominantly excitatory relationships [8] and our model is targeted toward unearthing these.

Table I
Example of a conditional probability table in an excitatory network.

	Π			$P[X_A = 1 \mid a_j]$
	X_B	X_C	X_D	
a_0	0	0	0	$\epsilon < \frac{1}{2}$
a_1	0	0	1	$\epsilon_1 \geq \epsilon$
a_2	0	1	0	$\epsilon_2 \geq \epsilon$
a_3	0	1	1	$\epsilon_3 \geq \epsilon, \epsilon_1, \epsilon_2$
a_4	1	0	0	$\epsilon_4 \geq \epsilon$
a_5	1	0	1	$\epsilon_5 \geq \epsilon, \epsilon_1, \epsilon_4$
a_6	1	1	0	$\epsilon_6 \geq \epsilon, \epsilon_2, \epsilon_4$
$a_7(a^*)$	1	1	1	$\phi > \epsilon, \frac{1}{2}, \epsilon_j \forall j$

The excitatory assumption manifests as a set of constraints on the conditional probability tables associated with the DBN. Consider a node[1] X_A (an indicator variable for event-type $A \in \mathcal{E}$) and let Π denote a parent-set for X_A. In excitatory networks, the probability that A occurs, conditioned on the occurrence of all the events associated with Π, should be *at least as high as* the corresponding conditional probability when only some (though not all) of the events of Π occur. Further, the probability of A is less than 0.5 when none of the events of Π occur and greater than 0.5 when all the events of Π occur. For example, let $\Pi = \{X_B, X_C, X_D\}$. The conditional probability table for A given Π is shown in Table I. The different conditioning contexts come about by the occurrence or otherwise of each of the events in Π. These are denoted by $a_j,\ j = 0, \ldots, 7$. So while a_0 represents the all-zero assignment (i.e. none of the events B, C or D occur), a_7 (or a^*) denotes the all-ones assignment (i.e. all the events B, C and D occur). The last column of the table lists the corresponding conditional probabilities along with the associated excitatory constraints. The baseline propensity

[1]To facilitate simple exposition, time-stamps of random-variables are dropped from the notation in this discussion.

of A is denoted by ϵ and the only constraint on it is that it must be less than $\frac{1}{2}$. Conditioned on the occurrence of any event of Π, the propensity of A can only increase, and hence, $\epsilon_j \geq \epsilon \ \forall j$ and $\phi \geq \epsilon$. Similarly, when both C and D occur, the probability must be at least as high as that when either C or D occurred alone (i.e. we must have $\epsilon_3 \geq \epsilon_1$ and $\epsilon_3 \geq \epsilon_2$). Finally, for the all-ones case, denoted by $\Pi = a^*$, the conditional probability must be greater than $\frac{1}{2}$ and must also satisfy $\phi \geq \epsilon_j \ \forall j$.

In the context of DBN structure learning, excitatory networks ensure that event-types will occur *frequently* after their respective parents (with suitable delays). This will allow us to estimate DBN structure using frequent pattern discovery algorithms (which have been a mainstay in data mining for many years). We now have a simple necessary condition on the probability (or frequency) of parent-sets in an excitatory network.

Theorem 4.1: Let X_A denote a node in the Dynamic Bayesian Network corresponding to the event-type $A \in \mathcal{E}$. Let Π denote a parent-set with excitatory influence on A. Let ϵ^* be an upper-bound for conditional probabilities $P[X_A = 1 \mid \Pi = a]$ for all $a \neq a^*$ (i.e. for all but the all-ones assignment in Π). If the mutual information $I[X_A\,;\,\Pi]$ exceeds $\vartheta (> 0)$, then the joint probability of an occurrence of A along with all events of Π satisfies $P[X_A = 1, \Pi = a^*] \geq P_{min}\Phi_{min}$, where

$$P_{min} = \frac{P[X_A = 1] - \epsilon^*}{1 - \epsilon^*} \quad (2)$$

$$\Phi_{min} = h^{-1}\left[\min\left(1, \frac{h(P[X_A = 1]) - \vartheta}{P_{min}}\right)\right] \quad (3)$$

and where $h(\cdot)$ denotes the binary entropy function $h(q) = -q\log q - (1-q)\log(1-q)$, $0 < q < 1$ and $h^{-1}[\cdot]$ denotes its pre-image greater than $\frac{1}{2}$.

Proof: Under the excitatory model we have $P[X_A = 1 \,|\, \Pi = a^*] > P[X_A = 1 \mid \Pi = a] \ \forall a \neq a^*$. First we apply ϵ^* to terms in the expression for $P[X_A = 1]$:

$$\begin{aligned} P[X_A = 1] &= P[\Pi = a^*]P[X_A = 1 \mid \Pi = a^*] \\ &+ \sum_{a \neq a^*} P[\Pi = a]P[X_A = 1 \mid \Pi = a] \\ &\leq P[\Pi = a^*] + (1 - P[\Pi = a^*])\epsilon^* \end{aligned}$$

This gives us $P[\Pi = a^*] \geq P_{min}$ (see Fig. 2). Next, since we are given that mutual information $I[X_A\,;\,\Pi]$ exceeds ϑ, the corresponding conditional entropy must satisfy:

$$\begin{aligned} H[X_A \mid \Pi] &= P[\Pi = a^*]h(P[X_A = 1 \mid \Pi = a^*]) \\ &+ \sum_{a \neq a^*} P[\Pi = a]h(P[X_A = 1 \mid \Pi = a]) \\ &< H[X_A] - \vartheta = h(P[X_A = 1]) - \vartheta \end{aligned}$$

