Cengiz Pehlevan and Dmitri B. Chklovskii

Neuroscience-Inspired Online Unsupervised Learning Algorithms

Artificial neural networks

©ISTOCKPHOTO.COM/JUST_SUPER

Digital Object Identifier 10.1109/MSP.2019.2933846
Date of current version: 29 October 2019

Inventors of the original artificial neural networks (ANNs) derived their inspiration from biology [1]. However, today, most ANNs, such as backpropagation-based convolutional deep-learning networks, resemble natural NNs only superficially. Given that, on some tasks, such ANNs achieve human or even superhuman performance, why should one care about such dissimilarity with natural NNs? The algorithms of natural NNs are relevant if one's goal is not just to outperform humans on certain tasks but to develop general-purpose artificial intelligence rivaling that of a human. As contemporary ANNs are far from achieving this goal and natural NNs, by definition, achieve it, natural NNs must contain some "secret sauce" that ANNs lack. This is why we need to understand the algorithms implemented by natural NNs.

Motivated by this argument, we have been developing ANNs that could plausibly model natural NNs on the algorithmic level. In our ANNs, we do not attempt to reproduce many biological details, as in existing biophysical modeling work. Rather, we try to develop algorithms that respect major biological constraints. For example, biologically plausible algorithms must be formulated in the online (or streaming), rather than offline (or batch), setting. This means that input data are streamed to the algorithm sequentially, and the corresponding output must be computed before the next input sample arrives. Furthermore, memory accessible to a biological algorithm is limited so that no significant fraction of previous inputs or outputs can be stored.

Another key constraint is that in biologically plausible NNs, learning rules must be local: a biological synapse can update its weight based on the activity of only the two neurons that the synapse connects. Such locality of the learning rule is violated by most ANNs, including backpropagation-based deep-learning networks. In contrast, our NNs employ exclusively local learning rules. Such rules are also helpful for hardware implementations of ANNs in neuromorphic chips [2], [3].

We derive the algorithms performed by our NNs from optimization objectives. In addition to deriving learning rules for synaptic weights, as is done in existing ANNs, we also derive the architecture, activation functions, and dynamics of neural activity from the same objectives. To do this, we postulate only a cost

function and an optimization algorithm, which, in our case, is alternating stochastic gradient descent/ascent [4]. The steps of this algorithm map to an NN, specifying its architecture, activation functions, dynamics, and learning rules. Viewing both weight and activity updates as the steps of an online optimization algorithm allows us to predict the output of our NNs to a wide range of stimuli without relying on exhaustive numerical simulation.

To derive local learning rules, we employ optimization objectives operating with pairwise similarities of inputs and outputs of an NN rather than individual data points. Typically, our objectives favor similar outputs for similar inputs, hence the name *similarity-matching objectives*. The transformation of dissimilar inputs in the NN depends on the optimization constraints. Despite using pairwise similarities, we still manage to derive online optimization algorithms.

Our focus is on unsupervised learning. This is not a hard constraint but rather a matter of priority. Whereas humans are clearly capable of supervised learning, most of our learning tasks lack big labeled data sets. On the mechanistic level, most neurons lack a clear supervision signal.

Background

Single-neuron online principal component analysis

In a seminal 1982 paper [5], Oja proposed that a biological neuron can be viewed as an implementation of a mathematical algorithm solving a computational objective. Specifically, he modeled a neuron by an online principal component analysis (PCA) algorithm. As PCA is a workhorse of data analysis used for dimensionality reduction, denoising, and latent factor discovery, Oja's model offers an algorithmic-level description of biological NNs.

Oja's single-neuron online PCA algorithm works as follows. At each time step, t, it receives an input data sample, $\mathbf{x}_t \in \mathbb{R}^n$. As our focus is on the online setting, we use the same variable, t, to measure time and index the data points. Then, the algorithm computes and outputs the corresponding top principal component value, $y_t \in \mathbb{R}$:

$$y_t \leftarrow \mathbf{w}_{t-1}^{\top}\mathbf{x}_t, \tag{1}$$

where $\mathbf{w}_{t-1} \in \mathbb{R}^n$ is the feature vector computed at time step, $t-1$. Here and ahead, lowercase italic letters are scalar variables, and lowercase boldfaced letters designate vectors. At the same time step t, after computing the principal component, the algorithm updates the (normalized) feature vector with a learning rate η:

$$\mathbf{w}_t \leftarrow \mathbf{w}_{t-1} + \eta(\mathbf{x}_t - \mathbf{w}_{t-1}y_t)y_t. \tag{2}$$

If data are drawn independently from a stationary distribution with a mean vector of zero, the feature vector, $\mathbf{w}$, converges to the eigenvector corresponding to the largest eigenvalue of input covariance [5].

