Check for updates

The MIT Press

RESEARCH

An application of neighbourhoods in digraphs to the classification of binary dynamics

Pedro Conceição[1], Dejan Govc[2], Jānis Lazovskis[3], Ran Levi[1] (iD), Henri Riihimäki[5], and Jason P. Smith[4]

[1]Institute of Mathematics, University of Aberdeen, Aberdeen, UK
[2]Faculty of Mathematics and Physics, University of Ljubljana, Ljubljana, Slovenia
[3]Riga Business School, Riga Technical University, Riga, Latvia
[4]Department of Mathematics and Physics, Nottingham Trent University, Nottingham, UK
[5]Department of Mathematics, KTH, Stockholm, Sweden

ABSTRACT

A binary state on a graph means an assignment of binary values to its vertices. A time-dependent sequence of binary states is referred to as binary dynamics. We describe a method for the classification of binary dynamics of digraphs, using particular choices of closed neighbourhoods. Our motivation and application comes from neuroscience, where a directed graph is an abstraction of neurons and their connections, and where the simplification of large amounts of data is key to any computation. We present a topological/graph theoretic method for extracting information out of binary dynamics on a graph, based on a selection of a relatively small number of vertices and their neighbourhoods. We consider existing and introduce new real-valued functions on closed neighbourhoods, comparing them by their ability to accurately classify different binary dynamics. We describe a classification algorithm that uses two parameters and sets up a machine learning pipeline. We demonstrate the effectiveness of the method on simulated activity on a digital reconstruction of cortical tissue of a rat, and on a nonbiological random graph with similar density.

AUTHOR SUMMARY

We explore the mathematical concept of a closed neighbourhood in a digraph in relation to classifying binary dynamics on a digraph, with particular emphasis on dynamics on a neuronal network. Using methodology based on selecting neighbourhoods and vectorising them by combinatorial and topological parameters, we experimented with a dataset implemented on the Blue Brain Project reconstruction of a neocortical column, and on an artificial neural network with random underlying graph implemented on the NEST simulator. In both cases the outcome was run through a support vector machine algorithm reaching classification accuracy of up to 88% for the Blue Brain Project data and up to 81% for the NEST data. This work is open to generalisation to other types of networks and the dynamics on them.

INTRODUCTION

A *binary state* on a graph means an assignment of binary values to its vertices. A motivating example in this article appears in the context of neuroscience. If one encodes the connectivity of a neuronal network as a directed graph, then the spikes produced by the neurons at an instant of time is a binary state on the encoding graph. Allowing time to vary and recording the spiking patterns of the neurons in the network produces an example of a *binary dynamics* on the encoding graph, namely a one-parameter family of binary states on its vertices. A network of neurons that receives external signals and responds to those signals thus generates a binary dynamics. Binary dynamics appear in other contexts as well (Gleeson, 2008; Samuelsson & Socolar, 2006), but in this paper we use networks of spiking neurons as a primary example.

Binary state on a graph:
An assignment of binary values to the vertices of the graph.

Binary dynamics on a graph:
A one-parameter family of binary states on the graph.

One-parameter family:
A family of objects or numbers that can be indexed by a single parameter, for example, time.

The *signal classification problem*, that is, the task of correctly pairing a signal injected into a neuronal network with the response of the network, or in other words, identifying the incoming signal from the response, is generally very challenging. This paper proposes a methodology by which this task can be approached and provides scenarios in which this methodology is successful.

Considering raw binary states on a large graph is generally quite problematic for a number of reasons. First, the sheer number of theoretically possible states makes analysing a collection of them a daunting task (Churchland & Abbott, 2016; Fan & Markram, 2019). Moreover, natural systems such as neuronal networks tend to be very noisy, in the sense that the emerging dynamics from the same stimulus may take a rather large variety of forms (Cunningham & Yu, 2014; Stein, Gossen, & Jones, 2005). Finally, it is a general working hypothesis in studying network dynamics that the network structure affects its function (Bargmann & Marder, 2013; Chambers & MacLean, 2016; Curto & Morrison, 2019; Rubinov & Sporns, 2010). This paradigm in neuroscience is often encapsulated by the slogan "*neurons that fire together tend to wire together.*" Hence, when studying dynamics on a neuronal network, it makes sense to examine assemblies of vertices, or subgraphs, and the way in which they behave as dynamical subunits, instead of considering individual vertices in the network (Babichev, Ji, Mémoli, & Dabaghian, 2016; Curto & Itskov, 2008; Milo et al., 2002).

In previous studies we considered cliques in a directed graph, with various orientations of the connections between nodes, as basic units from which one could extract information about binary dynamics (Govc, Levi, & Smith, 2021; M. W. Reimann et al., 2017). However, the results in these papers fell short of suggesting an efficient classifier of binary dynamics (Govc et al., 2021, Sections 4.1–4.2). Indeed, when we applied the methods of Govc et al. (2021) and M. W. Reimann et al. (2017) to the main dataset used in this paper, we obtained unsatisfactory classification accuracy. This suggests that in a graph that models a natural system, cliques may be too small to carry the amount of information required for classification of a noisy signal. This motivates us to build our classification strategy on neuron assemblies, where the richer structure serves a dual purpose of amalgamating dynamical information and regulating the noise inherent in single neurons or cliques.

The guiding hypothesis of this paper is that a collection of vertex assemblies, forming a subgraph of the ambient connectivity graph encoding a network, can be used in classification of binary dynamics on the network. A network of spiking neurons is our primary example. Taking this hypothesis as a guideline, we introduce a very flexible feature generation methodology that takes as input binary dynamics on a digraph $\mathcal{G}$ induced on a preselected collection of subgraphs of $\mathcal{G}$, and turns it into a feature vector, which an then be used in machine learning classification. The neighbourhood of a vertex v in the graph $\mathcal{G}$, namely the subgraph of $\mathcal{G}$ that is induced by v and all its neighbours in $\mathcal{G}$, suggests itself naturally as a type of subgraph to be

Digraph:
A finite collection of points connected to each other by directed edges, without loops or multiple edges in the same direction.

considered in this procedure, and is a central object of study in this paper. Vertex neighbourhoods have been studied extensively in graph theory and its applications (Kartun-Giles & Bianconi, 2019). An outline is given below and a full description in Methods.

The way we apply the method can be summarised as follows. Given a directed graph $\mathcal{G}$, we use a variety of real valued vertex functions that we refer to as *selection parameters* and are derived from the neighbourhood of each vertex, to create a sorted list of the vertices. With respect to each such parameter, we pick the "top performing" vertices and select their neighbourhoods. To that collection of subgraphs we apply our feature generation method, which is based again on applying the same parameters to the selected neighbourhoods, now in the role of *feature parameters*. All the parameters we use are invariant under isomorphism of directed graphs, that is, graph properties that remain unchanged when the vertices are permuted while leaving their connectivity intact. Therefore we occasionally refer to certain parameters as "graph invariants."

The choice of parameters is related to measures of network connectivity and architecture. For instance, the parameters **fcc** and **tcc** (see Table 1) are examples of measures of functional segregation (Rubinov & Sporns, 2010). The parameters we refer to as *spectral parameters* arise in spectral graph theory (Chung, 2005) and are prevalent in many applications, including in neuroscience. For instance, the paper de Lange, de Reus, and van den Heuvel (2014) studies the Laplacian spectrum of the macroscopic anatomical neural networks of macaques and cats, and the microscopic network of the *C. elegans*. The topological parameters, such as the Euler characteristic **ec** and Betti numbers are classical topological invariants. In M. W. Reimann et al. (2017) these were used in various ways to extract information on structure and function

Laplacian:
A matrix associated to a digraph that informs on various qualitative properties of the digraph. In this article two versions of the Laplacian matrix are used.

Betti numbers and homology:
A sequence of algebraic objects associated to a topological space that measures, figuratively speaking, for each n the number of n-dimensional cavities or holes in the space. That number is called the n-th Betti number.

Table 1. A partial list of the selection and feature parameters examined in this project

Abbreviation	Short Description
fcc	Clustering coefficient (Fagiolo)
tcc	Transitive clustering coefficient
ec	Euler characteristic
nbc	Normalised Betti coefficient
size	Number of vertices in the graph
asg	Adjacency spectral gap
asr	Adjacency spectral radius
blsg	Bauer Laplacian spectral gap
blsr	Bauer Laplacian spectral radius
clsg	Chung Laplacian spectral gap
clsr	Chung Laplacian spectral radius
tpsg	Transition probability spectral gap
tpsr	Transition probability spectral radius

Note. See Supporting Information Table S1 for additional parameters.

and their interaction in the Blue Brain Project reconstruction on the neocortical column. The parameter **size** is a natural parameter associated to any graph and is closely related to firing rate in neuroscience. However, most of the parameters we tested were never examined in a neuroscientific context. Our aim was to investigate which parameters may prove useful in classification of binary dynamics without making any assumptions about their relevance. It is exactly this approach that allowed us to discover that certain spectral parameters perform strongly as selection parameters, while others do not. At the same time a newly introduced topological parameter, "normalised Betti coefficient" **nbc** shows strong performance as a feature parameter when tested on neighbourhoods with low selection parameter values, but not on high selection values.

The primary test of our methods in this paper is done on data generated by the Blue Brain Project that was also used in M. Reimann et al. (2021) for signal classification by established neuroscience methodology. The data consists of eight families of neuronal stimuli that are injected in a random sequence to the digital reconstruction of the neocortical column of a young rat. This reconstructed microcircuit consists of approximately 31,000 neurons and 8,000,000 synaptic connections, and is capable of receiving neuronal signals and responding to them in a biologically accurate manner (Markram et al., 2015). We used 60% of the data to train a support vector machine, and the remaining 40% for classification. With our methods we are able to achieve classification accuracy of up to 88%.

In this article we did not attempt to explain the relevance of any of the mathematical concepts we use to neuroscience, as our main aim was to discover and investigate the utility of various concepts. However, in M. Reimann et al. (2021) the same dataset is studied by standard techniques of computational neuroscience combined with the ideas presented in this paper. In particular, it is shown that an informed choice of neighbourhood improves classification accuracy when compared to traditional methods. Interestingly, the selection of neighbourhoods that improved performance with the technique presented in M. Reimann et al. (2021) show reduced performance with the techniques presented in this article, and vice versa. In both projects a classification accuracy of nearly 90% was achievable, but with different selection parameters (see Results). This suggests that considering vertex neighbourhoods as computational units can be beneficial in more than one way.

