Interpretable Machine Learning for Predicting the Synchronization of Coupled Oscillators

REU 2022

Agam Goyal
Bella Wu
Binhao Chen
Bryan Xu
Advisor: Hanbaek Lyu

Department of Mathematics
University of Wisconsin - Madison

August 31, 2022

Table of Contents

Table of Contents

Background

• **Problem of Interest:** Theoretically understanding the large-scale behavior of any complex systems that consisting of locally interacting dynamic agents. In particular, the long-term global synchronization of coupled oscillators.

• **Difficulty:** In spite of several sufficient conditions for model parameters (e.g., large coupling strength) or initial configuration (e.g., phase concentration being an open semicircle) are known, it often seems intractable to obtain analytical or asymptotic solutions to prediction problems.

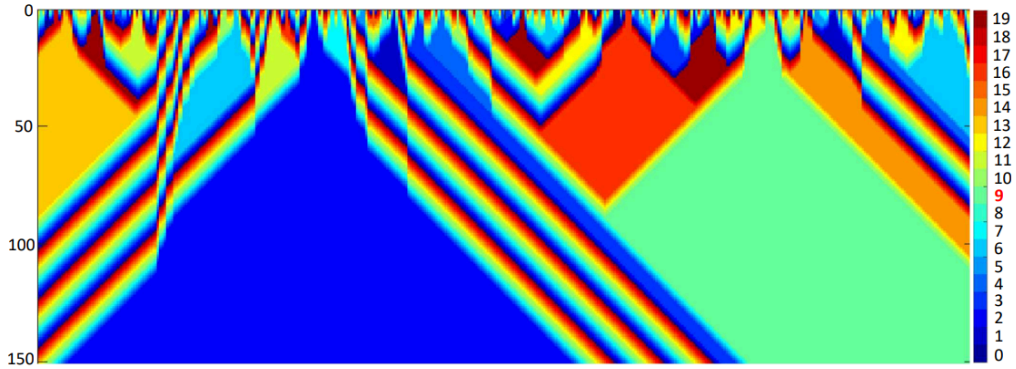


Figure: Example of Partial Synchronization of a System of Coupled Oscillators. (Simulation of 20-color FCA on a path of 400 nodes for 150×20)

Prior Work & Our Approach

• **Prior Work:** *Learning to Predict Synchronization* (L2PSync) views the synchronization prediction problem as a binary classification task. This framework shows fundamental classification algorithms trained on large enough datasets of initial dynamics can successfully predict whether a system on highly heterogeneous sets of unknown graphs will eventually synchronize with surprisingly high accuracy.

• **Our Work: Interpretable L2PSync.** Given any connected graph $G = (V, E)$, coupling ϕ, and fixed parameters $n \in \mathbb{N}, T \gg r > 0$. Develop a machine learning method that can predict the following indicator function $\mathbb{1}\,(X_T$ is synchronized$)$[1] while also output discriminating features that are used for classification, based on the initial trajectory $(X_t)_{0 \le t \le r}$ that are determined by graph topology and the coupling ϕ and optionally also with statistics of graph G.

• **Our Approach:** Our work could be viewed as a multi-objective optimization problem, which is hoping to combine two main ingredients, namely, *cellular automata* and *supervised dictionary learning*, to achieve a decent accuracy of the classification task, similar to the methods deployed in L2PSync, while providing reasonable guarantees of the interpretability of features selected by the dictionary learning algorithm simultaneously.

[1] Throughout this work, $\mathbb{1}(\cdot)$ denotes the indicator function.

Table of Contents

Models of Coupled Oscillators

- Main barrier: Most traditional oscillator and clock synchronization models assume that each oscillator is continuous-time and continuous-state $\Rightarrow$ the dynamics of the oscillators quickly become intractable on heterogeneous underlying graphs.

- Solution: Discretize the time and state of the model by using *Cellular Automata* instead of the usual continuous model.

- Three mathematically-justified models of Coupled Oscillators:
  - (1) Kuramoto Model (KM)
  - (2) Firefly Cellular Automata (FCA)
  - (3) Greenberg-Hastings Model (GHM)

Kuramoto Model

Consider a graph $G = (V, E)$, and a continuous phase space $\Omega = \mathbb{R}/2\pi\mathbb{Z}$. The evolution of the phase dynamics of the initial phase configuration $X_0 : V \to \Omega$ is determined by the following system of ordinary differential equations.

$$\frac{d}{dt}X_t(v) = \omega_v + K \sum_{u \in \mathcal{N}(v)} \sin(X_t(u) - X_t(v)) \qquad \forall\ v \in V \tag{1}$$

where $\mathcal{N}$(v) represents the set of nodes neighboring v in G, ω_v denotes the intrinsic frequency of v, and K denotes the *coupling strength* of the model.

Kuramoto Model

One of the most well-studied continuous-state oscillator models over the years.

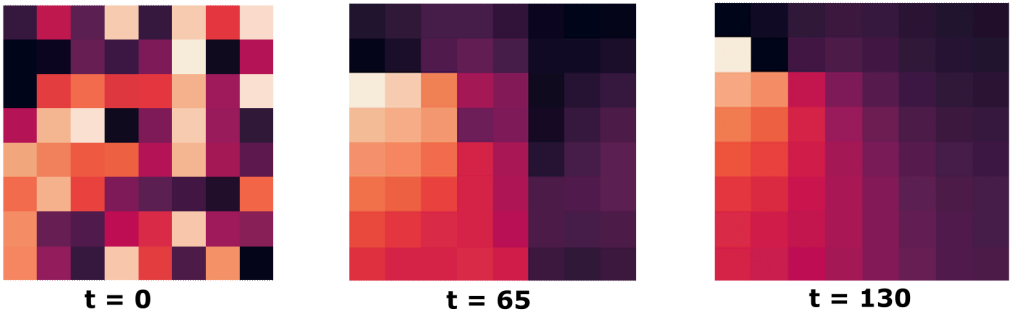


Figure: Simulation of the Kuramoto Model on an 8×8 2D-Grid graph (8-by-8 2D Lattice). Each heap map represents the phase configuration X_t of the system at the corresponding iteration t mentioned below the figure.

Firefly Cellular Automata (FCA)

Assume a finite simple graph $G = (V, E)$. The number of nodes is denoted by n. The map X maps the set of vertices to a corresponding state, written as $X : V \to Z/\kappa Z$, where the later is a cyclic group taking order $0 < 1 < \cdots < \kappa - 1$. At any specific time t, the node v takes the state/coloring of $X_t(v)$. Define the neighbors of blinking states as a set $N(b)$, the blinking state as $b(\kappa) = \lfloor \frac{\kappa - 1}{2} \rfloor$, then the transition rule of FCA writes:

$$\text{(FCA)} \qquad X_{t+1}(v) = \begin{cases} X_t(v) & X_t(v) > b(\kappa), \text{and } X_t(u) = b(\kappa) \text{ for some } u \in N(v), \\ X_t(v) + 1 & \text{otherwise} \end{cases} \tag{2}$$

Firefly Cellular Automata (FCA)

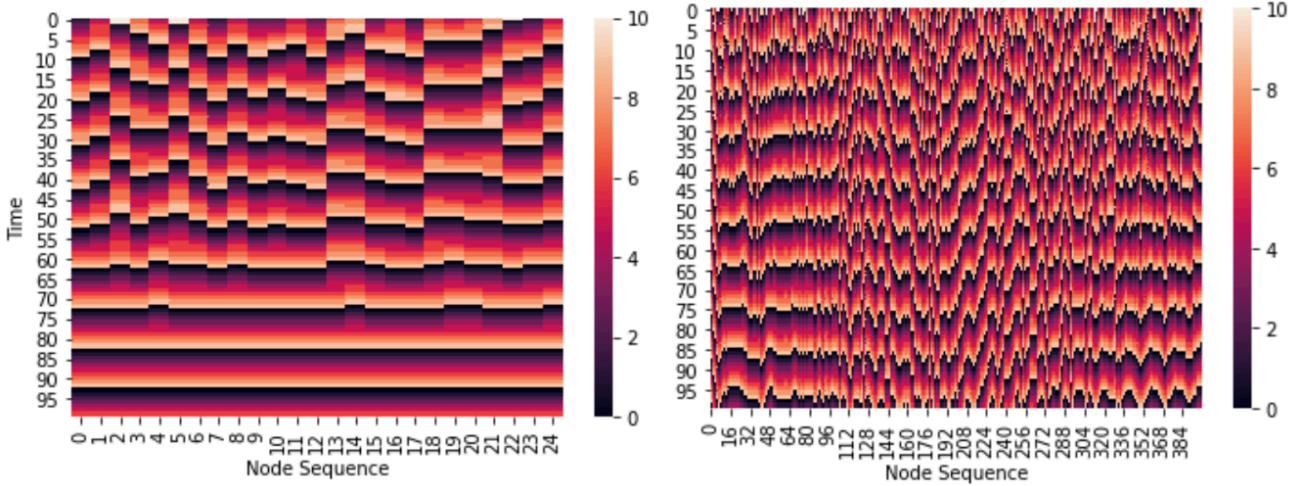


Figure: Illustrative example of FCA($\kappa = 10$). Left: A synchronizing example of FCA model on 5×5 2d grid. Right: A non-synchronizing example of FCA model on 20×20 2d grid.

.

Greenberg-Hastings Model (GHM) Transition Rule

GHM: Emulates an excitable media, each element of which has excitation potential. The neighbors affect each other in a diffusive local transportation fashion. e.g. forest fire

GHM Transition Rule: With a graph $G(V, E)$ and a phase space $\Omega = \mathbb{Z}/\kappa\mathbb{Z} = \{0, 1, ...\kappa - 1\}$, we obatin the following phase mapping $X_t : V \to \Omega$. The time evolution of the phase states for each node v in graph G follows the following rules where $N(v)$ gives the neighbor set of node v.

$$\text{(GHM)} \qquad X_{t+1}(v) = \begin{cases} 0 & \text{if } X_t(v) = 0 \quad \& \quad X_t(u) \neq 1 \, \forall u \in N(v) \\ 1 & \text{if } X_t(v) = 0 \quad \& \quad \exists \, u \in N(v) \text{ s.t. } X_t(u) = 1 \\ (X_t(v) + 1) mod(\kappa) & \text{otherwise} \end{cases} \tag{3}$$

Hereafter we will refer $X_t(v) = 0$ as quiescent state, $X_t(v) = 1$ as excited state, and $X_t(v)$ otherwise as refractory states.

Greenberg-Hastings Model (GHM) Dynamics Visualization

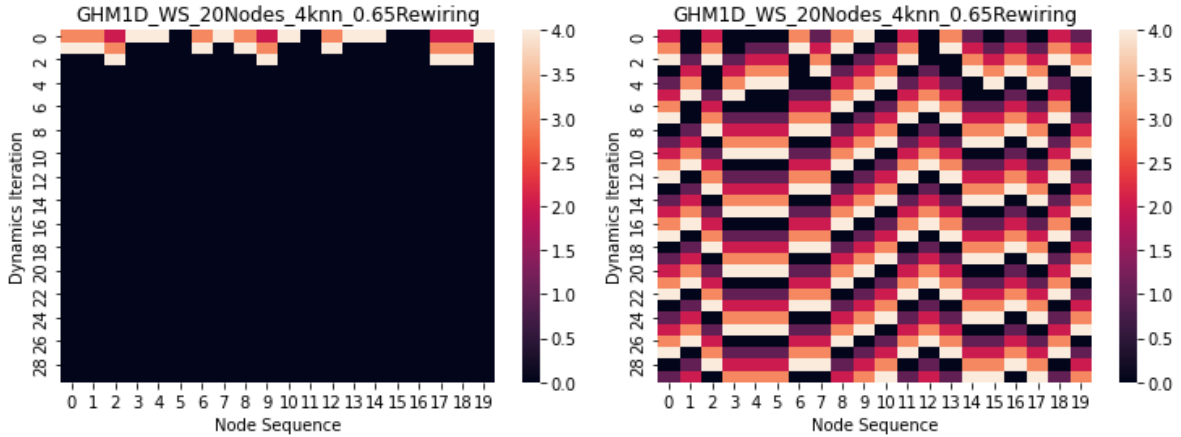


Figure: Simulation of the GHM synchronizing and non-synchronizing cases on 20-node Watts Strogatz graph sof 4 nearest neighbors with 0.65 rewiring probability.

Graph Embedding Algorithms

Graph Embedding: The transformation of any given property graphs to a vector, or sometimes a set of several vectors. This kind of embedding algorithms aim to capture the major characteristics of the graph, including graph topology, vertex-to-vertex relationship, and also other relevant information of the graphs, subgraphs, and vertices.

- Make Prediction of the synchronization problem of the coupled oscillator system exclusively using graph features, i.e., no information about the dynamics of each node is included.

- Check whether these advanced graph embedding algorithms would yield a surprising classification accuracy or not, which might be of interest in verifying that dynamics is a crucial part of the information to ensure a decent classification accuracy.

Spectral Embedding

Spectral embedding is an approach to calculating a nonlinear embedding by using nonlinear dimensionality reduction.

- Normally this algorithm implements Laplacian Eigenmaps, which finds a low dimensional representation of the data using a spectral decomposition of the graph Laplacian. It forms an affinity matrix given by the specified function and applies spectral decomposition to the corresponding graph laplacian.

- Minimization of a cost function based on the graph ensures that points close to each other on the manifold are mapped close to each other in the low dimensional space, preserving local distances.

Node2Vec

Node2Vec is a *semi-supervised* [2] framework for learning continuous feature representations for nodes in networks, having the following features

- **Objective:** Learn a mapping of nodes to a low dimensional feature space that maximizes the likelihood of preserving network neighborhoods of nodes.
- **Flexibility:** The notion of a network neighborhood is flexible by using a biased random walk procedure using different sampling techniques like BFS and DFS
- **Use cases:** Node2Vec performs better than the state-of-the-art algorithms on the following tasks:
  - Multi-label Classification of nodes in a network
  - Link prediction between nodes of a network

[2] uses small amount of labeled data combined with large amount of unlabeled data

Fundamentals of Node2Vec

Let $G = (V, E)$ be any (un)directed and (un)weighted network and let $f : V \to \mathbb{R}^d$ be the mapping function from nodes to the feature representation of dimension d.

For every node $u \in V$, we define a *network neighborhood* $N_S(u) \subset V$ that is generated through a sampling strategy S - biased random walk.

Using the idea of the *skip-gram* architecture, we solve a **modified version**[3] of the following optimization problem using Stochastic Gradient Descent, which maximizes the log-likelihood of observing a network neighborhood $N_S(u)$ for node u conditioned on its feature representation:

$$\max_f \sum_{u \in V} log\ Pr(N_S(u) \mid f(u))$$

[3] using conditional probability and negative sampling

Graph2Vec

Graph2Vec is an *unsupervised* framework for learning distributed representations of arbitrary sized graphs, having the following features:

- **Objective:** Learn low dimensional graph embeddings/distributed representations without the need of class labels primarily for graph classification and clustering
- **Novelty:** Two features of Graph2Vec makes it stand out as opposed to existing frameworks like *Graph Kernels*[4]:
  - Allows the use of ML algorithms because of a generic representation
  - Promotes data-driven embeddings as opposed to handcrafted decompositions (like Hamiltonian path in our case)
- **Structural equivalence:** Graph2Vec samples non-linear substructures in form of rooted subgraphs, which helps in yielding similar embeddings for structurally similar graphs.

[4] evaluate the similarity (using kernel function) between a pair of graphs G and G' by recursively decomposing them into substructures

Fundamentals of Graph2Vec

Let $\mathbb{G} = \{G_1, G_2, \cdots\}$ be a set of graphs and δ a positive integer, we want to learn $\delta-$dimensional distributed representations for every graph $G_i \in \mathbb{G}$. The matrix representation of embeddings is denoted as $\Phi \in \mathbb{R}^{|G| \times \delta}$.