The steps of the Oja algorithm in (1) and (2) naturally correspond to the operations of a biological neuron. Assuming that the components of the input vector are represented by the activities (firing rates) of the upstream neurons, (1) describes a weighted summation of the inputs by the output neuron. Such weighted summation can be naturally implemented by storing the components of feature vector, $\mathbf{w}$, in the corresponding synaptic weights. If the activation function is linear, the output, y_t, is simply the weighted sum. The weight update (2) is a biologically plausible local synaptic learning rule. The first term of the update, $\mathbf{x}_t y_t$, is proportional to the correlation of the presynaptic and postsynaptic neurons' activities. The second term, $\mathbf{w}_t y_t^2$, also local, asymptotically normalizes the synaptic weight vector to one. In neuroscience, synaptic weight updates proportional to the correlation of the presynaptic and postsynaptic neurons' activities are called *Hebbian*.

Minimization of the reconstruction error yields biologically implausible multineuron networks

Next, we would like to build on Oja's insightful identification of biological processes with the steps of the online PCA algorithms by computing multiple principal components using multineuron NNs. Instead of trying to extend the Oja model heuristically, we will derive them by using optimization of a principled objective function. Specifically, we postulate that the algorithm minimizes the reconstruction error, derive an online algorithm optimizing such objective, and map the steps of the algorithm onto biological processes.

In the conventional reconstruction error minimization approach, each data sample, $\mathbf{x}_t \in \mathbb{R}^n$, is approximated as a linear combination of each neuron's feature vector weighted by its activity [4]. Then the minimization of the reconstruction (or coding) error can be expressed as follows:

$$\min_{\mathbf{W}} \sum_{t=1}^{T} \min_{\mathbf{y}_t} \|\mathbf{x}_t - \mathbf{W}\mathbf{y}_t\|_2^2, \tag{3}$$

where matrix $\mathbf{W} \in \mathbb{R}^{n\times k}$, $k < n$, is a concatenation of feature column vectors and T is both the number of data samples and (in the online setting) the number of time steps.

In the offline setting, a solution to the optimization problem (3) is PCA: the columns of optimum $\mathbf{W}$ are a basis for the k-dimensional principal subspace [6]. Elements of $\mathbf{W}$ could be constrained to avoid unreasonably low or high values. In the online setting, (3) can be solved by alternating minimization [4]. After the arrival of data sample, $\mathbf{x}_t$, the feature vectors are kept fixed while the objective (3) is minimized with respect to the principal components by running the following gradient-descent dynamics until convergence:

$$\dot{\mathbf{y}}_t = \mathbf{W}_{t-1}^{\top}\mathbf{x}_t - \mathbf{W}_{t-1}^{\top}\mathbf{W}_{t-1}\mathbf{y}_t, \tag{4}$$

where $\cdot$ is a derivative with respect to a continuous-time variable that runs within a time step, t. Unlike a closed-form output of a single Oja neuron in (1), (4) is iterative.

After the output, $\mathbf{y}_t$ converges at the same time step, t, the feature vectors are updated according to the following gradient-descent step, with respect to $\mathbf{W}$ on the total objective:

$$\mathbf{W}_t \leftarrow \mathbf{W}_{t-1} + \eta(\mathbf{x}_t - \mathbf{W}_{t-1}\mathbf{y}_t)\mathbf{y}_t^{\top}. \tag{5}$$

If there were a single output channel, the algorithm in (4) and (5) would reduce to that in (1) and (2), provided that the scalar $\mathbf{W}_{t-1}^{\top}\mathbf{W}_{t-1}$ is rescaled to unity. In NN implementations of the algorithm in (4) and (5), feature vectors are represented by synaptic

weights and components by the activities of output neurons. Then (4) can be implemented by a single-layer NN, as seen in Figure 1(a), in which activity dynamics converge faster than the time interval between the arrival of successive data samples. The lateral connection weights, $-\mathbf{W}_{t-1}^{\top}\mathbf{W}_{t-1}$ decorrelate neuronal feature vectors by suppressing activities of correlated neurons.

However, implementing the update in (5) in the single-layer NN architecture, as shown in Figure 1(a), requires nonlocal learning rules, making it biologically implausible. Indeed, the last term in (5) implies that updating the weight of a synapse requires the knowledge of output activities of all other neurons that are not available to the synapse. Furthermore, the matrix of lateral connection weights, $-\mathbf{W}_{t-1}^{\top}\mathbf{W}_{t-1}$, in the last term of (4) is computed as a Gramian of feedforward weights, clearly a nonlocal operation. This problem is not limited to PCA, and it arises in networks of nonlinear neurons as well [4], [11].

To respect the local learning constraint, many authors constructed biologically plausible single-layer networks using heuristic local learning rules that were not derived from an objective function [12], [13]. However, we think that abandoning the optimization approach creates more problems than it solves. Alternatively, NNs with local learning rules can be derived if one introduces a second layer of neurons [14]. However, such architecture does not map naturally on biological networks.

We outlined how the conventional reconstruction approach fails to generate biologically plausible multineuron networks for online PCA. In the next section, we introduce an alternative approach that overcomes this limitation. Furthermore, this approach suggests a novel view of neural computation, leading to many interesting extensions.