Every term in the expression for $H[X_A \,|\, \Pi]$ is non-negative, and hence, each term (including the first one) must be less than $(h(P[X_A = 1]) - \vartheta)$. Using $(P[\Pi = a^*] \geq P_{min})$ in the above inequality and observing that $P[X_A = 1 \,|\, \Pi = a^*]$ must be greater than 0.5 for an excitatory network, we now get $(P[X_A = 1 \mid \Pi = a^*] > \Phi_{min})$. This completes the proof. □

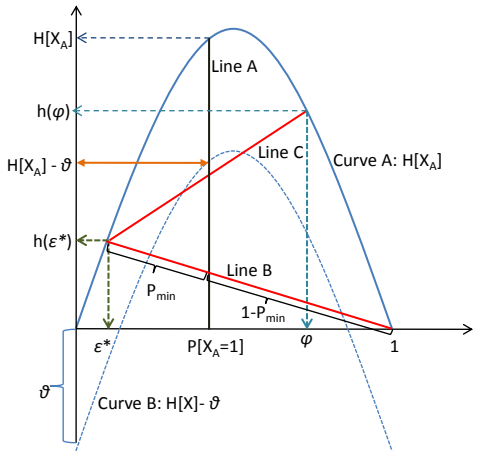


Figure 2. An illustration of the results obtained in Theorem 4.1. x-axis of the plot is the range of $P[X_A = 1]$ and y-axis is the corresponding range of entropy $H[X_A]$. For the required mutual information criteria, the conditional entropy must lie below $H[X_A] - \vartheta$ and on line A. At the boundary condition [$\phi = 1$], Line A splits Line B in the ratio $P_{min} : (1 - P_{min})$. This gives the expression for P_{min}.

V. Fixed-delay episodes

In the framework of frequent episode discovery [9] the data is a single long stream of events over a finite alphabet (cf. Sec. III and *Example 1*). An ℓ-node (serial) episode, α, is defined as a tuple, $(V_\alpha, <_\alpha, g_\alpha)$, where $V_\alpha = \{v_1, \ldots, v_\ell\}$ denotes a collection of nodes, $<_\alpha$ denotes a *total order*[2] such that $v_i <_\alpha v_{i+1},\ i = 1, \ldots, (\ell - 1)$. If $g_\alpha(v_j) = A_{i_j}, A_{i_j} \in \mathcal{E}, j = 1, \ldots, \ell$, we use the graphical notation $(A_{i_1} \rightarrow \cdots \rightarrow A_{i_\ell})$ to represent α. An occurrence of α in event stream, $s = \langle (E_1, \tau_1), (E_2, \tau_2), \ldots, (E_n, \tau_n)\rangle$, is a map $h : V_\alpha \rightarrow \{1, \ldots, n\}$ such that (i) $E_{h(v_j)} = g(v_j)\ \forall v_j \in V_\alpha$, and (ii) for all $v_i <_\alpha v_j$ in V_α, the times of occurrence of the i^{th} and j^{th} events in the occurrence satisfy $\tau_{h(v_i)} \leq \tau_{h(v_j)}$ in s.

Example 2: Consider a 3-node episode $\alpha = (V_\alpha, <_\alpha, g_\alpha)$, such that, $V_\alpha = \{v_1, v_2, v_3\}$, $v_1 <_\alpha v_2$, $v_2 <_\alpha v_3$ and $v_1 <_\alpha v_3$, and $g_\alpha(v_1) = A$, $g_\alpha(v_2) = B$ and $g_\alpha(v_3) = C$. The graphical representation for this episode is $\alpha = (A \rightarrow B \rightarrow C)$, indicating that in every occurrence of α, an event of type A must appear before an event of type B, and the B must appear before an event of type C. For example, in sequence (1), the subsequence $\langle (A, 1), (B, 3), (C, 9)\rangle$ constitutes an occurrence of $(A \rightarrow B \rightarrow C)$. For this

[2]In general, $<_\alpha$ can be any partial order over V_α. We focus on only total orders here and show how multiple such total orders can be used to model DBNs of arbitrary -arity. In [9], such total orders are referred to as *serial* episodes.

occurrence, the corresponding h-map is given by, $h(v_1) = 1$, $h(v_2) = 2$ and $h(v_3) = 5$.

There are many ways to incorporate explicit time constraints in episode occurrences like the windows-width constraint of [9]. Episodes with inter-event *gap* constraints were introduced in [10]. For example, the framework of [10] can express the temporal pattern "B must follow A *within* 5 time-ticks and C must follow B *within* 10 time-ticks." Such a pattern is represented using the graphical notation, $(A \overset{[0-5]}{\longrightarrow} B \overset{[0-10]}{\longrightarrow} C)$. In this paper, we use a simple sub-case of the inter-event gap constraints, in the form of *fixed* inter-event time-delays. For example, $(A \overset{5}{\rightarrow} B \overset{10}{\rightarrow} C)$ represents a fixed-delay episode, every occurrence of which must comprise an A, followed by a B *exactly* 5 time-ticks later, which in-turn is followed by a C *exactly* 10 time-ticks later.

Definition 5.1: An ℓ-node *fixed-delay episode* is defined as a pair, $(\alpha, \mathcal{D})$, where $\alpha = (V_\alpha, <_\alpha, g_\alpha)$ is the usual (serial) episode of [9], and $\mathcal{D} = (\delta_1, \ldots, \delta_{\ell-1})$ is a sequence of $(\ell - 1)$ non-negative delays. Every occurrence, h, of the fixed-delay episode in an event sequence $s$must satisfy the inter-event constraints, $\delta_i = (\tau_{h(v_{i+1})} - \tau_{h(v_i)}),\ i = 1, \ldots, (\ell - 1)$. $(A_{j_1} \overset{\delta_1}{\longrightarrow} \cdots \overset{\delta_{\ell-1}}{\longrightarrow} A_{j_\ell})$ is the graphical notation for inter-event episode, $(\alpha, \mathcal{D})$, where $A_{j_i} = g_\alpha(v_i),\ i = 1, \ldots, \ell$.