For each node $n \in N_i$ in graph $G_i \in \mathbb{G}$, get a rooted subgraph using the Weisfeiler-Lehman (WL) kernel. Rooted subgraphs are used because compared to lower order substructures like nodes, they capture graph features better, and are non-linear leading to structural equivalence.

Again using *skip-gram* architecture, we solve the following optimization problem, again using Stochastic Gradient Descent, which maximizes the log-likelihood of observing a rooted subgraph neighborhood $sg_n^{(d)}$ for node n with degree d, conditioned on its embedded representation:

$$\max_{\Phi} \sum_{n \in N_i} log\ Pr(sg_n^{(d)} \mid \Phi(\mathbb{G}))$$

Dictionary Learning

- Dictionary Learning is a machine learning technique that is used to learn interpretable latent structures of complex data sets in order to realize what features of the data the model considers to be the most relevant ones for its task. It consists of two main tasks:
  - Sampling a large number of structured subsets (usually square patches) of a data set.
  - Applying *non-negative matrix factorization* as described below, to find a set of basis elements that form our *dictionary*.

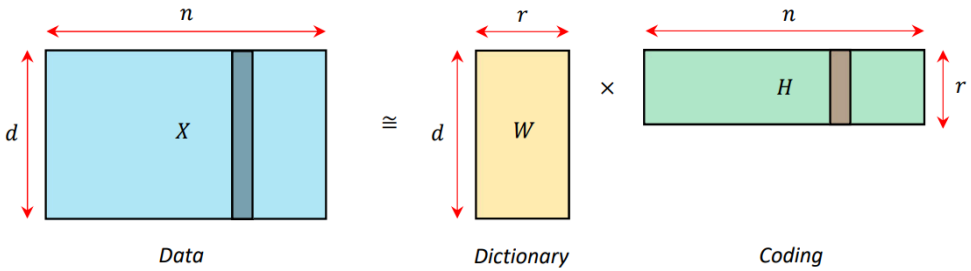


Figure: Illustration of matrix factorization.

Non-negative Matrix Factorization

- Given a large data matrix $\mathbf{X}$, can we find some small number of "dictionary vectors" so that we can represent each column of the data matrix as a linear combination of dictionary vectors? More precisely, given a data matrix $\mathbf{X} \in \mathbb{R}^{d \times n}$ and a rank parameter $r \in \mathbb{N}$, we wish to factorize X into the product of $\mathbf{W} \in \mathbb{R}^{d \times r}$ and $\mathbf{H} \in \mathbb{R}^{r \times n}$ by solving the following optimization problem.

$$\inf_{\mathbf{W} \in \mathbb{R}^{d \times r}, \mathbf{H} \in \mathbb{R}^{r \times n}} \|\mathbf{X} - \mathbf{W}\mathbf{H}\|_F^2$$

Here $\mathbf{W}$ is called the dictionary and $\mathbf{H}$ is the code of data $\mathbf{X}$ using dictionary $\mathbf{W}$.

- (Dictionary Learning)

$$\inf_{\mathbf{W} \in \mathbb{R}^{d \times r}, \mathbf{H} \in \mathbb{R}^{r \times n}_{\geq 0}} \|\mathbf{X} - \mathbf{W}\mathbf{H}\|_F^2$$

- (Non-negative Matrix Factorization)

$$\inf_{\mathbf{W} \in \mathbb{R}^{d \times r}_{\geq 0}, \mathbf{H} \in \mathbb{R}^{r \times n}_{\geq 0}} \|\mathbf{X} - \mathbf{W}\mathbf{H}\|_F^2$$

Supervised Dictionary Learning

- Supervised dictionary learning (SDL) provides systematic approaches to balance some degree of trade-off between dictionary learning and classification, the objective of SDL can naturally be formulated as a multi-objective optimization problem as below:

$$\min_{\mathbf{W},\mathbf{H},\boldsymbol{\beta}} L(\mathbf{W},\mathbf{H},\boldsymbol{\beta}) := \left(\sum_{i=1}^{n} \ell\left(y_i, \mathbf{g}\left(\mathbf{a}\left(\boldsymbol{x}_i, \mathbf{W}, \mathbf{h}_i, \boldsymbol{\beta}\right)\right)\right) \right) + \xi \left\| \mathbf{X}_{\text{data}} - \mathbf{W}\mathbf{H} \right\|_F^2$$

subject to: Constraints on $\mathbf{W} \in \mathbb{R}^{p\times r}, \mathbf{H} \in \mathbb{R}^{r\times n}$, and $\boldsymbol{\beta} \in \mathbb{R}^{r\times\kappa}$

where $\mathbf{X}_{\text{data}} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{p\times n}, \mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_n] \in \mathbb{R}^{r\times n}$, and $\ell(\cdot)$ is a classification loss and is usually taken as the negative log likelihood

$$\ell\left(y_i, \mathbf{g}\left(\mathbf{a}\left(\boldsymbol{x}_i, \mathbf{W}, \mathbf{h}_i, \boldsymbol{\beta}\right)\right)\right) := -\sum_{j=0}^{1} \mathbf{1}\left(y_i = j\right) \log\left\{g_j\left(\mathbf{a}\left(\boldsymbol{x}_i, \mathbf{W}, \mathbf{h}_i, \boldsymbol{\beta}\right)\right)\right\}.$$

Here, the tuning parameter ξ controls the trade-off between the two objectives of classification and dictionary learning. We allow to put desired constraints on the parameters $\{\mathbf{W}, \mathbf{H}, \boldsymbol{\beta}\}$. In particular, we will consider nonnegativity constraints on $\mathbf{W}$ and $\mathbf{H}$ as in the supervised nonnegative matrix factorization (SNMF) model to enjoy the nice interpretability of NMF in the supervised setting.

Concentration Principle and Baseline Predictor

Concentration Principle: Let G be an arbitrary connected graph. For the Kuramoto Model (KM) with identical intrinsic frequency and for Firefly Cellular Automata (FCA), the given dynamics on G synchronize if all phases at any given time are confined in an open half-circle[5] in the phase space Ω. Furthermore, if all states used in the configuration X_t are confined in an open half-circle for any $t \geq 1$, then the trajectory on G eventually synchronizes.

For the Greenberg-Hastings Model, the concentration principle doesn't hold, so we define a phase X_t to be *concentrated* if X_t is synchronized.

Baseline Predictor: *Given the set of dynamics $(X_t)_{0 \leq t \leq r}$ and $T \geq r$, predict "synchronization" of X_T if X_t is "concentrated" for any $1 \leq t \leq r$. Otherwise, flip a fair coin to choose between the two labels of "synchronization" or "non-synchronization".*

[5] Refers to any arc of length $< \pi$ for the continuous phase space $\Omega = \mathbb{R}/2\pi\mathbb{Z}$ and any interval of $< \kappa/2$ consecutive integers (mod κ) for the discrete phase space $\Omega = \mathbb{Z}/\kappa\mathbb{Z}$. This confinement in an open half-circle is what we define as being "concentrated".

Table of Contents

Effect of Sub-Graph Size – Experimental Setup

- Three global networks: `UCLA`, `Caltech`, and `NWS`.

Table: Graph Statistics of the Networks used for sampling subgraphs

Networks	`UCLA`	`Caltech`	`NWS`
Number of Nodes	20467	769	20000
Number of Edges	747613	16656	16702185
Edge Density	0.0036	0.0564	0.0835
Average Clustering Coefficient	0.2149	0.4092	0.3092

Effect of Sub-Graph Size – Experimental Setup

- Three global networks: `UCLA`, `Caltech`, and `NWS`.

Table: Graph Statistics of the Networks used for sampling subgraphs

Networks	`UCLA`	`Caltech`	`NWS`
Number of Nodes	20467	769	20000
Number of Edges	747613	16656	16702185
Edge Density	0.0036	0.0564	0.0835
Average Clustering Coefficient	0.2149	0.4092	0.3092

- Generated k–paths from each networks for $10 \leq k \leq 40$, sampled 100 sub-graphs for each path, and ran the Kuramoto dynamics on each, to compute the ratio of graphs on which the Kuramoto dynamics synchronize.

Effect of Sub-Graph Size – Experimental Setup

- Three global networks: `UCLA`, `Caltech`, and `NWS`.

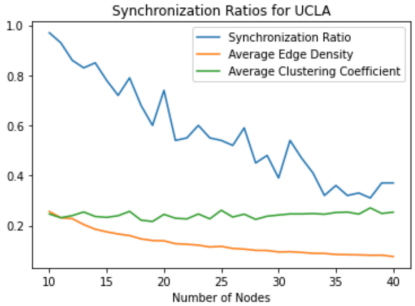
Table: Graph Statistics of the Networks used for sampling subgraphs

Networks	`UCLA`	`Caltech`	`NWS`
Number of Nodes	20467	769	20000
Number of Edges	747613	16656	16702185
Edge Density	0.0036	0.0564	0.0835
Average Clustering Coefficient	0.2149	0.4092	0.3092

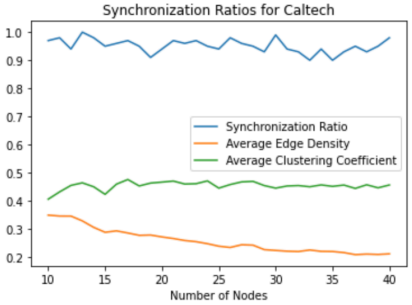
- Generated k–paths from each networks for $10 \leq k \leq 40$, sampled 100 sub-graphs for each path, and ran the Kuramoto dynamics on each, to compute the ratio of graphs on which the Kuramoto dynamics synchronize.

- Also noted the average Clustering Coefficient and Edge Density for each iteration to learn correlations.

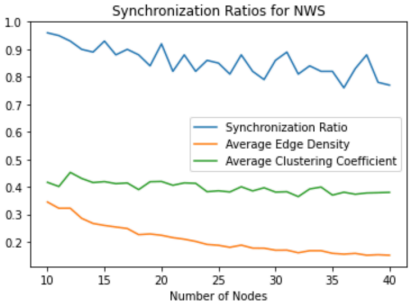
Effect of Sub-Graph Size – Results



(a) UCLA Synchronization Ratio



(b) Caltech Synchronization Ratio



(c) NWS Synchronization Ratio

Figure: Plots showing the *Average Synchronization Ratio* for the 100 subgraphs sampled from UCLA, Caltech and NWS networks with the number of nodes ranging from 10 to 40. Also superimposed on the plot is the *Average Edge Density* and the *Average Clustering Coefficient* for the 100 graphs sampled at each new iteration.

Effect of Sub-Graph Size – Observations

- Synchronization ratio decreases the quickest for `UCLA`, moderately for `NWS` and in the case of `Caltech`, remains almost the same (above $\sim 90\%$)

Effect of Sub-Graph Size – Observations

- Synchronization ratio decreases the quickest for `UCLA`, moderately for `NWS` and in the case of `Caltech`, remains almost the same (above $\sim 90\%$)
- The average clustering coefficient roughly remains the same for each network sub-graphs, at a value slightly higher than the average clustering coefficients for the original network (See table 1)

Effect of Sub-Graph Size – Observations

- Synchronization ratio decreases the quickest for `UCLA`, moderately for `NWS` and in the case of `Caltech`, remains almost the same (above $\sim 90\%$)
- The average clustering coefficient roughly remains the same for each network sub-graphs, at a value slightly higher than the average clustering coefficients for the original network (See table 1)
- The average edge density of the network sub-graphs, decay in an asymptotic fashion as the number of nodes in the sub-graphs increase

Effect of Sub-Graph Size – Observations

- Synchronization ratio decreases the quickest for `UCLA`, moderately for `NWS` and in the case of `Caltech`, remains almost the same (above $\sim 90\%$)
- The average clustering coefficient roughly remains the same for each network sub-graphs, at a value slightly higher than the average clustering coefficients for the original network (See table 1)
- The average edge density of the network sub-graphs, decay in an asymptotic fashion as the number of nodes in the sub-graphs increase
- Inference:
  - `UCLA`: Sparse – 0.35% edge density (low) and 0.25 Average Clustering Coefficient (low)
  - `NWS`: Quite dense – 8.5% edge density (high) and 0.31 Average Clustering Coefficient (moderate)
  - `Caltech`: Quite dense – 5.5% edge density (moderate to high) and 0.41 Average Clustering Coefficient (high)

Non-negative Matrix Factorization – Experimental Setup

- For each of the three networks, we sample 1600 $50-$node sub-graphs and simulate the Kuramoto dynamics on them.

Non-negative Matrix Factorization – Experimental Setup

- For each of the three networks, we sample 1600 $50-$node sub-graphs and simulate the Kuramoto dynamics on them.

- Representation matrix is a $k^2 \times N$ matrix with N columns (here 1600) each containing the "flattened-out" adjacency matrix of dimension k^2 in the column-major fashion.

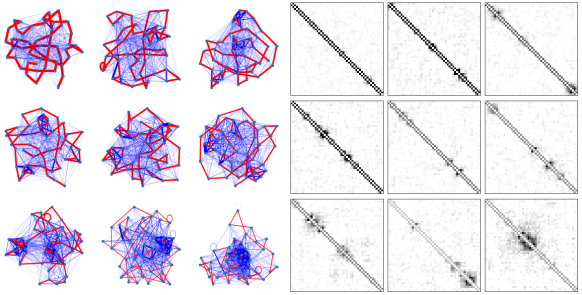
Non-negative Matrix Factorization – Experimental Setup

- For each of the three networks, we sample 1600 $50-$node sub-graphs and simulate the Kuramoto dynamics on them.

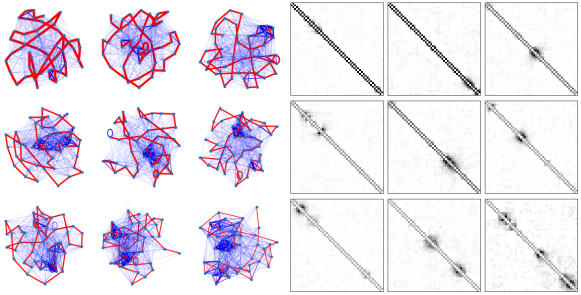
- Representation matrix is a $k^2 \times N$ matrix with N columns (here 1600) each containing the "flattened-out" adjacency matrix of dimension k^2 in the column-major fashion.

- Next, we separate out the cases of synchronization and non-synchronization into two separate matrices while ensuring a balanced split. We learn 9 dictionary atoms using non-negative matrix factorization on these matrices containing feature spaces for synchronized and non-synchronized cases separately.

Non-negative Matrix Factorization – UCLA Results



(a) UCLA Synchronization Dictionaries

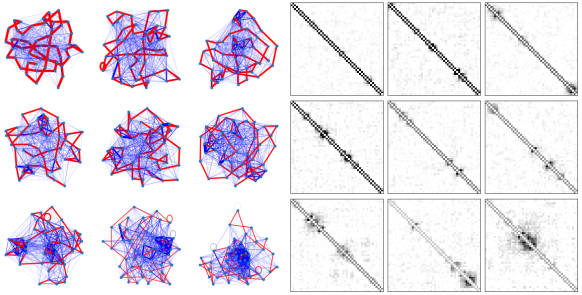


(b) UCLA Non-Synchronization Dictionaries

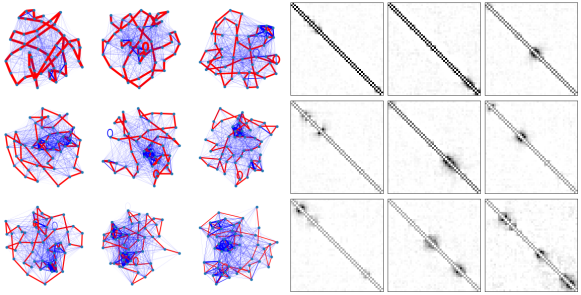
Figure: Figures showing the Dictionary atoms learned using Non-negative Matrix Factorization and their Graphical Representations for UCLA network.