Similarity-based approach to linear dimensionality reduction

In this section, we propose a different objective function for the optimization approach to constructing PCA NNs, which we term *similarity matching*. From this objective function, we derive an online algorithm implementable by an NN with local learning rules. Then, we introduce other similarity-based algorithms for linear dimensionality reduction that include more biological features, such as different neuron classes.

Similarity-matching objective function

We start by stating an objective function that will be used to derive NNs for linear dimensionality reduction. The similarity of a pair of inputs, $\mathbf{x}_t$ and $\mathbf{x}_{t'}$, both in $\mathbb{R}^n$, can be defined as their dot product, $\mathbf{x}_t^{\top}\mathbf{x}_{t'}$. Analogously, the similarity of a pair of outputs, which live in $\mathbb{R}^k$ with $k < n$, is $\mathbf{y}_t^{\top}\mathbf{y}_{t'}$. *Similarity matching,* as its name suggests, learns a representation where the similarity between each pair of outputs matches that of the corresponding inputs

$$\min_{\mathbf{y}_1,\ldots,\mathbf{y}_T} \frac{1}{T^2}\sum_{t=1}^{T}\sum_{t'=1}^{T}(\mathbf{x}_t^{\top}\mathbf{x}_{t'} - \mathbf{y}_t^{\top}\mathbf{y}_{t'})^2. \tag{6}$$

This offline objective function, previously employed for multidimensional scaling, is optimized by the projections of inputs onto the principal subspace of their covariance (i.e., performing PCA up to an orthogonal rotation). Furthermore, (6) has no local minima other than the principal subspace solution [7], [15]. The similarity-matching objective in (6) may seem like a strange choice for deriving an online algorithm implementable by an NN. In (6), pairs of inputs and outputs from different time steps interact with each other. However, in the online setting, an output must be computed at each time step without accessing inputs or outputs from other time steps. In addition, synaptic weights do not appear explicitly in (6), seemingly precluding mapping onto an NN.

Variable substitution trick

Both of the aforementioned concerns can be resolved by a simple math trick akin to completing the square [16]. We first focus on the cross-term in the expansion of the square in (6), which we call *similarity alignment*. By introducing a new variable, $\mathbf{W} \in \mathbb{R}^{k\times n}$, we can rewrite the cross-term

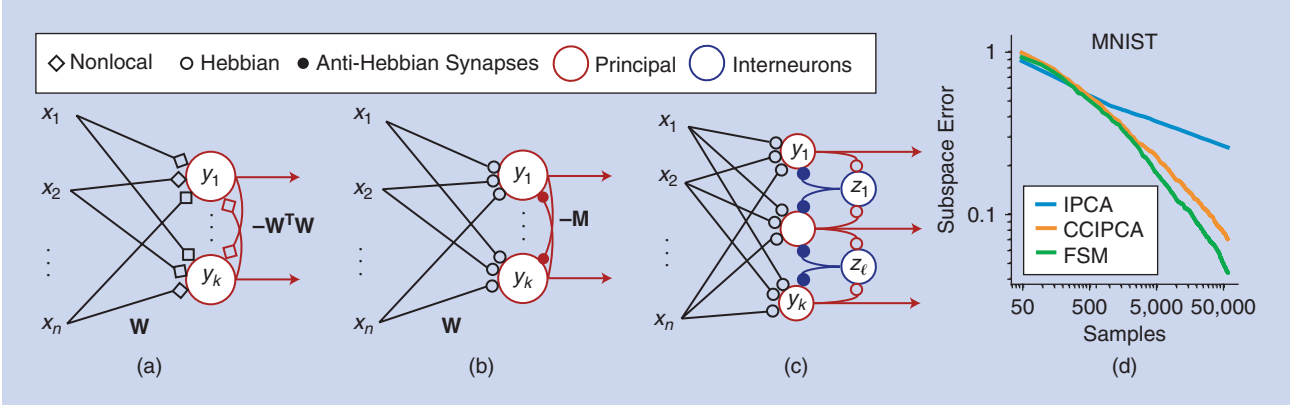

FIGURE 1. (a) The single-layer NN implementation of the multineuron online PCA algorithm derived using the reconstruction approach requires nonlocal learning rules. (b) A Hebbian/anti-Hebbian network derived from similarity matching. (c) A biologically plausible NN with multiple populations of neurons for whitening inputs, derived from a constrained similarity-alignment cost function [7]. (d) The performance of the FSM algorithm, compared to state-of-the-art online PCA algorithms [8], [9] for the MNIST data set reduced to $k = 16$ dimensions (see [10] for details and error definitions). MNIST: Modified National Institute of Standards and Technology database; IPCA: incremental principal component analysis; CCIPCA: candid covariance-free IPCA.