Definition 5.2: Two occurrences, h_1 and h_2, of a fixed-delay episode, $(\alpha, \mathcal{D})$, are said to be *distinct*, if they do not share any events in the data stream, s. Given a user-defined, $W > 0$, *frequency* of $(\alpha, \mathcal{D})$ in s, denoted $f_s(\alpha, \mathcal{D}, W)$, is defined as the total number of distinct occurrences of $(\alpha, \mathcal{D})$ in s that terminate strictly after W.

In general, counting distinct occurrences of episodes suffers from computational inefficiencies [11]. (Each occurrence of an episode $(A \rightarrow B \rightarrow C)$ is a substring that looks like $A * B * C$, where $*$ denotes a *variable-length* don't-care, and hence, counting all distinct occurrences in the data stream can require memory of the same order as the data sequence which typically runs very long). However, in case of fixed-delay episodes, it is easy to track distinct occurrences efficiently. For example, when counting frequency of $(A \overset{3}{\longrightarrow} B \overset{5}{\longrightarrow} C)$, if we encounter an A at time t, to recognize an occurrence involving this A we only need to check for a B at time $(t + 3)$ and for a C at time $(t + 8)$. In addition to being attractive from an efficiency point-of-view, we show next in Sec. V-A that the distinct occurrences-based frequency count for fixed-delay episodes will allow us to interpret relative frequencies as probabilities of DBN marginals. (Note that the W in *Definition 5.2* is same as length of the history window used in the constraint **A1**. Skipping occurrences terminating in the first W time-ticks makes it easy to normalize the frequency count into a probability measure).

A. Marginals from episode frequencies

In this section, we describe how to compute mutual information from the frequency counts of fixed-delay episodes. For this, every subset of event-indicators in the network is associated with a fixed-delay episode.

Definition 5.3: Let $\{X_j(t) : j = 1, \ldots, M;\ t = 1, \ldots, T\}$ denote the collection of event-indicators used to model event stream, $s = \langle (E_1, \tau_1), \ldots (E_n, \tau_n) \rangle$, over alphabet, $\mathcal{E} = \{A_1, \ldots, A_M\}$. Consider an ℓ-size subset, $\mathcal{X} = \{X_{j_1}(t_1), \ldots, X_{j_\ell}(t_\ell)\}$, of these indicators, and without loss of generality, assume $t_1 \leq \cdots \leq t_\ell$. Define the $(\ell - 1)$ inter-event delays in $\mathcal{X}$ as follows: $\delta_j = (t_{j+1} - t_j),\ j = 1, \ldots, (\ell - 1)$. The fixed-delay episode, $(\alpha(\mathcal{X}), \mathcal{D}(\mathcal{X}))$, that is associated with the subset, $\mathcal{X}$, of event-indicators is defined by $\alpha(\mathcal{X}) = (A_{j_1} \rightarrow \cdots \rightarrow A_{j_\ell})$, and $\mathcal{D}(\mathcal{X}) = \{\delta_1, \ldots, \delta_{\ell-1}\}$. In graphical notation, the fixed-delay episode associated with $\mathcal{X}$ can be represented as follows:

$$(\alpha(\mathcal{X}), \mathcal{D}(\mathcal{X})) = (A_{j_1} \overset{\delta_1}{\rightarrow} \cdots \overset{\delta_{\ell-1}}{\rightarrow} A_{j_\ell}) \tag{4}$$

For computing mutual information we need the marginals of various subsets of event-indicators in the network. Given a subset like $\mathcal{X} = \{X_{j_1}(t_1), \ldots, X_{j_\ell}(t_\ell)\}$, we need estimates for probabilities of the form, $P[X_{j_1}(t_1) = a_1, \ldots, X_{j_\ell}(t_\ell) = a_\ell]$, where $a_j \in \{0, 1\}$, $j = 1, \ldots, \ell$. The fixed-delay episode, $(\alpha(\mathcal{X}), \mathcal{D}(\mathcal{X}))$, that is associated with $\mathcal{X}$ is given by *Definition 5.3* and its frequency in the data stream, s, is denoted by $f_s(\alpha(\mathcal{X}), \mathcal{D}(\mathcal{X}), W)$ (as per *Definition 5.2*) where W denotes length of history window as per **A1**. Since an occurrence of the fixed-delay episode, $(\alpha(\mathcal{X}), \mathcal{D}(\mathcal{X}))$, can terminate in each of the $(T - W)$ time-ticks in s, the probability of an all-ones assignment for the random variables in $\mathcal{X}$ is given by:

$$P[X_{j_1}(t_1) = 1, \ldots, X_{j_\ell}(t_\ell) = 1] = \frac{f_s(\alpha(\mathcal{X}), \mathcal{D}(\mathcal{X}), W)}{T - W} \tag{5}$$

For all other assignments (i.e. for assignments that are not all-ones) we use inclusion-exclusion to obtain corresponding probabilities. Inclusion-exclusion has been used before in data mining, e.g., in [12], to obtain exact or approximate frequency counts for arbitrary boolean queries using only counts of *frequent* itemsets in the data. In our case, counting distinct occurrences of fixed-delay episodes facilitates use of the inclusion-exclusion formula for obtaining the probabilities needed for computing mutual information of different candidate parent-sets. Consider the set, $\mathcal{X} = \{X_{j_1}(t_1), \ldots, X_{j_\ell}(t_\ell)\}$, of ℓ event-indicators, and let $\mathcal{A} = (a_1, \ldots, a_\ell),\ a_j \in \{0, 1\},\ j = 1, \ldots, \ell$, be an assignment for the event-indicators in $\mathcal{X}$. Let $\mathcal{U} \subset \mathcal{X}$ denote the subset of indicators out of $\mathcal{X}$ for which corresponding assignments (in $\mathcal{A}$) are 1's, i. e. $\mathcal{U} = \{X_{j_k} \in \mathcal{X} : k \text{ s.t.}$