- Dictionaries for both the synchronizing and non-synchronizing cases have "hubs" lying along the Hamiltonian path.

Non-negative Matrix Factorization – UCLA Results



(a) UCLA Synchronization Dictionaries

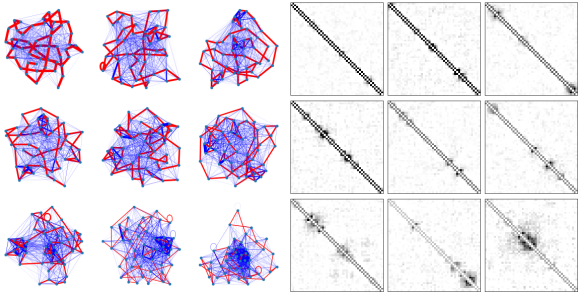


(b) UCLA Non-Synchronization Dictionaries

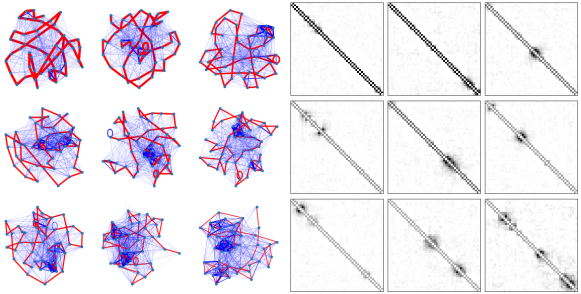
Figure: Figures showing the Dictionary atoms learned using Non-negative Matrix Factorization and their Graphical Representations for UCLA network.

- Dictionaries for both the synchronizing and non-synchronizing cases have "hubs" lying along the Hamiltonian path.
- Dictionary atoms indicate that the feature space contains sparse graphs, which is in accordance with our understanding of the UCLA network.

Non-negative Matrix Factorization – UCLA Results



(a) UCLA Synchronization Dictionaries

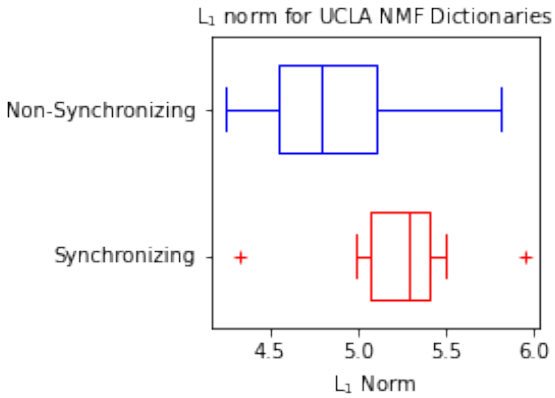


(b) UCLA Non-Synchronization Dictionaries

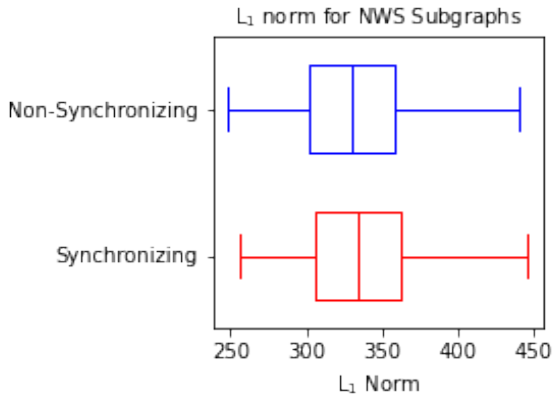
Figure: Figures showing the Dictionary atoms learned using Non-negative Matrix Factorization and their Graphical Representations for UCLA network.

- Dictionaries for both the synchronizing and non-synchronizing cases have "hubs" lying along the Hamiltonian path.
- Dictionary atoms indicate that the feature space contains sparse graphs, which is in accordance with our understanding of the UCLA network.
- Outside the Hamiltonian path, dictionaries in the synchronizing cases have a higher edge density compared to those in the non-synchronizing cases (Graphs look denser)

Non-negative Matrix Factorization – UCLA Results



(a) UCLA Dictionaries L_1 norm Box Plot

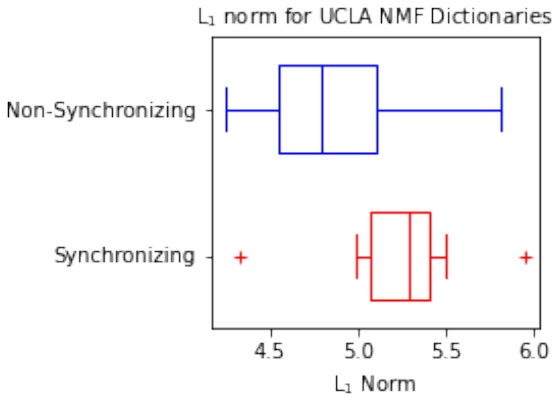


(b) UCLA Subgraphs L_1 norm Box Plot

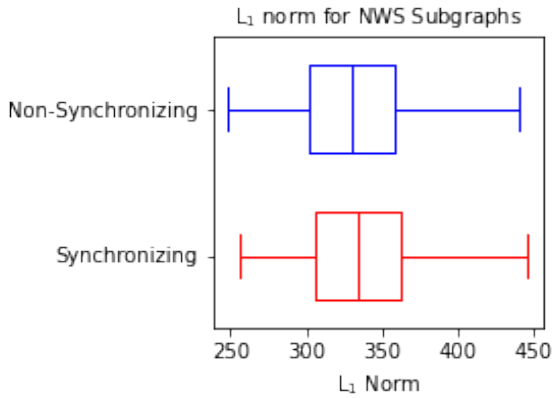
Figure: UCLA Dictionaries L_1 norm Box Plot in comparison to the L_1 norm Box Plot for the Sub-Graphs for UCLA

- The median L_1 norm for synchronizing dictionaries lies higher than not just the median but even the 75^{th} percentile norm for the non-synchronizing cases.

Non-negative Matrix Factorization – UCLA Results



(a) UCLA Dictionaries L_1 norm Box Plot

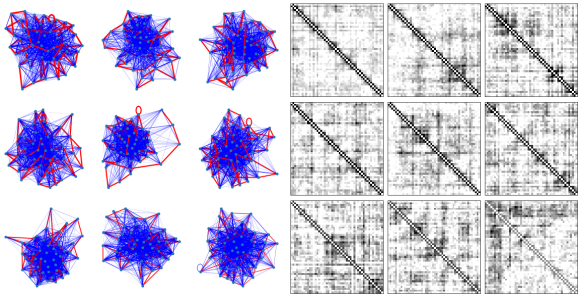


(b) UCLA Subgraphs L_1 norm Box Plot

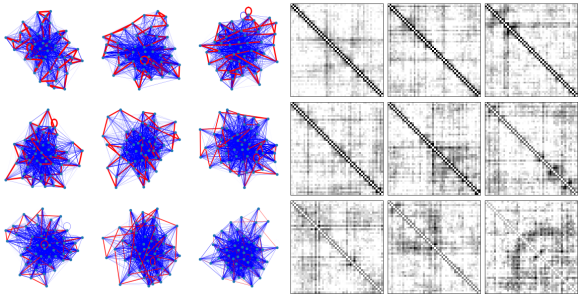
Figure: UCLA Dictionaries L_1 norm Box Plot in comparison to the L_1 norm Box Plot for the Sub-Graphs for UCLA

- The median L_1 norm for synchronizing dictionaries lies higher than not just the median but even the 75^{th} percentile norm for the non-synchronizing cases.
- On constructing a similar box plot for the original sub-graphs, see that synchronizing sub-graphs indeed have a higher L_1 norm, though will have to conduct a t–test to confirm.

Non-negative Matrix Factorization – Caltech Results



(a) Caltech Synchronization Dictionaries

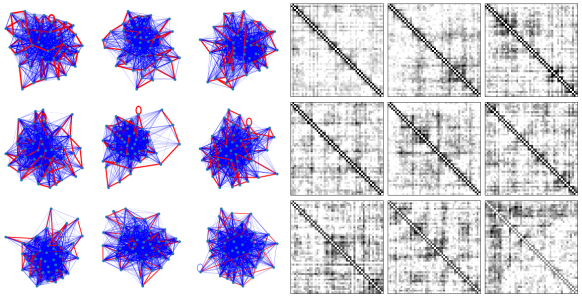


(b) Caltech Non-Synchronization Dictionaries

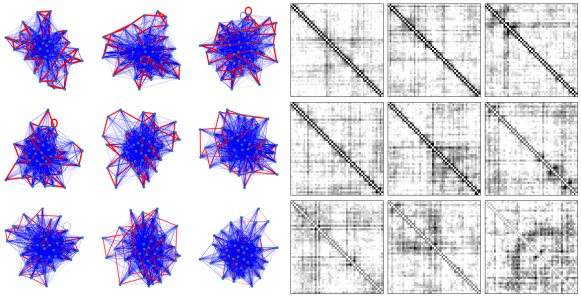
Figure: Figures showing the Dictionary atoms learned using Non-negative Matrix Factorization and their Graphical Representations for Caltech network.

- No significant hubs like seen in the case of UCLA. However, both the synchronizing and non-synchronizing cases seem to have one or more "pivot nodes" lying along the Hamiltonian path.

Non-negative Matrix Factorization – Caltech Results



(a) Caltech Synchronization Dictionaries

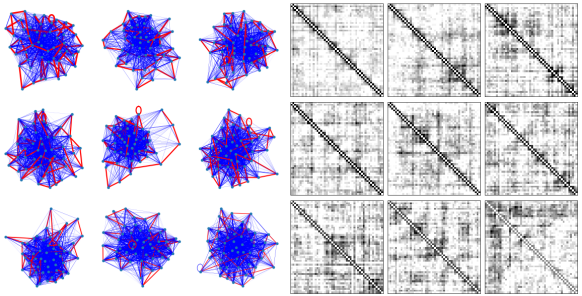


(b) Caltech Non-Synchronization Dictionaries

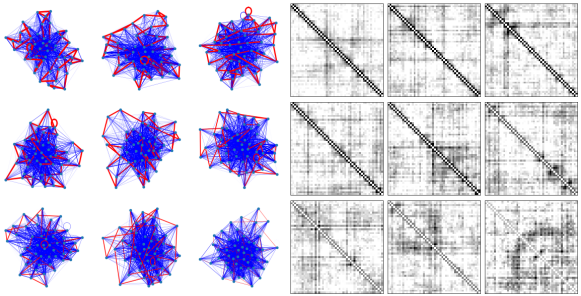
Figure: Figures showing the Dictionary atoms learned using Non-negative Matrix Factorization and their Graphical Representations for Caltech network.

- No significant hubs like seen in the case of UCLA. However, both the synchronizing and non-synchronizing cases seem to have one or more "pivot nodes" lying along the Hamiltonian path.
- Dictionary atoms indicate that the feature space contains dense graphs, which is in accordance with our understanding of the Caltech network.

Non-negative Matrix Factorization – Caltech Results



(a) Caltech Synchronization Dictionaries

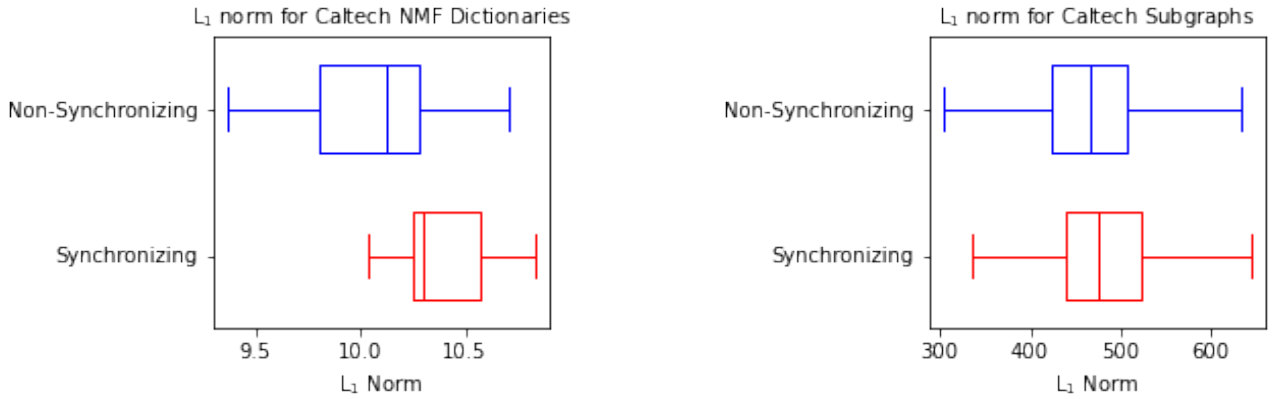


(b) Caltech Non-Synchronization Dictionaries

Figure: Figures showing the Dictionary atoms learned using Non-negative Matrix Factorization and their Graphical Representations for Caltech network.

- No significant hubs like seen in the case of UCLA. However, both the synchronizing and non-synchronizing cases seem to have one or more "pivot nodes" lying along the Hamiltonian path.
- Dictionary atoms indicate that the feature space contains dense graphs, which is in accordance with our understanding of the Caltech network.
- Harder to notice significant difference in dictionary atoms. Both cases produce dense dictionaries, however dictionaries for synchronizing arguably contain larger number of pivot nodes.

Non-negative Matrix Factorization – Caltech Results



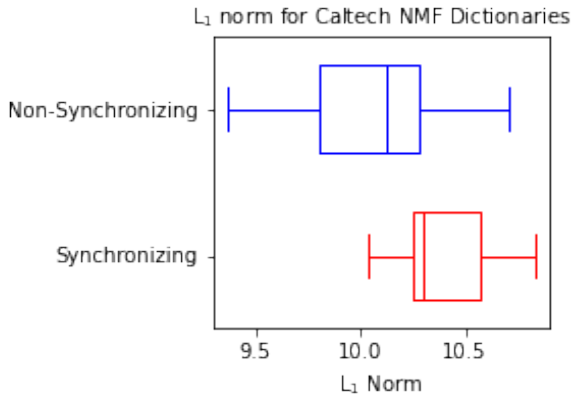
(a) Caltech Dictionaries L_1 norm Box Plot

(b) Caltech Subgraphs L_1 norm Box Plot

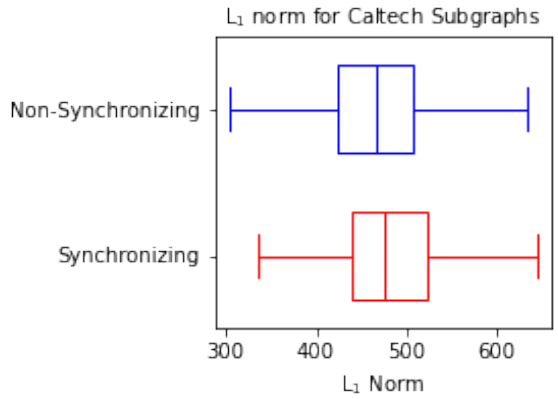
Figure: Caltech Dictionaries L_1 norm Box Plot in comparison to the Box Plot for the Sub-Graphs for Caltech

- The median L_1 norm for synchronizing dictionaries lies higher than not just the median but even the 75^{th} percentile norm for the non-synchronizing cases, similar to UCLA

Non-negative Matrix Factorization – Caltech Results



(a) Caltech Dictionaries L_1 norm Box Plot

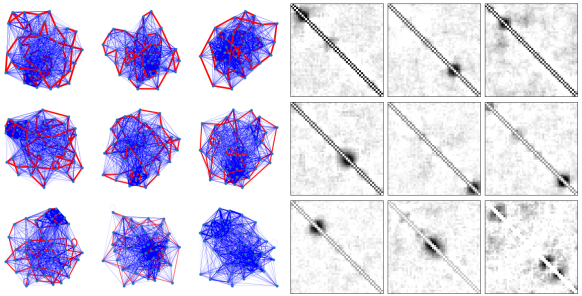


(b) Caltech Subgraphs L_1 norm Box Plot

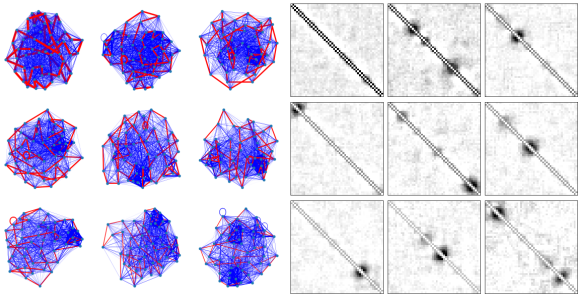
Figure: Caltech Dictionaries L_1 norm Box Plot in comparison to the Box Plot for the Sub-Graphs for Caltech

- The median L_1 norm for synchronizing dictionaries lies higher than not just the median but even the 75^{th} percentile norm for the non-synchronizing cases, similar to UCLA
- On constructing a similar box plot for the original sub-graphs, see that synchronizing sub-graphs indeed have a higher L_1 norm, though will have to conduct a t–test to confirm.

