Neural Networks 157 (2023) 160–175

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Inferring the location of neurons within an artificial network from their activity

Alexander J. Dyer [a,*], Lewis D. Griffin [b]

[a] Research Department of Cell and Developmental Biology, University College London, UK
[b] Department of Computer Science, University College London, UK

ARTICLE INFO

Article history:
Received 4 May 2022
Received in revised form 14 August 2022
Accepted 11 October 2022
Available online 20 October 2022

Keywords:
Artificial neural networks
Network inference
Supervised learning
Correlation

ABSTRACT

Inferring the connectivity of biological neural networks from neural activation data is an open problem. We propose that the analogous problem in artificial neural networks is more amenable to study and may illuminate the biological case. Here, we study the specific problem of assigning artificial neurons to locations in a network of known architecture, specifically the LeNet image classifier. We evaluate a supervised learning approach based on features derived from the eigenvectors of the activation correlation matrix. Experiments highlighted that for an image dataset to be effective for accurate localisation, it should fully activate the network and contain minimal confounding correlations. No single image dataset was found that resulted in perfect assignment, however perfect assignment was achieved using a concatenation of features from multiple image datasets.

1. Introduction

A struck planar domain ('drum') resonates at a spectrum of frequencies determined by its shape. Kac (1966) famously asked ("Can one hear the shape of a drum?") whether that spectrum uniquely determine the shape. The question was answered in the negative by Gordon et al. (1992), by constructing a pair of nonisometric but isospectral shapes. However, in later work the authors comment that the existence of such pairs is atypical and that in some sense "most" geometrical shapes are spectrally "solitary" (Gordon & Webb, 1996). Furthermore, in any case the area of the drumhead can be inferred using Weyl's formula. It is in this spirit of systematically exploring an inverse problem that we wish to approach the problem of network inference, by which we mean the inference of the connectivity and weights of a network (analogous to the shape of the drum) from activity at its nodes (analogous to the sound of the drum).

Inferring the connectivity of biological neural networks (BNNs) using activation data recorded from the nodes is an open problem in neuroscience. Improving on the performance of existing inference algorithms would be of great practical and theoretical utility. This paper begins by proposing a research agenda for developing better inference algorithms through a series of intermediary inference tasks performed on artificial neural networks (ANNs). We then analyse and successfully complete the first of these intermediary tasks. In doing so we show that the activation data of ANNs contains accessible information about the network connectivity and hope to demonstrate the benefits of the artificial setting for algorithm development.

In the biological setting, the full network inference task is to use high-dimensional recordings of neural activity to infer the connectivity (or "wiring diagram") of the corresponding network and the weights of its connections. We take it for granted that algorithms for performing this task would be useful to the neuroscience community. Although various computational methods for network inference have been developed (see Magrans de Abril et al., 2018, for a review) none are able to achieve full reconstruction.

Instead, we propose to study the BNN task in the controlled, synthetic setting of ANNs. The advantages of the synthetic setting are that: (i) ground truth connectivity is known and manipulatable, (ii) the effect of noise, quantity and completeness of activation data can be readily studied, (iii) the role of the stimulus set driving the activations can be easily investigated, and (iv) results are deterministic and reproducible. Of course, even if effective algorithms for the ANN task can be developed there will no doubt be additional problems to solve with real biological data, but until we can perform well in the synthetic domain it is implausible that we can perform well in the biological one.

Our goals are:

(1) To present a research agenda in which a series of intermediary (artificial) inference task are to be attempted as a strategy for approaching the full (biological) network inference task.

* Corresponding author.
E-mail address: alexander.dyer.18@ucl.ac.uk (A.J. Dyer).

https://doi.org/10.1016/j.neunet.2022.10.012

(2) To demonstrate the utility of performing inference tasks in the ANN setting.
(3) To demonstrate that the activity of ANNs contains recoverable information about the structure of the network. Our successful performance on the chosen inference task achieves this aim and represents a first step in the proposed research agenda.

The rest of this paper is organised as follows. In Section 2 we present our proposed research agenda. In 3 we provide a brief background to the network inference task in the BNN setting and review existing methods in a general setting. In 4 we characterise the ANN we experiment with, first describing its architecture, then investigating its activations in response to images and then the correlations between those activations. In 5 we describe our methods including the assignment procedure and performance measure. In 6 we discuss the results of our various experiments including how we achieved an assignment with zero error, and in 7 we conclude.

2. Intermediary inference tasks

We propose to incrementally approach the full network inference task in the synthetic setting through a sequence of intermediary inference challenges. Methods used to solve one task may not generalise to another task, and so a given task's solution may not provide direct insight into the full inference problem; but, if an intermediary task cannot be solved then it is implausible that the hardest task of full network inference can be achieved. It may be the case however that know-how developed in pursuing each task improves our performance in latter tasks, but this is not our primary motivation.

Having chosen ANNs as our object of study there are then a range of possibilities for inference tasks depending on what information we have access to and what we are trying to infer. We separate the network information into its:

- Inputs (I) — the stimulus set.
- Responses (R) — the values that each neuron takes, paired with the stimulus that triggered them.
- Architecture (A) — the connectivity information.
- Weights (W) — the weight of each connection (edge) in the network. The amount of information available about W is limited by the amount given about A.
- Location (L) — where in the network each neuron is located, specified in terms of the network architecture. The amount of information available about L is limited by the amount given about A.

Information about each of I, R, A, W and L may be total, partial or missing. Inferring A-W-L given only I-R is the full network inference task. Intermediary tasks, for example inferring W-L given partial access to A and full access to I-R, are all special cases of the full network inference task and as such should be possible if the full inference task is possible.

In Fig. 1 we suggest a sequence of milestone tasks of increasing challenge. Each is named based on the primary type of information to be inferred, but we also specify partial inference of the remaining data types. Since we are working in the ANN setting, we always assume access to the input/stimulus information (I).

The first panel in Fig. 1 illustrates the *response localisation* inference task explored in this paper. This task requires localisation of network responses, given other information about the network. We consider the specific case in which we have full I, R and A information, and partial L information, and must infer the remaining L — we are neither given, nor attempt to infer W information. This is much simpler than inferring A — N neurons can be localised to N locations in N! ways, whereas N neurons can connected into 2^{N^2} possible architectures.

Once response localisation has been shown to be possible, we suggest *weight synchronisation* as the next task. As shown in Fig. 1 this involves inferring all the weight information given full access to I, R and A and some L information. Note that we assume algorithms developed in previous tasks can be usefully leveraged in future tasks, in this case inferring some of the missing L information to improve inference of the missing W.

Next, we suggest attempting *architecture inference*: given partial architecture information (e.g. the number and types of layers) inferring the remainder of the architecture (the specifics of how each neuron is connected).

Finally, having achieved good performance in the previous tasks we suggest attempting full *network inference*, by which we mean inferring some of A, W and L given I and partial information on R. This represents a more realistic analogue of the data available in biological experiments, in particular the fact that we only have responses (R) from a fraction of the network.

Having proposed a sequence of inference tasks for approaching full network inference in ANNs, in the next section we provide some background to the network inference task in the biological setting and subsequently network inference methods in general.

3. Background

3.1. Networks of biological neurons

The anatomical *connectivity* of neural elements is crucial to their joint operation. This connectivity can be represented using *graphs* of nodes connected by edges and can be applied at different scales of abstraction. At fine scale, nodes represent individual neurons and edges represent synaptic connections. At coarser scale, nodes can represent collections of neurons defined by volume elements, anatomical region (cortical area or layer) and/or sub-type, and edges can represent, for example, nerve tracts between regions.

Nodes in a brain network may have experimentally collected activation data associated with them. Dependent on scale, such data can represent the firing rate of individual neurons or the aggregated firing of a group of neurons such as those present in a volume element. Activation data can take the form of time series following a stimulus, or as a vector of firing rates for different elements of a stimulus set. *Structural* methods determine connectivity without using such activation data, by directly detecting the presences of edges; while *functional* methods infer the presence of edges from nodal activation data (Friston, 2011).

In brain networks, the gold standard structural method is *Electron microscopy* (EM) which can directly image synaptic connections, but can only be applied ex vivo and on tiny volumes of neural tissue (Bae et al., 2021; Helmstaedter, 2013; Shapson-Coe et al., 2021). *Viral tracing* (Nectow & Nestler, 2020; Saleeba et al., 2019) and *Polarised Light Imaging* (Larsen et al., 2007) of gross histological sections, can discover connectivity over much longer distances, but are similarly ex vivo. *Diffusion MRI* infers nerve tracts connecting brain regions entirely noninvasively, but is limited to millimetre scale spatial resolution which is much coarser than individual fibres (Novikov et al., 2019).

A range of methods exist for measuring activation data suitable for inferring connectivity: for example at fine scale *implanted microelectrodes* (Steinmetz et al., 2021); at medium scale *optical imaging* (Fan et al., 2019); and at coarse scale *functional MRI* (Lurie et al., 2020). Given activation data, connectivity is inferred from correlated (or, more generally, statistically dependent) activity. This is a difficult inference, as neural activations can be correlated even when the neurons are not directly anatomically

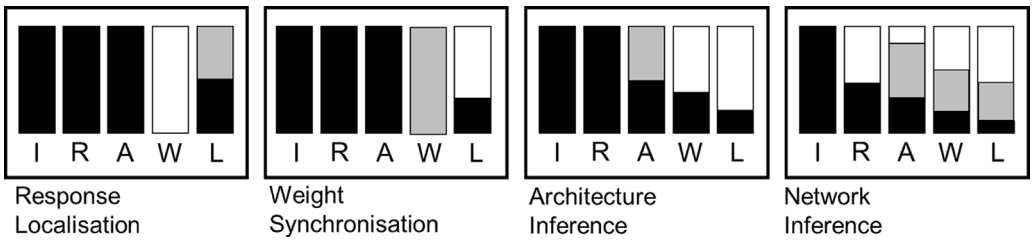

Fig. 1. Suggested sequence of intermediary inference tasks to be performed in the artificial neural network setting. Each bar represents the amount of a given type of network information, where the letters stand for (I) input, (R) responses, (A) architecture, (W) weights and (L) locations. Black represents given information, white missing, and grey to be inferred. The first panel depicts the task completed in this work.

connected. On top of this, it may require very specific stimuli to reveal the activation dependence between neurons which are anatomically connected.

Comparing structural and functional methods for determining connectivity at fine scale, we conclude that anatomical imaging is challenging because of the *number* of such connections (orders of magnitude more than the number of neurons) and the small size of synapses. Improving anatomical imaging methods so that they can achieve sufficiently fine resolution to resolve connections *and* a sufficiently large field-of-view to measure a significant network may be very difficult to achieve. In contrast, functional methods require only measurements of neuron activity. The quality of the inferred connectivity is then limited by the range and number of the stimuli used to drive the activations, the noise in the measurement of activations, and by the algorithms used to make the inference.

3.2. Methods for network inference

In the domain of network science, methods have been developed to characterise network connectivity statistically (Costa et al., 2007; Newman, 2003), for example by the distribution of the node degrees. These methods have been used to study the networks associated with, for example: rumour spreading in social networks (Bakshy et al., 2012), neural activity in schizophrenic patients (van den Heuvel et al., 2013), and the structure of human language (Solé et al., 2010).

These methods are not applicable when the network topology is unknown, but for such scenarios methods have been proposed to infer the graph from the available data (Dong et al., 2019; Timme & Casadiego, 2014). In general graph inference is an ill-posed problem, with many possible graphs consistent with the data. In this context there are several ways one can proceed, depending on whether none or some of the network is already known.