Procedure 1 Overall Procedure

Input: Alphabet $\mathcal{E}$, event stream $s = \langle (E_1, \tau_1), \ldots, (E_n, \tau_n = T) \rangle$, length W of history window, conditional probability upper-bound ϵ^*, mutual information threshold, ϑ

Output: DBN structure (parent-set for each node in the network)

1: **for all** $A \in \mathcal{E}$ **do**
2: $X_A :=$ event-indicator of A at any time $t > W$
3: Set $f_{min} = (T - W) P_{min} \Phi_{min}$, using Eqs. (2)-(3)
4: Obtain set, $\mathcal{C}$, of fixed-delay episodes ending in A, with frequencies greater than f_{min} (cf. Sec. VI-B, *Procedure 2*)
5: **for all** fixed-delay episodes $(\alpha, \mathcal{D}) \in \mathcal{C}$ **do**
6: $\mathcal{X}_{(\alpha,\mathcal{D})} :=$ event-indicators corresponding to $(\alpha, \mathcal{D})$
7: Compute mutual information $I[X_A \, ; \, \mathcal{X}_{(\alpha,\mathcal{D})}]$
8: Remove $(\alpha, \mathcal{D})$ from $\mathcal{C}$ if $I[X_A \, ; \, \mathcal{X}_{(\alpha,\mathcal{D})}] < \vartheta$
9: Prune $\mathcal{C}$ using conditional mutual information criteria to distinguish direct from indirect influences (cf. Sec. VI-C)
10: Return (as parent-set for X_A) event-indicators corresponding to episodes in $\mathcal{C}$

Procedure 2 $pattern\_grow(\alpha, \mathcal{D}, \mathcal{L}_{(\alpha,\mathcal{D})})$

Input: ℓ-node episode $(\alpha, \mathcal{D}) = (A_{j_1} \xrightarrow{\delta_1} \cdots \xrightarrow{\delta_{\ell-1}} A_{j_\ell})$ and event sequence $s = \langle (E_1, \tau_1), \ldots, (E_n, \tau_n = T) \rangle$, Length of history window W, Frequency threshold f_{min}.

1: $\Delta = W - span(\alpha, \mathcal{D})$
2: **for all** $A \in \mathcal{E}$ **do**
3: **for** $\delta = 0$ to Δ **do**
4: **if** $\delta = 0$ **and** $(A_{j_1} > A$ **or** $\ell = 1)$ **then**
5: **continue**
6: $(\alpha', \mathcal{D}') = A \xrightarrow{\delta} \alpha$; $\mathcal{L}_{(\alpha',\mathcal{D}')} = \{\}$; $f_s(\alpha', \mathcal{D}') = 0$
7: **for all** $\tau_i \in \mathcal{L}_{(\alpha,\mathcal{D})}$ **do**
8: **if** $\exists (E_j, \tau_j)$ such that $E_j = A$ and $\tau_i - \tau_j = \delta$ **then**
9: Increment $f_s(\alpha', \mathcal{D}')$
10: $\mathcal{L}_{(\alpha',\mathcal{D}')} = \mathcal{L}_{(\alpha',\mathcal{D}')} \cup \{\tau_j\}$
11: **if** $f_s(\alpha', \mathcal{D}') \geq f_{min}$ **then**
12: Add $(\alpha', \mathcal{D}')$ to output set $\mathcal{C}$
13: **if** $span(\alpha', \mathcal{D}') \leq W$ **then**
14: $pattern\_grow(\alpha', \mathcal{D}', \mathcal{L}_{(\alpha',\mathcal{D}')})$

$a_k = 1$ in $\mathcal{A}, 1 \leq k \leq \ell\}$. Inclusion-exclusion is used to compute the probabilities as follows:

$$P[X_{j_1} = a_1, \ldots, X_{j_\ell} = a_\ell] = \sum_{\substack{\mathcal{Y} \text{ s.t.} \\ \mathcal{U} \subseteq \mathcal{Y} \subseteq \mathcal{X}}} (-1)^{|\mathcal{Y} \setminus \mathcal{U}|} \left(\frac{f_s(\mathcal{Y})}{T - W} \right) \quad (6)$$

where $f_s(\mathcal{Y})$ is short-hand for $f_s(\alpha(\mathcal{Y}), \mathcal{D}(\mathcal{Y}), W)$, the frequency (cf. *Definition 5.2*) of the fixed-delay episode, $(\alpha(\mathcal{Y}), \mathcal{D}(\mathcal{Y}))$.

VI. Algorithms

A. Overall approach

In Secs. III-V, we developed the formalism for learning an optimal DBN structure from event streams by using distinct occurrences-based counts of fixed-delay episodes to compute the DBN marginal probabilities. The top-level algorithm (cf. Sec. III) for discovering the network is to fix any time $t > W$, to consider each $X_j(t)$, $j = 1, \ldots, M$, in-turn, and to find its set of parents in the network. Due to the translation invariance assumption **A2**, we need to do this *only once for each event-type* in the alphabet.