Non-negative Matrix Factorization – NWS Results



(a) NWS Synchronization Dictionaries

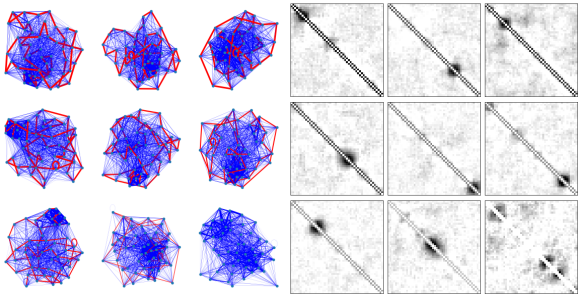


(b) NWS Non-Synchronization Dictionaries

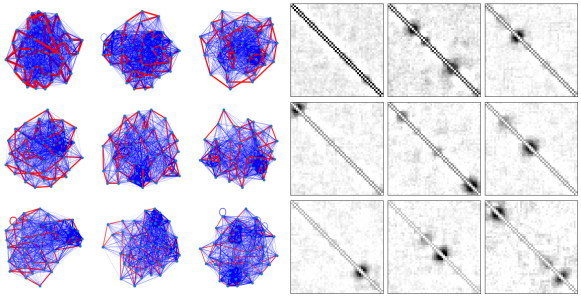
Figure: Figures showing the Dictionary atoms learned using Non-negative Matrix Factorization and their Graphical Representations for NWS network.

- Dictionaries for both the synchronizing and non-synchronizing cases have "hubs" lying along the Hamiltonian path, like seen in the case of UCLA.

Non-negative Matrix Factorization – NWS Results



(a) NWS Synchronization Dictionaries

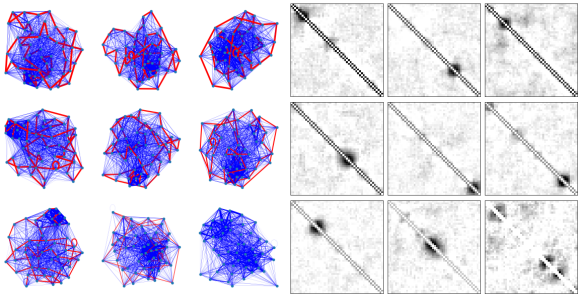


(b) NWS Non-Synchronization Dictionaries

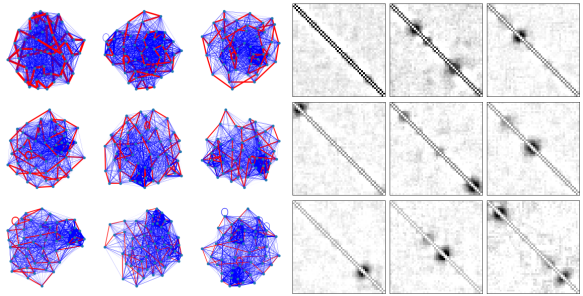
Figure: Figures showing the Dictionary atoms learned using Non-negative Matrix Factorization and their Graphical Representations for NWS network.

- Dictionaries for both the synchronizing and non-synchronizing cases have "hubs" lying along the Hamiltonian path, like seen in the case of UCLA.
- Dictionary atoms indicate that the feature space contains denser graphs than UCLA, which is in accordance with our understanding of the NWS network.

Non-negative Matrix Factorization – NWS Results



(a) NWS Synchronization Dictionaries

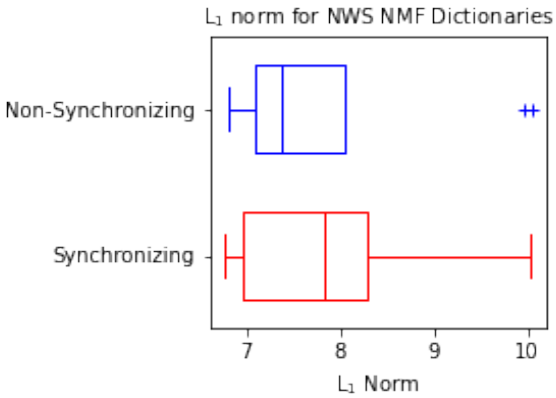


(b) NWS Non-Synchronization Dictionaries

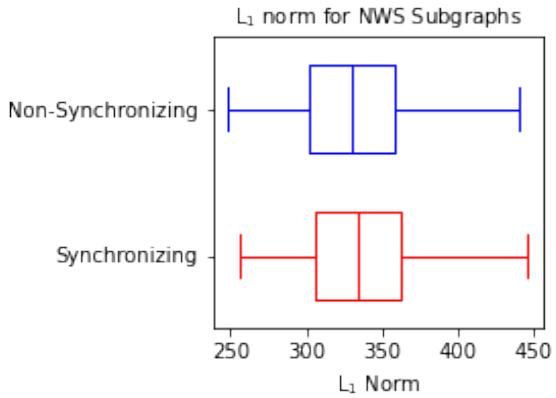
Figure: Figures showing the Dictionary atoms learned using Non-negative Matrix Factorization and their Graphical Representations for NWS network.

- Dictionaries for both the synchronizing and non-synchronizing cases have "hubs" lying along the Hamiltonian path, like seen in the case of UCLA.
- Dictionary atoms indicate that the feature space contains denser graphs than UCLA, which is in accordance with our understanding of the NWS network.
- Harder to notice significant difference in dictionary atoms. Both cases produce dense dictionaries outside of the Hamiltonian path, however usually one single major dense hub for synchronizing cases rather than multiple ones.

Non-negative Matrix Factorization – NWS Results



(a) NWS Dictionaries L_1 norm Box Plot

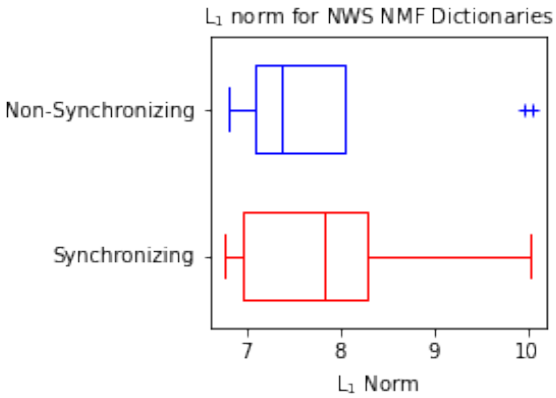


(b) NWS Subgraphs L_1 norm Box Plot

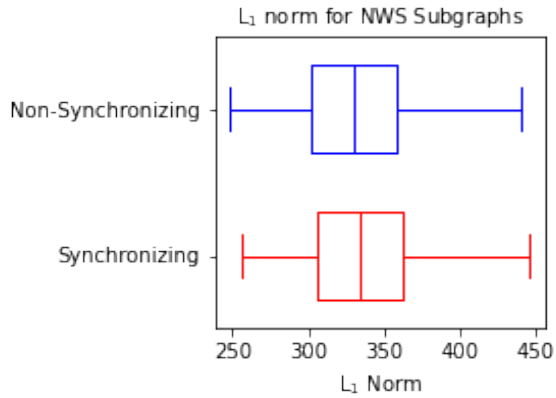
Figure: NWS Dictionaries L_1 norm Box Plot in comparison to the Box Plot for the Sub-Graphs for NWS

- The median L_1 norm for synchronizing dictionaries lies close to the 75^{th} percentile norm for the non-synchronizing cases, and also the 75^{th} percentile is greater than the maximum

Non-negative Matrix Factorization – NWS Results



(a) NWS Dictionaries L_1 norm Box Plot



(b) NWS Subgraphs L_1 norm Box Plot

Figure: NWS Dictionaries L_1 norm Box Plot in comparison to the Box Plot for the Sub-Graphs for NWS

- The median L_1 norm for synchronizing dictionaries lies close to the 75^{th} percentile norm for the non-synchronizing cases, and also the 75^{th} percentile is greater than the maximum
- On constructing a similar box plot for the original sub-graphs, see that synchronizing sub-graphs indeed have a higher L_1 norm, though will have to conduct a t–test to confirm.

Encoding Dynamics – Need for an Alternative Approach

- Kuramoto dynamics usually take iterations in the order of 100s to show synchronizing behavior compared to Firefly Cellular Automata (FCA) and Greenberg-Hastings-Model (GHM)

Encoding Dynamics – Need for an Alternative Approach

- Kuramoto dynamics usually take iterations in the order of 100s to show synchronizing behavior compared to Firefly Cellular Automata (FCA) and Greenberg-Hastings-Model (GHM)
- For this reason, applying NMF to a feature representation encoding the dynamics and adjacency matrix together is a task that requires a different approach

Encoding Dynamics – Need for an Alternative Approach

- Kuramoto dynamics usually take iterations in the order of 100s to show synchronizing behavior compared to Firefly Cellular Automata (FCA) and Greenberg-Hastings-Model (GHM)
- For this reason, applying NMF to a feature representation encoding the dynamics and adjacency matrix together is a task that requires a different approach
- Consider using `r` iterations of dynamics as the input to Supervised Dictionary Learning (SDL) algorithm, in the similar manner that we undertook for NMF, for a sub-graph with k–nodes

Encoding Dynamics – Need for an Alternative Approach

- Kuramoto dynamics usually take iterations in the order of 100s to show synchronizing behavior compared to Firefly Cellular Automata (FCA) and Greenberg-Hastings-Model (GHM)

- For this reason, applying NMF to a feature representation encoding the dynamics and adjacency matrix together is a task that requires a different approach

- Consider using `r` iterations of dynamics as the input to Supervised Dictionary Learning (SDL) algorithm, in the similar manner that we undertook for NMF, for a sub-graph with k–nodes

$$\underbrace{(k^2 \times r)}_{\text{number of rows}} \quad \times \quad \underbrace{N}_{\text{number of columns}}$$

quickly reaches the order of 10^7 or 10^8 for the vectorized matrix during the process of NMF

Encoding Dynamics – Need for an Alternative Approach

- Kuramoto dynamics usually take iterations in the order of 100s to show synchronizing behavior compared to Firefly Cellular Automata (FCA) and Greenberg-Hastings-Model (GHM)

- For this reason, applying NMF to a feature representation encoding the dynamics and adjacency matrix together is a task that requires a different approach

- Consider using `r` iterations of dynamics as the input to Supervised Dictionary Learning (SDL) algorithm, in the similar manner that we undertook for NMF, for a sub-graph with k–nodes

$$\underbrace{(k^2 \times r)}_{\text{number of rows}} \quad \times \quad \underbrace{N}_{\text{number of columns}}$$

quickly reaches the order of 10^7 or 10^8 for the vectorized matrix during the process of NMF

- Moreover, on applying SDL to such a setting leads to unclear and uninterpretable results, which calls for some alternative approaches

Color-coded Adjacency Matrix at time t

• Let $G = (V, E)$ be a graph, and let X_t be the configuration of the Kuramoto dynamics at time-t. Let $v_i, v_j \in V$ be two vertices in G. We define the color-coded matrix as follows

$$(i,j)^{\text{th}},\ (j,i)^{\text{th}} entry = \begin{cases} 0 & (v_i, v_j) \notin E \\ \min\ \{|X_t(v_j) - X_t(v_i)|, |2\pi + X_t(v_i) - X_t(v_j)|\} & (v_i, v_j) \in E \end{cases}$$

Color-coded Adjacency Matrix at time t

• Let $G = (V, E)$ be a graph, and let X_t be the configuration of the Kuramoto dynamics at time-t. Let $v_i, v_j \in V$ be two vertices in G. We define the color-coded matrix as follows

$$(i,j)^{\text{th}},\ (j,i)^{\text{th}} entry = \begin{cases} 0 & (v_i, v_j) \notin E \\ \min\ \{|X_t(v_j) - X_t(v_i)|, |2\pi + X_t(v_i) - X_t(v_j)|\} & (v_i, v_j) \in E \end{cases}$$

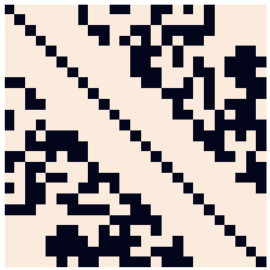
• Consider the following two graphs - (1) 25-Node NWS Graph (Synchronizing) and (2) 50-Node Cycle Graph (Non-Synchronizing). Here's their adjacency matrix

Color-coded Adjacency Matrix at time t

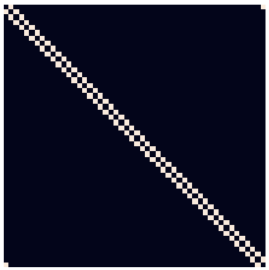
• Let $G = (V, E)$ be a graph, and let X_t be the configuration of the Kuramoto dynamics at time-t. Let $v_i, v_j \in V$ be two vertices in G. We define the color-coded matrix as follows

$$(i,j)^{\text{th}},\ (j,i)^{\text{th}} entry = \begin{cases} 0 & (v_i, v_j) \notin E \\ \min\ \{|X_t(v_j) - X_t(v_i)|, |2\pi + X_t(v_i) - X_t(v_j)|\} & (v_i, v_j) \in E \end{cases}$$

• Consider the following two graphs - (1) 25-Node NWS Graph (Synchronizing) and (2) 50-Node Cycle Graph (Non-Synchronizing). Here's their adjacency matrix



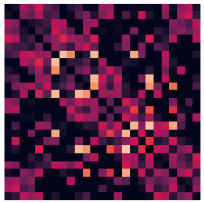
(a) 25-Node `NWS` Adjacency Matrix



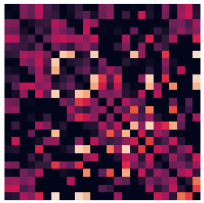
(b) `Cyclic Graph` Adjacency Matrix

Figure: Adjacency mtrices of the two graphs to illustrate the concept of Color-Coded Adjacency Matrix

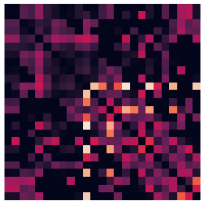
NWS Color-Coded Adjacency Matrix – Synchronizing