$$-\frac{1}{T^2}\sum_{t=1}^{T}\sum_{t'=1}^{T}\mathbf{y}_t^\top\mathbf{y}_{t'}\mathbf{x}_t^\top\mathbf{x}_{t'}=\min_{\mathbf{W}\in\mathbb{R}^{k\times n}}-\frac{2}{T}\sum_{t=1}^{T}\mathbf{y}_t^\top\mathbf{W}\mathbf{x}_t+\mathrm{Tr}\mathbf{W}^\top\mathbf{W}. \quad (7)$$

To prove this identity, we find optimal $\mathbf{W}$ by taking a derivative of the expression on the right with respect to $\mathbf{W}$ and setting it to zero, and then we substitute the optimal $\mathbf{W}^*=(1/T)\sum_{t=1}^{T}\mathbf{y}_t\mathbf{x}_t^\top$ back into the expression. Similarly, for the quartic $\mathbf{y}_t$ term in (6)

$$\frac{1}{T^2}\sum_{t=1}^{T}\sum_{t'=1}^{T}\mathbf{y}_t^\top\mathbf{y}_{t'}\mathbf{y}_t^\top\mathbf{y}_{t'}=\max_{\mathbf{M}\in\mathbb{R}^{k\times k}}\frac{2}{T}\sum_{t=1}^{T}\mathbf{y}_t^\top\mathbf{M}\mathbf{y}_t-\mathrm{Tr}\mathbf{M}^\top\mathbf{M}. \quad (8)$$

By substituting (7) and (8) into (6), we get

$$\min_{\mathbf{W}\in\mathbb{R}^{k\times n}}\max_{\mathbf{M}\in\mathbb{R}^{k\times k}}\frac{1}{T}\sum_{t=1}^{T}[2\mathrm{Tr}(\mathbf{W}^\top\mathbf{W})-\mathrm{Tr}(\mathbf{M}^\top\mathbf{M})+\min_{\mathbf{y}_t\in\mathbb{R}^{k\times 1}}l_t(\mathbf{W},\mathbf{M},\mathbf{y}_t)], \quad (9)$$

where

$$l_t(\mathbf{W},\mathbf{M},\mathbf{y}_t)=-4\mathbf{x}_t^\top\mathbf{W}^\top\mathbf{y}_t+2\mathbf{y}_t^\top\mathbf{M}\mathbf{y}_t. \quad (10)$$

In the resulting objective function in (9) and (10), optimal outputs at different time steps can be computed independently, making the problem amenable to an online algorithm. The price paid for this simplification is the appearance of the minimax optimization problem in variables $\mathbf{W}$ and $\mathbf{M}$. Minimization over $\mathbf{W}$ aligns output channels with the greatest variance directions of the input, and maximization over $\mathbf{M}$ diversifies the output by decorrelating output channels similarly to the Gramian, $\mathbf{W}^\top\mathbf{W}$, used previously. This competition in a gradient descent/ascent algorithm results in the principal subspace projection, which is the only stable fixed point of the corresponding dynamics [17].

Online algorithm and NN

We are ready to derive an algorithm for optimizing (6) online. First, we minimize (10) with respect to the output variables, $\mathbf{y}_t$, while holding $\mathbf{W}$ and $\mathbf{M}$ fixed using gradient-descent dynamics

$$\dot{\mathbf{y}}_t=\mathbf{W}\mathbf{x}_t-\mathbf{M}\mathbf{y}_t. \quad (11)$$

As before, dynamics in (11) converge within a single time step, t, and outputs $\mathbf{y}_t$. After the convergence of $\mathbf{y}_t$, we update $\mathbf{W}$ and $\mathbf{M}$ by the gradient descent of (7) and gradient ascent of (8), respectively

$$W_{ij}\leftarrow W_{ij}+\eta(y_ix_j-W_{ij}),\quad M_{ij}\leftarrow M_{ij}+\frac{\eta}{2}(y_iy_j-M_{ij}). \quad (12)$$

The algorithm in (11) and (12), first derived in [17], can be naturally implemented by a biologically plausible NN, as shown in Figure 1(b). Here, activity of the upstream neurons corresponds to input variables. Output variables are computed by the dynamics of activity in (11) in a single layer of neurons. Variables $\mathbf{W}$ and $\mathbf{M}$ are represented by the weights of synapses in feedforward and lateral connections, respectively. The learning rules in (12) are local. That is, the weight update, ΔW_{ij}, for the synapse between the jth input neuron and the ith output neuron depends only on the activities, x_j, of the jth input neuron and y_i of the ith output neuron, and the synaptic weight. In neuroscience, learning rules in (12) for synaptic weights $\mathbf{W}$ and $-\mathbf{M}$ [here, the minus sign indicates inhibitory synapses; see (11)] are called *Hebbian* and *anti-Hebbian*, respectively.

To summarize this section so far, starting with the similarity-matching objective, we derived a Hebbian/anti-Hebbian NN for dimensionality reduction. The minimax objective can be viewed as a zero-sum game played by the weights of feedforward and lateral connections [16], [18]. This demonstrates that synapses with local updates can still collectively work together to optimize a global objective. A similar, although not identical, NN was proposed by Földiak [12] heuristically. The advantage of our optimization approach is that the offline solution is known.