The algorithm is outlined in *Procedure 1*. For each $A \in \mathcal{E}$, we first compute the minimum frequency for episodes ending in A based on the relationship between mutual information and joint probabilities as per *Theorem 4.1* (line 3, *Procedure 1*). Then we use a pattern-growth approach (see *Procedure 2*) to discover all patterns terminating in A (line 4, *Procedure 1*). Each frequent pattern corresponds to a set of event-indicators (line 6, *Procedure 1*). The mutual information between this set of indicators and the node X_A is computed using exclusion-exclusion formula and only sets for which this mutual information exceeds ϑ are retained as candidate parent-sets (lines 5-8, *Procedure 1*). Finally, we prune out candidate parent-sets which have only *indirect influences* on A and return the final parent-sets for nodes corresponding to event-type A (lines 9-10, *Procedure 1*). This pruning step is based on some conditional mutual information criteria (to be described later in Sec. VI-C).

B. Discovering fixed-delay episodes

We employ a pattern-growth algorithm (Procedure 2) for mining frequent fixed-delay episodes because, unlike *Apriori*-style algorithms, pattern-growth procedures allow use of different frequency thresholds for episodes ending in different alphabets. This is needed in our case, since, in general, *Theorem 4.1* prescribes different frequency thresholds for nodes in the network corresponding to different alphabets. The recursive procedure is invoked with $(\alpha, \mathcal{D}) = (A, \phi)$ and frequency threshold $f_{min} = (T - W) P_{min} \phi_{min}$ (Recall that in the main loop of *Procedure 1*, we look for parents of nodes corresponding to event-type $A \in \mathcal{E}$).

The pattern-growth algorithm listed in *Procedure 2* takes as input, an episode $(\alpha, \mathcal{D})$, a set of start times $\mathcal{L}_{(\alpha,\mathcal{D})}$, and the event sequence s. $\mathcal{L}_{(\alpha,\mathcal{D})}$ is a set of time stamps τ_i such that there is an occurrence of $(\alpha, \mathcal{D})$ starting at τ_i in s. For example, if at level 1 we have $(\alpha, \mathcal{D}) = (C, \phi)$, then $\mathcal{L}_{(C,\phi)} = \{1, 4, 5, 8, 9\}$ in the event sequence s shown in Fig 3. The algorithm obtains counts for all episodes like $(\alpha', \mathcal{D}')$

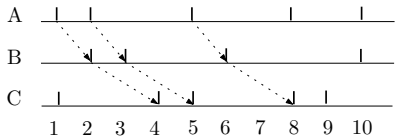


Figure 3. An event sequence showing 3 distinct occurrences of the episode $A \xrightarrow{1} B \xrightarrow{2} C$.

generated by extending $(\alpha, \mathcal{D})$ e.g. $B \xrightarrow{1} C$, ..., $A \xrightarrow{5} C$ etc. For an episode say $(\alpha', \mathcal{D}') = B \xrightarrow{2} C$, the count is obtained by looking for occurrences of event B at times $\tau_j = \tau_i - 2$ where $\tau_i \in \mathcal{L}_{(C,\phi)}$. In the example such B's at $\tau_j \in \mathcal{L}_{B \xrightarrow{2} C}$ = $\{2, 3, 6\}$. The number of such occurrences $(= 3)$ gives the count of $B \xrightarrow{2} C$. At every step the algorithm tries to grow an episode with count $f_s > f_{min}$ otherwise stops.

C. Conditional MI criteria

The final step in determining the parents of a node X_A involves testing of some conditional mutual information criteria (cf. line 10, *Procedure 1*). The input to the step is a set $\mathcal{C}$ of (frequent) fixed-delay episodes ending in A. Each episode in $\mathcal{C}$ is associated with a set of event-indicators whose mutual information with X_A exceeds ϑ. Consider two such sets $\mathcal{Y}$ and $\mathcal{Z}$, each having sufficient mutual information with X_A. Our conditional mutual information criterion is: *remove $\mathcal{Y}$ from the set of candidate parents (of X_A), if* $I[X_A\ ;\ \mathcal{Y} \mid \mathcal{Z}] = 0$[3]. We repeat this test for every pair of episodes in $\mathcal{C}$.

To understand the utility of this criterion, there are two cases to consider: (i) either $\mathcal{Y} \subset \mathcal{Z}$ or $\mathcal{Z} \subset \mathcal{Y}$, and (ii) both $\mathcal{Y} \not\subset \mathcal{Z}$ and $\mathcal{Z} \not\subset \mathcal{Y}$. In the first case, our conditional mutual criterion will ensure that we pick the larger set as a parent only if it brings more information about X_A than the smaller set. In the second case, we are interested in eliminating sets which have a high mutual information with X_A because of indirect influences. For example, if the network were such that C excites B and B excites A, then X_C can have high mutual information with X_A, but we do not want to report X_C as a parent of X_A, since C influences A only *through* B. Our conditional mutual information criterion will detect this and eliminate X_C from the set of candidate parents (of X_A) because it will detect $I[X_A\ ;\ X_C \mid X_B] = 0$.

VII. Experimental Results

We present results on data gathered from both mathematical models of spiking neurons as well as real neuroscience datasets.

A. Neuronal network model

The approach here is to model each neuron as an inhomogeneous Poisson process whose firing rate is a function of the input received by the neuron is recent past [13]:

$$\lambda_i(t) = \frac{\lambda}{1 + exp(-I_i(t) + \delta)} \tag{7}$$

Eq. (7) gives the firing rate of the i^{th} neuron at time t. The network inter-connect allowed by this model gives it the amount of sophistication required for simulating higher-order interactions. More importantly, the model allows for variable delays which mimic the delays in conduction pathways of real neurons.

$$I_i(t) = \sum_j \beta_{ij} Y_{j(t-\tau_{ij})} + \ldots + \sum_{ij\ldots l} \beta_{ij\ldots l} Y_{j(t-\tau_{ij})} \ldots Y_{l(t-\tau_{il})} \tag{8}$$

In Eq. (8), $Y_{j(t-\tau_{ij})}$ is the indicator of the event of a spike on j^{th} neuron τ_{ij} time earlier and the $\beta_{(.)}$s are the weight parameters for the interactions. The higher order terms in the input contribute to the firing rate only when the i^{th} neuron received inputs from all the neurons in the term with corresponding delays. With suitable choices of parameters $\beta_{(.)}$, one can simulate a wide range of networks.