When none of the network is known, graph inference methods can be classified as either model-free or model-based. In a model-free approach the network is inferred one edge at a time depending on pairwise measures e.g. an edge is deemed present if and only if the Pearson's correlation of nodal activity exceeds a specified threshold. Disambiguating direct from indirect correlations is challenging and such methods are sensitive to noise (Feizi et al., 2013). Brain networks and gene regulatory networks, where the edges are difficult to observe reliably, are natural settings for this model-free approach (Pernice et al., 2011).

Alternatively, one can assume an underlying model that governs the nodal dynamics, constraining the space of possible graphs. Model-based methods can be separated into three categories: statistical, physically-motivated and graph signal models (Dong et al., 2019). Statistical models assume that the node features represent a set of random variables and graph inference corresponds to learning a factorisation of the joint probability, using regularisation methods like graphical lasso (Friedman et al., 2008). Physically-motivated models assume a physical process, such as infection spread, which determines the nodal dynamics. Graph inference then involves finding the topology that best explains the observed data given the underlying physics. In graph signal models, node features are explicitly considered as signals supported by the underlying graph structure. The eigenvectors of the graph adjacency matrix can be used as a basis to represent the signal, just as Fourier components can be used as a basis for a signal over a Euclidean domain. This approach potentially allows smoothness of the signal over the graph to be used as a regularising constraint to improve graph inference (Dong et al., 2019; Mateos et al., 2019; Shuman et al., 2012).

When a large fraction of the network topology is already known, one can predict missing/future/spurious links by posing the problem as a binary classification task, deciding for each possible edge whether or not it is present in the graph. Features to support these decisions can be derived directly from the known topology (e.g. node degree), and/or from the data on the nodes (e.g. 'likes' of a user in a social network, or correlation between time series arising from dynamical processes across the graph) (Hasan et al., 2006).

More recently, machine learning approaches have been applied to learn to directly predict the full graph, represented as the adjacency matrix. For example Zhang et al. (2019) use a graph neural network approach in which they learn the adjacency matrix from time series data of a dynamical system and use it to predict the dynamics at the subsequent time step, with better performance implying a more accurate adjacency matrix. As such their method learns in an unsupervised manner and is model-free. However, the method was limited to networks of a few hundred nodes.

A different scenario occurs when studying a system which permits interventions, such that the effect of targeted perturbations are informative about the underlying graph. This is the case both in the field of network tomography (Castro et al., 2004) where the context is typically telecommunication networks, and in the field of dynamical systems where nodes are often modelled as coupled oscillators (Timme & Casadiego, 2014).

The body of work most similar to ours, are the responses to the Chalearn Connectomics challenge (Orlandi et al., 2015) in which participants were tasked with inferring the connectivity of a simulated network of neurons from time-series of their activations. The simulated network was based on data recorded from neuronal cultures. The synthetic network consisted of 1000 neurons grouped into 10 clusters via preferential attachment with approximately 15,000 connections in total. Ground truth connectivity was known, and participants used methods like partial correlations, transfer entropy and deep learning for edge prediction. The organising authors went to considerable lengths to produce biologically realistic data and would presumably have preferred to use real biological data were such a dataset, with groundtruth connectivity, available to them. In contrast we are proposing to systematically explore the network inference problem in the synthetic setting of ANNs and suggest only to attempt

network inference for simulated biological networks once substantial progress has been made. The top scoring method reported by Orlandi et al. (2015) achieved an AUROC of 0.942, but considering that the actual network edges make up only 1.5% of the possible edges, this means that many more false-positive edges will be detected than true-positives. As such there is still plenty of room for improving these inference algorithms.

In conclusion, although reconstruction has been achieved with high accuracy in numerous settings, graph inference in general remains an open problem. Existing methods are limited by their sensitivity to noise, the strictness of their modelling assumptions, their reliance on labelled ground truth data, their inability to scale to larger networks or the requirement to intervene in the system. It is our opinion that ANNs, with their ease of generation, modifiability and accessibility of ground truth, provide an ideal testbed for developing network reconstruction algorithms. To the best of our knowledge no prior work has attempted to infer the graph topology of an ANN directly from its activity; nor has the simpler preliminary task (placing neurons within a known architecture) that we attempt here been previously studied.

4. Network characterisation

In this section we detail our approach to the response localisation task, the first inference task in our suggested research program.

Concretely, we consider an ANN with known architecture, and its activations in response to a particular input dataset. We will assume that we know the correct *training* locations of a set of *training* neurons. Our task is to assign the remaining *test* neurons correctly into the remaining *test* locations. A variety of algorithmic approaches to this task are plausible, but the one we will pursue here is based on learning classifiers each of which predict some information about the likely location of a neuron based on features computed from its activation data. Specifically, one classifier attempts to identify which layer a neuron belongs to, while another attempts to classify what the x-coordinate of the neuron is (assuming it belongs to, say, the first convolution layer), and so on. Combining the confidences from these classifiers gives a confidence for each neuron for each possible location in the network. We then compute the one-to-one assignment of test neurons to test network locations that maximises the overall confidence.

4.1. Architecture

In this section we summarise the architecture of the artificial neural network (ANN) we have chosen for this initial study, describing the patterns of its connectivity, the types of nodes present and their weights and biases.

We chose the classic ANN LeNet (LeCun et al., 1989), a 2D convolutional neural network (CNN) which takes greyscale images as input, and outputs a vector of class membership confidences for the handwritten digits 0 through to 9. LeNet is small and simple compared to many ANNs that are commonly used but does contain the three key layer types used in CNNs: convolutional, pooling and fully-connected. With only seven layers and a total of 19,694 neurons (see Fig. 1), it is small enough to allow for rapid prototyping and testing of the various steps of our response localisation algorithm. As an image classifier it is also fairly intuitive and its behaviour is somewhat interpretable.

We denote the seven layers of the network, in depth order, as: INPUT, CONVRAMP1, POOLING1, CONVRAMP2, POOLING2, LINEARRAMP and LINEARSOFT. The INPUT layer neurons correspond to the pixels of the input images and so can naturally be arranged in a 2D grid, with their activations equalling the pixel intensities. The CONVRAMP layers correspond to convolving a fixed set of weight matrices (filters) in parallel over the previous layer, with a separate *family* of neurons for each convolutional mask. After the linear convolution the resulting values are passed through the nonlinear function, $ramp\,(x) = \max(0, x)$. Both POOLING layers down-sample the activations of the preceding CONVRAMP layer by computing the maximum activation over 2×2 neurons from the same family. Each neuron in the LINEARRAMP and LINEARSOFT layers is fully-connected to all neurons in the preceding layer, and compute a linear function of their activations followed by a *ramp* or *softmax* non-linearity, respectively.

The full architecture details of LeNet are shown in Table 1. The location of a neuron in the network can be described by a 4-tuple of $(layer, family, x, y)$, with for example (3,10,5,6) referring to the POOLING1 neuron, in the 10th family, at 2D position (5,6).

The range of valid family, x and y values vary with layer, with some layers limited to just a single value for one or more parameters. The INPUT layer has only a single family with many neurons, layers from CONVRAMP1 to POOLING2 have many families with fewer neurons per family, and then LINEARRAMP has even more families but with just a single neuron per family. The trend of reducing x, y dimensionality and increasing family number reflects the local-to-global integration performed by the network.

The x and y coordinates of a neuron are ordinal values, in that similar values correspond to closer locations with the network architecture. In contrast the family coordinate, although represented as an integer, is really a categorical descriptor; since, for example, families 2 and 3 are no more related than families 2 and 12.

For both biological and artificial networks, the region of input space from which a neuron receives information is referred to as its receptive field (RF). The LeNet architecture was directly inspired by the pioneering work of Hubel & Wiesel (Hubel & Wiesel, 1962), in which they discovered neurons in the feline visual system whose RFs were a small, localised patch of the retina. They also produced a neural circuit model in which the RFs of "complex" cells deeper in the network, were composed from the RFs of "simple" cells, earlier in the network.

We consider the LeNet input pixels as modelling sensory neurons, and so we use the term RF to refer to the subset of these from which a neuron in later layers potentially receives information. Since LeNet is a feed-forward architecture with no self- or within-layer connections, the RF of a neuron is the union of the RFs of the neurons in the preceding layer that feed into it.

Finally, we mention the network weights and biases. Only edges to the CONVRAMP or LINEAR layers have weights, and only neurons in those layers have biases, and they are the only learnable parameters in the network. Distinct neurons with identical layer and x, y indices, but different family indices, connect to the same neurons in the preceding layer, so their activity varies only by virtue of the different weights and biases associated with their family.

We train the network on 60k images from the MNIST dataset of handwritten digits, the task for which LeNet was originally designed. After training, all weights fell within the range $(-1,1)$ and the resulting network achieved a classification accuracy, on unseen test data, of 99.0% compared to chance performance of 10%.

4.2. Network responses

In the previous section we described the static network architecture. We now consider the activations of the trained network, in response to input images, with a view to forming features that can be used to determine where neurons are located within the architecture.

Table 1
Architectural details. A "family" is the set of neurons in a layer connected to the previous layer with the same pattern of weights. For example, in CONVRAMP1 there are 20 convolutional filters, hence 20 families.

Layer name	Output size	Family number	Receptive field size	Activation function	Filter size	Stride
INPUT	(1,28 × 28)	1	1 × 1	–	–	–
CONVRAMP1	(20,24 × 24)	20	5 × 5	Ramp	5 × 5	(1,1)
POOLING1	(20,12 × 12)	20	6 × 6	Max Pooling	2 × 2	(2,2)
CONVRAMP2	(50,8 × 8)	50	14 × 14	Ramp	5 × 5	(1,1)
POOLING2	(50,4 × 4)	50	15 × 15	Max Pooling	2 × 2	(2,2)
LINEARRAMP	(500)	500	28 × 28	Ramp	800	–
LINEARSOFT	(10)	10	28 × 28	Softmax	500	–

Throughout the paper we will often compare results for two image datasets which we refer to as MNIST and GAUSSIAN (see Fig. 3). The images of the MNIST dataset are disjoint from those used to train the network. Their pixel values are in the range [0,1]. The images of the GAUSSIAN dataset have gaussian-distributed values in each pixel, with zero mean and unit standard deviation. The pixel values in MNIST images have significant spatial correlation, whereas the pixels values in the GAUSSIAN images are independently distributed. Unless explicitly stated otherwise we use $N_{images} = 10{,}000$ in each set.

We generate an *activation vector* (dimension $N_{neurons}$) for the network by feeding an image through it and recording the activity level of each neuron as the network progressively, layer after layer, computes its final output. In contrast, we can consider a single neuron and compute its activation value in response to a set of images, which we call its *response vector* (dimension N_{images}).

To visualise example activation vectors, we represent each vector element as a pixel, forming images by appropriately organising neurons to reveal the functional hierarchy of the network (see Fig. 4). Considering first an activation vector for an MNIST image, we see that in CONVRAMP1 we can interpret some families as implementing oriented edge detectors; while in CONVRAMP2 the spatial correspondence with the input is present but less clear, and it is more difficult to interpret the sensitivity of each family. Both POOLING1 and POOLING2 are readily interpretable as non-linearly downsampled versions of the preceding CONVRAMP layer. LINEARRAMP is seen to generate a sparse output and LINEARSOFT constrains its output to be a probability distribution; in this case only the first neuron is strongly active, indicating correct classification of the digit '0'.

The statistics of the MNIST activations are very different to those for GAUSSIAN input. In early layers the MNIST activations are sparser than the GAUSSIAN activations and in the later layers the converse is true, with the exception of the softmax layer where the network does not confidently categorise the GAUSSIAN image into a single class. For the GAUSSIAN input it is much harder to interpret a neuron's function from its activity, despite the network being unchanged.