Although no proof of convergence exists in the online setting, the algorithm in (11) and (12) performs well on large-scale data. A recent paper [10] introduced an efficient, albeit nonbiological, modification of the similarity-matching algorithm, fast similarity matching (FSM), and demonstrated its competitiveness with the state-of-the-art principal subspace projection algorithms in both processing speed and convergence rate, as shown in Figure 1(d). FSM produces the same output $\mathbf{y}_t$ for each input $\mathbf{x}_t$ as similarity matching by optimizing (10) by matrix inversion, $\mathbf{y}_t=\mathbf{M}^{-1}\mathbf{W}\mathbf{x}_t$. It achieves extra computational efficiency by keeping in memory and updating the $\mathbf{M}^{-1}$ matrix rather than $\mathbf{M}$; see [10] for suggestions on the implementation of these algorithms. A package with implementations of these algorithms can be found at https://github.com/flatironinstitute/online_psp and https://github.com/flatironinstitute/online_psp_matlab.

Other similarity-based objectives and linear networks

As the algorithm in (11) and (12) and the NN in Figure 1(b) were derived from the similarity-matching objective in (6), they project data onto the principal subspace but do not necessarily recover principal components per se. To derive PCA algorithms, we modified the objective function in (6) to encourage orthogonality of $\mathbf{W}$ [19], [20]. Such algorithms are implemented by NNs of the same architecture, as in Figure 1(b), but with slightly different local learning rules.

Although the similarity-matching NN in Figure 1(b) relies on biologically plausible local learning rules, it lacks biological realism in several other ways. For example, computing output requires recurrent activity that must settle faster than the time scale of the input variation, which is unlikely in biology. To respect this biological constraint, we modified the dimensionality reduction algorithm to avoid recurrence [20].

Another nonbiological feature of the NN in Figure 1(b) is that the output neurons compete with each other by communicating via lateral connections. In biology, such interactions are not direct but are mediated by interneurons. To reflect these observations, we modified the objective function by introducing a whitening constraint

$$\min_{\mathbf{y}_1,\ldots,\mathbf{y}_T}-\frac{1}{T^2}\sum_{t=1}^{T}\sum_{t'=1}^{T}\mathbf{y}_t^\top\mathbf{y}_{t'}\mathbf{x}_t^\top\mathbf{x}_{t'},\quad \text{s.t.}\ \ \frac{1}{T}\sum_t\mathbf{y}_t\mathbf{y}_t^\top=\mathbf{I}_k, \quad (13)$$

where $\mathbf{I}_k$ is the k-by-k identity matrix. Then, by implementing the whitening constraint using the Lagrange formalism, we derived

NNs where interneurons appear naturally. Their activity is modeled by the Lagrange multipliers, $\mathbf{z}_t^\top \mathbf{z}_{t'}$ [see Figure 1(c)] [7]

$$\min_{\mathbf{y}_1,\ldots,\mathbf{y}_T} \max_{\mathbf{z}_1,\ldots,\mathbf{z}_T} -\frac{1}{T^2}\sum_{t=1}^{T}\sum_{t'=1}^{T}\mathbf{y}_t^\top \mathbf{y}_{t'}\mathbf{x}_t^\top \mathbf{x}_{t'} + \frac{1}{T^2}\sum_{t=1}^{T}\sum_{t'=1}^{T}\mathbf{z}_t^\top \mathbf{z}_{t'}(\mathbf{y}_t^\top \mathbf{y}_{t'} - \delta_{t,t'}), \tag{14}$$

where $\delta_{t,t'}$ is the Kronecker delta. Notice how (14) contains the **y**-**z** similarity-alignment term similar to (7). We can now derive learning rules for the **y**-**z** connections using the variable substitution trick, leading to the network in Figure 1(c). In addition to dimensionality reduction, such a network can whiten the input data. For details of this and other NN derivations, see [7].

Nonnegative similarity-matching objective and nonnegative independent component analysis

So far, we considered similarity-based NNs comprising linear neurons. But many interesting computations require nonlinearity, and biological neurons are not linear. To construct more realistic and powerful similarity-based NNs, we note that the output of biological neurons is nonnegative (the firing rate cannot be less than zero). Hence, we modified the optimization problem by requiring that the output of the similarity-matching cost function in (6) is nonnegative

$$\min_{\mathbf{y}_1,\ldots,\mathbf{y}_T \geq 0} \frac{1}{T^2}\sum_{t=1}^{T}\sum_{t'=1}^{T}(\mathbf{x}_t^\top \mathbf{x}_{t'} - \mathbf{y}_t^\top \mathbf{y}_{t'})^2. \tag{15}$$

Here, the number of output dimensions, k, may be greater than the number of input dimensions, n, leading to a dimensionally expanded representation. Equation (15) can be solved by the same online algorithm as in (6) except that the output variables are projected onto the nonnegative domain. Such an algorithm maps onto the same network and learning rules as in (12) [see Figure 1(b)] but with rectifying neurons [21], [25], [26], as shown in Figure 2(a).