B. Types of Networks

In this section we demonstrate the effectiveness of our approach in unearthing different types of networks. Each of these networks was simulated by setting up the appropriate inter-connections, of suitable order, in our mathematical model.

Causative chains and higher order structures: A higher-order chain is one where parent sets are not restricted to be of cardinality one. In the example network of Fig. 4(a), there are four disconnected components with two of them having cycles. (Recall this would be 'illegal' in a static Bayesian network formulation.) Also the component consisting of nodes $18, 19, 20, 21$ exhibits higher-order interactions. The node 20 fires with high probability when node 18 has fired 4 ms before and node 18 has fired 5 ms before. Similarly node 21 is activated by node $18, 19, 20$ firing at respective delays. The complete network consists of 100 nodes (with the remainder of the nodes firing independently). Spike train data is generated for runs of 60 sec using the multi-neuronal simulator. The base firing rate for neurons is set at 20Hz and the activation probability of a child node (i.e. the conditional probability that the child node fires given its parents) is varied form 0.6 to 0.8 (by suitably selecting β_{ij}'s). Our algorithm reports good precision and recall over a range of ϑ $(0.05 \leq \vartheta \leq 0.5)$ and $\epsilon*$ $(0.02 \leq \epsilon* \leq 0.04)$. For lower values of (simulation) conditional probability, recall gradually drops but precision remains high (100%). Details are shown in Table II.

Overlapping causative chains: The graph shown in Fig. 4(a) (b) has two chains $0 \rightarrow 1 \rightarrow 2 \rightarrow 3$ and $12 \rightarrow 1 \rightarrow 13 \rightarrow 14 \rightarrow 15 \rightarrow$ which share the node 1. Here 1 can be independently excited by 0 or 12. Also 2 is activated by $0, 1$ together and 13 is activated by $13, 1$.

[3]We use a small threshold parameter to ascertain this equality.

Table II
RESULTS FOR NETWORK SHOWN IN FIG. 4(A) FOR VARYING CONDITIONAL PROBABILITY (USED IN GENERATION) AND MIN. MI ϑ; BASE FIRING RATE = 20HZ [PROBABILITY=0.02 IN 1MS BINS]; BASE RATE THRESHOLD $\epsilon^* = .03$.

Cond. prob	ϑ	Time (sec)	Recall (%)	Precision (%)
.6	.05	7.5	69.6	100
.6	.075	7.4	60.9	100
.6	.1	7.1	4.3	100
.6	.25	2.8	0.0	100
.9	.05	41.8	100	92.0
.9	.075	42.6	87.0	95.2
.9	.1	42.0	69.6	100
.9	.25	30.0	47.8	100
.9	.5	26.0	0.0	100

Table III
RESULTS FOR NETWORK SHOWN IN FIG. 4(D) FOR VARYING BASE RATE THRESHOLD ϵ^* AND MIN. MI ϑ; BASE FIRING RATE = 20HZ AND COND. PROB. = 0.9.

ϵ^*	ϑ	Time (in sec)	Recall (%)	Precision (%)
0.005	0.04	56.0	80.0	66.7
0.01	0.04	56.3	80.0	66.7
0.03	0.04	45.8	66.7	66.7
0.06	0.04	0.9	0.0	100.0
0.1	0.04	0.9	0.0	100.0
0.03	0.05	44.2	66.7	66.7
0.03	0.075	6.1	26.7	80.0
0.03	0.1	6.0	13.3	100.0
0.03	0.25	1.2	0.0	100.0
0.03	0.5	0.8	0.0	100.0

Thus a firing event on 0 excites the chain $0 \to 1 \to 2 \to 3$ where as a firing event on 12 excites the other chain. This shows one possible way in which neurons can participate in several different circuits at the same time (e.g. polychronous circuits [14]). Depending on the stimulus sequence, the same neurons can participate in different cascade firing events (encoding completely unrelated pieces of information). For each node, our formulation reports multiple sets of nodes that satisfy the minimum mutual information ϵ^*, thus unearthing 0 and 12 as two sets activating 1. $0, 1$ and $12, 1$ are also found to be the parent sets of 2 and 3 respectively (with high precision and recall).

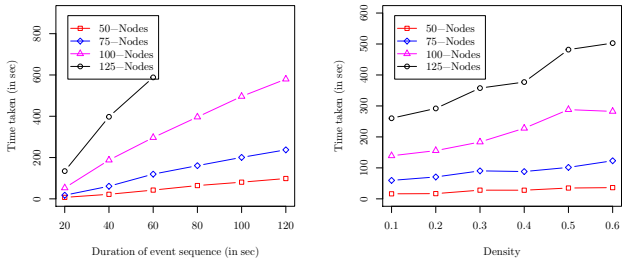
Syn-fire chains: Another important pattern often reported in neuronal spike train data is that of synfire chains. This consists of groups of synchronously firing neurons strung together repeating over time. In [10], it was noted that discovering such patterns required a combination of serial and parallel episode mining. But the DBN approach applies more naturally to mining such network structures. Again we are able to find the structure for a wide range of parameters. For larger histories or influence windows W, many combinations of nodes are frequent, slowing down the mining process. (For instance, network 4(c) takes 180 sec to mine as compared to 60 sec for network 4(a), on a dual core 3GHz Windows Vista computer with 3GB RAM.)