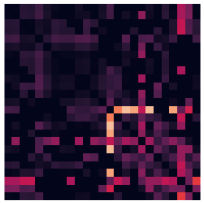
(a) $\mathbf{t = 0}$



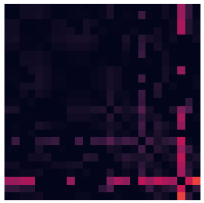
(b) $\mathbf{t = 50}$



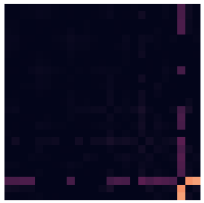
(c) $\mathbf{t = 100}$



(d) $\mathbf{t = 150}$



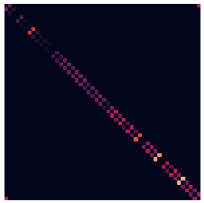
(e) $\mathbf{t = 200}$



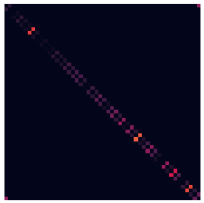
(f) $\mathbf{t = 250}$

Figure: Kuramoto dynamics on NWS at different iterations mentioned below the figure

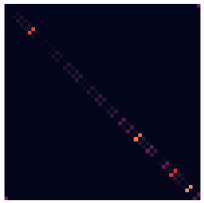
Cycle Graph Color-Coded Adjacency Matrix – Non-Synchronizing



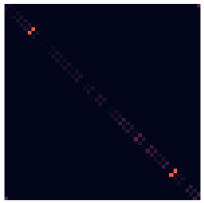
(a) $\mathbf{t = 0}$



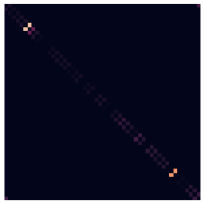
(b) $\mathbf{t = 50}$



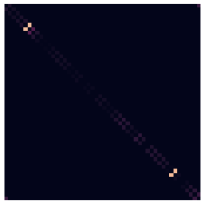
(c) $\mathbf{t = 100}$



(d) $\mathbf{t = 150}$



(e) $\mathbf{t = 200}$



(f) $\mathbf{t = 250}$

Figure: Kuramoto dynamics on Cycle Graph at different iterations mentioned below the figure

Action Plan - I

- Use (online) Tensor-Factorization algorithms on the color-coded adjacency matrix as a whole, and apply **CP-decomposition** to it to learn latent structures in this 3D-tensor

Action Plan - I

- Use (online) Tensor-Factorization algorithms on the color-coded adjacency matrix as a whole, and apply **CP-decomposition** to it to learn latent structures in this 3D-tensor

- **CP-Decomposition** – factorizes a tensor into a linear combination of rank one tensors, rather than vectorizing it into a single vector.

Action Plan - I

• Use (online) Tensor-Factorization algorithms on the color-coded adjacency matrix as a whole, and apply **CP-decomposition** to it to learn latent structures in this 3D-tensor

• **CP-Decomposition** – factorizes a tensor into a linear combination of rank one tensors, rather than vectorizing it into a single vector.

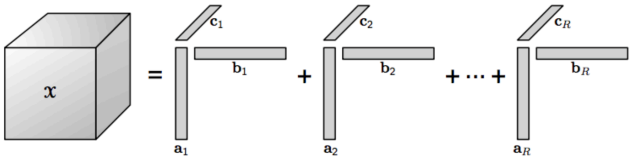


Figure: Illustrating CP-decomposition for a 3D tensor

• Plan to use this on our color-coded adjacency matrix time series, since it can also be represented as a 3D-tensor to learn latent features in combination with classification

Action Plan - I

• Use (online) Tensor-Factorization algorithms on the color-coded adjacency matrix as a whole, and apply **CP-decomposition** to it to learn latent structures in this 3D-tensor

• **CP-Decomposition** – factorizes a tensor into a linear combination of rank one tensors, rather than vectorizing it into a single vector.

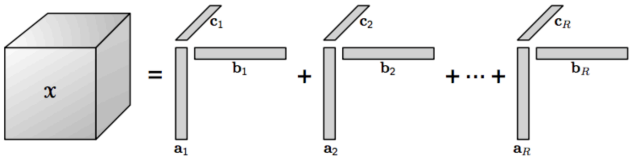


Figure: Illustrating CP-decomposition for a 3D tensor

• Plan to use this on our color-coded adjacency matrix time series, since it can also be represented as a 3D-tensor to learn latent features in combination with classification

Action Plan - II

- Use Convolutional Neural Network (CNN) adaptations like the Long-term Recurrent Convolutional Network (LRCN)

Action Plan - II

- Use Convolutional Neural Network (CNN) adaptations like the Long-term Recurrent Convolutional Network (LRCN)

- Unlike in Tensor-factorization algorithms, where we would use the matrix format, here we use the color-coded adjacency matrices as images to the neural network.

Action Plan - II

- Use Convolutional Neural Network (CNN) adaptations like the Long-term Recurrent Convolutional Network (LRCN)

- Unlike in Tensor-factorization algorithms, where we would use the matrix format, here we use the color-coded adjacency matrices as images to the neural network.

- The Long Short-Term Memory (LSTM) could prove especially useful keeping in mind the time series nature of these color-coded adjacency matrices.

Action Plan - II

- Use Convolutional Neural Network (CNN) adaptations like the Long-term Recurrent Convolutional Network (LRCN)
- Unlike in Tensor-factorization algorithms, where we would use the matrix format, here we use the color-coded adjacency matrices as images to the neural network.
- The Long Short-Term Memory (LSTM) could prove especially useful keeping in mind the time series nature of these color-coded adjacency matrices.
- Used it on weighted adjacency-matrices in `L2PSync`, might prove to be useful here

Table of Contents

FCA Dynamics Datasets

Datasets	NWS	UCLA26
# nodes	20	20
Avg of # edges	41.02	25.93
Std of # edges	9.53	4.75
Avg diameter	5.28	11.18
Std of diameter	1.22	3.47
r (training iter)	50	50
T (prediction iter)	200	200
# Sync.	7548	7553
# Nonsync.	2452	2447

Table: Dynamics datasets generated for FCA($\kappa = 8$) on 20-node connected sub-graphs of a large NWS graph and UCLA26. The large NWS graph is generated with 20000 nodes, nearest neighbors of 1000, and shortcut edge probability of 0.7. Each dataset contains 10000 underlying graph structures.

Black-Box Model: Random Forest

1. Dynamics
2. Width of dynamics
3. Shifted dynamics

Shifted Dynamics

Let $G = (V, E)$ be a graph, and let $X : V \to \mathbb{Z}_\kappa$ be a κ-configuration.

- $X_{shifted} = X + \text{argmin}_{0 \leq a < \kappa}(\max(X + a) \mod \kappa)$

Shifted Dynamics

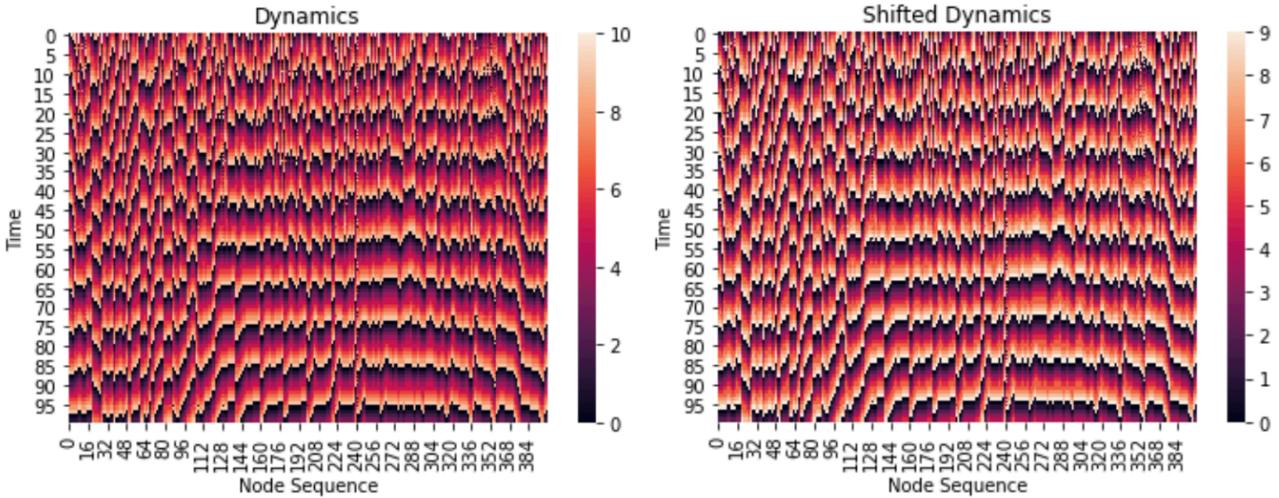


Figure: Left: Heatmap of dynamics from time 0 to time 100 on 20×20 2d grid networks. Right: Heatmap of shifted dynamics from time 0 to time 100 on 20×20 2d grid networks

.

Black-Box Model: Random Forest

1. Dynamics
2. Width of dynamics
3. Shifted dynamics
4. Dynamics + Adjacency matrix
5. Dynamics + node2vec
6. Dynamics + graph2vec
7. Dynamics + spectral embedding
8. Dynamics + Graph features

Graph Features

	Features
Graph-level	# edges, # nodes, min degree, max degree, diameter, degree assortativity coefficient, # cliques, average clustering coefficient, density
Node-level	Degree centrality, eigenvector centrality, betweenness centrality, closeness centrality, clustering coefficient, degree

Table: Graph features calculated for sub-graphs of a large `NWS` graph and `UCLA26`. The graph-level feature is applied to each underlying graph. The node-level feature is applied to each node of each underlying graph.

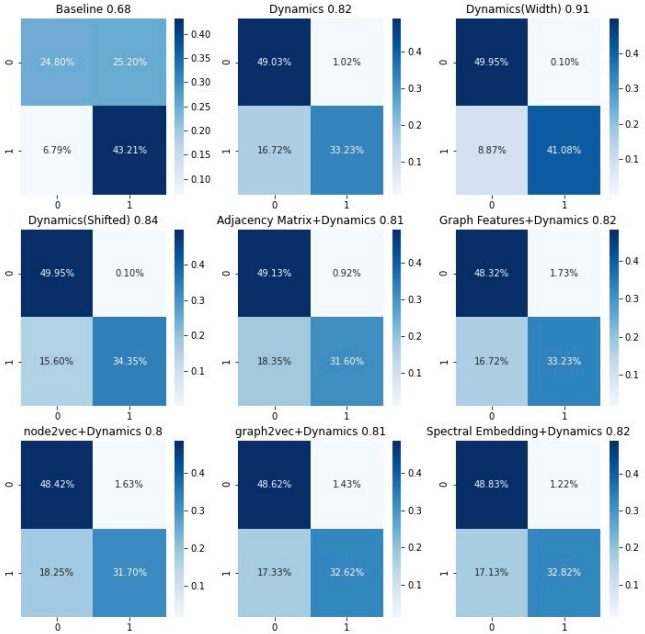


Figure: RF performance on different feature inputs for sub-graphs of a large NWS network. The number in the title is the accuracy score. For each confusion matrix, the x-axis is the predicted value and the y-axis is the actual value. If it is labeled as 0, then it is non-synchronizing. If it is labeled as 1, then it is synchronizing. The baseline model is defined in Section **??**

.

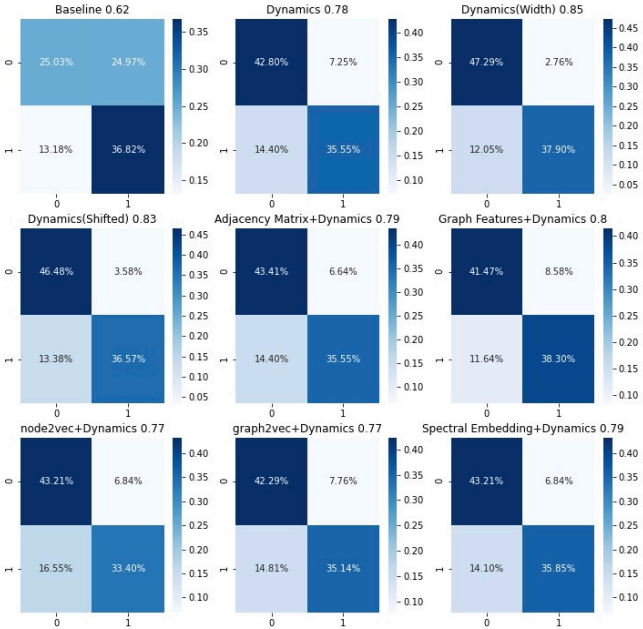


Figure: RF performance on different feature inputs for sub-graphs of UCLA26. The number in the title is the accuracy score. For each confusion matrix, the x-axis is the predicted value and the y-axis is the actual value. If it is labeled as 0, then it is non-synchronizing. If it is labeled as 1, then it is synchronizing. The baseline model is defined in **??**

NMF to compare synchronizing and nonsynchronizing pairs

- Dynamics + Adjacency matrix
- apply separately on synchronizing and nonsynchronizing pairs

NMF to compare synchronizing and nonsynchronizing pairs

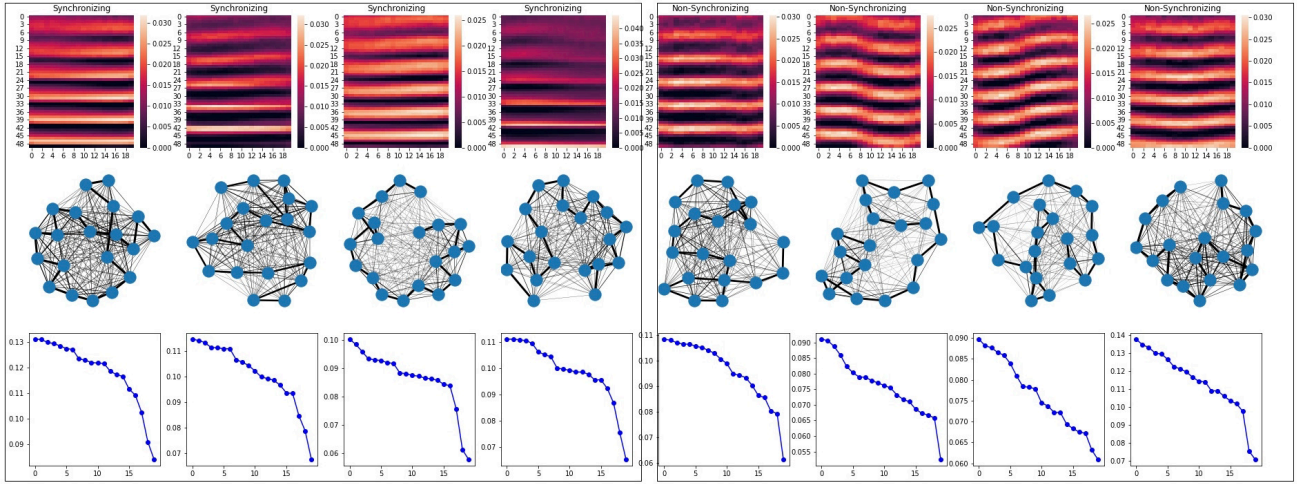


Figure: The dictionary elements learned from NMF for synchronizing pairs and non-synchronizing pairs of sub-graphs from a large NWS network. The left four dictionary elements are from the synchronizing pairs, and the right four dictionary elements are from the non-synchronizing pairs. Each column is a dictionary element. The heatmaps are the dictionary elements learned from dynamics, where x-axis is the ordering of node and the y-axis is time. The networks are the dictionary elements learned from adjacency matrices, where the weights of edges are set to be the values in the dictionary elements. The degree distribution corresponds to the network above it.