Our chosen task is to assign a set of test neurons to their correct locations in the network LeNet based on activation data. To do so we will train a series of classifiers for each of the parameters in the location 4-tuple ($layer$, $family$, x, y). We hypothesise that response vectors, and more generally the response matrix, contain information that will support these classifications. However, since both the vectors and the matrix are high-dimensional (N_{images} and $N_{images} \times N_{neurons}$ respectively) it is likely that features based on these will be more effective for classification than the complete raw data.

Since the order of presentation of images to the network is random, there is no meaningful way to convert response vectors into images, as we did for the activation vector. Instead, we consider histograms of responses (see Fig. 5). For input neurons, whose activations are pixel values of the images, the responses can be positive or negative; whilst for all other layers the responses can only be non-negative due to the *ramp()* and *softmax()* non-linearities. For most neurons of these later layers, zero-valued responses are common, giving a left peak to the histogram; but for a small subset of neurons, zero is the response to *all* images, we refer to such neurons as *silent neurons*.

Silent neurons will be difficult to localise within the network since they are indistinguishable based on their responses, so we have studied their number and location carefully. First, we observe that MNIST input produces ∼1k silent neurons and GAUSSIAN input produces ∼2k, out of 19,694 in the network. The two sets of silents have only ∼100 neurons in common — close to what would be expected by chance if they were independent sets.

The locations of the silent neurons are not uniform across the network and the distribution differs between image datasets (see Fig. 6 for histograms of silent neuron number and proportion by layer and Fig. 7 for the positions of silent neurons in the network). We observe that there is a strong tendency for the silent neurons to occur in the same families. For example, pooling results over 25 independent trainings of LeNet (not shown), the histogram of the number of silent neurons within a particular layer is bimodally peaked at zero and at the total number of neurons within a family for that layer. Whilst families are more likely to be totally silent, or totally active rather than an intermediary level of activity, there is no bias as to which family indices are majority silent across repeated training instances.

We have chosen not to exclude these silent neurons from our analysis for simplicity and to ensure that results across different conditions are comparable. In conclusion, the number of silent neurons, and their precise location in the network varies greatly due to the stimulus set and to a lesser extent due to random factors during training.

To assess whether individual response vectors provide information useful for localising neurons we produce a scatter plot of their means and standard deviation. Fig. 8 shows that for GAUSSIAN input these moments allow good differentiation between most layers of the network, and between most families within a layer (inset b), but no differentiation by x or y coordinate within a family and layer (inset c). The lack of indication as to x or y coordinate is unsurprising given the spatially stationary statistics of the GAUSSIAN image dataset.

The distribution of response moments for MNIST images (see Fig. 8d) is very different to that for GAUSSIAN images (compare to Fig. 8c). For MNIST images there is much less layer distinguishability (not shown), very little family distinguishability (not shown), but there is substantial x–y distinguishability. This is because MNIST images are centred and so on average there is more activity in the centre of the input space than at the edges leading to higher means and standard deviations.

For GAUSSIAN input the mean and standard deviation of response vectors separate the layers and families because the network connectivity and weights associated with these structures is what has caused the variation in the moments. In contrast, for MNIST input the variation in the moments is *also* caused by

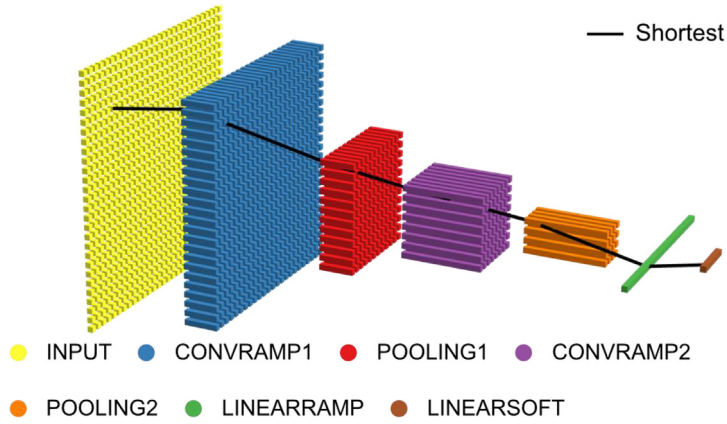

Fig. 2. Graphical representation of LeNet architecture with neurons coloured by layer membership. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

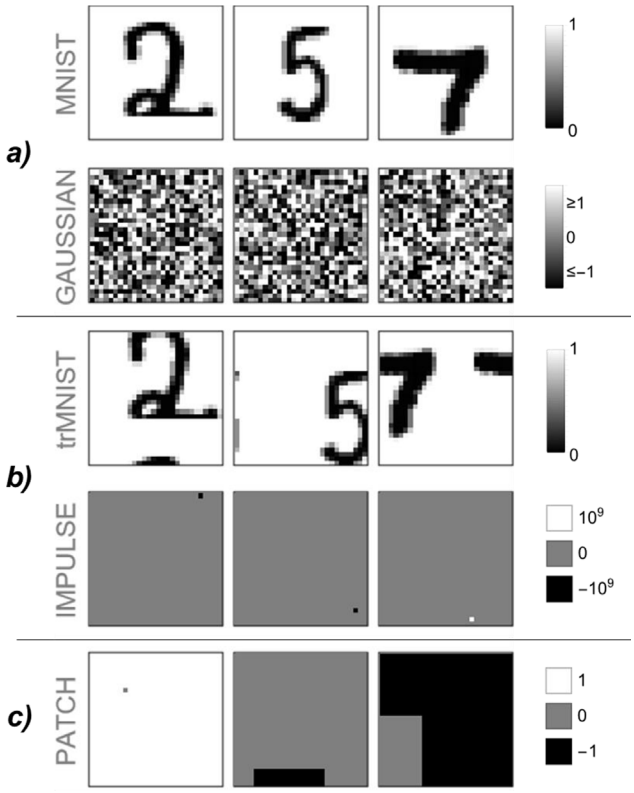

Fig. 3. Examples from various input image datasets used in this work. (**a**) MNIST and GAUSSIAN, see Section 3.2. (**b**) trMNIST and IMPULSE, introduced in Section 4.3. (**c**) PATCH, introduced in section 5.5.

the image dataset, in particular the x–y variation of the images allows the x–y location of some neurons to be more readily distinguished. This points to two fundamentally different types of image datasets: those that contain intrinsic structure, that is subsequently inherited by the network activity and those that are structureless, such that any structure in the network activity arises solely due to the network architecture.

4.3. Network correlations

In the previous section we saw that response vectors of *individual* neurons support features (mean and standard deviation) which are useful for classifying the layer, family, x and y indices of neurons to varying degrees. In this section we consider what useful features can be extracted from the correlations of the responses of *pairs* of neurons. To that end we consider the *response correlation matrix,* whose *ijth* entry is the Pearson Correlation Coefficient between the response vectors for neurons i and j.

Two computational notes. First, for computational convenience we perturb all responses with a random value in the range $[-10^{-9}, 10^{-9}]$ before computing correlations; this prevents undefined correlations when one or both of the neurons is silent. We note that the random perturbation step we use creates spurious absolute correlations between neurons, at least one of which is silent, with an average value of only 0.008. Second, we have found that *absolute* correlation is more effective for the localisation task than signed, so in the remainder, *correlation will always refers to the absolute values of the response correlation matrix.*

We first consider the distribution of the values in the correlation matrix. We order each row from largest to smallest, and then average over columns to yield an ordered list of the mean *nth* largest correlation values. For example, for MNIST images the value in 3rd position is 0.63, which is the mean correlation between a neuron and the neuron which is the third most correlated with it. Fig. 9a shows that these mean correlations are larger for all ranks when MNIST images are used as network inputs rather than GAUSSIAN images.

We then investigate whether the degree of correlation is related to the graph distance between neurons. To do so we created an undirected graph representation of the network with which we can compute the length of the shortest path between two locations in the network (an example shortest path traversing the whole network is shown in Fig. 2). The mean graph distance between pairs of neurons is 4.6. Fig. 9b shows that the average correlation of neuron pairs separated by a graph distance of n, is only above chance for $n < 10$ for MNIST and $n < 4$ for GAUSSIAN input. The maximum graph-distance in LeNet is 10, so we see that MNIST images induce correlations across almost the entire network, whereas only neurons relatively close in the network are correlated for GAUSSIAN input. Importantly, for both image sets we see a clear relationship between *average* correlation and graph distance which is promising. However, the substantial variability in the correlation means that graph distance cannot be accurately estimated from correlation; and a threshold on correlation to predict direct connectivity performs poorly.

To analyse the correlation structure further we consider the correlations between a specific target neuron's response vector and the response vectors of each of the neurons in the network. This *correlation vector* is a single row of the correlation matrix. We visualise the correlation vector in the same way as with the

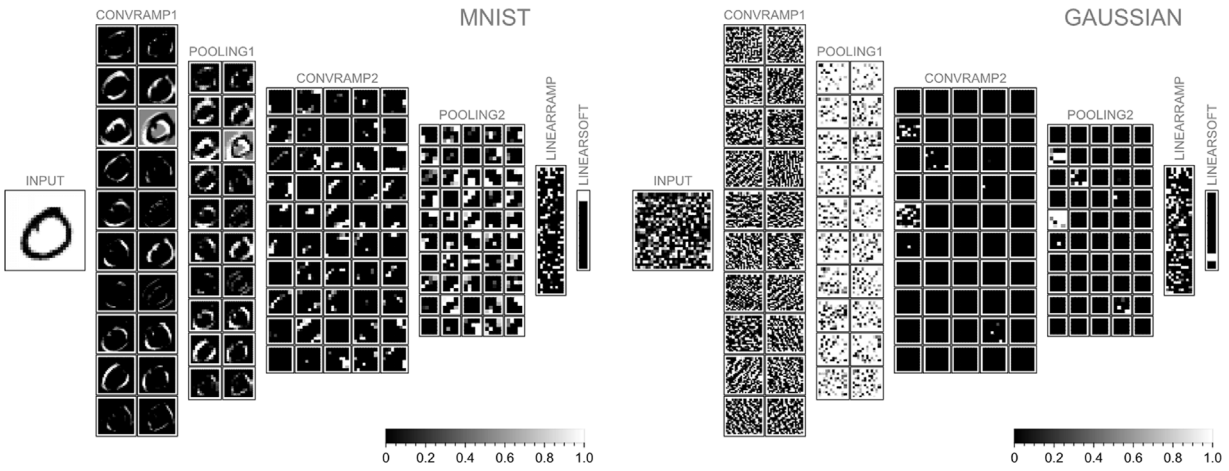

Fig. 4. Visualisation of the activation vector for an MNIST input and a GAUSSIAN input. Pixel intensity corresponds to neuron activation value, rescaled between [0,1] for visual clarity.

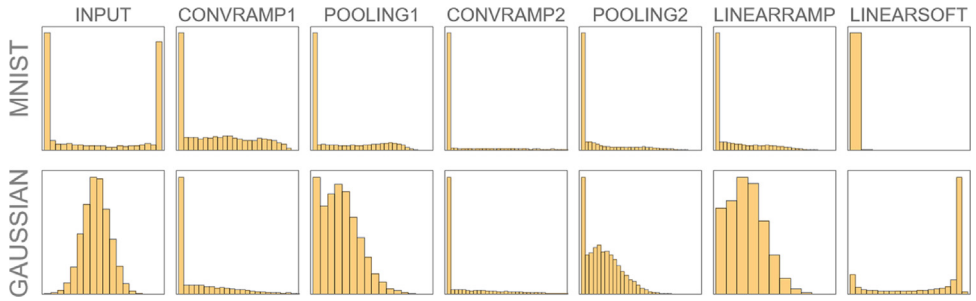

Fig. 5. Histograms of responses for *the neuron with the largest standard deviation from each layer* for MNIST and GAUSSIAN input. Neurons in different layers are seen to have different response statistics and behave differently for different input image datasets.