Polychronous circuits: Groups of neurons that fire in a time-locked manner with respect to each other are referred to as polychronous groups. This notion was introduced in [14] and gives rise to an important class of patterns. Once again, our DBN formulation is a natural fit for discovering such groups from spike train data. A polychronous circuit is shown in Fig 4(d). We are also able to discover overlapping polychronous circuits (where different sets of nodes can excite the same node). For relatively deep networks (having nodes with long ancestry) recall drops mainly because the nodes lower down in the graph are not excited sufficiently often (and hence do not meet the mutual information threshold). Detailed results are listed in Table III.

C. Scalability

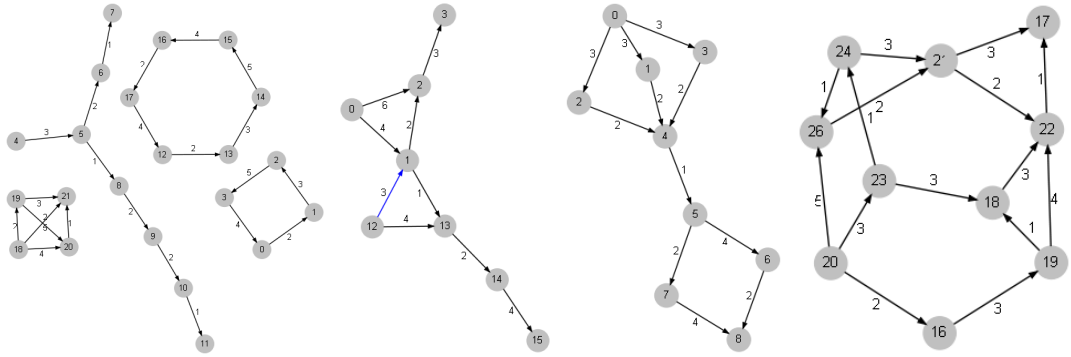
The scalability of our approach with respect to data length and number of variables is shown in Fig 5(a). Here four different networks with 50, 75, 100 and 125 variables respectively were simulated for time durations ranging from 20 sec to 120 sec. The base firing rate of all the networks was fixed at 20 Hz. In each network 40% of the nodes were chosen to have upto three parents. The parameters of the DBN mining algorithm were chosen such that recall and precision are both high ($> 80\%$). It can be seen in the figures that for a network with 125 variables, the total run-time is of the order of few minutes along with recall $> 80\%$ and precision at almost 100%.

Another way to study scalability is w.r.t. the density of the network, defined as the ratio of the number of nodes that are descendants for some other node to the total number of nodes in the network. Fig 5(b) shows the time taken for mining DBNs when the density is varied from 0.1 to 0.6, averaged over 36 datasets. We observe near linear growth in time taken and the absolute figures can be improved using native implementation (currently our algorithms are implemented in Python).



(a) Varying data length in sec (b) Varying network density

Figure 5. Plot of total time taken for DBN discovery



(a) Higher-order causative chains (b) Overlapping causative chains (c) Syn-fire Chains (d) Polychronous Circuits

Figure 4. Four classes of DBNs investigated in our experiments.

D. Sensitivity

Finally, we discuss the sensitivity of the DBN mining algorithm to the parameters ϑ, ϵ^* and cond. mutual information threshold (used to check for MI=zero). To obtain precision-recall curves for our algorithm applied to data sequences with different characteristics, we vary the parameter ϑ in the range [0.04-0.06] and repeat for different cond. mutual information threshold values. The data sequence for this experiment is generated from the multi-neuronal simulator using different settings of base firing rate, conditional probability, number of nodes in the network, and the density of the network.

The set of precision-recall curves are shown in Fig 6. The general trends observed here show that for a range of settings of the conditional probability, base firing rates, and network topology, both high precision and high recall can be obtained. As the stringency of the conditional mutual information threshold is increased (compare Fig. 6 (top) to Fig. 6 (bottom), we observe a deterioration of performance only w.r.t. the base rate threshold.

E. Cortical cultures

Multi-electrode arrays provide high throughput recordings of the spiking activity in neuronal tissue and are hence rich sources of event data where events correspond to specific neurons being activated. We use data from dissociated cortical cultures gathered by Steve Potter's laboratory at Georgia Tech [15] which gathered data over several days. The mining is done with mutual information threshold $\vartheta = 0.001$ with DBN search parameter $\epsilon^* = 0.02$.

In order to establish the significance of the networks discovered we run our algorithm on several surrogate spike trains generated by replacing the neuron labels of spikes in the real data with randomly chosen labels. These surrogates break the temporal correlations in the data and yet preserve the overall summary statistics. No network structure was found in such surrogate sequences. We are currently in the process of characterizing and interpreting the usefulness of such networks found in real data. An example network is shown in Fig. 7, reflecting the sustained bursts observed in this culture by Wagenaar *et al* [15].

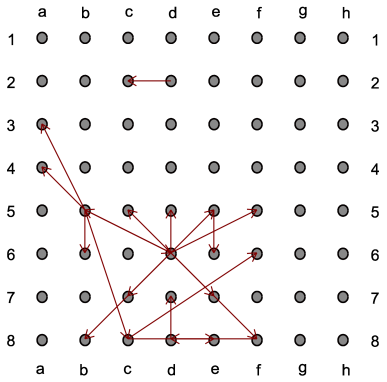


Figure 7. DBN structure discovered from first 15 min of spike train recording on day 35 of culture 2-1 [15].

VIII. Discussion

Our work marries frequent pattern mining with probabilistic modeling for analyzing discrete event stream datasets. DBNs provide a formal probabilistic basis to model relationships between time-indexed random variables but are intractable to learn in the general case. Conversely, frequent episode mining is scalable to large datasets but does not exhibit the rigorous probabilistic interpretations that are the mainstay of the graphical models literature. We have presented the beginnings of research to relate these two diverse threads and demonstrated its potential to mine excitatory networks with applications in spike train analysis.

