Identifying a phase transition between emergent stable dynamics in stochastic CTLNs

Caitlin Lienkaemper, Eric Han

Abstract. The Combinatorial Linear-Threshold Network (CTLN) model is a simplified mathematical model that focuses on the intricate connectivity between neurons as the key factor affecting the resulting dynamics of the network. Emergent dynamics are nonlinear and ultimately coalesce to one of two stable states. By electing to represent neural networks as graphs, we are able to exploit graph structures to determine the behavior of specific network structures. Previous studies of CTLNs have focused on characterizing the dynamics of specific graph structures. We take a stochastic approach that characterizes graphs based on two key parameters, and analyze the dynamics of graphs with specific characteristics rather than focusing on any specific graph structure. We identify a phase transition between the two stable states of random networks of varying sizes.

Contents

1 Preliminaries

The dynamics of threshold-linear networks are governed by the following ODE:

$$\frac{dx_i}{dt} = -x_i + \left[\sum_{j=1}^{n} W_{ij}x_j + \theta\right]_+ \text{, i = 1, 2,..., n,}$$

where n is the number of neurons. x_i represents the firing rate of the i-th neuron and $\theta > 0$ is a constant external input. W_{ij} is an $n \times n$ matrix that describes the connective strength between neurons i, j. The threshold non-linearity as described by $[\cdot]_+ := \max\{0, \cdot\}$ produces nonlinear dynamics within the network. CTLNs are a special case of the TLN model where we restrict connection strengths in W_{ij}. The dynamics exhibited can coalesce to one of two steady states[1], either a fixed point or a chaotic attractor.

[1] is this guaranteed? proof where?

Before continuing, we introduce a few important definitions.

Definition 1.1 (Directed Graph)**.** A graph where every edge has a direction. A graph where all edges are bidirectional is called an *undirected* graph.

Definition 1.2 (Clique)**.** A set of vertices of size k, σ_k, such that they are all connected with bidirectional edges.

Definition 1.3 (Target of a Clique)**.** A vertex that receives an edge from every vertex in σ_k.

Definition 1.4 (Maximality)**.** A clique is maximal if it has no targets.

Definition 1.5 (Fixed Point Support)**.** A set of vertices with no targets, that is, a maximal clique.

Definition 1.6 (Symmetry)**.** Let v_i, v_j be any two vertices with an edge between them in a directed graph. i and j are considered *symmetric* if there exists a bidirectional edge between them. A graph is considered fully symmetric if every edge in the graph is a bidirectional edge; or if there exist no edges in the graph.

In order to analyze this as a stochastic system, we must be able to parametrize characteristics of the CTLN that affect the dynamics manifested. The easiest and most intuitive way to do this is to parametrize edge connection probability as well as the symmetry of the graph. That is, we assign values to how "bidirectional" our network is. A completely symmetric network would be an undirected graph, while a completely asymmetric graph would only contain unidirectional edges. We then generate random graphs with these parameters and attempt to discern the average dynamics of a size n system with specific combinations of edge connection probability and symmetry.

To that end, we have 4 cases for possible edge generation: no edge, a forward edge from $v_i \to v_j$, a backward edge from $v_j \to v_i$, and a bidirectional edge. We assign the values $\left\{1-p, p\left(\frac{1-q}{2}\right), p\left(\frac{1-q}{2}\right), pq\right\}$ respectively. p serves as our edge connection parameter, and q serves as our symmetry parameter. It is important to note here that edge generation is independent.[2]

[2] justify this. I don't know how.

A program was written in Python that simulates graph generation and numerically simulates dynamics of those graphs generated. Random initial conditions are used. In order to detect whether a specific network coalesces to a fixed point or not, we explicitly calculate the numerical derivative for the last few timesteps of our solution and check to see if it falls within an acceptable threshold. We use this algorithm to simulate dynamics for the full spectrum of edge connection probabilities and symmetry parametrizations.

Currently, we are looking at the expected number of stable fixed points in any size n graph by using the expected number of target free cliques as a function of the edge connection probability p and the symmetry parameter q. The aim is to identify the expected number of fixed point supports (e.g. maximal cliques) as we take the large N limit of these stochastic networks.