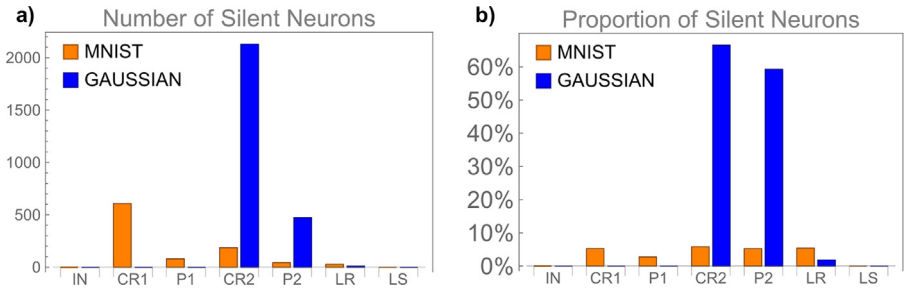

Fig. 6. Histograms of both the number **(a)** and proportion **(b)** of silent neurons by layer.

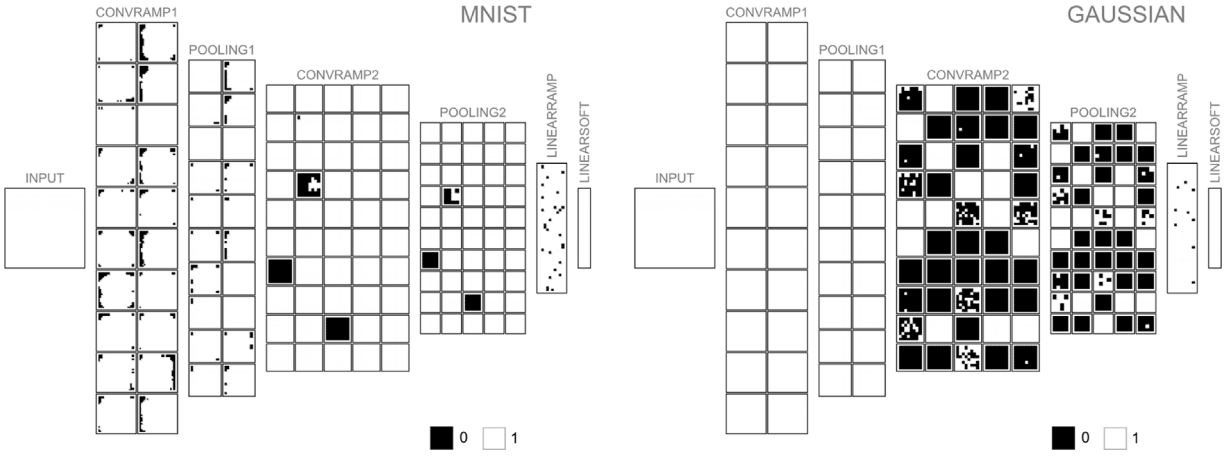

Fig. 7. Unitised (0 if 0, else 1) visualisation of the absolute activation matrix averaged over the image dataset for both MNIST and GAUSSIAN input. Silent neurons are shown in black.

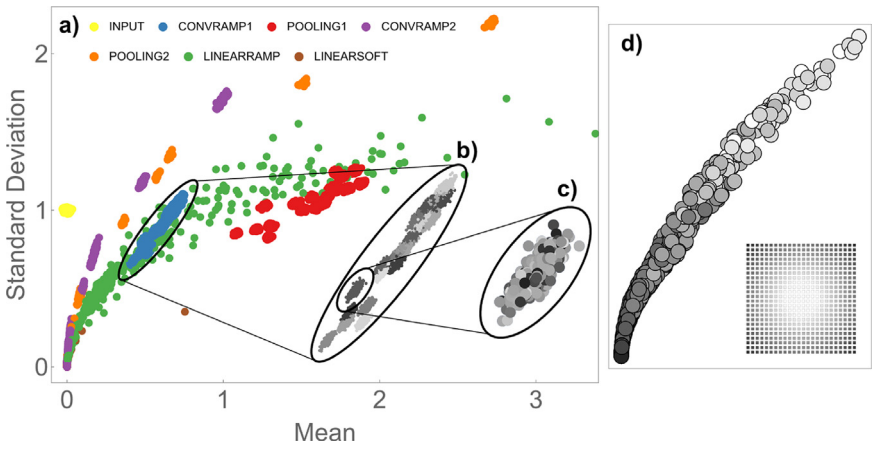

Fig. 8. **(a)** Mean vs. standard deviation of the response vectors for GAUSSIAN input colour coded by layer membership. Expanded views of: **(b)** all CONVRAMP1 neurons randomly colour coded (greyscale) by family, **(c)** a single family of CONVRAMP1 neurons colour coded (greyscale) by distance from the centre of the output space, **(d)** the same as (c) but for MNIST input. Inset in (d) shows (greyscale) colour map used in (c,d). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

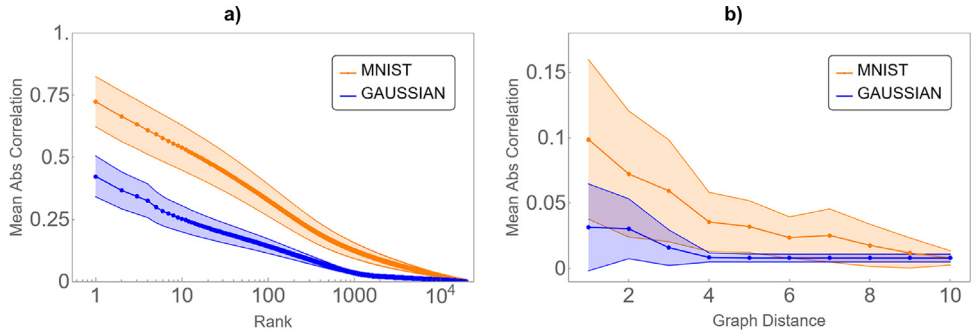

Fig. 9. **(a)** Each row of the (absolute) correlation matrix is sorted in descending order, and we average over these sorted lists. **(b)** Entries of the (absolute) correlation matrix are grouped by the graph distance between the two corresponding neurons and the average over each group is taken. The interval bands indicate the standard deviation across target neurons.

activation vector. Each pixel in the image represents a single neuron in the network, with its intensity in this case corresponding to the correlation between the response vector associated with that neuron and that of the target neuron. As before we show a single example vector for both MNIST and GAUSSIAN input (see Fig. 10) corresponding to a centrally-located neuron in the POOLING1 layer. For MNIST input this neuron is seen to be highly correlated with a large fraction of the network, as expected. The correlated neurons extend across layers but also across the spatial domain. In contrast, for GAUSSIAN input the same POOLING1 neuron is strongly correlated with only a localised patch in each layer, and the strength of correlation decreases rapidly as one progresses to deeper layers.

Having seen that for GAUSSIAN input neurons are correlated only in a localised region, we compare the correlation structure to the pattern of receptive fields (RF) in the network, which we remind are defined as the input locations that potentially drive a neuron. Fig. 11 focuses on the areas of Fig. 10 close to the spatial location of the target neuron. The coloured bounding boxes indicate the relation of the RFs of the displayed neurons to the RF of the target neuron in POOLING1. Neurons within red boundaries have RFs which are a subset of the target RF; within green boundaries an overlapping RF; and within blue boundaries a superset RF.

Since the GAUSSIAN image dataset itself is uncorrelated, correlations induced between neurons in the network are only present when the RFs of two neurons overlap in some way. In contrast, for MNIST input, neurons are correlated far outside the RF overlap region. This is due to correlations within the MNIST images themselves. Given this correspondence between correlation and patterns of RFs, along with the spatial organisation of RFs in LeNet, we expect the correlation matrix to contain useful information for distinguishing neuron location.

Rows of the correlation matrix represent neuron-specific views of the correlational structure of the network. To form an integrated view of the entire structure of the correlation matrix we compute its eigenvectors. As before we present a side-by-side visualisation, this time of an eigenvector of the correlation matrix for each of the two image datasets (see Fig. 12).

For MNIST input the illustrated eigenvector reflects the spatial structure of MNIST digits. While for GAUSSIAN input the eigenvector has a more systematic spatial structure, reminiscent of the eigenmodes of the Laplacian operator on a 2D domain. The GAUSSIAN eigenvector also has some variation across families within a layer; but this is less obvious as the change is not systematic with family number, which is an arbitrary index. The pattern of the eigenvector values across the INPUT layer is roughly preserved across subsequent layers and families, whenever they are sufficiently active. As such we need only visualise the eigenvector values over the INPUT layer to get a sense of its structure. Concretely, silent neurons will not contribute to the eigenvector pattern.

In Fig. 13, we compare eigenvectors of the *response* correlation matrix to eigenvectors of the *stimulus* correlation matrix, i.e. the matrix of correlations between INPUT neurons only. These are both defined across the INPUT layer so can be compared there.

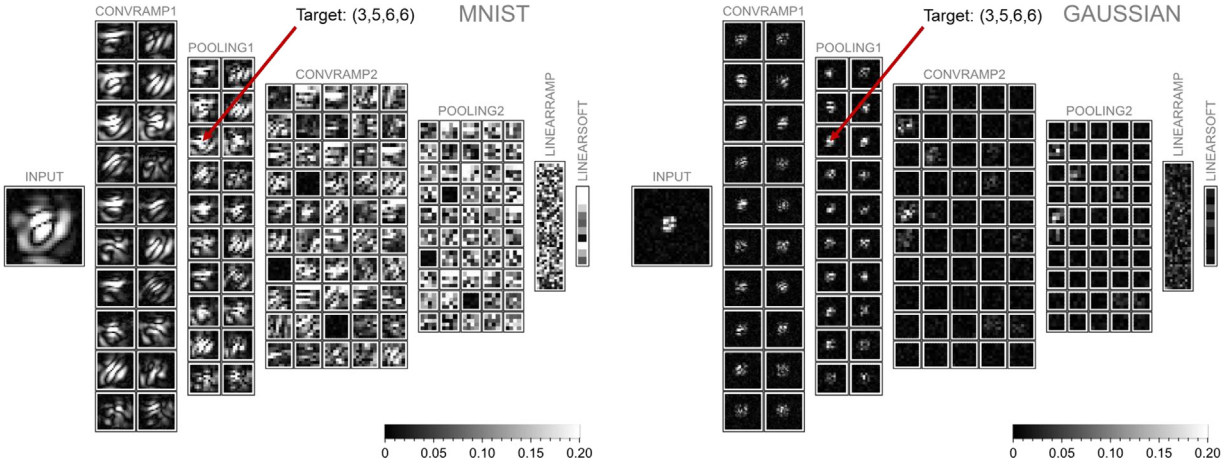

Fig. 10. Visualisation of example correlation vectors. Intensity corresponds to the (absolute) correlation between a target neuron (3,5,6,6) (indicated with red arrow), and the neuron represented by that pixel. Values are clipped to between [0.0,0.2] so that weak, but non-negligible correlations are visible. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