To further test our methods in different settings, we used the NEST neural simulation tool (Jordan et al., 2019) to generate neuronal networks. This software package simulates network models of spiking neurons by using simplified neuron models to allow more flexibility and faster processing speed. We created a collection of eight families of stimuli, but on random graphs with varying densities, and applied our machinery to that dataset. Here again we obtained classification accuracy of up to 81%.

Important work on (open) vertex neighbourhoods was reported recently in Kartun-Giles and Bianconi (2019). Our approach is independent of this work and is different from it in a number of ways. Most significantly, we do not study the structure of the entire graph and its dynamical properties by means of its full neighbourhood structure. Instead, we aim to infer dynamical properties of the graph from a relatively small collection of vertices, selected by certain graph theoretic and topological properties, and their neighbourhoods.

High-resolution figures and supplementary material is available at the Aberdeen Neurotopology Group webpage. In particular, we included a comprehensive visualization of spectral graph invariants of the Blue Brain Project graph, as well as other types of stochastically generated graphs, animations of some of the background work for this project, and a list of links to software implementing the methodology described in this paper.

RESULTS

We start with a brief description of the mathematical formalism used in this article and our approach to classification tasks. This is intended to make the section accessible to readers without a strong mathematical background. We then proceed by describing our main data source and the setup and implementation of our experiments. Following this preparation we present our results, validation experiments, and an application of the same techniques in a different setup.

A Brief Introduction to the Mathematical Formalism

In this article a *digraph* will always mean a finite collection of vertices (nodes) V and a finite collection of oriented edges (arcs) E. Reciprocal edges between a pair of vertices are allowed, but multiple edges in the same orientation between a fixed pair of vertices and self-loops are not allowed.

The fundamental mathematical concept essential for our discussion is that of the neighbourhood of a vertex in a digraph (Figure 1). Let $\mathcal{G}$ be a digraph, and let v_0 be any vertex in $\mathcal{G}$. The *neighbours* of v_0 in $\mathcal{G}$ are all vertices that are "one step away" from v_0, in either direction. The *neighbourhood* of v_0 in $\mathcal{G}$ is the subgraph of $\mathcal{G}$ induced by v_0 and all its neighbours, which we denote by $N_{\mathcal{G}}(v_0)$. The vertex v_0 is referred to as the *centre* of its neighbourhood.

Numerical invariants of digraphs can be found in pure and applied graph theory literature, many of those found their uses in theoretical neuroscience (see Rubinov & Sporns, 2010, for a good survey). Some such invariants are used in this article, and a few are introduced here for the first time (e.g., transitive clustering coefficient). Other parameters we used are defined by using topological constructions that arise from digraphs. Such constructions are typically invariant under digraph isomorphism. Standard tools of algebraic topology can then be used to extract numerical invariants of graphs in ways that take emerging higher dimensional structure into account.

Algebraic topology:
The mathematical discipline of studying certain properties of topological spaces by means of associated algebraic objects.

There are many ways in which one can associate a topological space with a digraph. In this article we use the *directed flag complex*. It is a topological space made out of gluing together *simplices* in different dimensions, starting at 0-simplices (points), 1-simplices (edges), 2-simplices (triangles), 3-simplices (tetrahedra), etc. The *n*-simplices in a directed flag complex associated to a digraph are its directed (*n* + 1)-cliques, namely the ordered subsets of vertices $\{v_0, v_1, ..., v_n\}$, such that there is an edge from v_i to v_j for all $i < j$. Figure 2 shows the directed flag complex associated to a small digraph. The directed flag complex was introduced and used for topologically analysing structural and functional properties of the Blue Brain Project reconstruction of the neocortical columns of a rat (M. W. Reimann et al., 2017). The interested reader may find a comprehensive survey of directed flag complexes and other topological concepts in the Materials and Methods section of M. W. Reimann et al. (2017). If v_0 is a vertex in $\mathcal{G}$, we denote by $TR_{\mathcal{G}}(v_0)$ the directed flag complex of $N_{\mathcal{G}}(v_0)$.
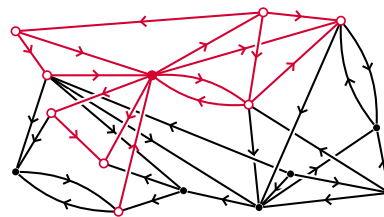

Figure 1. A neighbourhood in a digraph, marked in red, with its centre marked solid colour.
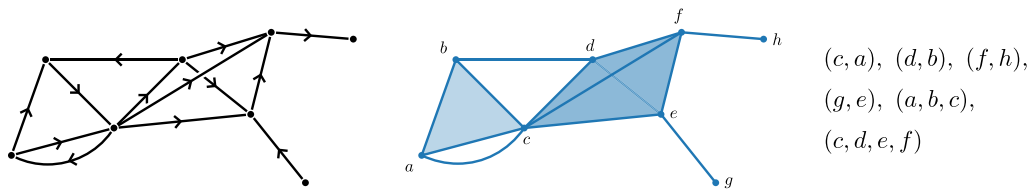
Figure 2. A digraph (left), the associated directed flag complex as a topological space (centre), and its maximal cliques (right).

The Classification Method

We now describe briefly our approach to classification of binary dynamics. For a precise mathematical definition of what we mean by binary dynamics see Methods. The task at hand can be described as follows. We are given a large set of instantiations of binary dynamics on a fixed digraph $\mathcal{G}$, each of which is labelled by a symbol from some relatively small set. The label of each binary dynamics is unique and known. The aim is to produce a machine learning compatible topological summary for each binary dynamics, so that when the summaries are introduced in a random order, one can train on part of the data with known labels and predict the unknown labels of the remaining part.

The *first step* is selection of neighbourhoods. For each vertex v in the digraph $\mathcal{G}$ we consider its neighbourhood $N_\mathcal{G}(v)$ and the associated directed flag complex $\mathrm{Tr}_\mathcal{G}(v)$. We then compute a variety of numerical graph parameters of $N_\mathcal{G}(v)$ and topological parameters of $\mathrm{Tr}_\mathcal{G}(v)$. These parameters are used to create a ranked list of vertices in $\mathcal{G}$. We then select for each parameter 50 vertices that obtained the top (or bottom) values with respect to that parameter. We now have a set of 50 neighbourhoods corresponding to each parameter. A parameter that is used in this step is referred to as a *selection parameter*, and we denote it by P. A short summary of the main parameters we used with their abbreviations is in Table 1. A detailed description of the parameters is given in Methods.

Graph parameter:
A real or complex number that can be associated to any given graph by some mathematical procedure.

In the *second step* we introduce binary dynamics in $\mathcal{G}$. Each instantiation of the dynamics consists of several consecutive time bins (in our experiments we used two, but there is no limitation). For each time bin we consider the neurons that were active and the subgraph that they induce in each of the neighbourhoods we preselected. This gives us, for each selection parameter and each time bin, a set of 50 subgraphs that correspond to a particular instantiation of binary dynamics on $\mathcal{G}$.

The *third step* is vectorising the data, that is, a computation of the same graph parameters and topological parameters for each of the subgraphs resulting from the second step. When we use our parameters in the vectorisation process they are referred to as *feature parameters*, and are denoted by Q. This now gives a vector corresponding to each instantiation of the dynamics, and the pair (P, Q) of selection and feature parameters.

The *fourth and final step* is injecting the data into a support vector machine. In this project we used 60% of the data for training and the remaining for testing. See Figure 3 for a schematic summary of the process.
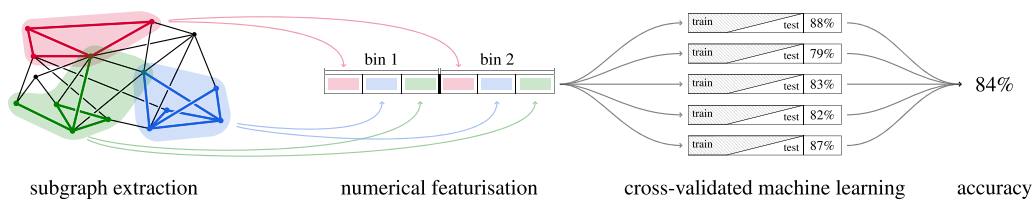

Figure 3. A schematic description of the vector summary and classification pipeline.

We note that the method described here is an example of a much more general methodology that is described in detail in the Methods section of this article. In particular, the graph and topological parameters that we chose to work with are selected from within the abundance of mathematical concepts that arise in graph theory, combinatorics, and topology. We do not attempt in this article to associate a neuroscientific meaning to these parameters.

The Data

Our main source of data is a simulation that was run on a Blue Brain Project reconstruction of the microcircuitry of the somatosensory cortex in the brain of a rat (Markram et al., 2015). From this model we extract the connectivity of the microcircuit in the form of a digraph whose vertices correspond to neurons, and with an edge from v to u if there is a synaptic connection from the neuron corresponding to v to the one corresponding to u. We denote the Blue Brain Project digraph by $\mathcal{G}$. The digraph consists of 31,346 vertices and 7,803,528 edges. The connectivity matrix of this specific circuit, as well as 41 other instantiations of the reconstruction, is accessible on the Digital Reconstruction of Neocortical Microcircuitry website.

The binary dynamics we experimented with consists of eight stimuli families labelled 0–7. For each stimulus a random subset (10%) of afferent neurons is activated. The stimuli differ with respect to which subset of afferent neurons is activated, where afferents can be shared between stimuli. The probability of a given afferent being associated with two given stimuli is 1%. In each stimulation time window one and only one stimulus is presented. The stimuli were injected into the circuit in a random sequence of 200 milliseconds per stimulus, and 557 repeats for each stimulus label. The dataset thus consists of 4,456 binary dynamics functions. The task is to determine the label of that stimulus, that is, the expected output is an integer from 0 to 7. Thus, the chance level performance would be 12.5%. More detail on the source of data, biological analysis, and an alternative approach to classification of the same data is in M. Reimann et al. (2021).

Setup

We computed all the graph parameters listed in Table 1, as well as additional parameters listed in the Supporting Information, for all neighbourhoods in the digraph (see Supporting Information, Data and Code, for a brief description of computational methods and links to software). We fixed a positive integer M, and for each selection parameter P we selected the vertices v_1, v_2, ..., v_M, whose neighbourhoods $N_{\mathcal{G}}(v_1)$, ..., $N_{\mathcal{G}}(v_M)$ obtained the top (or bottom) M values of the parameter P (see Step II in Methods). We experimented with $M = 20, 50, 100,$ and 200. Here we report on the results we obtained for $M = 50$, which provided the highest classification accuracy. For $M = 20$ performance was strong as well, but for $M = 100$ and 200 the improvement compared to $M = 50$ was relatively minor, and not worth the additional time and computation needed.