NMF to compare synchronizing and nonsynchronizing pairs

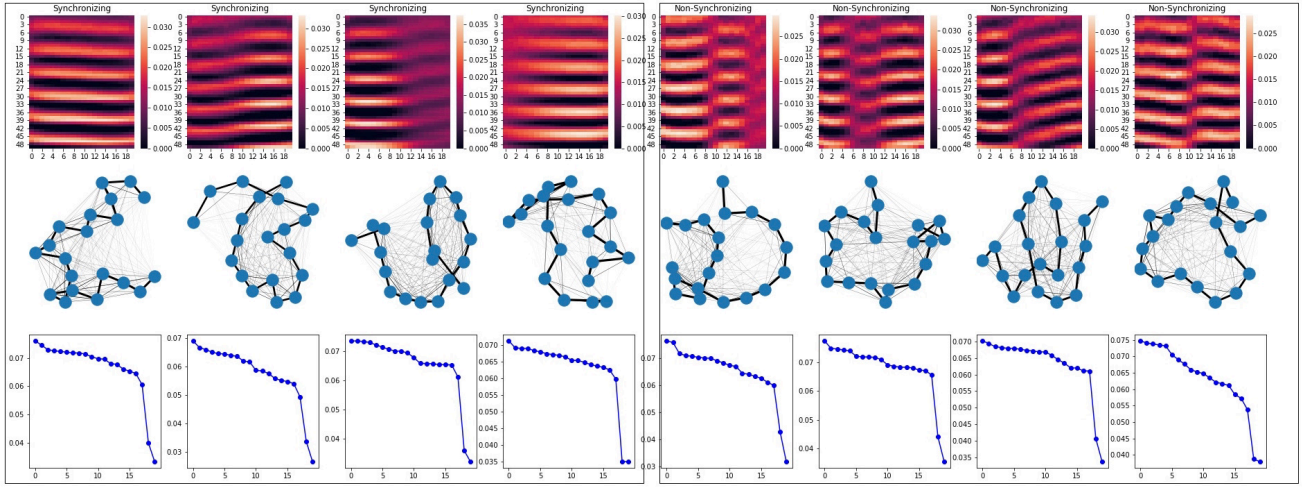


Figure: The dictionary elements learned from NMF for synchronizing pairs and non-synchronizing pairs of sub-graphs from UCLA26. The left four dictionary elements are from the synchronizing pairs, and the right four dictionary elements are from the non-synchronizing pairs. Each column is a dictionary element. The heatmaps are the dictionary elements learned from dynamics, where x-axis is the ordering of node and the y-axis is time. The networks are the dictionary elements learned from adjacency matrices, where the weights of edges are set to be the values in the dictionary elements. The degree distribution corresponds to the network above it.

SDL on predicting synchronization

1. Dynamics
2. Shifted dynamics
3. Dynamics + Adjacency matrix
4. Colored adjacency matrix

Colored Adjacency Matrix at time t

Let $G = (V, E)$ be a graph, where $v_i, v_j \in V$, and let $X_t : V \to \mathbb{Z}_\kappa$ be a κ-configuration at time t.

$$\text{(i j -th entry)}\Psi_t(i,j) = \begin{cases} 0 & \text{if } (v_i, v_j) \notin E \\ \min(|X_t(v_i) - X_t(v_j)|, |X_t(v_i) - X_t(v_j) + \kappa|) & \text{if } (v_i, v_j) \in E \end{cases} \quad (4)$$

Colored Adjacency Matrix at time t

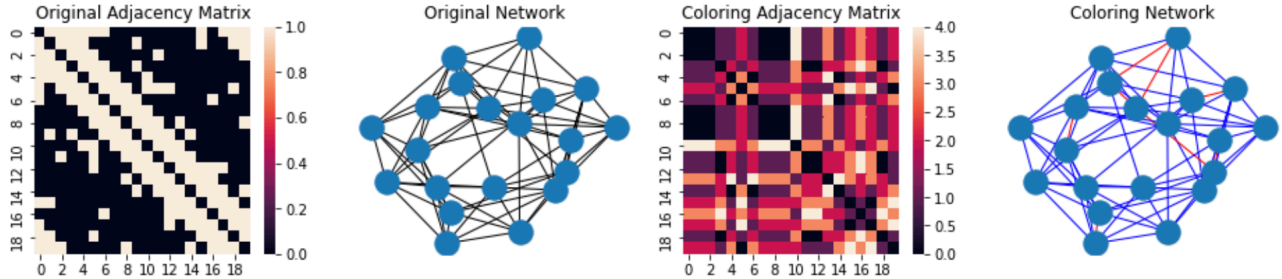


Figure: Left: Original adjacency matrix and network plot on a randomly generated 20-node NWS network. Right: Colored adjacency matrix and colored network plot on the same NWS network. Red edges indicate there is no color difference between two vertices, while blue edges indicate there is color different between two vertices.

.

SDL on predicting synchronization

Feature	Baseline	RF	SDL
Dynamics	0.68	0.82	0.68
Shifted dynamics	0.68	0.84	0.68
Dynamics + adjacency matrix	0.68	0.81	0.61
Colored adjacency matrix	0.68	0.83	0.80

Table: The classification accuracy of the baseline model, RF model, and SDL model based upon different feature inputs from the dataset of NWS

SDL on predicting synchronization

Feature	Baseline	RF	SDL
Dynamics	0.61	0.78	0.63
Shifted dynamics	0.61	0.83	0.68
Dynamics + adjacency matrix	0.61	0.79	0.50
Colored adjacency matrix	0.61	0.82	0.74

Table: The classification accuracy of the baseline model, RF model, and SDL model based upon different feature inputs from the dataset of UCLA26

SDL on predicting synchronization

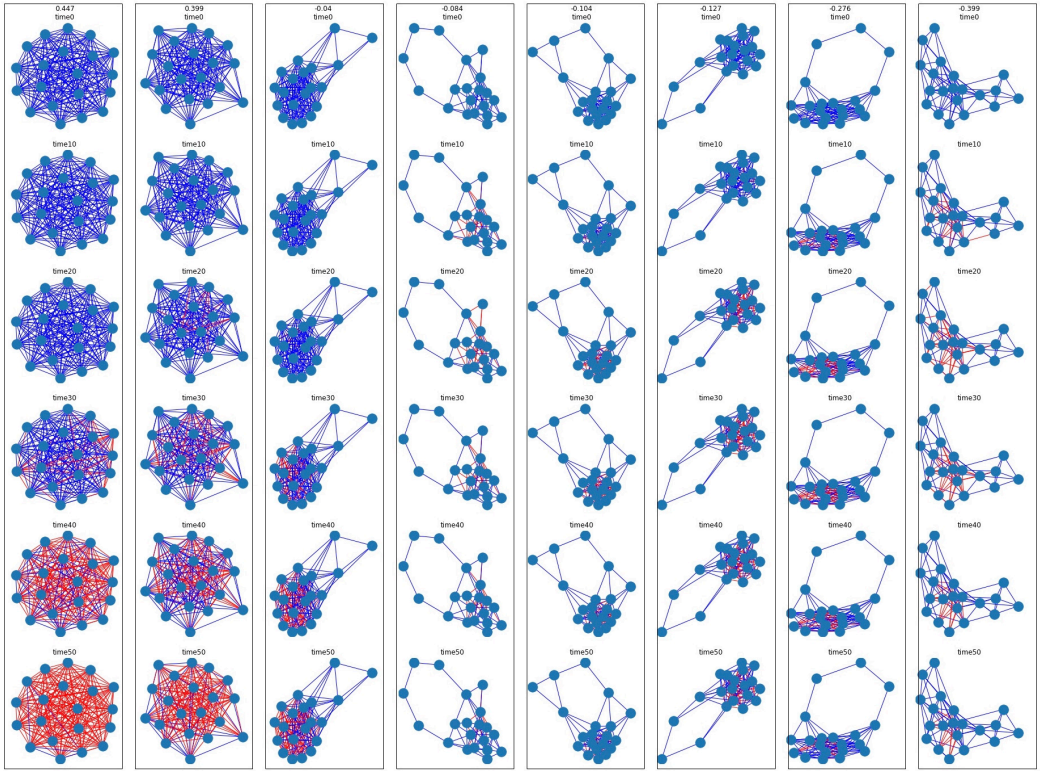


Figure: The dictionary elements learned from SDL on the colored adjacency matrix from dataset of NWS.

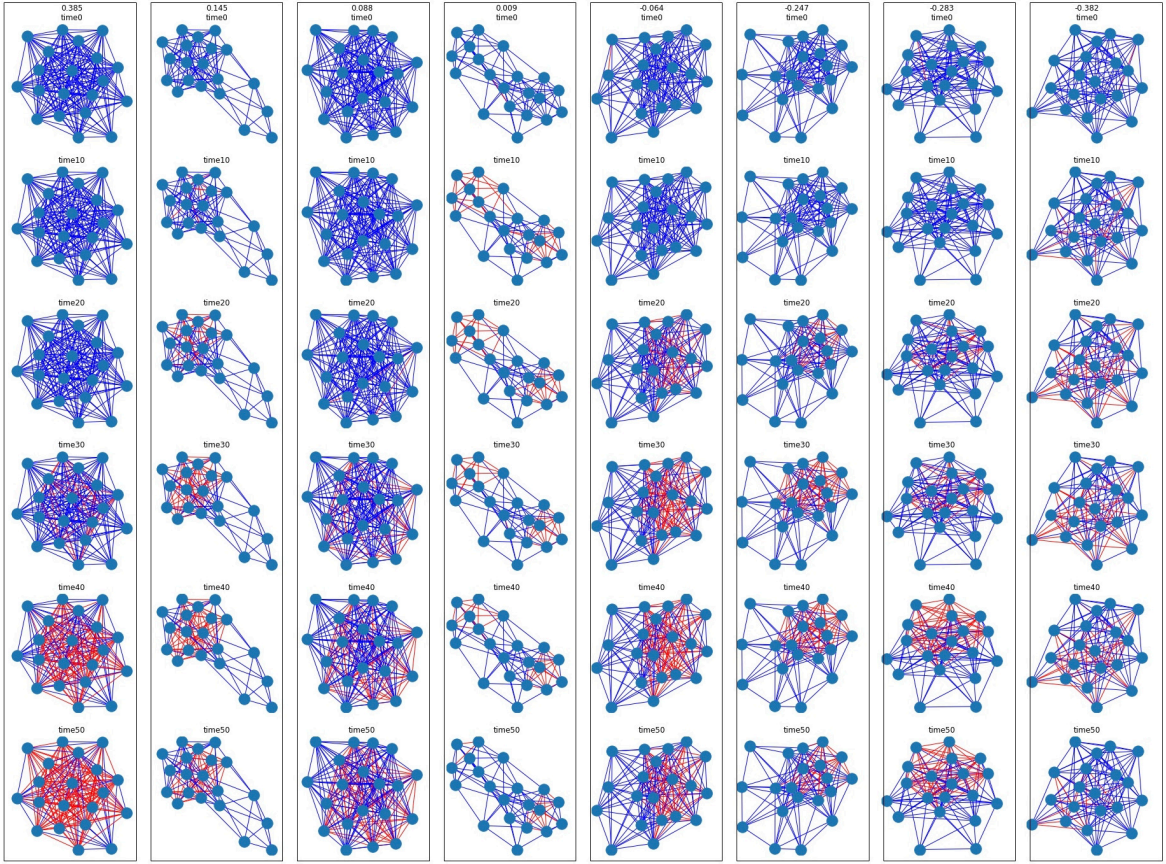


Figure: The dictionary elements learned from SDL on the colored adjacency matrix from dataset of UCLA26.

Table of Contents

Experimental Setup

1. Goal: Explore relationship between number of nodes and synchronization behavior for GHM dynamics on subgraphs of three types of orginal graphs. The three full networks has been introduced in Kuramoto previously.

Experimental Setup

1. Goal: Explore relationship between number of nodes and synchronization behavior for GHM dynamics on subgraphs of three types of orginal graphs. The three full networks has been introduced in Kuramoto previously.
2. Step1: Generate 100 k-node subgraphs for $5 \leq k \leq 40$ for all three full networks.

Experimental Setup

1. Goal: Explore relationship between number of nodes and synchronization behavior for GHM dynamics on subgraphs of three types of orginal graphs. The three full networks has been introduced in Kuramoto previously.
2. Step1: Generate 100 k-node subgraphs for $5 \leq k \leq 40$ for all three full networks.
3. Step2: Check synchronization and record average clustering coefficient and transitivity of 100 subgraphs for each node number. The density of each graph is also plotted

Results and observation

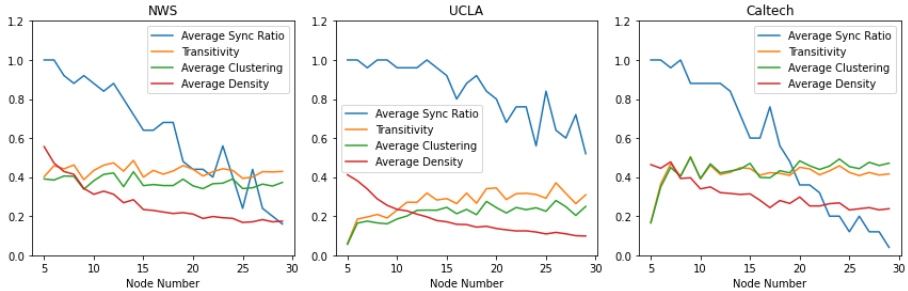


Figure: Synchronization ratio, average clustering coefficient, average transitivity, and average density trend with different node number in NWS, UCLA, and Caltech networks

Background

1. GHM in general synchronizes very fast or falls into infinite periodic wave-like behavior.

Background

1. GHM in general synchronizes very fast or falls into infinite periodic wave-like behavior.
2. GHM seldom synchronizes on 2-D grid graphs.