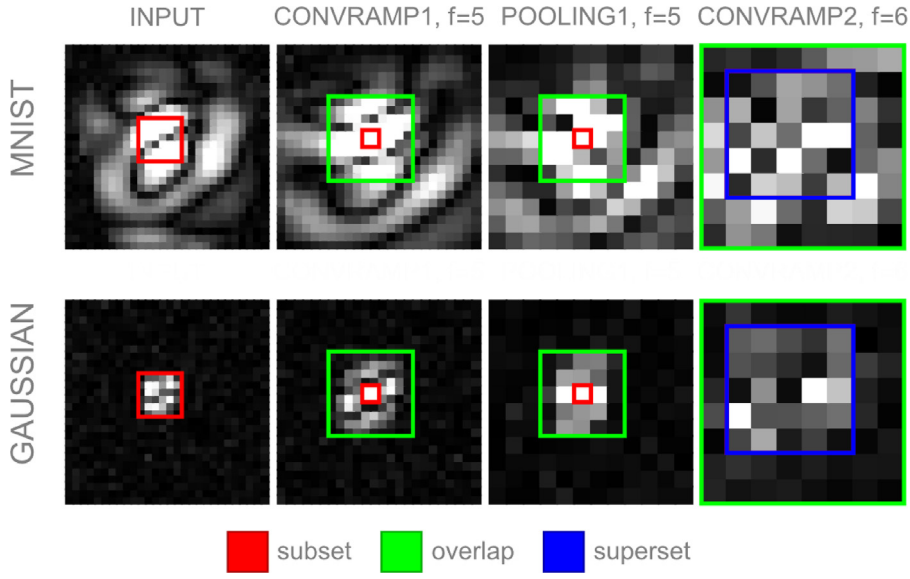

Fig. 11. Pixel intensities represent (absolute) correlation of the corresponding neuron with a target neuron, with network coordinates (3,5,6,6). Lighter pixels are more strongly (anti)correlated. Coloured boundaries delineate neurons whose receptive field (RF) have a particular relationship with the RF of the target neuron: red — subset, green — overlap and blue — superset. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

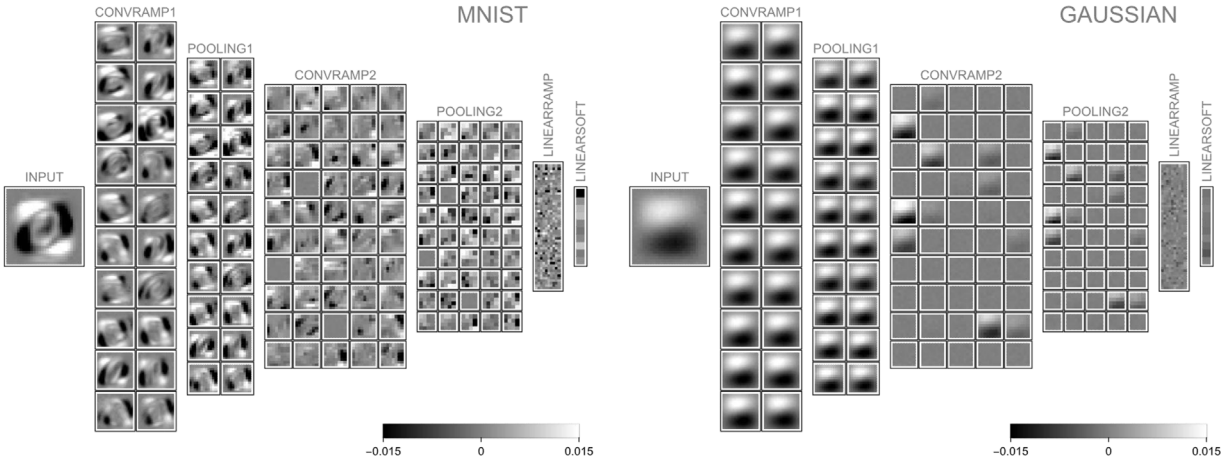

Fig. 12. Pixel intensity corresponds to the value of that neuron entry in the 2nd (chosen as an example) eigenvector of the (absolute) correlation matrix.

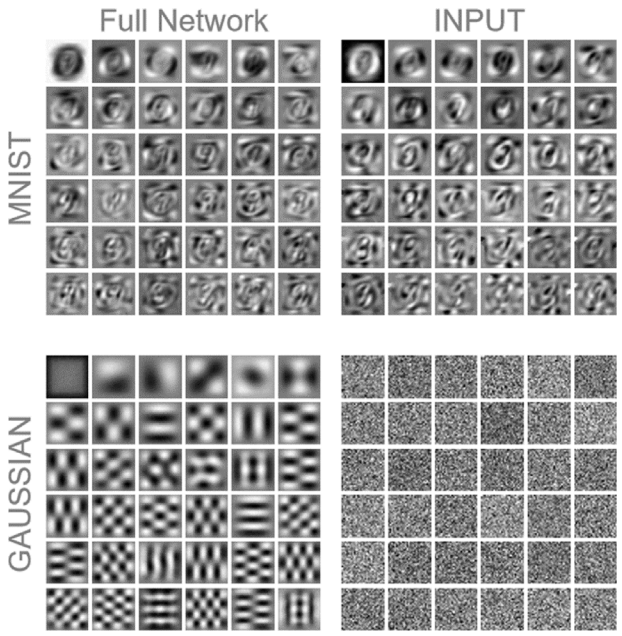

Fig. 13. Eigenvector values of the (absolute) correlation matrix plotted over the INPUT neurons and reshaped to form images. In the first column the entire response correlation matrix was used to compute the eigenvectors. In the second column the stimulus correlation matrix was used to form eigenvectors, i.e. using the correlations between all INPUT neurons only.

For MNIST images, the two types of eigenvectors are qualitatively similar, but for the GAUSSIAN images they are very different. The difference arises because, for MNIST images the correlations of the stimulus set are at least as strong as the correlations that arise within the network; whereas for GAUSSIAN images the stimulus set is free of correlational structure, so the eigenvectors of the response correlation matrix arise purely due to the network architecture.

Finally, to illustrate how eigenvectors vary with rank, the first 36 of them are shown in Fig. 13 (first column) for both input image datasets. It can be seen that for both MNIST and GAUSSIAN images, these tend to increasingly high frequency content for higher numbered eigenvectors. For both image datasets, the smoothness of the low rank eigenvectors with respect to graph distance make them look potentially effective for inferring neuron location, the discovery of such features having been the aim of this section.

5. Methods

In the previous section we identified candidate features to use in our classifiers — response moments, and values of low rank eigenvectors of the response correlation matrix. In the following Section 5.1 we describe how we use the classifiers operating on these features to infer the assignment of neurons to network locations. In 5.2 we introduce our assignment score and in 5.3 we specify which feature sets we will use in our analysis, including those derived from two further image datasets.

5.1. Assignment

The assignment problem we have devised requires N neurons to be matched 1-to-1 to N locations. We proceed by first computing an assignment confidence for each neuron-location pair, and then finding the optimal assignment of neurons to locations given those confidences.

To compute confidences, we take advantage of the regularity and hierarchical organisation of the structure of LeNet. Specifically, we train a set of multi-class classifiers to infer: the layer to which a neuron belongs, the family given that it belongs to layer CONVRAMP1, the x-coordinate given that it belongs to INPUT, etc. By summing log confidences from different subsets of this bank of classifiers we can arrive at an overall log confidence for each neuron n to belong at each location (l, f, x, y) i.e.

$$log\, P_{l,f,x,y} = log\, P_l + log\, P_{f|l} + log\, P_{x|l} + log\, P_{y|l} \quad (1)$$

We chose to pool neurons from all the families in a given layer when training the corresponding x and y classifiers, so that the total number of training examples was larger. If we had trained separate spatial classifiers for each family then, for example, the x-classifier for POOLING2, family 1, would only have 8 training neurons available (given a 50:50 train-test split) with which to learn 4 classes. By including all families, the number of training neurons is increased 50-fold for POOLING2.

We note that there are two special circumstances when these classifiers are degenerate. An example of the first type is for the classifier inferring family for the INPUT layer — since there is only one family, the probability of being in that family ($f = 1$) given the neuron is in the INPUT layer is one. An example of the second type is for the classifier inferring the family for the LINEARRAMP layer — since there is only one test or train neuron for each family for this layer, and since there is no systematic structure across the family, inference is impossible. We simply define the probability of being in a particular family as $1/k$ where k is the number of test neurons in LINEARRAMP.

In preparation for classifier training we first separate the neurons into an overall train-test split. From the full set of training neurons, we construct separate training sets for each of the 15 classifiers we construct. For the layer classifier, we construct a layer-balanced set by sampling with replacement from the training neuron set. For the family and x/y classifiers we use only training neurons of the appropriate layer. We train Random Forest (Breiman, 2001) multi-class classifiers to infer layer, family, x or y from the features: response mean, response standard deviation and the values of the first 16 eigenvectors. Other classifiers perform no better and are slower to compute.

Trained classifiers are then applied to the feature vectors for the test neurons, and the resulting log-confidences are combined to produce overall confidences between each test neuron and each test location. The 1-to-1 assignment that maximises the sum of overall log-confidences is then computed using the Hungarian algorithm (Kuhn, 1955).

5.2. Performance score

We need to choose a measure to score the success of our assignment of neuron responses to locations. Rather than base the score on whether a neuron is correctly assigned or not, we prefer a more powerful measure that uses graph distance to gauge how far from correct an erroneous assignment is, and we prefer root-mean-square (RMS) graph distance error rather than mean error so that large errors have extra impact. To prevent our score being dominated by performance in the layers of the network with more neurons (CONVRAMP1 for example contains 58% of the network's neurons) we compute the RMS graph error separately for each layer, and report the mean of these scores as our final error metric.

As described previously, some neurons are silent; and since all silent neurons are indistinguishable, this will inevitably lead to some assignment error. To determine how much of our error is due to silent neurons, we decompose the total error into that caused by *active* vs. *silent* neurons. As a baseline error we use random assignment. The value of this is a function of the network architecture and is the RMS graph distance between all pairs of neurons. For LeNet this is 4.6.

5.3. Feature sets

Based on our preliminary analysis we use as our feature set the mean and standard deviation of the response vectors and the values of the first 16 eigenvectors of the correlation matrix. Our classifier bank will receive as training input feature vectors corresponding to a 50% split of the total neuron population.

We compare our algorithm's performance on feature sets generated from four image datasets: GAUSSIAN and MNIST as described previously, as well as MNIST images which have been uniformly randomly translated with wrap-around (denoted trMNIST), and a dataset for which each image is all zeros except for a single pixel impulse of magnitude $\pm 10^9$, with two images per 784 pixel positions (denoted IMPULSE, see Fig. 3b for example images). Additionally, we use an image dataset – denoted MIXED – which contains all the GAUSSIAN, MNIST, trMNIST and IMPULSE images. Finally, we also evaluate assignment using features which are a concatenation of the features arising from the four individual datasets (denoted CONCAT).

To conclude Section 5, we summarise the key steps of our assignment algorithm:

Algorithm 1 (*Assignment Algorithm*)**.**

(1) Compute feature vectors consisting of: response mean, response standard deviation and the values of the first 16 eigenvectors of the (absolute) response correlation matrix, for all neurons.
(2) Split the feature vectors into a training set and a test set.
(3) Individually train all 15 classifiers on the training feature vectors.
(4) Input test feature vectors into the trained classifiers to compute an $N \times N$ matrix of log-confidences for each test neuron belonging to each test location.
(5) Apply the Hungarian Algorithm to the matrix of log-confidences to optimally solve the linear assignment problem of neurons to network locations.
(6) Compute the layerwise RMS graph distance errors of the assignment for each layer and take the mean of these values to give the overall error.