Vector Summaries

Each binary dynamics in our dataset has time parameter t between 0 and 200 milliseconds. The subinterval [0, 60] is where almost all the spiking activity is concentrated across the interval. Furthermore, the bulk of the stimulus is injected in the first 10 ms. Since we aimed to classify the response to the stimulus rather than the stimulus itself, we chose $\Delta = [10, 60]$ and divided that interval into two 25 ms subintervals, as experimentation showed that these choices provide the highest classification accuracy (see Step I in Methods).

We denote each instantiation of binary dynamics on $\mathcal{G}$ by B^n, for $n = 1, ..., 4,456$. Each instantiation consists of two binary states B_1^n, B_2^n corresponding to the neurons that fired in each of the 25 ms subintervals. For each selection parameter P, and each of the corresponding neighbourhoods $N_{\mathcal{G}}(v_m)$, $m = 1, ..., 50$, we computed the subgraphs $N_{m,k}$ of $N_{\mathcal{G}}(v_m)$ induced by the binary state B_k^n, that is, the subgraph induced by the neurons that fired in the given interval. This gave us, for each binary dynamics B^n and each graph parameter P, a 2×50 matrix U_n^P of subgraphs of $\mathcal{G}$, whose (m, k) entry is $N_{m,k}^n$ (see Step II in Methods).

Finally, for each graph parameter Q (from the same list of parameters) we applied Q to the entries of the matrix U_n^P to obtain a numerical feature matrix $U_n^{P,Q}$ corresponding to the binary dynamics function B^n, the selection parameter P, and the feature parameter Q. The matrix $U_n^{P,Q}$ is a vector summary of the binary dynamics B^n (see Step III in Methods).

Classification

For each pair of graph parameters (P, Q) the vector summaries $\{U_n^{P,Q}\}$ were fed into a support vector machine (SVM) algorithm. Our classification pipeline was implemented in Python using the `scikit-learn` package and the SVC implementation therein. The SVC was initialised with default settings and we used a 60/40 train/test split. The kernel used was Radial Basis function. We used one-versus-one approach for multiclass classification. For cross-validation we used standard fivefold cross-validation in `scikit-learn`, https://scikit-learn .org/stable/about.html. The results are presented in Figure 4.

For each of the selection parameters we tested, we considered both the neighbourhoods that obtained the top 50 values and those that obtained the bottom 50 values. In all the experiments, four parameters gave markedly better performance when used as feature parameters than all other parameters: Euler characteristic (**ec**), normalised Betti coefficient (**nbc**), **size**, and Bauer Laplacian spectral radius (**blsr**). All four perform significantly better than other feature parameters when the neighbourhoods were selected by bottom value parameters. With respect to top value selection parameters, **ec** and **size** performed well, while **nbc** and **blsr** were significantly weaker as feature parameters, except when coupled with Chung Laplacian spectral gap (**clsg**). The neighbourhoods selected by top values of selection parameters gave best
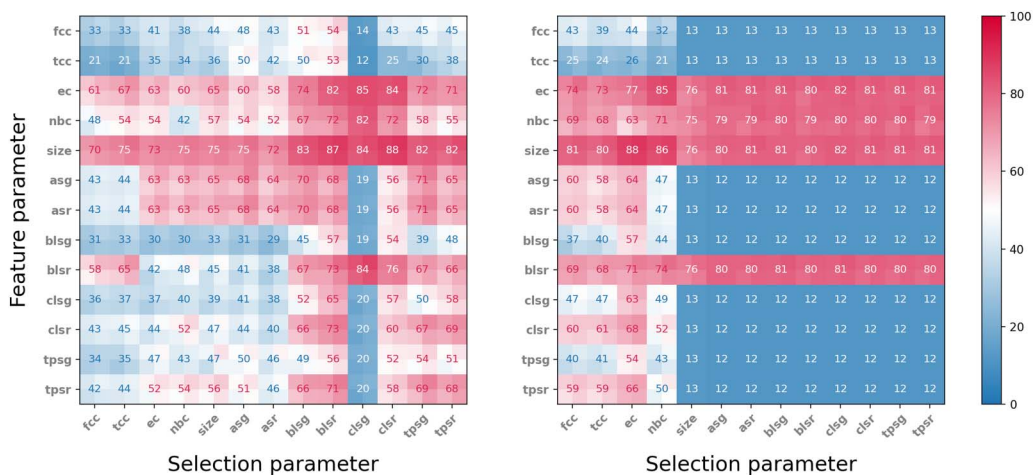

Figure 4. Results of eight stimuli classification experiments. Range of cross-validated accuracy is indicated by four smaller squares in each square. Left: Classification accuracy selecting the 50 neighbourhoods with highest parameter value. Right: Classification accuracy selecting the 50 neighbourhoods with lowest parameter value. Compare with Supporting Information Figure S3.

results when the selection parameter was one of the spectral graph invariants, while selecting by bottom value of selection parameters, the two types of clustering coefficients (**cc** and **tcc**) and Euler characteristic (**ec**) performed best.

Interestingly, the two best performing feature parameters, Euler characteristic and size, gave good results across all selection parameters, and performed almost equally well, regardless of whether the neighbourhoods were selected by top or bottom selection parameter value. This suggests that, at least in this particular network, the choice of feature parameter plays a much more important role in classification accuracy than any specific selection parameter. On the other hand, examining the rows of the best performing feature parameters, in Figure 4, we see a difference of up to 27% (top **ec**), 40% (top **nbc**), and 18% (top **size**) in classification accuracy, depending on which selection parameter is used, suggesting that, within a fixed choice of a feature parameter, the selection parameter may play an important role in the capability of the respective neighbourhoods to encode binary dynamics. Note that randomly classifying the eight stimuli gives an accuracy of 12.5%.

Validation

In order to validate our methods, we created five experiments, the results of which we then compared to a subset of the original tests. In each case we retrained the SVM algorithm and then retested.

A motivating idea in neuroscience in general, and in this work in particular, is that structure is strongly related to function. Our approach, using neighbourhoods sorted by graph parameters and using the same graph parameters as feature parameters is proposed in this article as a useful way of discovering combinations of parameters that achieve good classification results of binary dynamics. To test the validity of this proposal, we challenged our assumptions in five different ways, as described below.

Random selection. In this simple control experiment we test the significance of the selection parameter by comparing the results to a random choice of 50 vertices and performing the same vector summary procedure on their neighbourhoods. Twenty iterations of this experiment were performed, and the results for each feature parameter were compared to the outcome for the same feature parameter and the selection parameter with respect to which this feature parameter performed best. The results are described in Figure 5.
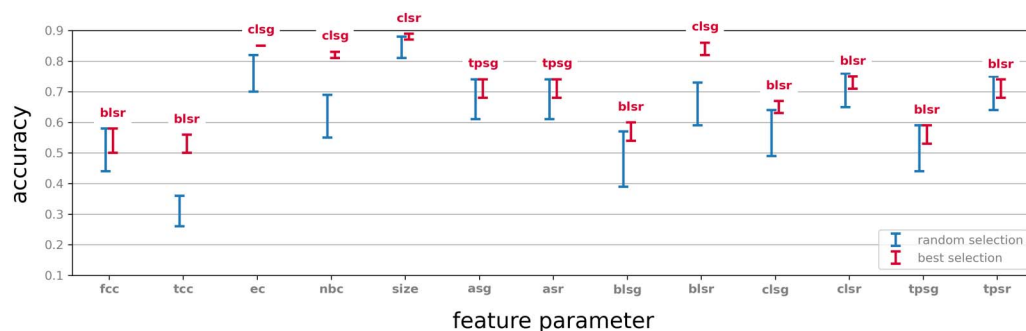

Figure 5. The classification performance based on the neighbourhoods of 50 randomly selected vertices (blue), compared to the performance of neighbourhoods selected by graph parameters with respect to a selection of feature parameters (red). Errors bars indicate range over 20 iterations. Labels on the red error bars indicate the selection parameter that performed best with respect to the indicated feature parameter. Compare with Supporting Information Figure S2.

We observe that in almost all cases reported here, a choice of neighbourhoods determined by a selection parameter outperforms a random choice (in some cases marginally). We also note that in all those cases the performance of a choice informed by one of these selection parameters exhibits a more consistent behaviour in terms of classification accuracy. This can be seen from the considerably larger error bars in the case neighbourhoods are selected at random. On the other hand, for some feature parameters a random choice does not seem to be a disadvantage, even compared to the best selection parameter with respect to this feature parameter (Supporting Information Figure S3). This suggests that while selection and generation of vector summary by objective parameters are advantageous, experimentation is generally necessary in order to decide which parameters best fit the classification task.

Neighbourhood versus centre. A working hypothesis in this paper is that neighbourhoods carry more information about a binary dynamics than individual vertices. We examined for each selection of 50 neighbourhoods by a graph parameter, as described above, the classification capability of the centres of these neighbourhoods. Specifically, this experiment is identical to the original classification experiment, except for each selection parameter P the two rows of the corresponding feature matrix have binary values, where the j-th entry in row i is set to be 1 if the j-th neuron in the sorted list fired in the i-th time bin at least once and 0 otherwise. These feature vectors were then used in the classification task by using the same train and test methodology. For each of the selection parameters we tested, we considered both the top 50 and the bottom 50 neurons in the corresponding sorted list.

The results of this experiment were compared with the original experiments and are shown in Figure 6. We note that in all cases a very significant drop in performance occurs. Interestingly, some vertices in the top 50 of a sorted list show classification accuracy that is far better than random, while the bottom 50 give performance comparable to random (for example, **fcc**). In some cases, however, the bottom 50 vertices give better performance than the top 50. This suggests that the selection parameters play a role in classification accuracy even before considering the activity in the neighbourhood of a vertex.

We also note that for almost all top valued selection parameters recorded in Figure 6 and some of the bottom valued ones, the classification performance using the centre alone is significantly better than random. This observation reinforces the idea that selection parameters inform on the capability of neurons to inform on activity.