2 Initial Results

We have generated heatmaps of averaged network dynamics that showcase general patterns for size-n networks across the full spectrum of edge connection probabilities and degrees of symmetry. Utilizing these heatmaps, we can see a distinct zone of phase transition where the dynamics go from coalescing to a fixed point to exhibiting strongly chaotic behavior.[3]

[3] It should be noted that chaotic behavior persists for longer for larger n graphs.

2.1 Undirected Graphs

Additionally, we have also identified that for networks of increasing size n, $n \to \infty$ results in the expected number of any fixed k-size clique going to 0, $\sigma_k \to 0$.

The main approach utilized requires that we can identify the expected number of maximal cliques for any set of parameters p and q. The expected number of maximal cliques is given by

$$\mathbb{E}_u[\text{maximal } \sigma_k] = \binom{N}{k} p^{\binom{k}{2}} \tag{1}$$

where p is the probability of an edge between any two vertices in the network.[4] Additionally, the probability that a clique is maximal is given by

[4] I believe this might need some refinement.

$$\mathbb{P}_u(\sigma_k \text{ is maximal}|\sigma_k \text{ exists}) = (1-p^k)^{N-k}. \tag{2}$$

Together, these two gives us the expected number of maximal cliques in any N-size undirected graph. The scheme for a directed graph is a little more complicated, but arises naturally.

2.2 Directed Graphs

First, we consider the expected number of maximal cliques. Instead of looking for general edge generation probability, we now look specifically at the probability that a bidirectional edge is generated, pq. So, the expected number of maximal cliques is given by

$$\mathbb{E}_d(\text{maximal } \sigma_k) = \binom{N}{k} pq^{\binom{k}{2}}. \tag{3}$$

Similarly, in order to find the probability that any k-clique is maximal, recall that we are essentially finding the complement of the probability that it is *not* maximal. That is, we require the probability that the clique has any targets. Thus, we identify the probability of there existing a feed-forward edge to a vertex outside of the clique, $p\left(\frac{1-q}{2}\right)$ and pq. So,

$$\mathbb{P}_d(\sigma_k \text{ is maximal}|\sigma_k \text{ exists}) = \left[1 - \left(pq + p\left(\frac{1-q}{2}\right)\right)^k\right]^{N-k}. \tag{4}$$

Moving forward, unless otherwise specified, we will always refer to the probabilities described for directed graphs.

The expected number of maximal cliques for an N-size network is given by the expected number of maximal cliques multiplied by the probability that any given clique is maximal given that it exists.

$$\mathbb{E}_d \cdot \mathbb{P}_d = \binom{N}{k} pq^{\binom{k}{2}} \left[1 - \left(pq + p\left(\frac{1-q}{2}\right)\right)^k\right]^{N-k}. \tag{5}$$

2.3 Numerical Results

Detailed below are the numerical methods used and their resulting simulations. These are the grounds that spurred the main goal[5] of the paper.

[5] Change to result once done!

To generate dynamics for individual graphs, we used a standard iterative method for numerically solving PDEs.

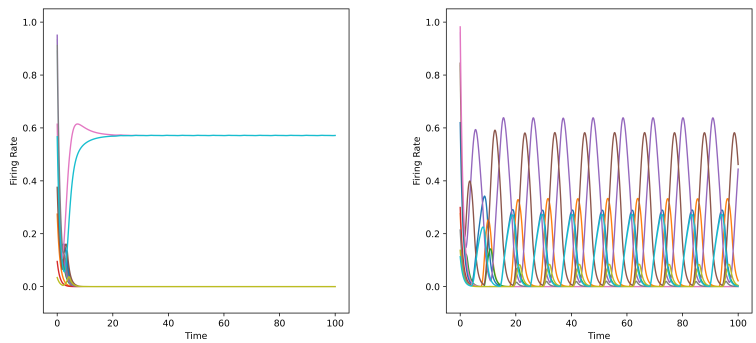

Figure 1: Two 15-networks demonstrating different behavior despite having the same parameters. $N = 15, p = 0.6, q = 0.1$. This highlights the variance in network dynamics, even in small networks. As network size is increased, outliers are less likely to occur, but coalescence to a steady state is much slower.

Accurate simulation of network dynamics means that we may average the behavior of randomly-generated networks across the entire spectrum of symmetry-edge connection values. To do this, we simply simulate the dynamics of networks with specific parameters multiple times in order to generate heatmaps that indicate how likely a specific pair of parameters is to result in a fixed point. We detect fixed points by numerically computing the derivatives at the final few[6] point in time.

[6] I forgot how many