6. Results

In Section 6.1 we present results for the best performing algorithm. In the subsections following we report auxiliary results: in 6.2 we vary hyperparameters, in 6.3 we test hypotheses that explain the observed pattern of performance, and in 6.4 we combine results from these analyses to produce a feature set which results in zero error.

6.1. Best performing hyperparameters

In this subsection we present the scores for our best performing hyperparameters, for our different feature sets, in a plot of silent vs. active error (see Fig. 14).

To separate out the variability in performance due to random train-test splits we use a "perfect" train-test split, wherein each spatial layer and family is divided in a checkerboard pattern such that there is an even sample of x and y locations, and then linear layers are sampled by taking the odd or even indices. The training split is then layer-balanced via oversampling, as described previously. For reference we also compute results for 10 random train-test splits, also with layer-balancing, and plot the standard deviation as an error bar. The random train-test splits cause a significant variation in performance, but not enough to change the ordering of the performance of each feature set.

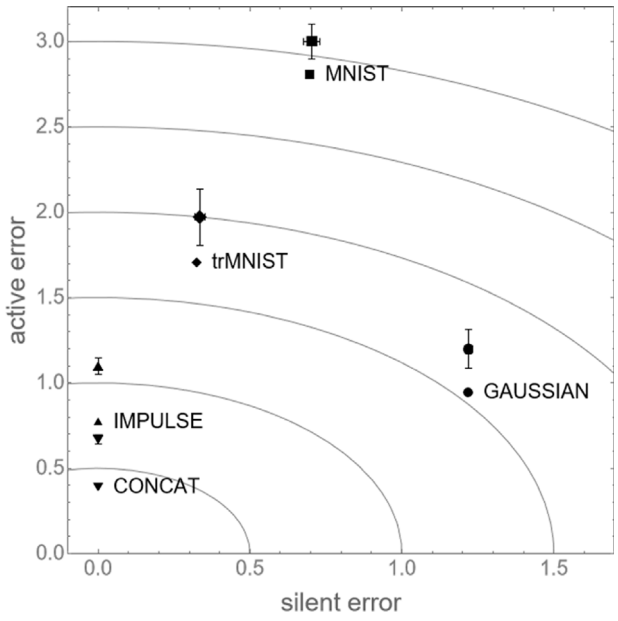

Fig. 14. Scatter plot of silent vs. active errors for various feature sets. Contours indicate overall error. Points with error bars indicate the standard deviation of errors over 10 random train-test splits. Those without error bars are the errors resulting from a perfect train-test split. Overall error for each feature set using perfect train-test split: MNIST — 2.9, trMNIST — 1.7, GAUSSIAN — 1.5, IMPULSE — 0.8, CONCAT — 0.4.

The overall errors for each feature set using perfect train-test split in descending order are: MNIST 2.9, trMNIST 1.7, GAUSSIAN 1.5, MIXED 0.8, IMPULSE 0.8, CONCAT 0.4.

We see that MNIST feature set produces the largest overall error and the CONCAT feature set produces the smallest (as indicated by the contours in Fig. 14). Despite the large error for MNIST images, we found that its inclusion in the concatenated feature set still improves performance. Of the feature sets derived from the four individual image datasets, IMPULSE results in the lowest overall error. It also produces zero silent neurons. Interestingly, trMNIST produces substantially fewer silent neurons than GAUSSIAN (210 compared to 2609), but because of its high active error, it has a worse overall error than GAUSSIAN. By decomposing the errors in this way, we see that activating more of the network is not guaranteed to improve performance.

We also observed that combining the feature sets themselves (CONCAT) yields better results than combining the image datasets (MIXED). With MIXED, the correlations induced by the IMPULSE image dataset dominate the correlations induced by the other image datasets and as such MIXED performs only marginally better than IMPULSE alone and so we exclude it from further analysis. In contrast, by concatenating the feature sets, the classifiers can make use of the information in the correlation matrices associated with each image dataset individually, resulting in the lowest error.

The superior performance of the CONCAT feature set could be caused by the fact that it used four times as many features as the other feature sets. To test this hypothesis, we increased both the number of moments and eigenvectors used as features for each of the single image datasets but found no significant improvements in the assignment error in any case. Each of the feature sets corresponding to GAUSSIAN, MNIST, trMNIST and IMPULSE appear to individually contain useful information, not present in the others, such that all the features must be leveraged to achieve the lowest overall error.

Fig. 15 shows that localisation error varies considerably with layer. This is not surprising given how much they vary in terms

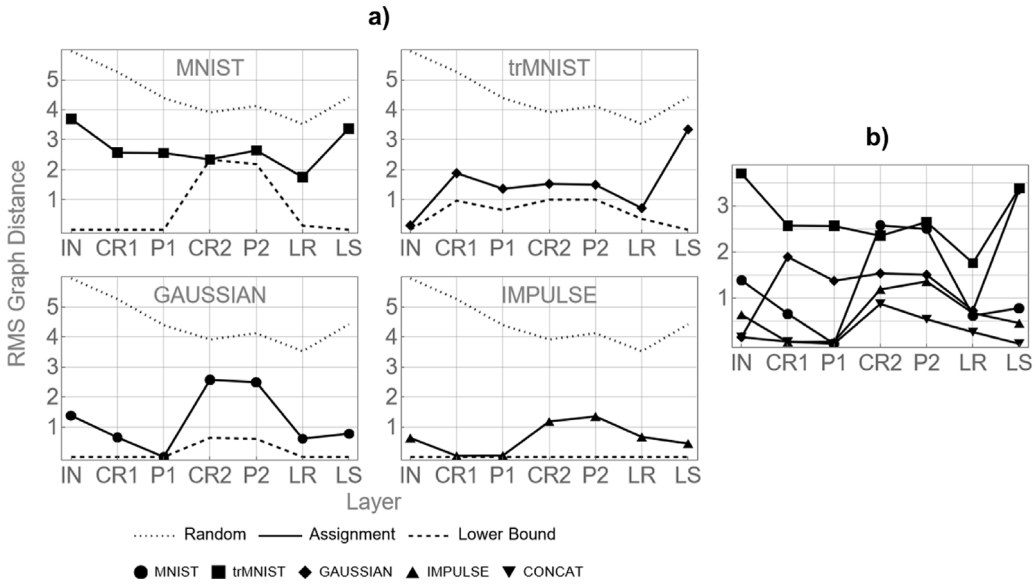

Fig. 15. **(a–d)** Layerwise RMS graph distance errors for all image datasets separately. The dotted line shows the baseline score when neurons are assigned randomly. The dashed line shows the score that would be achieved if assignment was perfect other than the error caused by silent neurons. **(e)** Comparison of layerwise scores for all four unmixed image datasets, and for the CONCAT features.

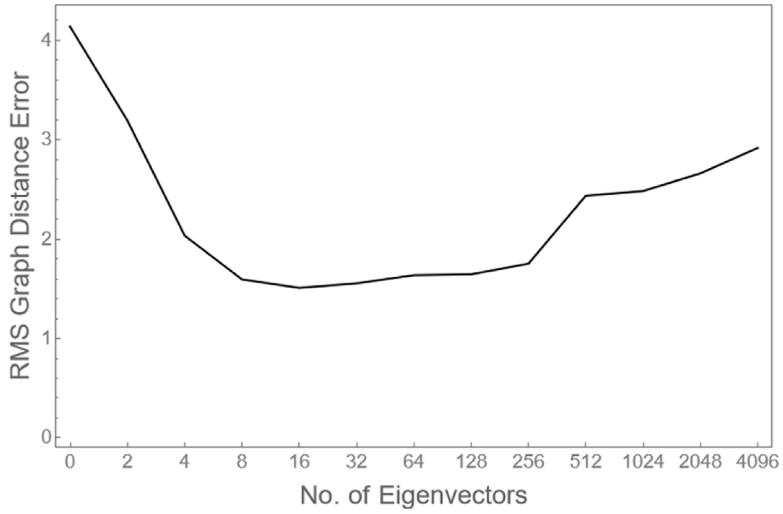

Fig. 16. Tuning curve for the number of eigenvectors used as feature vectors. The Mean and Standard Deviation of the response vectors were included as features by default, so that the scores were comparable to our final result. A single minimum is present at 16. A wide range of values for the total number of eigenvectors yield similar performance, only at the extremes is the error significantly increased.

of x–y range, number of family and pattern of connectivity with adjacent layers. CONVRAMP2 and POOLING2 neurons are consistently difficult to localise for all feature sets, even those for which all of these neurons are active (e.g. IMPULSE). The figure shows that no single image dataset has the best performance for every layer, though CONCAT is equal to or better than any single dataset for all layers.

6.2. Hyperparameter variations

For brevity we report the effect of the variations in the hyperparameters of our algorithm using only the GAUSSIAN dataset, and the perfect train-test split described in the previous subsection. First, we vary the number of eigenvectors used to create features (Fig. 16). We see that there is a single broad minimum at 16 eigenvectors. Given that eigenvectors above that number contain visible network structure we expect that they do provide potentially useful information for localisation, but the noise they also contain drowns out their utility.

In Table 2, we summarise some other basic modifications to the algorithm. We make the following comments:

- We find that artificially balancing the number of training neurons from different layers leads to a substantial improvement.
- Of the four classifier methods tested, Random Forest produced the lowest error. Although neural networks gave a comparable score it took approximately ten times longer to train. The simplicity and time efficiency of Random Forest make it an attractive method for this sort of exploratory work.
- The inclusion of the mean response as a feature is seen to lead to a significant improvement in performance. Additional moments (standard deviation, skewness, etc.) give

Table 2
Performance scores for variations in the algorithm. The hyperparameters used in the "best performing" algorithm, are indicated in bold. In some cases, the hyperparameter chosen is not one which results in the minimum error for GAUSSIAN but is a heuristic balance of low error across image datasets, parsimony, and computation time.

Layer balancing	Balanced **1.54**	Unbalanced 2.27	–	–	–
Classifier method	Random forest **1.54**	Neural network 1.61	Decision tree 2.30	Logistic regression 2.12	–
Moments	None 1.96	1st 1.59	1st–2nd 1.54	1st–3rd 1.53	1st–4th **1. 54**
Correlation Matrix (cm)	cm 3.62	Abs [cm] **1.54**	cm^2 1.48	Abs [cm^3] 1.49	cm^4 3.02
No. Images	1k 1.88	2.5k 1.69	5k 1.63	7.5k 1.54	10k **1.54**

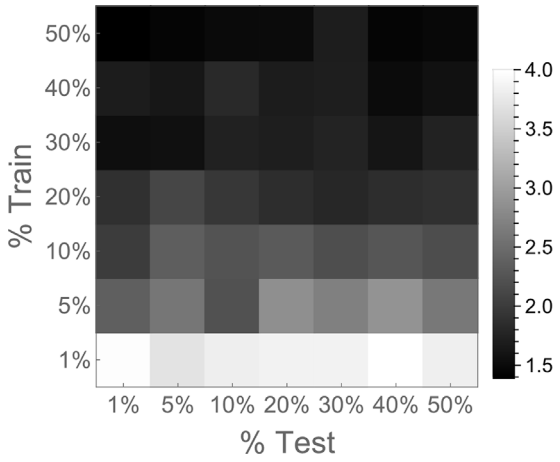

Fig. 17. The RMS graph distance error averaged over layers and averaged over 5 different random train-test splits for a range of training and test sample sizes where the percentage is of the total population of 19,694 neurons.

further improvements but with diminishing gain. The improvement is primarily due to the layer classifier. Without response moments it achieves only 40% accuracy; with moments 96% accuracy.
- Reducing the number of input images only modestly degrades performance.