Neighbourhoods versus arbitrary subgraphs. For each selection parameter we considered the degrees of the 50 selected centres. For a centre v_i of degree d_i we then selected at random d_i vertices in the ambient graph and considered the subgraph induced by those vertices and
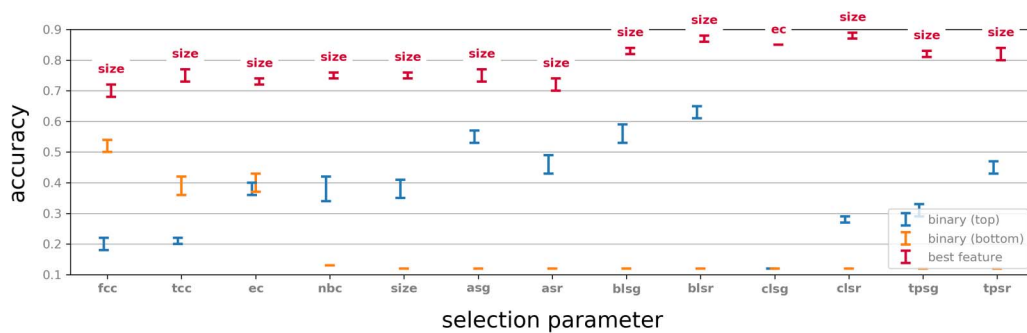

Figure 6. Classification results by binary vectors using only the centres of each of the top and bottom 50 neighbourhoods for each parameter. For comparison, the performance for each selection parameter classified by the highest performing feature parameter is included.
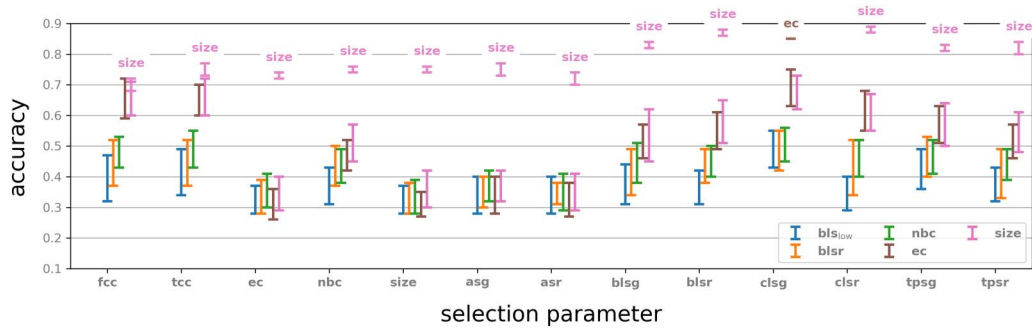
Figure 7. Classification by subgraphs of the same vertex count as the neighbourhoods selected by the specified selection parameters. The results of classification by the highest performing feature parameters are above each of the columns.

the centre v_i. We used these 50 subgraphs in place of the original neighbourhoods. In this way we create for each centre a new subgraph with the same vertex count as the original neighbourhoods that is unrelated to the centres in any other controllable way. We extracted feature vectors by using these subgraphs for each of the selection parameters and repeated the classification experiment. The results were compared to the original results with respect to the strongest performing feature parameter. Notice that these are always either **ec** or **size**, both of which can be applied to an arbitrary digraph, not necessarily a neighbourhood.

The results of this experiment were compared with the original experiments, and are shown in Figure 7. There is a clear drop in performance for all selection parameters except **fcc** (Fagiolo's clustering coefficient; see Methods). Furthermore, classification using these subgraphs shows considerably larger error bars. This suggests that using neighbourhoods with our methodology is advantageous. One explanation for this may be the tighter correlation of activity among neurons in a neighbourhood, compared to an arbitrary subgraph of the same size in the network, but we did not attempt to verify this hypothesis.

Fake neighbourhoods. In this experiment we considered for each centre its degree and selected at random the corresponding number of vertices from the ambient graph. We then modified the adjacency matrix of the ambient graph so that the centre is connected to each of the vertices selected in the appropriate direction, so as to preserve the centre's in- and out-degree. Computationally, this amounts to applying a random permutation to the row and the column of each of the centres. The result is a new ambient graph, where the old centres are now centres of new neighbourhoods. We extracted feature vectors using these "fake neighbourhoods" and repeated the classification experiment. The results were compared with the original classification. The outcome is illustrated in Figure 8.
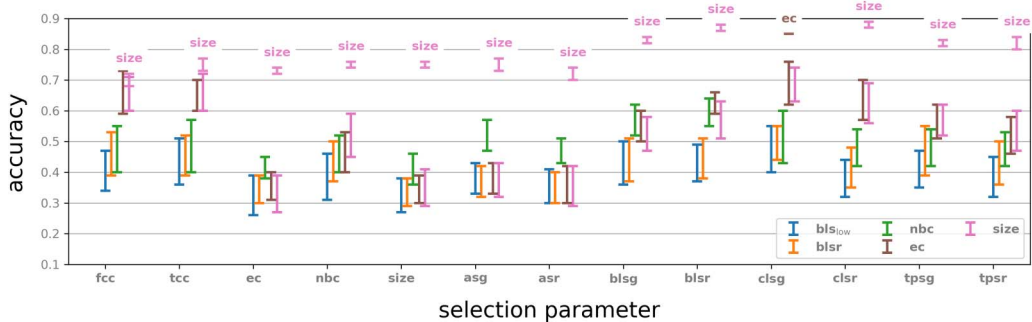

Figure 8. Classification by "fake neighbourhoods": Original classification with respect to best performing feature parameter is given for comparison.
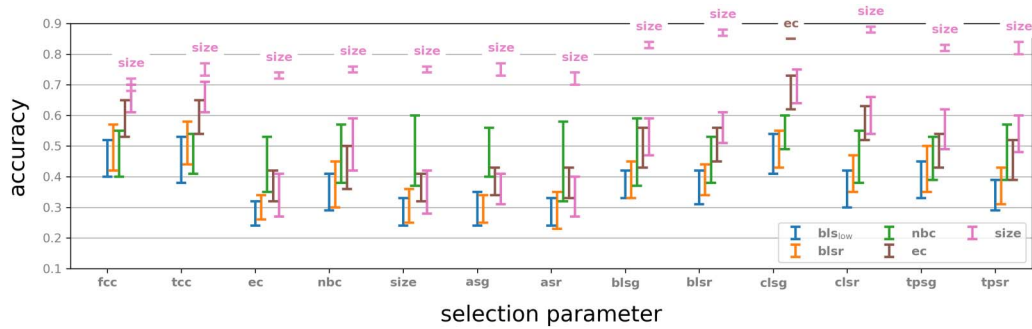
Figure 9. Classification of shuffled binary dynamics functions and comparison to the top results for the original dynamics.

We note that with respect to almost all selection parameters there is a significant drop in performance resulting from this modification. The one exception is **fcc**, where **ec** as a feature parameter actually sometimes gives slightly better results, but with a large error bar. It is interesting that the results are similar for some of the parameters to those observed in previous experiment (Figure 7), but quite different for others. However, the drop in performance is similar in both cases. We make no hypothesis attempting to explain these observations.

Shuffled activity. In this experiment we applied a random permutation σ of the neuron indices in the Blue Brain Project microcircuit, so that neuron $\sigma(i)$ now receives the spike train (sequence of spikes) of neuron i for each stimulus. That is, we precompose the binary dynamics with σ to get a new binary dynamics, which still appears in eight varieties, since the operation of permuting the neuron indices is bijective. In other words, we can reconstruct the original activity from the shuffled activity by applying the inverse permutation σ^{-1}. The same selection and feature parameters were used and the resulting data was used for training and testing. The results are shown in Figure 9.

We observe again that there is a significant drop in performance resulting from this shuffling. This is quite surprising since the shuffled activity spike train should give eight families of stimuli that carry some sort of internal resemblance, and since we retrained and tested with these stimuli, one could expect that the classification results will be comparable to those of the original experiments. That not being the case suggests that structure and function in the Blue Brain Project reconstruction are indeed tightly related.

Testing the Method on an Artificial Neuronal Network

To test our methods in a nonbiological binary dynamics setting, we conducted a set of experiments with the NEST simulator (Jordan et al., 2019). The NEST software simulates spiking neuronal network models and offers a vast simplification of neuronal networks that are based on the exact morphology of neurons (such as the Blue Brain Project reconstructions). It also provides great flexibility in the sense that it allows any connectivity graph to be implemented in it and any initial stimulation to be injected into the system with the response modulated by various flexible parameters.

To move as far as possible from a strict biological setup, we generated a number of Erdős–Rényi random digraphs on 1,000 vertices, which we implemented on NEST. We then created eight distinct stimuli, each enervating a random selection of 100 vertices of the graph. A random sequence of stimuli was then created, with each stimulus type repeated 500 times. Our experiment consisted of injecting the sequence of stimuli into the simulator, for a duration of 5 ms, one every 200 ms, to reduce the influence of one stimulus on the next. To introduce
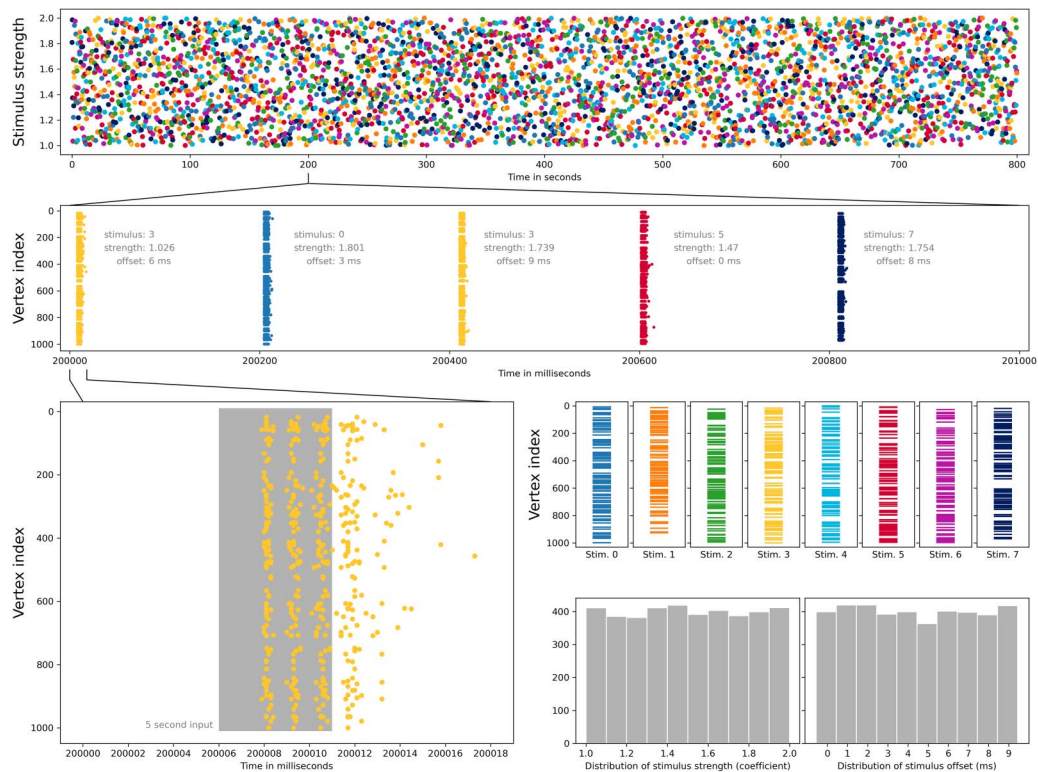
Figure 10. Eight types of input stimuli for Erdős–Rényi random digraphs, executed as a single 800 second experiment. Top row: Sequence of stimuli types, 500 of each, and relative strength of input for each stimulus. Second row: Spiking neurons on a 1,000-ms interval from the experiment. Bottom left: Spiking neurons and length of external input on a 18-ms interval. Third row right: Random selections of 100 vertices from 1,000 vertices, acting as receptors of external input. Bottom row right: Distribution of randomly selected relative strength and input stimulus time offset over the whole experiment.