GHM dynamics visualization on 2-D grid

1. Simulates a time evolution of 5-state GHM dynamics on 2-D 70-by-70 fully connected grid

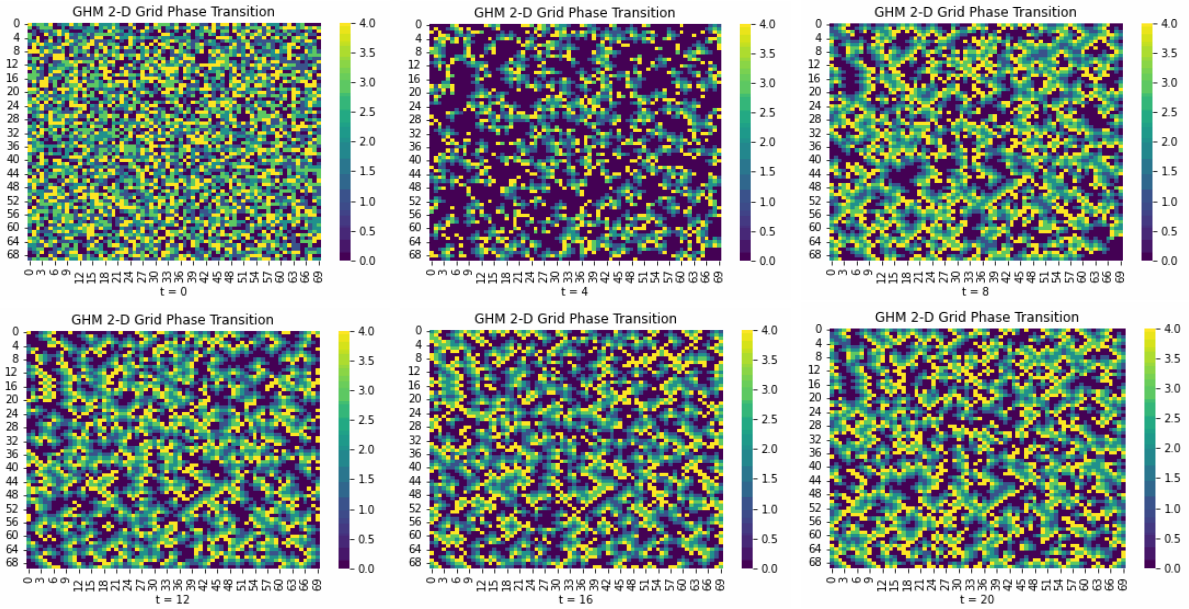


Figure: Time evolution of GHM dynamics on 70-by-70 grid

GHM dynamics visualization on 2-D grid

1. Simulates a time evolution of 5-state GHM dynamics on 2-D 70-by-70 fully connected grid
2. Notice the periodic behavior for $t = 8$ and $t = 20$

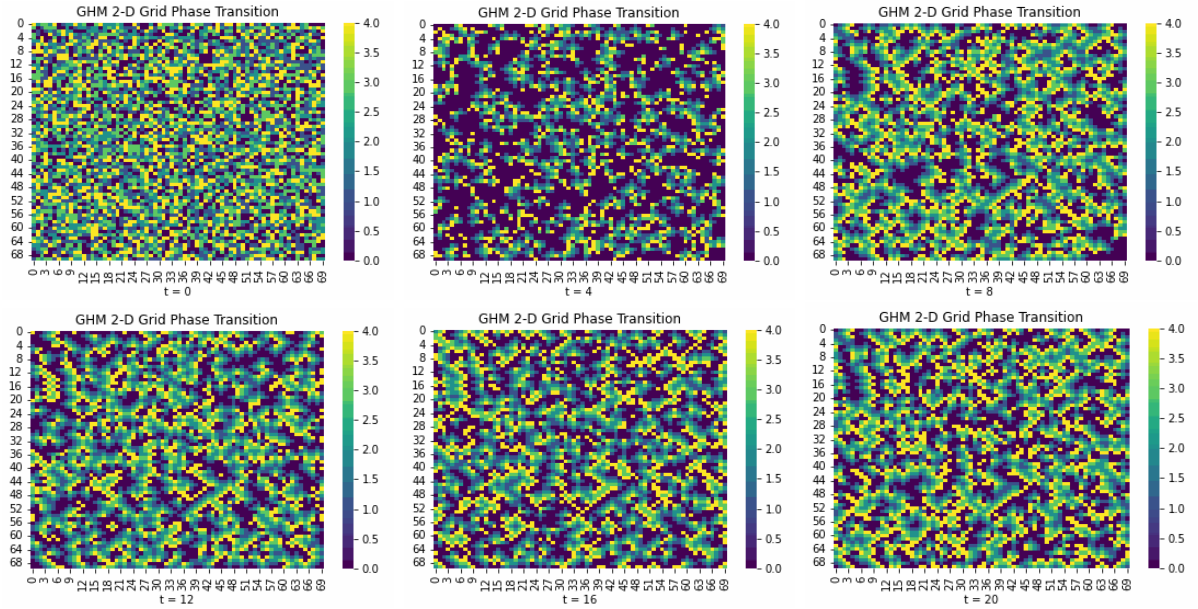


Figure: Time evolution of GHM dynamics on 70-by-70 grid

Stochasticity for GHM transition on 2-D

1. Add stochasticity to the original updating rule of GHM, but which line?

Stochasticity for GHM transition on 2-D

1. Add stochasticity to the original updating rule of GHM, but which line?
2. Is the quiescent or excited state more important for synchronization?

Stochasticity for GHM transition on 2-D

1. Add stochasticity to the original updating rule of GHM, but which line?
2. Is the quiescent or excited state more important for synchronization?
3. Randomly generate $P_{tv} \in [0, 1]$ at t-th iteration for node v. Tuning the threshold probability H to see different synchronizing behaviors.

Stochasticity for GHM transition on 2-D

1.

$$X_{t+1}(v) = \begin{cases} 0 & \text{if } X_t(v) = 0 \quad \& \quad X_t(u) \neq 1 \, \forall u \in N(v) \\ 0 & \text{if } X_t(v) = 0 \quad \& \quad \exists \, u \in N(v) \implies X_t(u) = 1 \quad \& \quad P_{tv} > H \\ 1 & \text{if } X_t(v) = 0 \quad \& \quad \exists \, u \in N(v) \implies X_t(u) = 1 \quad \& \quad P_{tv} \leq H \\ X_t(v) + 1 & \text{otherwise} \end{cases} \tag{5}$$

Stochasticity for GHM transition on 2-D

1. $$X_{t+1}(v) = \begin{cases} 0 & \text{if } X_t(v) = 0 \quad \& \quad X_t(u) \neq 1 \, \forall u \in N(v) \\ 0 & \text{if } X_t(v) = 0 \quad \& \quad \exists \, u \in N(v) \implies X_t(u) = 1 \quad \& \quad P_{tv} > H \\ 1 & \text{if } X_t(v) = 0 \quad \& \quad \exists \, u \in N(v) \implies X_t(u) = 1 \quad \& \quad P_{tv} \leq H \\ X_t(v) + 1 & \text{otherwise} \end{cases} \tag{5}$$

2. Is the quiescent or excited state more important for synchronization?

$$X_{t+1}(v) = \begin{cases} 0 & \text{if } X_t(v) = 0 \quad \& \quad X_t(u) \neq 1 \, \forall u \in N(v) \quad \& \quad P_{tv} \leq H \\ 1 & \text{if } X_t(v) = 0 \quad \& \quad X_t(u) \neq 1 \, \forall u \in N(v) \quad \& \quad P_{tv} > H \\ 0 & \text{if } X_t(v) = 0 \quad \& \quad \exists \, u \in N(v) \implies X_t(u) = 1 \quad \& \quad P_{tv} > H \\ 1 & \text{if } X_t(v) = 0 \quad \& \quad \exists \, u \in N(v) \implies X_t(u) = 1 \quad \& \quad P_{tv} \leq H \\ X_t(v) + 1 & \text{otherwise} \end{cases} \tag{6}$$

Result

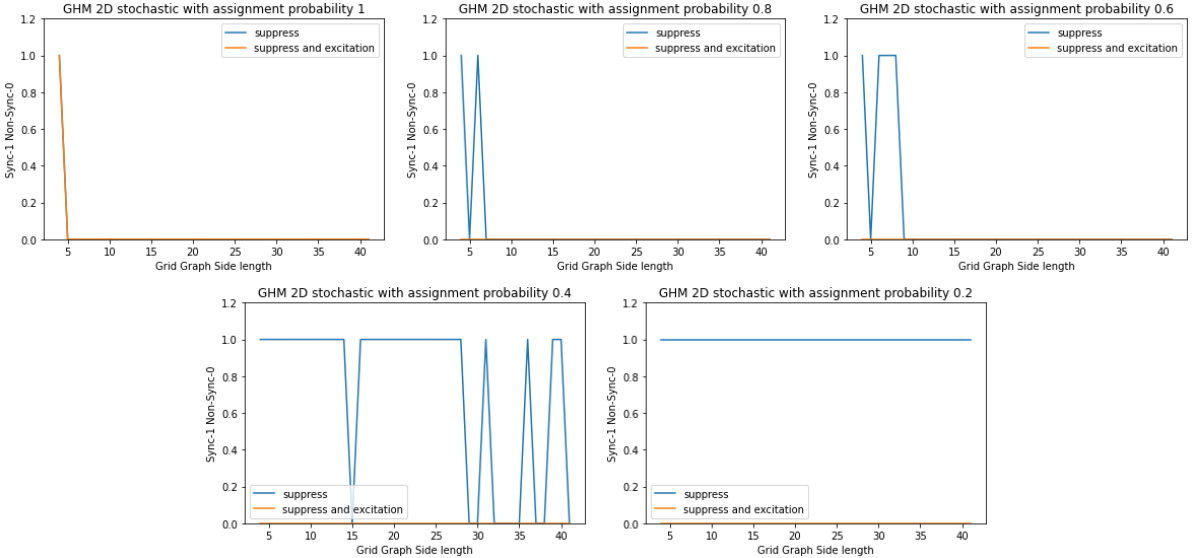


Figure: Synchronizing behavior for different transition probability applied to suppressing excitation only and both boosting and suppressing excitation on grid with side length 2 to 40

GHM NMF Setup

1. Perform Non-Negative Matrix Factorization over subgraphs of NWS, UCLA, and Caltech with rank-16 approximation. Those 16 dictionary elements are latent factors for reconstructing the initial data matrix. Here the features used are adjacency matrix only

GHM NMF Setup

1. Perform Non-Negative Matrix Factorization over subgraphs of NWS, UCLA, and Caltech with rank-16 approximation. Those 16 dictionary elements are latent factors for reconstructing the initial data matrix. Here the features used are adjacency matrix only
2. Generate the dictionary elements as symmetric adjacency matrix plot, then their corresponding graph visualization.

GHM NMF Setup

1. Perform Non-Negative Matrix Factorization over subgraphs of NWS, UCLA, and Caltech with rank-16 approximation. Those 16 dictionary elements are latent factors for reconstructing the initial data matrix. Here the features used are adjacency matrix only
2. Generate the dictionary elements as symmetric adjacency matrix plot, then their corresponding graph visualization.
3. Box plot the distribution of the L-1 norm of dictionary graphs and original adjacency matrices for all the graphs, split by sync or non-sync

GHM NWS NMF Visualization

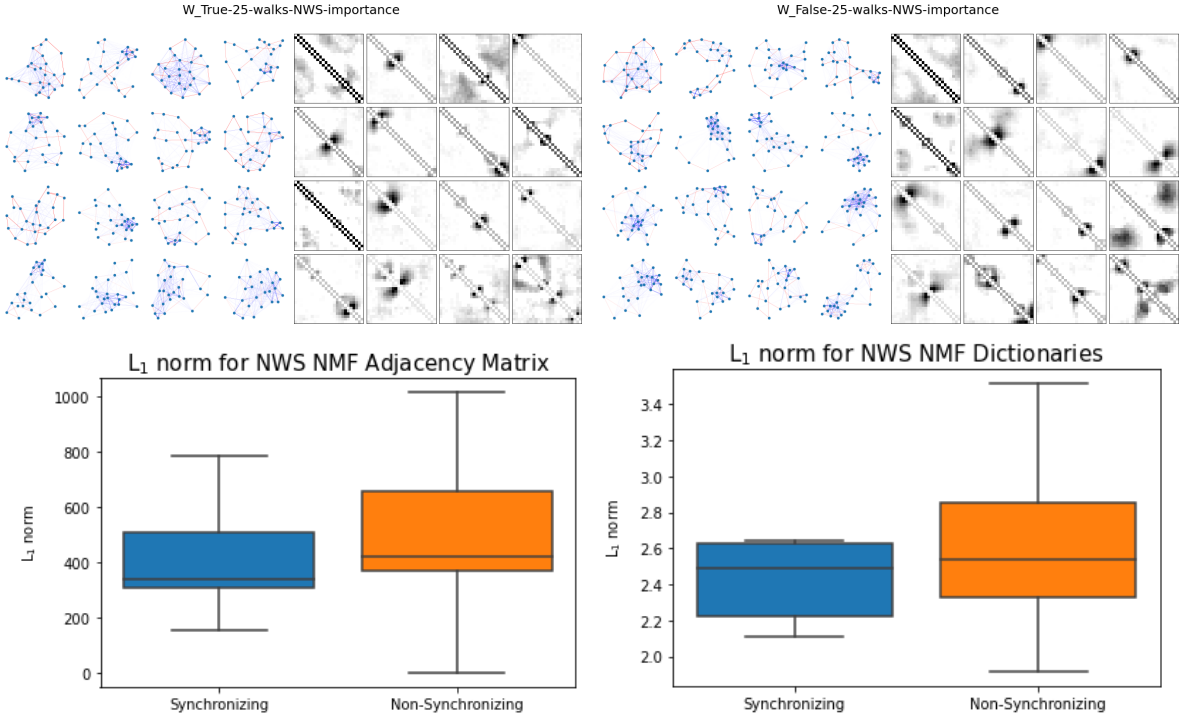


Figure: GHM NMF dictionary plots and box plots for dictionary elements and real graph adjacency matrices on NWS subgraphs

GHM NWS NMF Discussion

1. Hubs occur on both synchronizing and non-synchronizing graphs. Dense area around the main path.

GHM NWS NMF Discussion

1. Hubs occur on both synchronizing and non-synchronizing graphs. Dense area around the main path.
2. Generally non-sync graphs are much denser than sync ones for real adjacency matrix L-1 norm distribution. Same trend for dictionary box plots

GHM UCLA NMF Visualization

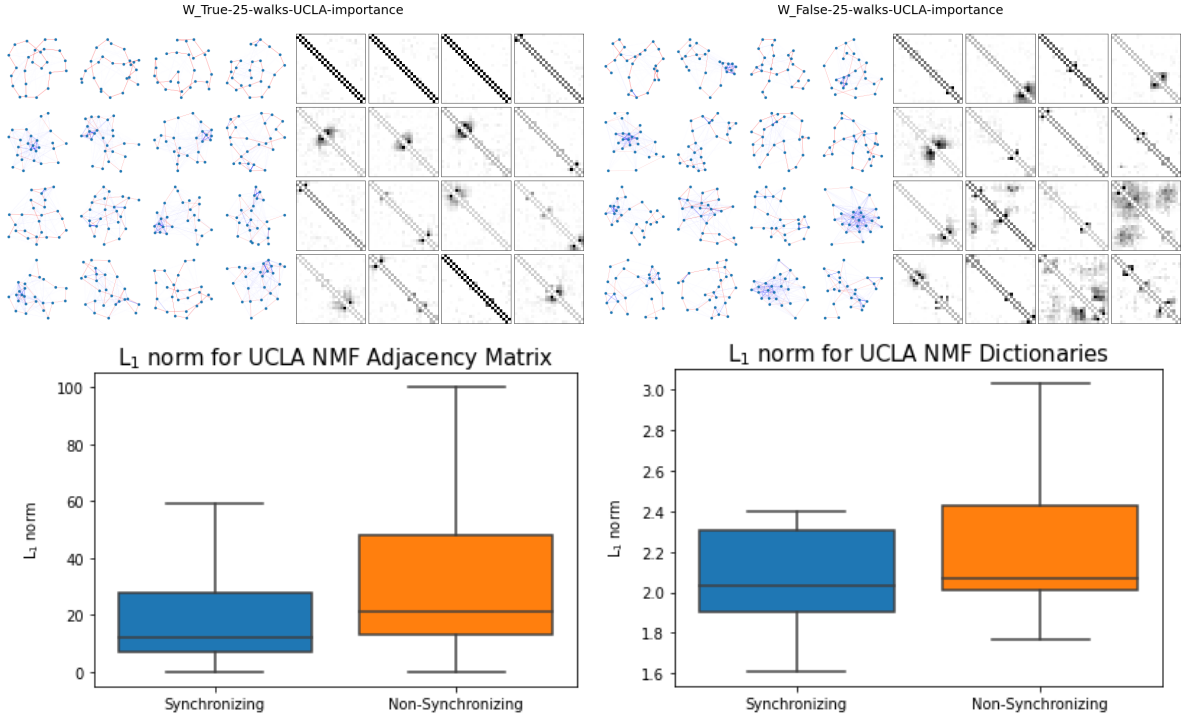


Figure: GHM NMF dictionary plots and box plots for dictionary elements and real graph adjacency matrices on UCLA subgraphs