We also systematically explored varying both the number of training and test samples finding that reducing the former led to an expected increase in error whereas varying the latter had no clear effect on performance (see Fig. 17). In these experiments a random test-train split was used rather than the perfect split used previously. We included a uniform class prior in our calculation of location probability to account for the fact that for low training percentages some classes may not be sampled at all. Features were created using a correlation matrix constructed *only* from the union of the training and test set neuron responses for that run.

6.3. Explaining performance

In this section we test three hypotheses to explain the variations in performance that we observe for different image datasets.

Hypothesis 1. Neurons with higher activation rates are more likely to be correctly localised.

Having observed the prevalence of silent neurons for various image datasets and acknowledged that they are difficult to localise, we speculated whether low activity in general would lead to poor localisation performance. To test this hypothesis, we recorded the distribution of errors for a particular assignment, binned by activation rates - defined as the fraction of input generating a non-zero activation. As Fig. 18 shows, there is not a simple relationship between activation rate and assignment error. As such we reject Hypothesis 1.

Hypothesis 2. More spatially uniform image datasets result in lower error.

The MNIST dataset is not translationally invariant, with most of the activity concentrated at the centre of the input space. With trMNIST, we randomly and uniformly translate the MNIST images to produce a more translationally invariant dataset. Here, we introduce another set of modifications, trMNIST σ, in which we randomly sample translations from a zero centred normal distribution for various values of σ, to investigate the transition between MNIST and trMNIST in terms of spatial uniformity. The overall errors for MNIST and trMNIST are 2.9 and 1.7 respectively. For $\sigma = (1, 2, 4, 8, 16)$ the overall errors of trMNIST σ are 2.6, 2.2, 2.1, 2.0 and 1.9 respectively. We see that the error smoothly decreases for increasing uniformity.

For the IMPULSE dataset score recorded previously, we used a balanced set with all 784 positions for the IMPULSE occurring twice, once for each of the $\pm 10^9$ magnitudes. We compare this to a dataset of 10k images where the IMPULSE positions are sampled randomly from a uniform distribution, denoted IMPULSE*. In this image dataset we also ensure a 50:50 split of positive to negative impulses. The overall errors for IMPULSE and IMPULSE* are 0.8 and 1.3 respectively. For this particular type of image dataset, the performance is very sensitive to the uniform balancing of activity over the input space. For IMPULSE*, the slight heterogeneity produced by the fact that we only have 10k images causes a significant degradation in performance.

In all the cases investigated, comparing image datasets drawn from similar distributions, the more spatially uniform image dataset results in lower error, supporting Hypothesis 2.

Hypothesis 3. Higher correlations in the image dataset leads to higher error.

We consider a dataset of constant images, where the constant value is drawn from a zero centred normal distribution. Such a dataset will induce strong correlations between neurons with nonoverlapping receptive fields. The score for such a dataset (denoted CONSTANT) is 4.3, marginally better than random assignment (4.9).

To smoothly vary from the CONSTANT image dataset to the GAUSSIAN dataset we consider GAUSSIAN images with a random offset added to each image, with the offset drawn from a normal distribution of zero mean and unit standard deviation, (denoted *off* GAUSSIAN). *off* GAUSSIAN results in an overall error of 2.1. As the value of σ is increased, the assignment error increases

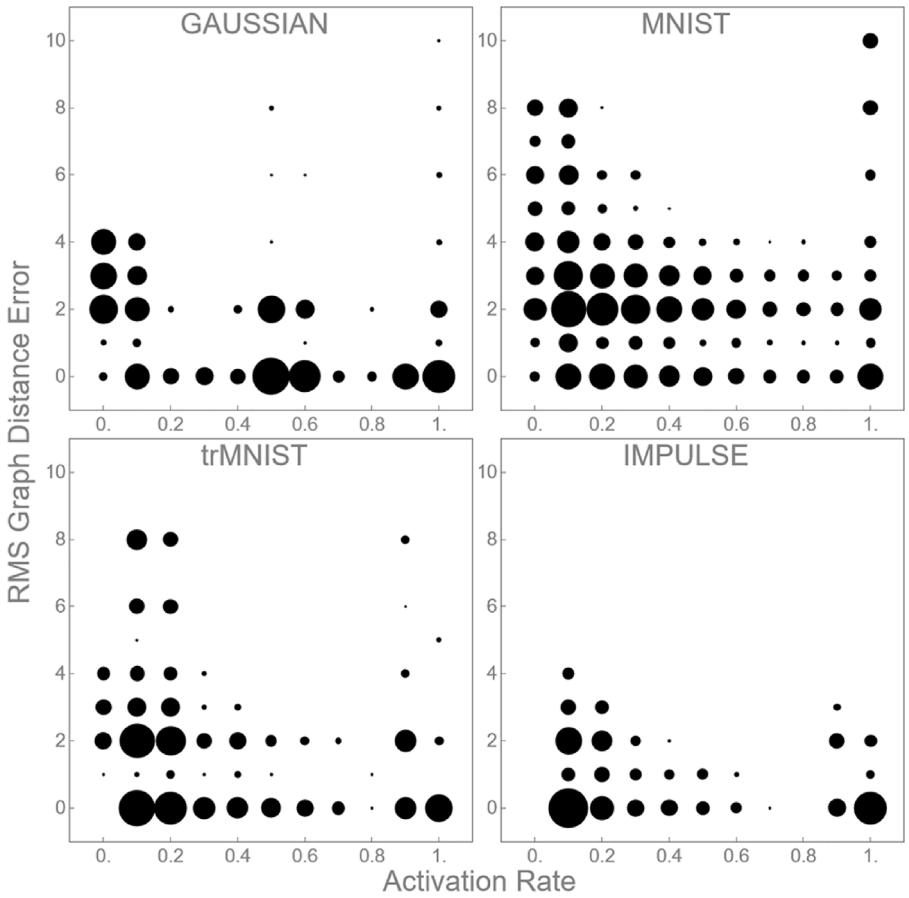

Fig. 18. 2D histograms of activation rate vs. error for each image dataset, where the area of the circles is proportional to the logarithm of the number of neurons in that bin. There is no clear relationship between the activity of a neuron and its assignment error.

until equal to the error produced by CONSTANT images. These modifications to the GAUSSIAN image dataset demonstrate the effect of increasing the correlation between all the neurons in the network simultaneously.

To investigate the impact of spatially-nonuniform input correlations we consider another modification to the GAUSSIAN image dataset. By tiling *identical* (14×14) arrays of gaussian noise into a (28×28) array to yield a single image, we create an image dataset (denoted *ti*GAUSSIAN) with large correlations between distant pixels, but indistinguishable point statistics from the GAUSSIAN image dataset. For *ti*GAUSSIAN, neurons within each tile are uncorrelated, but there is perfect correlation between each neuron and its copy in the other three tiles. The overall errors for GAUSSIAN and *ti*GAUSSIAN are 1.5 and 3.7 respectively. The strong correlations in the *ti*GAUSSIAN image dataset have resulted in an even higher error than for the MNIST image dataset. Note that for this instance, GAUSSIAN produces 2609 silent neurons and *ti*GAUSSIAN produces 2599, indicating that the drastic difference in performance is a result of the image dataset correlations and not due to any differential activation of the network. As such Hypothesis 3 is supported.

6.4. Perfect assignment

Having established our best performing algorithm and seen that concatenated feature sets, as well as increased spatial uniformity and decreased correlation of the image datasets result in lower error, we sought to leverage these insights to construct a set of features that would lead to zero error. We also use the perfect train-test split described previously.

With single pixel impulses yielding the lowest error of a single image dataset, as well as resulting in zero silent neurons, we investigated variations on this theme: varying amplitude, sign, background value and number of impulses per image as well as the number of moments and eigenvectors used as features. Concatenating features derived from such image sets as well as including variations on the GAUSSIAN image dataset, reduced the error even further, but it remained nonzero. Since it appeared that using only uncorrelated image sets was insufficient, we re-evaluated our Hypothesis 3, that correlations in the image dataset led to higher error, reasoning that if the induced correlations respect the underlying architecture, then they need not necessarily confound the assignment task.

Since the receptive fields (RF) in LeNet are all square, with widths $w \in (1,5,6,14,15,28)$, we construct image datasets that mirror this structure. Specifically, we consider images containing a square patch of constant value, surrounded by a background of a different constant value. To ensure spatial uniformity across the image dataset, square patches of each size are translated systematically across the image space, allowing patches to overhang the borders of the image (for example images see Fig. 3c). We refer to such image datasets as PATCH.

The total set of translated patch images for the patch widths we use has 9367 members, comparable to our previous image datasets with 10k images each. We create four such image datasets corresponding to foreground, background values $(f, b) \in ((1,0), (0,1), (-1,0), (0,-1))$.

For these image datasets we use the first four moments of the response vectors and the first 19 eigenvectors of the correlation matrix as our features. Concatenating the four feature sets we

achieve an overall assignment error of 0.08, using 92 features. This compares favourably with our previous best result, CONCAT, which produced an error of 0.40 using 72 features.

By randomly searching through similar features over the full range of possible width values [1,28], we were able to find less symmetric feature sets that resulted in an assignment error of exactly zero. We note that the set of features resulting in perfect assignment for a particular network instance will give a low, but nonzero, error for another network instance. However, the random search can be repeated and in our experience feature sets resulting in zero error for the particular network instance, could always be found.

7. Conclusion

We have investigated the task of assigning neurons to known locations in an Artificial Neural Network (ANN) based on the responses of neurons to a dataset of stimulus images. The aim of this was to develop insights into the task which could inform pursuit of the more ambitious goal of network inference from neuronal responses.

There were two stages to the results from this project. The first was to establish whether the activations of neurons in ANNs contained accessible information regarding their location in the network. Having confirmed that a significant fraction of neurons could be correctly assigned using features derived from the moments of the response vectors and the eigenvectors of their correlation matrix, the task then became assessing how best to stimulate the network to improve the assignment error.

Our assignment performance is sensitive to several factors but none so severely that having less than ideal conditions renders the assignment useless. The balance of classes in the train-test split causes variation in the performance, which is of course true in general for classification tasks. The spatial uniformity of the input image dataset is also important. We found that in general, image datasets consisting of uncorrelated images resulted in lower error. Our analysis suggests that this is primarily because correlations in the input confound correlations induced by the network.

Furthermore, when we decomposed the assignment error into errors caused by active and silent neurons separately, we saw that image datasets which activated the majority of the network (trMNIST) could nonetheless have similar overall error to image datasets that result in many silent neurons (GAUSSIAN), due to having high active error. Simply activating more of the network was not a sufficient condition for lower error.

For features derived from single image datasets, we achieved the lowest error using a dataset of single pixel impulses. Such images correlate only those neurons with overlapping receptive fields (RF), resulting in a correlation matrix with minimal noise and which activates every neuron in the network, unlike gaussian noise. By combining features derived from different image sets through concatenation, we avoid having to find a single image distribution which performs optimally across all neurons. Appending more features however is not an unalloyed good; eventually it degrades performance.

Having failed to achieve zero error using only feature sets based on uncorrelated image datasets (variations of GAUSSIAN and IMPULSE), we reasoned that correlations induced by the input are not necessarily deleterious if they do not confound the correlations induced by the network architecture. Since the receptive fields of LeNet are all square with fixed widths we created a series of image datasets in which a square patch of constant value and corresponding width is translated uniformly across the input space. Using features derived from such image datasets we were able to achieve a perfect assignment with zero overall error using 50% training data.