some randomness, the start time of each stimulus is randomly selected from the first 10 ms, the strength of each stimulus is multiplied by a random number between 1 and 2, and background noise is included (using NEST's `noise_generator` device with strength 3). For each 200-ms interval, the first 10 ms were not included in the classification. As a result some of the input may be included in the classified data, but never more than 4 ms, and for approximately 60% of the 4,000 stimuli the input is completely excluded from classification. The code used to create these experiments is available in Lazovskis (2021), and the experiments are presented visually in Figure 10.

The spikes from this simulation were then extracted and were run through the same pipeline as the Blue Brain Project data. We experimented with graph densities of 0.08, 0.01, and 0.005, and with selections of 10, 20, and 50 neighbourhoods. Figure 11 shows the performance by the selection parameters from Table 1. Size was used in all cases as a feature parameter. The best performance was obtained with 50 neighbourhoods, with graph density of 0.01 in almost all selection parameters. The results of experiments with all parameters can be seen in Supporting Information Figure S5.

Interestingly, the middle graph density of 0.01 consistently performed equally as well or better than both the denser 0.08 and less dense 0.005 across all feature parameters, except neighbourhood size (**size**) and adjacency spectral gap (**asg**). Another interesting observation is that the strongest selection parameter in this experiment turns out to be normalised Betti coefficient (**nbc**), or transitive clustering coefficient (**tcc**), depending on if "strongest" is taken
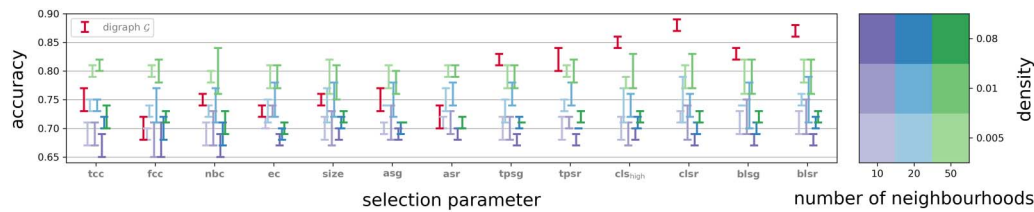
Figure 11. Classification of eight random signals on an Erdős–Rényi random digraph on 1,000 vertices and connection probabilities of 8%, 1%, and 0.5% and selection of 10, 20, and 50 neighbourhoods, modelled on a NEST simulator. Selection parameters are the same as in the main example and feature parameter is always **size**. Graph $\mathcal{G}$ means the BBP graph and its performance with respect to **size** as feature parameter is given for comparison. Compare with Supporting Information Figure S5.

to mean with the highest individual accuracy or with the highest average accuracy from cross-validation, respectively. Both of these selection parameters in the Blue Brain Project experiments exhibited rather mediocre performance (see Figure 4, left). This suggests that different networks and binary dynamics on them may require experimentation with a collection of selection (and feature) parameters, in order to optimise the classification accuracy.

DISCUSSION

Closed neighbourhood:
In a graph G, the closed neighbourhood of a vertex v refers to the set consisting of v and all its immediate neighbours, as well as the subgraph that this set induces in G.

In this paper we examined the concept of a closed neighbourhood in relation to the classification of binary dynamics on a digraph. Regardless of what the source of the binary dynamics is, but with the assumption that it is given in a time series of labelled instantiations, we ask how can the dynamics be read off and classified. In the context of neuroscience, which is our primary motivation for this study, this is a question on the boundary between computational neuroscience and machine learning. Our methods provide a method of addressing this question.

We proposed a methodology that will take as input binary dynamics on a digraph and produce a vector summary of the dynamics by means of combinatorial and/or topological parameters of a relatively small number of neighbourhoods. Using this methodology we experimented with a dataset implemented on the Blue Brain Project reconstruction of the neocortical column of a rat, and on an artificial neural network with random underlying graph implemented on the NEST simulator. In both cases the vector summaries were then run through a support vector machine algorithm that was able to achieve a classification accuracy of up to 88% for the Blue Brain Project data and up to 81% for the NEST data.

We used the same parameters both for selecting neighbourhoods and for the creation of feature vectors. We saw that certain spectral graph parameters used as selection parameters perform significantly better than more classical parameters such as degree and clustering coefficients. We also observed that the parameters that performed best as feature parameters were the simplest ones, namely *size* and *Euler characteristic*. Comparison to randomly selected neighbourhoods showed that the methodology works reasonably well even without selecting the neighbourhoods in an informed way, but that neighbourhoods selected in a way informed by graph parameters gives in general a better performance with a much smaller error range.

Our aim was to demonstrate that certain selections of subgraphs, informed by objective structural parameters, carry enough information to allow classification of noisy signals in a network of spiking neurons. In this article the subgraphs selected are closed neighbourhoods, and the selection criteria are our chosen selection parameters. We did not, however show, or attempt to demonstrate, that the use of neighbourhoods as a concept, or graph parameters as a selection mechanism, are the best methodology. The same techniques could be applied to

other subgraph selections and other vectorisation methods, which can be analysed by our pipeline with relatively small modifications.

Another aspect of our ideas that was not exploited at all in this project is the use of more than a single graph parameter in the selection procedure. We did show that different parameters are distributed differently in the Blue Brain Project graph, and hence one may hypothesise that optimising neighbourhood selection by two or more parameters may give improved classification accuracy.

As our aim was not to obtain the best classification, but rather to provide a good methodology for ingesting binary dynamics on a digraph and a producing machine learning digestible data stream, we did not experiment with other more sophisticated machine learning algorithms. It is conceivable that doing so may produce even better classification accuracy than what is achieved here.

Finally, our approach is closely related to graph neural networks where convolution is performed by aggregating information from neighbourhoods, that is, for every vertex, features are learned from all the adjacent vertices. The pipeline presented in this paper also takes as input sequences of neural firings and sequences of neuron assemblies that turn the firing patterns into feature values. The interaction of our work and the modelling perspectives from graph neural networks and sequence-to-sequence learning might thus pose an interesting future research question.

METHODS

Mathematical Concepts and Definitions

We introduce the basic concepts and notation that are used throughout this article. By a *digraph* we mean a *finite, directed simple graph*, that is, where reciprocal edges between a pair of vertices are allowed, but multiple edges in the same orientation between a fixed pair of vertices and self-loops are not allowed.

Topology is the study of topological spaces—a vast generalisation of geometric objects. In this paper we only consider spaces that are built out of simplices. Simplices occur in any dimension $n \geq 0$, where a 0-simplex is a point, a 1-simplex is a line segment, a 2-simplex is a triangle, a 3-simplex a tetrahedron, and so forth in higher dimensions. Simplices can be glued together to form a topological space. A good survey for this material intended primarily for readers with a neuroscience background can be found in the Materials and Methods section of M. W. Reimann et al. (2017).

We now describe a general setup that associates a family of topological objects with a digraph. A particular case of this setup is the main object of study in this paper.

Definition 1. *A topological operator on digraphs is an algorithm that associates with a digraph $\mathcal{G}$ a topological space $\Gamma(\mathcal{G})$, such that if $\mathcal{H} \subseteq \mathcal{G}$ is a subgraph then $\Gamma(\mathcal{H}) \subseteq \Gamma(\mathcal{G})$ as a closed subspace.*

That is, a topological operator on digraphs is a functor from the category of digraphs and digraph inclusions to the category of topological spaces and inclusions. The flag complex of $\mathcal{G}$ (ignoring orientation), the directed flag complex (Lütgehetmann, Govc, Smith, & Levi, 2020), and the flag tournaplex (Govc et al., 2021) are examples of such operators.

Definition 2. *Let $\mathcal{G} = (V, E)$ be a digraph, and let $v_0 \in V$ be any vertex.*

- *The neighbours of v_0 in $\mathcal{G}$ are all vertices $v_0 \neq v \in V$ that are incident to v_0.*

Topological space:
A generalisation of a geometric object. In this article topological spaces are objects built out of simplices.

Simplex (plural: simplices):
A basic building block for certain types of geometric objects. A 0-simplex is a point, a 1-simplex is a line segment, a 2-simplex is a triangle, and so on for all dimensions.
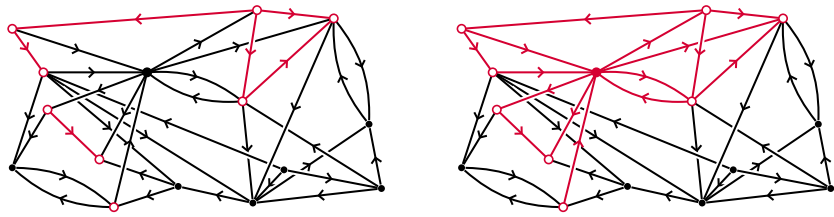
Figure 12. An open neighbourhood (left) and a closed neighbourhood (right) in a digraph, marked in red, with its central vertex marked solid colour.

- *The* open neighbourhood of v_0 is the subgraph of $\mathcal{G}$ induced by the neighbours of v_0 in $\mathcal{G}$. The closed neighbourhood of v_0 in $\mathcal{G}$ is the subgraph induced by the neighbours of v_0 and v_0 itself.