GHM UCLA NMF Discussion

1. Similar trend as NWS graph. The real graph adjacency matrices' average densities are much lower than NWS.

GHM Caltech NMF Visualization

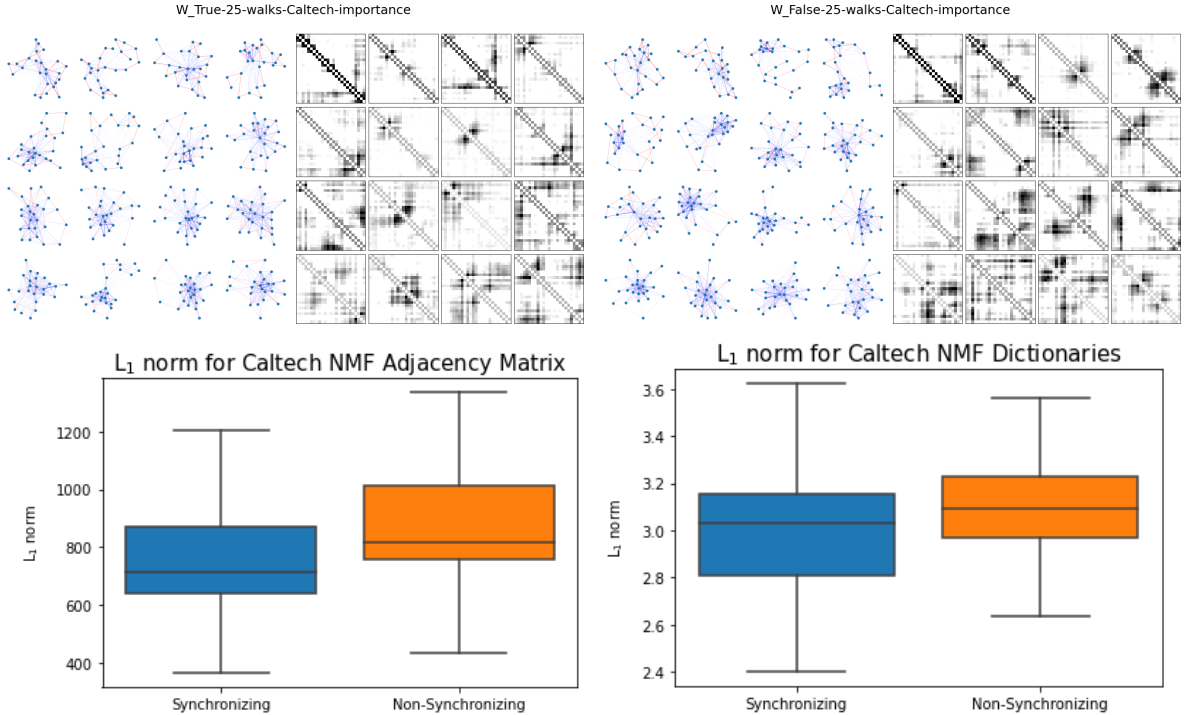


Figure: GHM NMF dictionary plots and box plots for dictionary elements and real graph adjacency matrices on Caltech subgraphs

GHM UCLA NMF Discussion

1. Similar trend to both NWS and UCLA graphs such that with higher graph density, it leans towards non-synchronization.

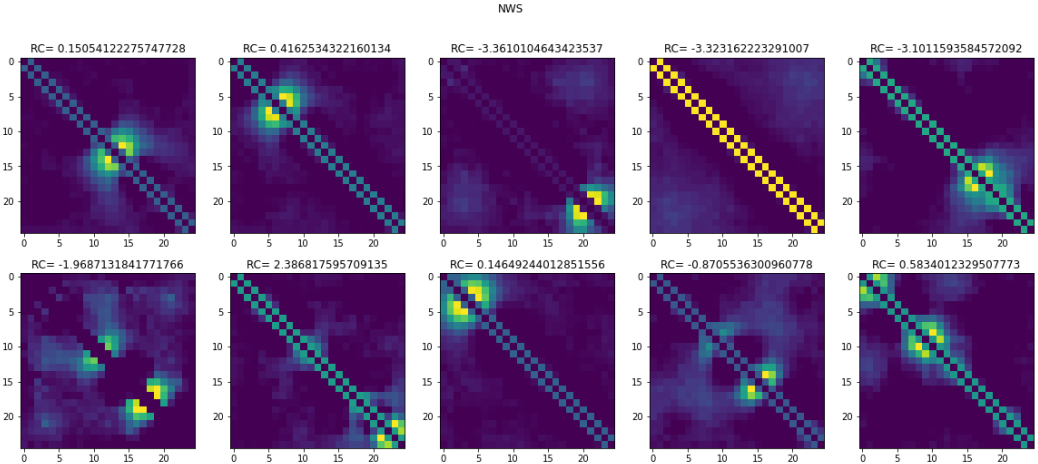
GHM UCLA NMF Discussion

1. Similar trend to both NWS and UCLA graphs such that with higher graph density, it leans towards non-synchronization.
2. Caltech has the densest subgraphs of all three networks.

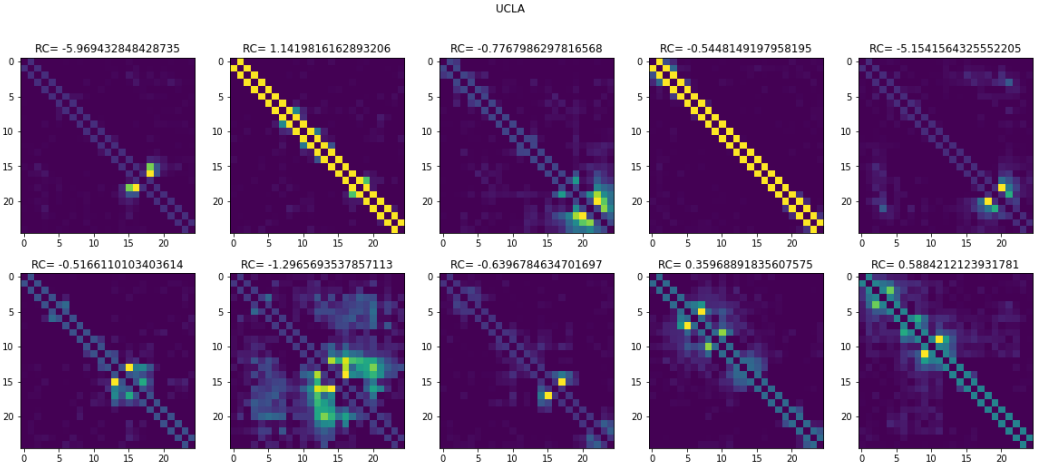
GHM Supervised Setup

- For supervised learning of GHM synchronization behavior, SVM and random forest methods will first be set as benchmarks of traditional classification algorithms.They will be used to train the data for the three netwroks and be compared to supervised dictionary learning techniques of SNMF and SDL-BCD.
- The random forest algorithm will spit out feature importance rank as an ordered list and we will visualize them using bar plot. The standard is based on mean decrease in impurity or known as Gini.
- Then we study the dictionary elements of the dictionary learning methods along with their regression coefficients.
- Apply color difference to the adjacency matrix and perform SDL

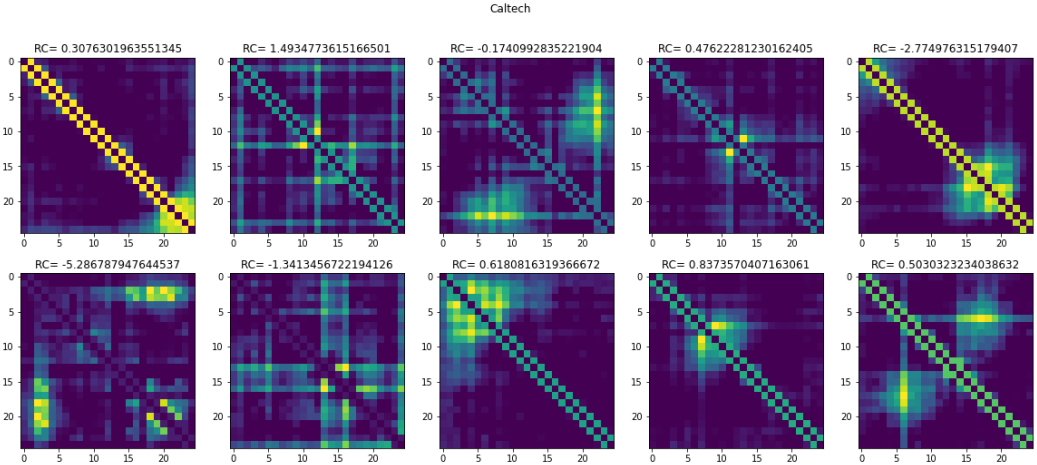
GHM SNMF Dictionary



GHM SNMF Dictionary



GHM SNMF Dictionary



GHM Supervised Training

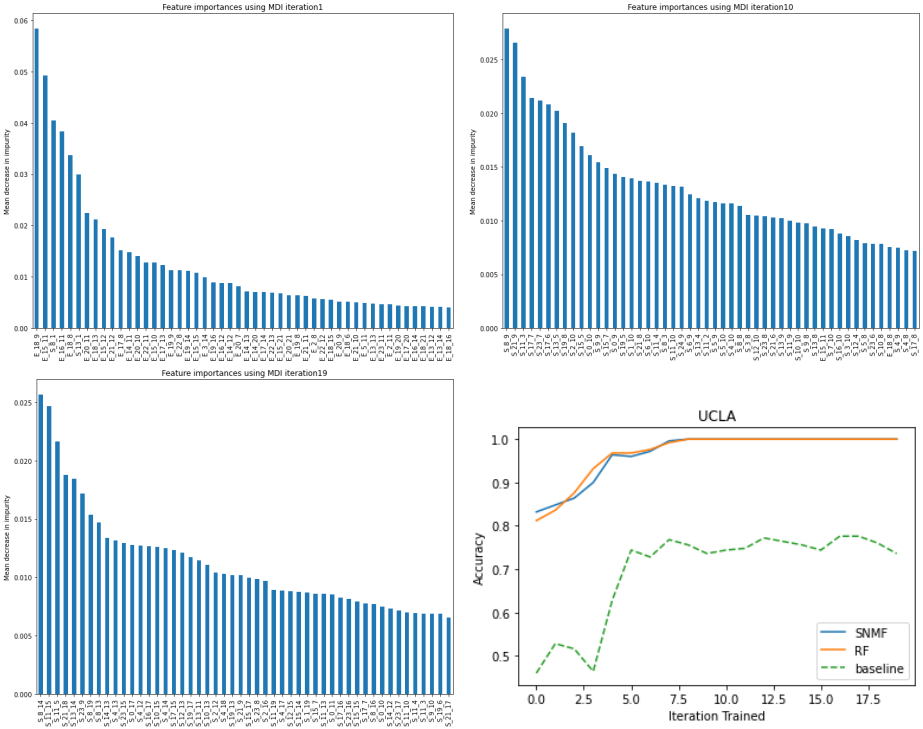


Figure: GHM feature importance and classification accuracy plot

GHM Supervised Training

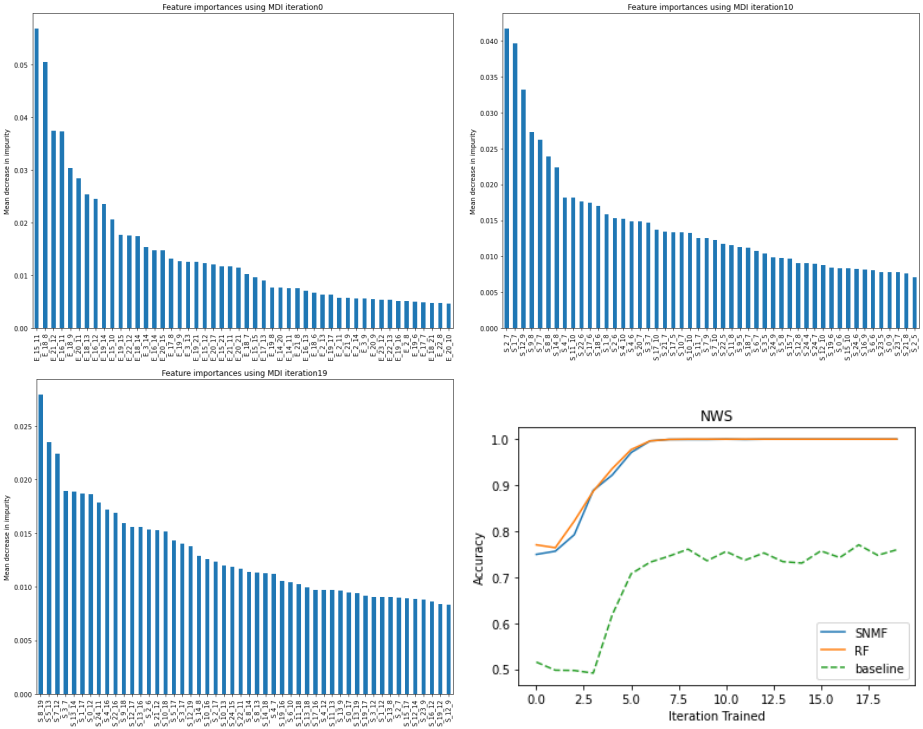


Figure: GHM feature importance and classification accuracy plot

GHM Supervised Training

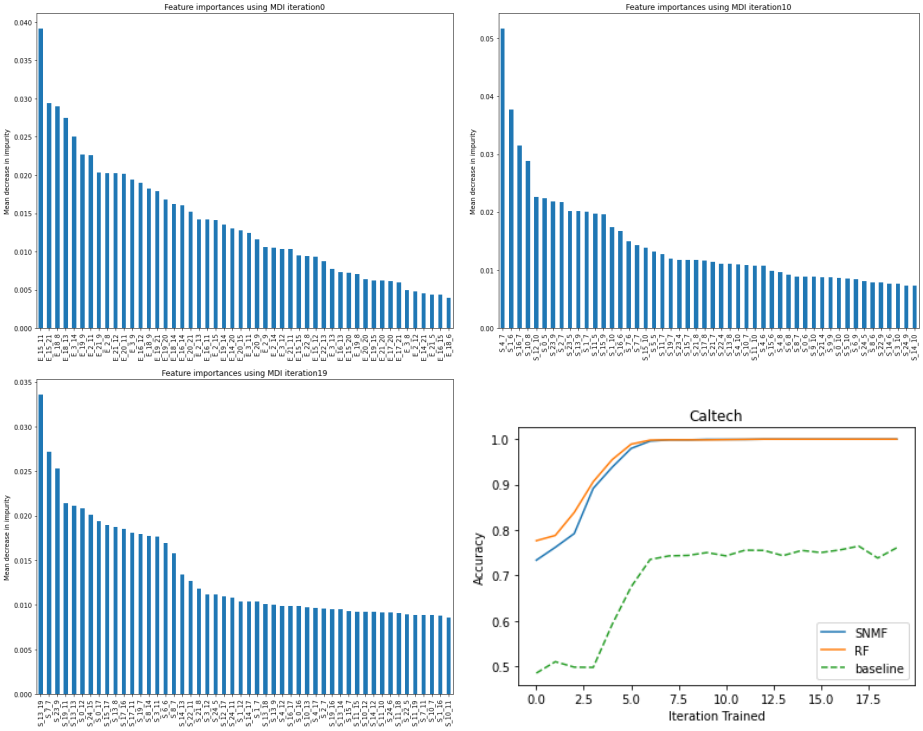


Figure: GHM feature importance and classification accuracy plot

Table of Contents

References I