Our solution to this task is not intended to be extended directly to BNNs, where parameterisation of network location information is inherently less clearly defined, ground truth connectivity is not typically available for the majority of the network and responses are both partial and noisy. Instead, we intend to provide evidence for the validity of developing inference algorithms in the synthetic setting of ANNs. A priori, given the presence of nonlinear activation functions and symmetries in the network architecture even the comparatively simple task of response localisation may have been impossible. Having clearly demonstrated that this is not the case, we can confidently progress onto more challenging inference tasks.

Given the substantial differences between ANNs and BNNs it may be the case that inference algorithms developed in the synthetic setting fail to transfer to the biological one. We argue that the productive cross-pollination between researchers developing ANNs and neuroscientists studying BNNs (exemplified by the emerging field of "Neuro-AI") continue to demonstrate that ANNs are a useful model of BNNs. Work like (Banino et al., 2018) provides implicit evidence for their functional similarity by revealing that the internal representations of agents based on ANNs engaged in a navigation task recapitulated those in the relevant biological circuits. Parallel to this are recent evolutionary arguments claiming that ANNs are more like biological neural processes that is often assumed (Hasson et al., 2020).

Short of the development of neural imaging technology with both high resolution and wide field of view, our best bet for investigating neural connectivity lies in leveraging recordings of neural activity (for which the number of simultaneously recorded neurons continues to grow) and improving the network inference algorithms that act on this activity. Algorithm development is best explored in a robust and controllable setting and ANNs are ideal for this task. Once robust network inference algorithms have been developed in the ANN setting then, if necessary, the gap between ANNs and BNNs can begin to be closed.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

This work was supported by the Engineering and Physical Sciences Research Council [grant number 548341].

References

Bae, J. A., Baptiste, M., Bodor, A. L., Brittain, D., Buchanan, J., Bumbarger, D. J., Castro, M. A., Celii, B., Cobos, E., Collman, F., Costa, N. M. da, Dorkenwald, S., Elabbady, L., Fahey, P. G., Fliss, T., Froudarakis, E., Gager, J., Gamlin, C., Halageri, A., & … Yu, S. (2021). Functional connectomics spanning multiple areas of mouse visual cortex. http://dx.doi.org/10.1101/2021.07.28.454025, (2021.07.28.454025).

Bakshy, E., Rosenn, I., Marlow, C., & Adamic, L. (2012). The role of social networks in information diffusion. In *Proceedings of the 21st international conference on world wide web - WWW'12* (pp. 519–528). http://dx.doi.org/10.1145/2187836.2187907.

Banino, A., Barry, C., Uria, B., Blundell, C., Lillicrap, T., Mirowski, P., Pritzel, A., Chadwick, M. J., Degris, T., Modayil, J., Wayne, G., Soyer, H., Viola, F., Zhang, B., Goroshin, R., Rabinowitz, N., Pascanu, R., Beattie, C., Petersen, S., & . Kumaran, D. (2018). Vector-based navigation using grid-like representations in artificial agents. *Nature*, *557*(7705), 429–433. http://dx.doi.org/10.1038/s41586-018-0102-6.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. http://dx.doi.org/10.1023/A:1010933404324.

Castro, R., Coates, M., Liang, G., Nowak, R., & Yu, B. (2004). Network tomography: Recent developments. *Statistical Science*, *19*(3), 499–517.

Costa, L. da F., Rodrigues, F. A., Travieso, G., & Villas Boas, P. R. (2007). Characterization of complex networks: A survey of measurements. *Advances in Physics*, *56*(1), 167–242. http://dx.doi.org/10.1080/00018730601170527.

Dong, X., Thanou, D., Rabbat, M., & Frossard, P. (2019). Learning graphs from data: A signal representation perspective. *IEEE Signal Processing Magazine*, *36*(3), 44–63. http://dx.doi.org/10.1109/MSP.2018.2887284.

Fan, J., Suo, J., Wu, J., Xie, H., Shen, Y., Chen, F., Wang, G., Cao, L., Jin, G., He, Q., Li, T., Luan, G., Kong, L., Zheng, Z., & Dai, Q. (2019). Video-rate imaging of biological dynamics at centimetre scale and micrometre resolution. *Nature Photonics*, *13*(11), 809–816. http://dx.doi.org/10.1038/s41566-019-0474-7.

Feizi, S., Marbach, D., Médard, M., & Kellis, M. (2013). Network deconvolution as a general method to distinguish direct dependencies in networks. *Nature biotechnology*, *31*(8), 726–733. http://dx.doi.org/10.1038/nbt.2635.

Friedman, J., Hastie, T., & Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, *9*(3), 432–441. http://dx.doi.org/10.1093/biostatistics/kxm045.

Friston, K. J. (2011). Functional and effective connectivity: A review. *Brain Connectivity*, *1*(1), 13–36. http://dx.doi.org/10.1089/brain.2011.0008.

Gordon, C., & Webb, D. (1996). You can't hear the shape of a drum. *American Scientist*, *84*(1), 46–55.

Gordon, C., Webb, D. L., & Wolpert, S. (1992). One cannot hear the shape of a drum. http://dx.doi.org/10.48550/arXiv.math/9207215, (arXiv:math/9207215). arXiv.

Hasan, M. A., Chaoji, V., Salem, S., & Zaki, M. (2006). Link prediction using supervised learning. In *Proc. of SDM 06 workshop on link analysis, counterterrorism and security*.

Hasson, U., Nastase, S. A., & Goldstein, A. (2020). Direct fit to nature: An evolutionary perspective on biological and artificial neural networks. *Neuron*, *105*(3), 416–434. http://dx.doi.org/10.1016/j.neuron.2019.12.002.

Helmstaedter, M. (2013). Cellular-resolution connectomics: Challenges of dense neural circuit reconstruction. *Nature Methods*, *10*(6), 501–507. http://dx.doi.org/10.1038/nmeth.2476.

Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, *160*(1), 106–154, 2.

Kac, M. (1966). Can one hear the shape of a drum?. *Am. Math. Mon.*, *73*, 1–23. http://dx.doi.org/10.1080/00029890.1966.11970915.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, *2*(1–2), 83–97. http://dx.doi.org/10.1002/nav.3800020109.

Larsen, L., Griffin, L. D., GRäzel, D., Witte, O. W., & Axer, H. (2007). Polarized light imaging of white matter architecture. *Microscopy Research and Technique*, *70*(10), 851–863. http://dx.doi.org/10.1002/jemt.20488.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, *1*(4), 541–551. http://dx.doi.org/10.1162/neco.1989.1.4.541.

Lurie, D. J., Kessler, D., Bassett, D. S., Betzel, R. F., Breakspear, M., Kheilholz, S., Kucyi, A., Liégeois, R., Lindquist, M. A., McIntosh, A. R., Poldrack, R. A., Shine, J. M., Thompson, W. H., Bielczyk, N. Z., Douw, L., Kraft, D., Miller, R. L., Muthuraman, M., Pasquini, L., & . Calhoun, V. D. (2020). Questions and controversies in the study of time-varying functional connectivity in resting fMRI. *Network Neuroscience*, *4*(1), 30–69. http://dx.doi.org/10.1162/netn_a_00116.

Magrans de Abril, I., Yoshimoto, J., & Doya, K. (2018). Connectivity inference from neural recording data: Challenges, mathematical bases and research directions. *Neural Networks*, *102*, 120–137. http://dx.doi.org/10.1016/j.neunet.2018.02.016.

Mateos, G., Segarra, S., Marques, A. G., & Ribeiro, A. (2019). Connecting the dots: Identifying network structure via graph signal processing. *IEEE Signal Processing Magazine*, *36*(3), 16–43. http://dx.doi.org/10.1109/MSP.2018.2890143.

Nectow, A. R., & Nestler, E. J. (2020). Viral tools for neuroscience. *Nature Reviews Neuroscience*, *21*(12), 669–681. http://dx.doi.org/10.1038/s41583-020-00382-z.

Newman, M. E. J. (2003). The structure and function of complex networks. *SIAM Review*, *45*(2), 167–256. http://dx.doi.org/10.1137/S003614450342480.

Novikov, D. S., Fieremans, E., Jespersen, S. N., & Kiselev, V. G. (2019). Quantifying brain microstructure with diffusion MRI: Theory and parameter estimation. *NMR in Biomedicine*, *32*(4), Article e3998. http://dx.doi.org/10.1002/nbm.3998.

Orlandi, J. G., Ray, B., Battaglia, D., Guyon, I., Lemaire, V., Saeed, M., Statnikov, A., Stetter, O., & Soriano, J. (2015). First connectomics challenge: From imaging to connectivity. In *Proceedings of the neural connectomics workshop at ECML 2014* (pp. 1–22). https://proceedings.mlr.press/v46/orlandi15.html.

Pernice, V., Staude, B., Cardanobile, S., & Rotter, S. (2011). How structure determines correlations in neuronal networks. *PLoS Computational Biology*, *7*(5), Article e1002059. http://dx.doi.org/10.1371/journal.pcbi.1002059.

Saleeba, C., Dempsey, B., Le, S., Goodchild, A., & McMullan, S. (2019). A student's guide to neural circuit tracing. *Frontiers in Neuroscience*, *13*(897), http://dx.doi.org/10.3389/fnins.2019.00897.

Shapson-Coe, A., Januszewski, M., Berger, D. R., Pope, A., Wu, Y., Blakely, T., Schalek, R. L., Li, P. H., Wang, S., Maitin-Shepard, J., Karlupia, N., Dorkenwald, S., Sjostedt, E., Leavitt, L., Lee, D., Bailey, L., Fitzmaurice, A., Kar, R., Field, B., & . Lichtman, J. W. (2021). A connectomic study of a petascale fragment of human cerebral cortex. http://dx.doi.org/10.1101/2021.05.29.446289, (2021.05.29.446289).

Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., & Vandergheynst, P. (2012). The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. http://dx.doi.org/10.1109/MSP.2012.2235192.

Solé, R. V., Corominas-Murtra, B., Valverde, S., & Steels, L. (2010). Language networks: Their structure, function, and evolution. *Complexity*, *15*(6), 20–26. http://dx.doi.org/10.1002/cplx.20305.

Steinmetz, N. A., Aydin, C., Lebedeva, A., Okun, M., Pachitariu, M., Bauza, M., Beau, M., Bhagat, J., Böhm, C., Broux, M., Chen, S., Colonell, J., Gardner, R. J., Karsh, B., Kloosterman, F., Kostadinov, D., Mora-Lopez, C., O'Callaghan, J., Park, J., & . Harris, T. D. (2021). Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings. *Science*, *372*(6539), eabf4588. http://dx.doi.org/10.1126/science.abf4588.

Timme, M., & Casadiego, J. (2014). Revealing networks from dynamics: An introduction. *Journal of Physics A: Mathematical and Theoretical*, *47*(34), Article 343001. http://dx.doi.org/10.1088/1751-8113/47/34/343001.

van den Heuvel, M. P., Sporns, O., Collin, G., Scheewe, T., Mandl, R. C. W., Cahn, W., Goñi, J., Hulshoff Pol, H. E., & Kahn, R. S. (2013). Abnormal rich club organization and functional brain dynamics in schizophrenia. *JAMA Psychiatry*, *70*(8), 783–792. http://dx.doi.org/10.1001/jamapsychiatry.2013.1328.

Zhang, Z., Zhao, Y., Liu, J., Wang, S., Tao, R., Xin, R., & Zhang, J. (2019). A general deep learning framework for network reconstruction and dynamics learning. *Applied Network Science*, *4*(1), 110. http://dx.doi.org/10.1007/s41109-019-0194-4.