We denote the open and closed neighbourhoods of v_0 in $\mathcal{G}$ by $N^{\circ}_{\mathcal{G}}(v_0)$ and $N_{\mathcal{G}}(v_0)$, respectively (Figure 12). More generally:

- Let $S \subseteq V$ be a subset of vertices. Then $N^{\circ}_{\mathcal{G}}(S)$ denotes the union of open neighbourhoods of all $v \in S$. Similarly $N_{\mathcal{G}}(S)$ is the union of all closed neighbourhoods of vertices $v \in S$.

Notice that if $S = \{v_0, v_1\}$, and v_0 and v_1 are incident in $\mathcal{G}$, then $N^{\circ}_{\mathcal{G}}(S) = N_{\mathcal{G}}(S)$. In this article we will mostly consider closed neighbourhoods. Neighbourhoods are also used in the paper M. Reimann et al. (2021), which is closely related to this article.

Terminology 1. Let $\mathcal{G}$ be a digraph and let S be a subset of vertices in $\mathcal{G}$. Unless explicitly stated otherwise, we shall from now on refer to the closed neighbourhood of S in $\mathcal{G}$ simply as the *neighbourhood* of S in $\mathcal{G}$. In the case where S contains a single vertex v_0, we will refer to v_0 as the *centre* of $N_{\mathcal{G}}(v_0)$.

The topological operator we consider in this article is the directed flag complex of a digraph which we recall next. See Figure 2 for an example.

Definition 3. A *directed n-clique* is a digraph, whose underlying undirected graph is an n-clique, and such that the orientation of its edges determines a linear order on its vertices. An *ordered simplicial complex* is a collection X of finite ordered sets that is closed under subsets. The n-simplices of an ordered simplicial complex X are the sets of cardinality $n + 1$. If $\mathcal{G}$ is a digraph, then the *directed flag complex* associated to $\mathcal{G}$ is the ordered simplicial complex whose n-simplices are the directed $(n + 1)$-cliques in $\mathcal{G}$. We denote the directed flag complex of a digraph $\mathcal{G}$ by $|\mathcal{G}|$.

Encoding Binary Dynamics on Neighbourhoods

We now describe our approach to classification of binary dynamics on a graph in general terms.

Definition 4. Let $\mathcal{G} = (V, E)$ be a graph (directed or undirected). A *binary state* on $\mathcal{G}$ is a function $\beta : V \to \{0, 1\}$. Equivalently, a binary state on $\mathcal{G}$ is a partition of V into two disjoint subsets that correspond to $\beta^{-1}(0)$ and $\beta^{-1}(1)$, or alternatively as a choice of an element of the power set $\mathcal{P}(V)$ of V. A *binary dynamics* on $\mathcal{G}$ is a function $B : \mathbb{R}_{\geq 0} \to \mathcal{P}(V)$ that satisfies the following condition:

- There is a partition of $\mathbb{R}_{\geq 0}$ into finitely many half open intervals $\{[a_i, b_i)\}_{i=1}^{P}$ for some $P \geq 1$, such that B is constant on $[a_i, b_i)$, for all $i = 1, ..., P$.

Simplicial complex:
A topological space made out of gluing simplices together.

Activity in a network of neurons, both natural and artificial, is a canonical example of a binary dynamics on a directed network.

Setup. The task we address in this section is a general classification methodology for binary dynamics functions. Namely, suppose one is given

- a set of binary dynamics functions $\{B_i \mid i \geq 1\}$ on a fixed ambient graph $\mathcal{G}$,
- a set of labels $\mathcal{L} = \{L_1, L_2, ..., L_n\}$, and
- a labelling function $L : \{B_i \mid i \geq 1\} \to \mathcal{L}$.

In addition, we operate under the assumption that *functions labeled by the same label are variants of the same event* (without specifying what the event is, or in what way its variants are similar). The aim is to produce a topological summary for each B_i in a way that will make the outcome applicable to standard machine learning algorithms. We next describe our proposed mechanism.

Creation of vector summary. Fix a graph $\mathcal{G}$ and a real-valued graph parameter Q, that is, a real-valued function taking digraphs as input and whose values are invariant under graph isomorphisms. Suppose that a set of labeled binary dynamics functions $\{B^n\}_{n=1}^{N}$ on $\mathcal{G}$ is given. Select an M-tuple $(\mathcal{H}_1, \mathcal{H}_2, ..., \mathcal{H}_M)$ of subgraphs of $\mathcal{G}$, for some fixed positive integer M.

Fix a time interval and divide it into time bins. In each bin, record the vertex set that showed the value 1, that is, was *active* at some point during that time bin. For each $1 \leq m \leq M$, restrict that set to $\mathcal{H}_m$ and record the subgraph induced by the active vertices. Apply Q to obtain a numerical M-tuple, and concatenate the vectors into a long vector, which encodes all time bins corresponding to the given dynamics.

We now describe the procedure more accurately in three steps.

I)   <u>Interval partition uniformising.</u> Fix an interval $I = [a, b] \subset \mathbb{R}_{\geq 0}$ and a positive integer K. Let $\Delta = \frac{b-a}{K}$. For $1 \leq k \leq K$, let I_k denote the subinterval

$$I_k \overset{\text{def}}{=} [a + (k-1)\Delta, a + k\Delta] \subseteq [a, b].$$

II) <u>Subgraph extraction.</u> For $1 \leq n \leq N$ and each $1 \leq m \leq M$, let $\beta_{m,k}^{n}$ denote the binary state on $\mathcal{H}_m$ defined by

$$\beta_{m,k}^{n} \overset{\text{def}}{=} \{v \in \mathcal{H}_m \mid \exists t \in I_k, \quad \text{such that} \quad v \in B^n(t)\}.$$

Let $\mathcal{H}_{m,k}^{n} \subseteq \mathcal{H}_m$ be the subgraph induced by all vertices in the set $\beta_{m,k}^{n}$. We refer to $\mathcal{H}_{m,k}^{n}$ as the *active subgraph* of $\mathcal{H}_m$ with respect to the binary dynamics function B^n.

III) <u>Numerical featurisation.</u> For each $1 \leq n \leq N$, let $q_{m,k}^{n}$ denote the value of Q applied to $\mathcal{H}_{m,k}^{n}$. Let F^n denote the $M \times K$ matrix corresponding to the binary dynamics function B^n, that is $(F^n)_{m,k} = q_{m,k}^{n}$.

For use in standard machine learning technology such as support vector machines, we turn the output of the procedure into a single vector by column concatenation. The output of this procedure is what we refer to as a *vector summary of the collection* $\{B^n\}_{n=1}^{N}$ (Figure 3). It allows great flexibility as its outcome depends on a number of important choices:

- the ambient graph $\mathcal{G}$,
- the selection procedure of subgraphs,

- the interval *I* and the binning factor *K*, and
- the graph parameter *Q*.

All these choices may be critical to the task of classifying binary dynamics functions, as our use case shows, and have to be determined by experimentation with the data.

Selection and Feature Parameters

In this section we describe the graph parameters used in this article. Some of these parameters are well known in the literature. All of them are invariant under digraph isomorphism. The parameters presented in this section are the primary parameters used for both selection and generation of vector summaries. We chose these particular parameters either because of their prevalence in the literature, or for their strong performance as either selection or feature parameters in classification tasks. Other parameters we examined are mentioned in the Supporting Information.

Throughout this section, we let $\mathcal{G} = (V, E)$ denote a locally finite digraph (that is, such that every vertex is of finite degree). For $k \geq 1$ and $v_0 \in V$, we let $S_k(v_0)$ denote the number of directed $(k + 1)$-cliques that contain v_0. In particular $S_1(v_0) = \deg(v_0)$.

Clustering coefficients. Watts and Strogatz (1998) introduced an invariant for undirected graphs they called *clustering coefficient*. For each vertex v_0 in the graph $\mathcal{G}$, one considers the quotient of the number tv_0 of triangles in $\mathcal{G}$ that contain v_0 as a vertex by the number $\binom{\deg(v_0)}{2}$ of triangles in the complete graph on v_0 and its neighbourhood in $\mathcal{G}$. The clustering coefficient of $\mathcal{G}$ is then defined as the average across all $v_0 \in \mathcal{G}$ of that number. Clustering coefficients are used in applied graph theory as measures of segregation (Rubinov & Sporns, 2010).

Clustering coefficient for digraphs. The Watts–Strogatz clustering coefficient was generalised by Fagiolo (2007) to the case of directed graphs. Fagiolo considers for a vertex v_0 every possible 3-clique that contains v_0, and then identifies pairs of them according to the role played by v_0, as a source, a sink, or an intermediate vertex (see Figure 13A–C) and (C). Fagiolo also considers cyclical triangles at v_0 and identifies the two possible cases of such triangles (see Figure 13D). The Fagiolo clustering coefficient at v_0 is thus the quotient of the number of equivalence classes of directed triangles at v_0, denoted by $\vec{t}_{v_0}$, by the number of such classes
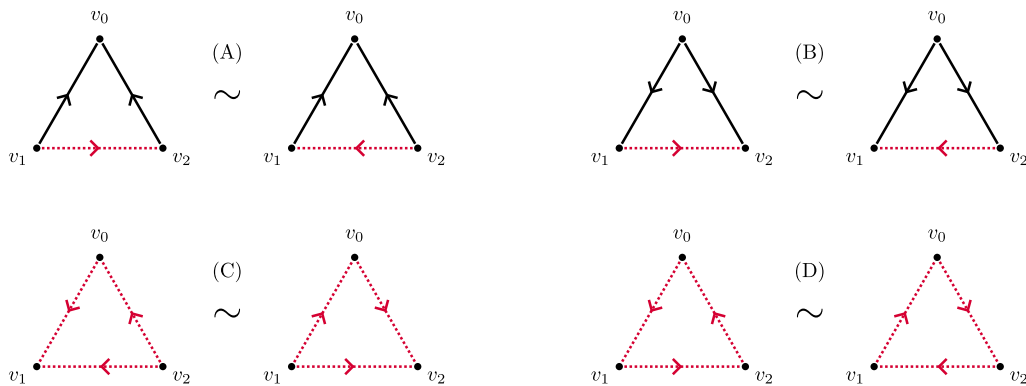

Figure 13. Eight possible directed triangles on the same three vertices. The pairs correspond to the identifications made by Fagiolo, with changes denoted by dotted edges. In the definition of the transitive clustering coefficient, the triangles in panels A, B, and C are counted individually, and those in panel D are ignored.

in the complete graph on v_0 and all its neighbours in $\mathcal{G}$. Thus, if v_0 is the *i*-th vertex in $\mathcal{G}$ with respect to some fixed ordering on the vertices, and $A = (a_{i,j})$ is the adjacency matrix for $\mathcal{G}$, then

$$\vec{t}_{v_0} \overset{\text{def}}{=} \frac{1}{2} \sum_{j,k} \left(a_{i,j} + a_{j,i}\right)\left(a_{i,k} + a_{k,i}\right)\left(a_{j,k} + a_{k,j}\right),$$

and the clustering coefficient at v_0 is defined by

$$CF(v_0) \overset{\text{def}}{=} \frac{\vec{t}_{v_0}}{\deg(v_0)(\deg(v_0) - 1) - 2\sum_j a_{i,j} a_{j,i}}.$$

Transitive clustering coefficient.   A directed 3-clique is also known in the literature as a *transitive 3-tournament*. Our variation on the clustering coefficient, the *transitive clustering coefficient* of a vertex v_0 in a digraph $\mathcal{G}$, is the quotient of the number of directed 3-cliques in $\mathcal{G}$ that contain v_0 as a vertex by the number of theoretically possible such 3-cliques.