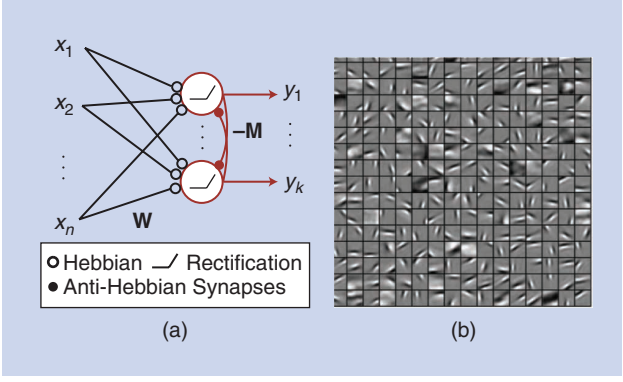

FIGURE 2. (a) A nonlinear Hebbian/anti-Hebbian network derived from NSM. (b) NSM learns edge filters from patches of whitened natural scenes. Learned filters are in small squares. See [21] for details of the simulations.

Table 1. The performance of unsupervised feature learning algorithms.

Algorithm	Accuracy
Convolutional NSM [24]	80.42%
k-means [22]	79.60%
Convolutional DBN [23]	78.90%

We list linear classification accuracy on CIFAR-10 using features extracted by NSM networks. We also list the best single-layer feature extractor (*k*-means) from an earlier study [22] and a deep-belief network (DBN) [23] on the same task. Detailed comparisons are available in [24].

A nonnegative similarity-matching (NSM) network learns features that are very different from PCA. For example, when the network is trained on whitened natural scenes, it extracts edge filters [21] (see Figure 2) as opposed to Fourier harmonics expected for a translationally invariant data set. Motivated by this observation, Bahroun and Soltoggio [24] developed a convolutional NSM network with multiple resolutions and used it as an unsupervised feature extractor for subsequent linear classification on the CIFAR-10 data set. They found that NSM NNs are superior to other single-layer unsupervised techniques [24], [27] (see Table 1).

As edge filters emerge also in the independent component analysis (ICA) of natural scenes [28], we investigated a connection of NSM with nonnegative ICA (NICA) used for blind source separation. The NICA problem is to recover independent, nonnegative, and well-grounded (finite probability density function in any positive neighborhood of zero) sources, $\mathbf{s}_t \in \mathbb{R}^d$, from observing only their linear mixture, $\mathbf{x}_t = \mathbf{A}\mathbf{s}_t$, where $\mathbf{A} \in \mathbb{R}^{n\times d}$ and $n \geq d$. Our solution of NICA is based on the observation that NICA can be solved in two steps [29], as shown in Figure 3(a). First, whiten the data and reduce it to d dimensions to obtain an orthogonal rotation of the sources (assuming that the mixing matrix is full rank). Second, find an orthogonal rotation of the whitened sources that yields a nonnegative output, as in Figure 3(a). The first step can be implemented by the whitening network in Figure 1(c). The second step can be implemented by the NSM network, as in Figure 2(a), because an orthogonal rotation does not affect dot-product similarities [25]. Therefore, NICA is solved by stacking the whitening and the NSM networks, as in Figure 3(b). This algorithm performs well compared to other popular NICA algorithms [25], as shown in Figure 3(c).

Nonnegative similarity-based networks for clustering and manifold tiling

NSM can also cluster well-segregated data [21], [30], and for data concentrated on manifolds, it can tile them [31]. To understand this behavior, we analyze the optimal solutions of nonnegative similarity-based objectives. Finding the optimal solution for a constrained similarity-based objective is rather challenging, as has been observed for the nonnegative matrix factorization problem. Here, we introduce a simplified similarity-based objective that allows us to make progress with the analysis and that admits an intuitive interpretation. First, we address the simpler clustering task, which, for highly segregated data, has a straightforward optimal solution. Second, we address manifold learning by viewing it as a soft-clustering problem.

A similarity-based cost function and NN for clustering

The key to our analysis is formulating a similarity-based cost function, an optimization of which will yield an online algorithm and an NN for clustering. The algorithm should assign inputs $\mathbf{x}_t$

to k clusters based on pairwise similarities and output cluster assignment indices $\mathbf{y}_t$. To arrive at a cost function, first consider a single pair of data points, $\mathbf{x}_1$ and $\mathbf{x}_2$. If $\mathbf{x}_1^\top \mathbf{x}_2 < \alpha$, where α is a preset threshold, then the points should be assigned to separate clusters—that is, $\mathbf{y}_1 = [1, 0]^\top$ and $\mathbf{y}_2 = [0, 1]^\top$—setting output similarity, $\mathbf{y}_1^\top \mathbf{y}_2$, to 0. If $\mathbf{x}_1^\top \mathbf{x}_2 > \alpha$, then the points are assigned to the same cluster, such as $\mathbf{y}_1 = \mathbf{y}_2 = [1, 0]^\top$. Such $\mathbf{y}_1$ and $\mathbf{y}_2$ are optimal solutions (although not unique) to the following optimization problem:

$$\min_{\mathbf{y}_1 \geq 0, \mathbf{y}_2 \geq 0} (\alpha - \mathbf{x}_1^\top \mathbf{x}_2) \mathbf{y}_1^\top \mathbf{y}_2, \quad \text{s.t.} \quad \|\mathbf{y}_1\|_2 \leq 1, \|\mathbf{y}_2\|_2 \leq 1. \tag{16}$$