Two key directions of future work are being explored. The excitatory assumption as modeled here posits an order over the entries of the conditional probability table but does not impose strict distinctions of magnitude over these entries. This suggests that, besides the conditional independencies inferred by our approach, there could potentially

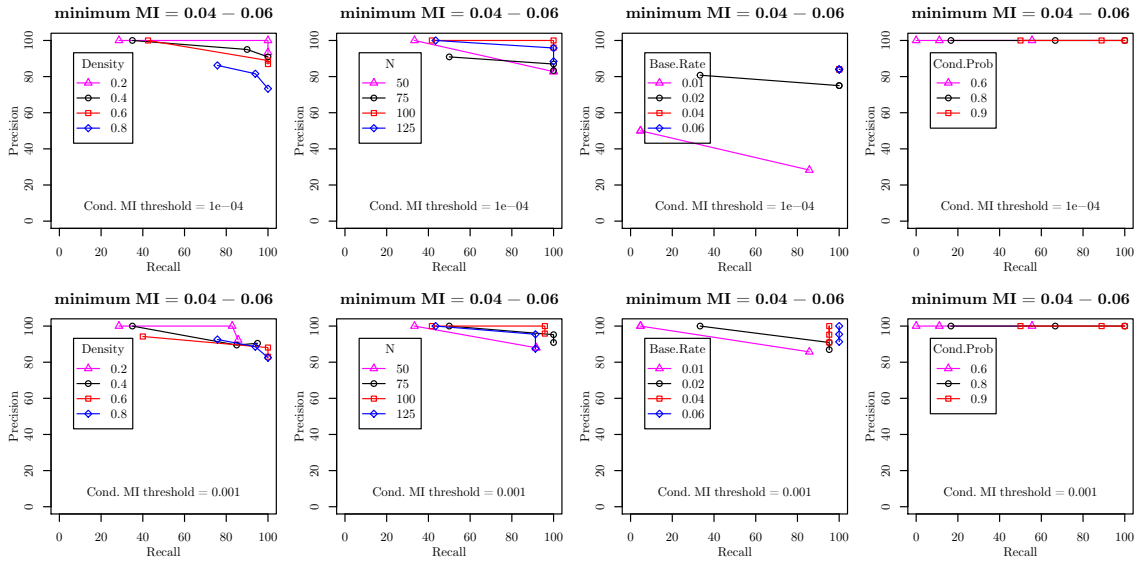


Figure 6. Precision-recall curves for different parameter values in the DBN mining algorithm.

be additional 'structural' constraints masquerading inside the conditional probability tables. We seek to tease out these relationships further. A second, more open, question is whether there are other useful classes of DBNs that have both practical relevance (like excitatory circuits) and which also can be tractably inferred using sufficient statistics of the form studied here.

ACKNOWLEDGMENT

This work is supported in part by General Motors Research, NSF grant CNS-0615181, and ICTAS, Virginia Tech. Authors thank V. Raajay and P. S. Sastry of Indian Institute of Science, Bangalore, for many useful discussions and for access to the neuronal spike train simulator of [13].

REFERENCES

[1] M. I. Jordan, Ed., *Learning in Graphical Models*. MIT Press, 1998.

[2] C. Chow and C. Liu, "Approximating discrete probability distributions with dependence trees," *IEEE Transactions on Information Theory*, vol. 14, no. 3, pp. 462–467, May 1968.

[3] J. Williamson, "Approximating discrete probability distributions with bayesian networks," in *Proc. Intl. Conf. on AI in Science & Technology, Tasmania*, 2000, pp. 16–20.

[4] M. Meila, "An accelerated chow and liu algorithm: Fitting tree distributions to high-dimensional sparse data," in *Proc. ICML'99*, 1999, pp. 249–257.

[5] N. Friedman, K. Murphy, and S. Russell, "Learning the structure of dynamic probabilistic networks," in *Proc. UAI'98*. Morgan Kaufmann, 1998, pp. 139–147.

[6] K. Murphy, "Dynamic Bayesian Networks: representation, inference and learning," Ph.D. dissertation, University of California, Berkeley, CA, USA, 2002.

[7] D. Patnaik, S. Laxman, and N. Ramakrishnan, "Inferring dynamic bayesian networks using frequent episode mining," *CoRR*, vol. abs/0904.2160, 2009.

[8] F. Rieke, D. Warland, R. Steveninck, and W. Bialek, *Spikes: Exploring the Neural Code*. The MIT Press, 1999.

[9] H. Mannila, H. Toivonen, and A. Verkamo, "Discovery of frequent episodes in event sequences," *Data Mining and Knowledge Discovery*, vol. 1, no. 3, pp. 259–289, 1997.

[10] D. Patnaik, P. S. Sastry, and K. P. Unnikrishnan, "Inferring neuronal network connectivity from spike data: A temporal data mining approach," *Scientific Programming*, vol. 16, no. 1, pp. 49–77, January 2007.

[11] S. Laxman, "Discovering frequent episodes: Fast algorithms, connections with hmms and generalizations," Ph.D. dissertation, IISc, Bangalore, India, September 2006.

[12] J. K. Seppanen, "Using and extending itemsets in data mining: Query approximation, dense itemsets and tiles," Ph.D. dissertation, Helsinki University of Technology, 2006.

[13] V. Raajay, "Frequent episode mining and multi-neuronal spike train data analysis," Master's thesis, IISc, Bangalore, 2009.

[14] E. M. Izhikevich, "Polychronization: Computation with spikes," *Neural Comput.*, vol. 18, no. 2, pp. 245–282, 2006.

[15] D. A. Wagenaar, J. Pine, and S. M. Potter, "An extremely rich repertoire of bursting patterns during the development of cortical cultures," *BMC Neuroscience*, 2006.