Let $\text{ind}(v_0)$ and $\text{oud}(v_0)$ denote the in-degree and out-degree of v_0. Let I_{v_0}, O_{v_0}, and R_{v_0} denote the number of in-neighbours (that are not out-neighbours), out-neighbours (that are not in-neighbours), and reciprocal neighbours of v_0, respectively. Notice that

$$\text{ind}(v_0) = I_{v_0} + R_{v_0} \quad \text{and} \quad \text{oud}(v_0) = O_{v_0} + R_{v_0}. \tag{1}$$

We introduce our variation on Fagiolo's clustering coefficient.

Definition 5. Define the transitive clustering coefficient at v_0 by

$$CT(v_0) \overset{\text{def}}{=} \frac{S_2(v_0)}{\deg(v_0)(\deg(v_0) - 1) - (\text{ind}(v_0)\text{oud}(v_0) + R_{v_0})}.$$

A justification for the denominator in the definition is needed and is the content of the Lemma 1 in Supporting Information.

Let $A = (a_{i,j})$ denote the adjacency matrix for $\mathcal{G}$ with respect to some fixed ordering on its vertices. Then for each vertex $v_0 \in \mathcal{G}$ that is the *i*-th vertex in this ordering, $S_2(v_0)$ can be computed by the formula

$$S_2(v_0) = \sum_{j,k}\left(a_{i,j} + a_{j,i}\right)\left(a_{i,k} + a_{k,i}\right)\left(a_{j,k} + a_{k,j}\right) - a_{i,j}a_{j,k}a_{k,i} = 2\vec{t}_{v_0} - \sum_{j,k} a_{i,j}a_{j,k}a_{k,i}. \tag{2}$$

Euler characteristic and normalised Betti coefficient.   The Betti numbers of the various topological constructions one can associate to a digraph have been shown in many works to give information about structure and function in a graph. A particular example, using Blue Brain Project data is M. W. Reimann et al. (2017).

Euler characteristic.   The Euler characteristic of a complex is possibly the oldest and most useful topological parameter, and has been proven to be useful to theory and applications. In the setup of a directed flag complex (or any finite semi-simplicial set) the Euler characteristic is given as the alternating sum of simplex counts across all dimensions:

$$EC(X) \overset{\text{def}}{=} \sum_{n \geq 0} (-1)^n |X_n|,$$

where $|X_n|$ is the number of *n*-simplices in X. Alternatively, the Euler characteristic can be defined using the homology of X by

$$EC(X) \overset{\text{def}}{=} \sum_{n \geq 0} (-1)^n \dim_{\mathbb{F}}(H_n(X, \mathbb{F})),$$

where $\mathbb{F}$ is any field of coefficients. The Euler characteristic is a homotopy invariant, and can take positive or negative values according to the dominance of odd- or even-dimensional cells in the complex in question.

Normalised Betti coefficient.   The *normalised Betti coefficient* is based on a similar idea to the Euler characteristic. It is invariant under graph isomorphism, but is not a homotopy invariant. Also, unlike the Euler characteristic, it is not independent of the chosen field of coefficients. We view the normalised Betti coefficient as a measure of how "efficient" a digraph is in generating homology, without reference to any particular dimension, but with giving increasing weight to higher dimensional Betti numbers.

Let $\mathcal{G}$ be a digraph, and for each $k \geq 0$, let $s_k(\mathcal{G})$ denote the number of k-simplices in the directed flag complex $|\mathcal{G}|$. Fix some field $\mathbb{F}$. By the *Betti number* β_i of $\mathcal{G}$ we mean the dimension of the homology vector space $H_i(|\mathcal{G}|, \mathbb{F})$.

Definition 6. Let $\mathcal{G}$ be a locally finite digraph. Define the *normalised Betti coefficient* of $\mathcal{G}$ to be

$$\mathfrak{B}(\mathcal{G}) \stackrel{\text{def}}{=} \sum_{i=0}^{\infty} \frac{(i+1)\beta_i(\mathcal{G})}{s_i(\mathcal{G})}.$$

Normalised Betti coefficients can be defined by any linear combination of Betti numbers, and also in a much more general context (simplicial posets), which we did not explore. Both the Euler characteristic and the normalised Betti coefficients are invariants of digraphs, and to use them as vertex functions we consider their value on the neighbourhood of a vertex.

Size (vertex count).   The *size* of a digraph can be interpreted in a number of ways. One standard way to do so is for a fixed simplicial object associated to a digraph, one counts the number of simplices in each dimension. This will typically produce a vector of positive integers, the (euclidean) size of which one can consider as the size of the digraph. Alternatively, the simplex count in any dimension can also be considered as a measure of size. In this article we interpret size as the number of vertices in the digraph. Thus by *size* of a vertex $v_0 \in \mathcal{G}$ we mean the vertex count in $N_{\mathcal{G}}(v_0)$. When working with binary states on a digraph, neighbourhood size means the number of vertices that obtain the value 1 in $N_{\mathcal{G}}(v_0)$.

Spectral invariants.   The *spectrum* of a (real valued) square matrix or a linear operator A is the collection of its eigenvalues. *Spectral graph theory* is the study of spectra of matrices associated to graphs. It is a well developed part of combinatorial graph theory and one that finds many applications in network theory, computer science, chemistry, and many other subjects (see a collection of web links on Applications of Spectral Graph Theory website: https://sites .google.com/site/spectralgraphtheory/home?authuser=0). The various versions of the Laplacian matrix associated to a graph plays a particularly important role. An interesting work relating neuroscience and the Laplacian spectrum is de Lange et al. (2014).

The *spectral gap* is generally defined as the difference between the two largest moduli of eigenvalues of A. In some situations, for instance in the case of the Laplacian matrix, the spectral gap is defined to be the smallest modulus of nonzero eigenvalues. Given a matrix and its spectrum, either number can be computed. As a standard in this article, spectral gaps are considered as the first type described above, except for the Chung Laplacian spectrum, where the spectral gap is defined to be the value of the minimal nonzero eigenvalue. However, in several cases we considered both options. To emphasise which option is taken we decorated the parameter codes from Table 1 with a subscript "high" (referring to the difference between

the two largest moduli) or "low" (referring to the smallest modulus of a nonzero eigenvalue). For example, Figures 7, 8, and 9 have **bls$_{\text{low}}$** as a parameter, indicating the lowest nonzero value in the Bauer Laplacian spectrum (that is, the minimal nonzero eigenvalue of the Bauer Laplacian matrix). Another variant of the standard concepts of spectra is what we call the *reversed* spectral gap (Definitions 7 and 9).

Yet another common invariant we considered is the *spectral radius*, which is the largest eigenvalue modulus of the matrix in question. We consider here four matrices associated to digraphs: the adjacency matrix, the transition probability matrix, the Chung Laplacian, and the Bauer Laplacian, with details to follow.

The adjacency and transition probability matrices. Let $\mathcal{G} = (V, E)$ be a weighted directed graph with weights $w_{u,v}$ on the edge (u, v) in $\mathcal{G}$, where $w_{u,v} = 0$ if (u, v) is not an edge in $\mathcal{G}$. Let $W_{\mathcal{G}} = (w_{u,v})$ denote the weighted adjacency matrix of $\mathcal{G}$. Let oud(u) denote the out-degree of a vertex u. The *transition probability matrix* for $\mathcal{G}$ is defined, up to an ordering of the vertex set V, to be the matrix $P_{\mathcal{G}}$, with

$$P_{\mathcal{G}} \stackrel{\text{def}}{=} D_{\text{out}}^{-1}(\mathcal{G}) \cdot W_{\mathcal{G}}, \tag{3}$$

where $D^{-1}(\mathcal{G})$ is the diagonal matrix with the reciprocal out-degree $1/\text{out}(u)$ as the (u, u) entry, if $\text{out}(u) \neq 0$, else the (u, u) entry is 0.

Definition 7. Let $\mathcal{G}$ be a digraph with adjacency matrix $A_{\mathcal{G}}$ and transition probability matrix $P_{\mathcal{G}}$. The *adjacency spectral gap* and the *transition probability spectral gap* of $\mathcal{G}$ are defined in each case to be the difference between the two largest moduli of its eigenvalues.

If we replace in the definition of $P_{\mathcal{G}}$ the matrix $D_{\text{out}}(\mathcal{G})$ by $D_{\text{in}}(\mathcal{G})$ of in-degrees, we obtain a variant of the transition probability matrix, which we denote by $P_{\mathcal{G}}^{\text{rev}}$, and its spectral gap is referred to as the *reversed transition probability spectral gap*.

For our specific application we considered the ordinary (as opposed to weighted) adjacency matrix, namely, where all weights $w_{u,v}$ are binary. We considered as parameters the spectral radius of the adjacency and transition probability matrices.

The Chung Laplacian. Chung defined the directed Laplacian for a weighted directed graph in Chung (2005). The Perron–Frobenius theorem (Horn & Johnson, 1990) states that any real valued irreducible square matrix M with nonnegative entries admits a unique eigenvector, all of whose entries are positive. The eigenvalue for this eigenvector is routinely denoted by ρ, and it is an upper bound for any other eigenvalue of M.

If $\mathcal{G}$ is strongly connected (that is, when there is a directed path between any two vertices in $\mathcal{G}$), then its transition probability matrix is irreducible, and hence satisfies the conditions of the Perron–Frobenius theorem. Thus $P_{\mathcal{G}}$ has an eigenvector, all of whose entries are positive. The *Perron vector* is such an eigenvector ϕ that is normalised in the sense that $\Sigma_{v \in V} \phi(v) = 1$. Let Φ denote the diagonal matrix with the v-th diagonal entry given by $\phi(v)$, and let P denote the transition probability matrix $P_{\mathcal{G}}$.