To obtain an objective function that would cluster the whole data set of T inputs, we simply sum (16) over all possible input pairs

$$\min_{\mathbf{y}_1 \geq 0, \ldots, \mathbf{y}_T \geq 0} \sum_{t=1}^{T} \sum_{t'=1}^{T} (\alpha - \mathbf{x}_t^\top \mathbf{x}_{t'}) \mathbf{y}_t^\top \mathbf{y}_{t'} \quad \text{s.t.} \quad \|\mathbf{y}_1\|_2 \leq 1, \ \ldots \ , \|\mathbf{y}_T\|_2 \leq 1. \tag{17}$$

Does optimization of (17) produce the desired clustering output? This depends on the data set. If a threshold, α, exists such that the similarities of all pairs within the same cluster are greater and similarities of pairs from different clusters are less than α, then the cost function in (17) is minimized by the desired hard-clustering output, provided that k is greater than or equal to the number of clusters. To solve the objective of (17) in the online setting, we introduce the constraints in the cost via Lagrange multipliers. Using the variable substitution trick, we can derive an NN implementation of this algorithm [31], as shown in Figure 4(a). The algorithm operates with local Hebbian and anti-Hebbian learning rules, whose functional form is equivalent to (12).

Manifold-tiling solutions

In many real-world problems, data points are not well-segregated, but they lie on low-dimensional manifolds. For such data, the optimal solution of the objective in (17) effectively tiles the data manifold [31]. We can understand such optimal solutions using soft clustering (i.e., clustering where each stimulus may be assigned to more than one cluster and assignment indices are real numbers between zero and one). Each output neuron is characterized by the weight vector of incoming synapses, which defines a centroid in the input data space. The response of a neuron is maximum when data fall on the centroid and decay away from it. Manifold-tiling solutions for several data sets are shown in Figure 5.