Definition 8. Let $\mathcal{G}$ be a strongly connected digraph. The *Chung Laplacian matrix* for $\mathcal{G}$ is defined by

$$\mathcal{L} \stackrel{\text{def}}{=} I - \frac{\Phi^{\frac{1}{2}} P \Phi^{-\frac{1}{2}} + \Phi^{-\frac{1}{2}} P^* \Phi^{\frac{1}{2}}}{2}, \tag{4}$$

where P^* denotes the Hermitian transpose of a matrix P. The *Chung Laplacian spectral gap* λ for a digraph $\mathcal{G}$ is defined to be the smallest nonzero eigenvalue of the Laplacian matrix.

The Chung Laplacian spectral gap λ of a strongly connected digraph $\mathcal{G}$ is related to the spectrum of its transition probability matrix P by Chung (2005, Theorem 4.3), which states that the inequalities

$$\min_{i \neq 0} \{1 - |\rho_i|\} \leq \lambda \leq \min_{i \neq 0} \{1 - \text{Re}(\rho_i)\} \tag{5}$$

hold, where the minima are taken over all eigenvalues of P. The theory in Chung (2005) applies for strongly connected graphs and we therefore defined the Laplacian spectral gap of a neighbourhood to be that of its largest strongly connected component.

We use the spectral gap of the Chung Laplacian for the largest strongly connected component of a neighbourhood as a selection parameter. When used as a feature parameter we consider the spectral gap of the largest strongly connected component of the active subgraph of the neighbourhood. We also use the spectral radius of the Chung Laplacian, both as selection and feature parameter.

The Bauer Laplacian.   The requirement that $\mathcal{G}$ is strongly connected is a nontrivial restriction, but it is required in order to guarantee that the eigenvalues are real. An alternative definition of a Laplacian matrix for directed graphs that does not require strong connectivity was introduced in Bauer (2012). Let $C(V)$ denote the vector space of complex valued functions on V. The Bauer Laplacian for $\mathcal{G}$ is the transformation $\Delta_\mathcal{G} : C(V) \to C(V)$ defined by

$$\Delta_\mathcal{G}(f)(v) \stackrel{\text{def}}{=} \begin{cases} f(v) - \dfrac{1}{\text{ind}(v)} \Sigma_v w_{v,u} f(u), & \text{if } \text{ind}(v) \neq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

If $\text{ind}(v) \neq 0$ for all $v \in V$, then $\Delta_\mathcal{G}$ corresponds to the matrix $\Delta_\mathcal{G} = I - D_{\text{in}}^{-1}(\mathcal{G}) \cdot W_\mathcal{G}$ where $D_{\text{in}}^{-1}(\mathcal{G})$ is defined analogously to $D_{\text{out}}^{-1}(\mathcal{G})$ in Definition 7, and $W_\mathcal{G}$ is the weighted adjacency matrix. In our case W is again taken to be the ordinary binary adjacency matrix.

Definition 9. The *Bauer Laplacian spectral gap* is the difference between the two largest moduli of eigenvalues in the spectrum.

We also considered the spectral radius of the Bauer Laplacian. Both are used as selection as well as feature parameters. If we replace in the definition $D_{\text{in}}(\mathcal{G})$ by $D_{\text{out}}(\mathcal{G})$, we obtain a matrix $\Delta_\mathcal{G}^{\text{rev}}$, whose spectral gap we refer to as the *reversed Bauer Laplacian spectral gap*.

ACKNOWLEDGMENTS

SUPPORTING INFORMATION

Supporting information for this article is available at https://doi.org/10.1162/netn_a_00228.

AUTHOR CONTRIBUTIONS

Ran Levi: Conceptualization; Funding acquisition; Methodology; Supervision; Writing – original draft; Writing – review & editing. Pedro Conceição: Conceptualization; Data curation; Formal analysis; Investigation; Validation. Dejan Govc: Data curation; Formal analysis; Investigation; Validation. Jānis Lazovskis: Conceptualization; Data curation; Formal analysis; Investigation; Software; Validation; Visualization; Writing – original draft. Henri Riihimäki:

Conceptualization; Data curation; Formal analysis; Investigation; Software; Validation; Writing – original draft. Jason P. Smith: Conceptualization; Data curation; Formal analysis; Investigation; Software; Validation; Writing – original draft.

REFERENCES

Babichev, A., Ji, D., Mémoli, F., & Dabaghian, Y. (2016). A topological model of the hippocampal cell assembly network. *Frontiers in Systems Neuroscience*, *10*, 50. https://doi.org/10.3389/fncom.2016.00050, PubMed: 27313527

Bargmann, C., & Marder, E. (2013). From the connectome to brain function. *Nature Methods*, *10*(6). https://doi.org/10.1038/nmeth.2451, PubMed: 23866325

Bauer, F. (2012). Normalized graph laplacians for directed graphs. *Linear Algebra and its Applications*, *436*, 4193–4222. https://doi.org/10.1016/j.laa.2012.01.020

Chambers, B., & MacLean, J. (2016). Higher-order synaptic interactions coordinate dynamics in recurrent networks. *PLoS Computational Biology*, *12*(8). https://doi.org/10.1371/journal.pcbi.1005078, PubMed: 27542093

Chung, F. (2005). Laplacians and the cheeger inequality for directed graphs. *Annals of Combinatorics*, *9*, 1–19. https://doi.org/10.1007/s00026-005-0237-z

Churchland, A., & Abbott, L. (2016). Conceptual and technical advances define a key moment for theoretical neuroscience. *Nature Neuroscience*, *19*(3). https://doi.org/10.1038/nn.4255, PubMed: 26906500

Cunningham, J., & Yu, B. (2014). Dimensionality reduction for large-scale neural recordings. *Nature Neuroscience*, *17*(1). https://doi.org/10.1038/nn.3776, PubMed: 25151264

Curto, C., & Itskov, V. (2008). Cell groups reveal structure of stimulus space. *PLoS Computational Biology*, *4*(10). https://doi.org/10.1371/journal.pcbi.1000205, PubMed: 18974826

Curto, C., & Morrison, K. (2019). Relating network connectivity to dynamics: Opportunities and challenges for theoretical neuroscience. *Current Opinion in Neurobiology*, *58*, 11–20. https://doi.org/10.1016/j.conb.2019.06.003, PubMed: 31319287

de Lange, S. C., de Reus, M. A., & van den Heuvel, M. P. (2014). The Laplacian spectrum of neural networks. *Frontiers in Computational Neuroscience*, *7*, 189. https://doi.org/10.3389/fncom.2013.00189, PubMed: 24454286

Fagiolo, G. (2007). Clustering in complex directed networks. *Physical Review E*, *76*, 026107. https://doi.org/10.1103/PhysRevE.76.026107, PubMed: 17930104

Fan, X., & Markram, H. (2019). A brief history of simulation neuroscience. *Frontiers in Neuroinformatics*, *13*, 32. https://doi.org/10.3389/fninf.2019.00032, PubMed: 31133838

Gleeson, J. P. (2008). Cascades on correlated and modular random networks. *Physical Review E*, *77*. https://doi.org/10.1103/PhysRevE.77.046117, PubMed: 18517700

Govc, D., Levi, R., & Smith, J. (2021). Complexes of tournaments, directionality filtrations and persistent homology. *Journal of Applied and Computational Topology*, *5*, 313–337. https://doi.org/10.1007/s41468-021-00068-0

Horn, R., & Johnson, C. (1990). *Matrix analysis* (2nd ed.). Cambridge, UK: Cambridge University Press.

Jordan, J., Mørk, H., Vennemo, S. B., Terhorst, D., Peyser, A., Ippen, T., … Plesser, H. E. (2019). *Nest 2.18.0*. Zenodo. https://doi.org/10.5281/zenodo.2605422

Kartun-Giles, A. P., & Bianconi, G. (2019). Beyond the clustering coefficient: A topological analysis of node neighbourhoods in complex networks. *Chaos, Solitons & Fractals: X*, *1*, 100004. https://doi.org/10.1016/j.csfx.2019.100004

Lazovskis, J. (2021). Neurotop-nest, GitHub, https://github.com/jlazovskis/neurotop-nest/

Lütgehetmann, D., Govc, D., Smith, J. P., & Levi, R. (2020). Computing persistent homology of directed flag complexes. *Algorithms*, *13*(1). https://doi.org/10.3390/a13010019

Markram, H., Muller, E., Ramaswamy, S., Reimann, M. W., Abdellah, M., Sanchez, C. A., … Schürmann, F. (2015). Reconstruction and simulation of neocortical microcircuitry. *Cell*, *163*, 456–492. https://doi.org/10.1016/j.cell.2015.09.029, PubMed: 26451489

Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., & Alon, U. (2002). Network motifs: Simple building blocks of complex networks. *Science*, *298*, 824–827. https://doi.org/10.1126/science.298.5594.824, PubMed: 12399590

Reimann, M., Riihimäki, H., Smith, J. P., Lazovskis, J., Pokorny, C., & Levi, R. (2021). Topology of synaptic connectivity constrains neuronal stimulus representation, predicting two complementary coding strategies. *PLoS ONE*, *17*(1), e0261702. https://doi.org/10.1371/journal.pone.0261702, PubMed: 35020728

Reimann, M. W., Nolte, M., Scolamiero, M., Turner, K., Perin, R., Chindemi, G., … Markram, H. (2017). Cliques of neurons bound into cavities provide a missing link between structure and function. *Frontiers in Computational Neuroscience*, *11*, 48. https://doi.org/10.3389/fncom.2017.00048, PubMed: 28659782

Rubinov, M., & Sporns, O. (2010). Complex network measures of brain connectivity: Uses and interpretations. *NeuroImage*, *52*, 1059–1069. https://doi.org/10.1016/j.neuroimage.2009.10.003, PubMed: 19819337

Samuelsson, B., & Socolar, J. E. S (2006). Exhaustive percolation on random networks. *Physical Review E*, *74*. https://doi.org/10.1103/PhysRevE.74.036113, PubMed: 17025714

Stein, R., Gossen, E., & Jones, K. (2005). Neuronal variability: Noise or part of the signal? *Nature Reviews Neuroscience*, *6*, 389–397. https://doi.org/10.1038/nrn1668, PubMed: 15861181

Watts, D., & Strogatz, S. (1998). Collective dynamics of 'small-world' networks. *Nature*, *393*, 440–442. https://doi.org/10.1038/30918, PubMed: 9623998