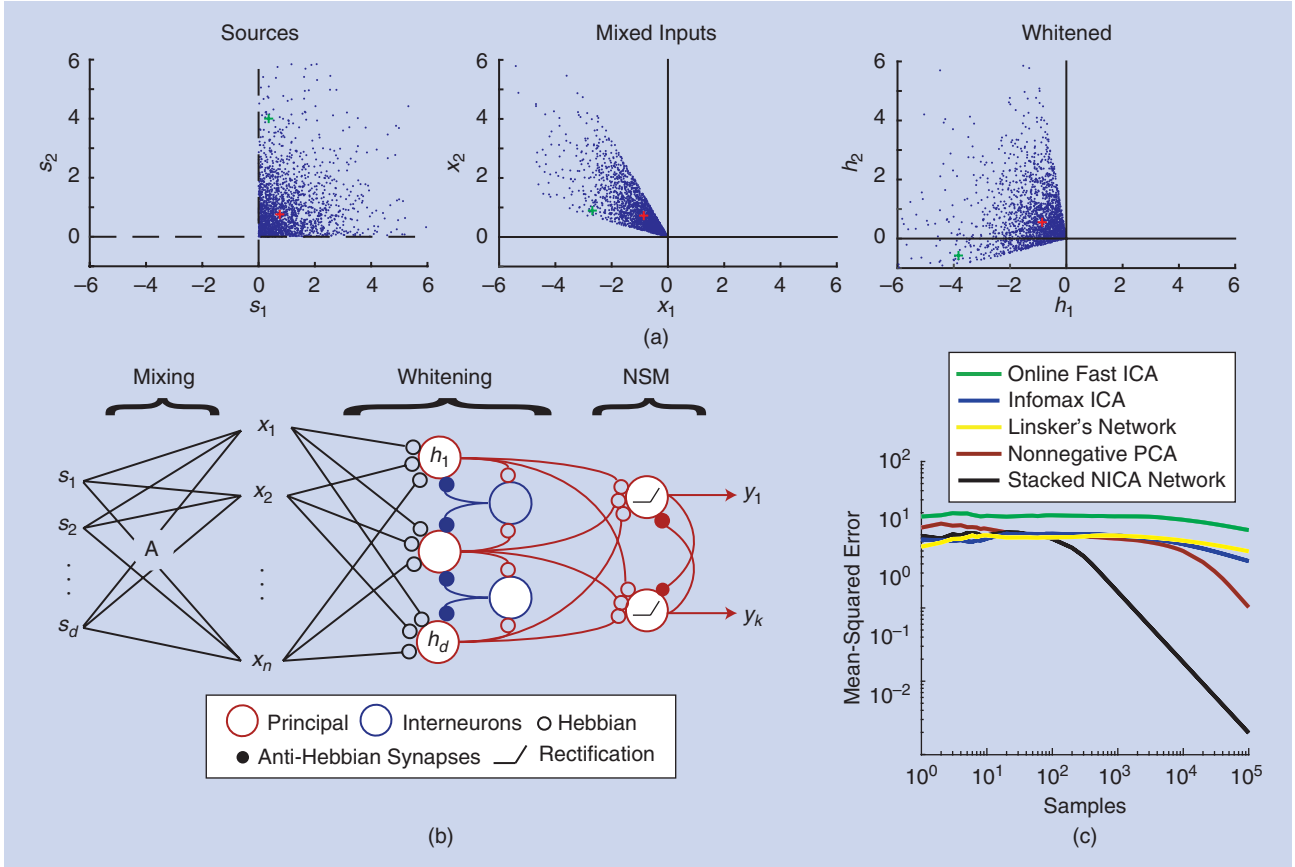

FIGURE 3. (a) An illustration of Plumbley's nonnegative ICA algorithm. Two source channels (left, each blue point shows one source vector, $\mathbf{S}_t$) are linearly mixed to a 2D mixture, which are the inputs to the algorithm (middle). Whitening (right) yields an orthogonal rotation of the sources. Sources are then recovered by solving the NSM problem. Green and red plus signs track two source vectors across mixing and whitening stages. (b) A stacked network for NICA. The network sees the mixtures $\mathbf{x}_t$ and aims to recover sources $\mathbf{S}_t$ at its output $\mathbf{y}_t$. (c) The performance of the stacked network for NICA (black) in reconstructing the source vector on a 10-dimensional artificial data set (see [25] for details). The performance metric is the squared error between the network's output and the original sources, averaged over all samples until that point.

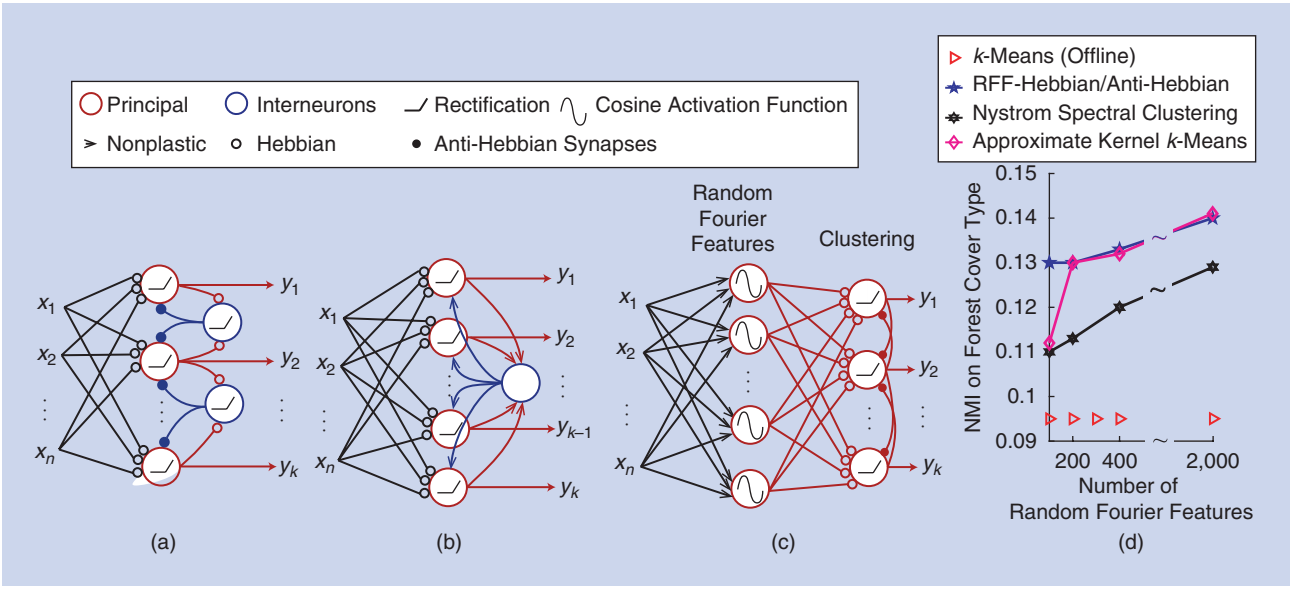

FIGURE 4. Biologically plausible NNs for clustering and manifold learning. (a) A biologically plausible excitatory-inhibitory NN implementation of the algorithm. In this version, anti-Hebbian synapses operate at a faster time scale than Hebbian synapses [31]. (b) The hard and soft k-means networks. Rectified neurons are perfect (hard k-means) or leaky (soft k-means) integrators. They have learned (homeostatic) activation thresholds and ephaptic couplings. (c) When augmented with a hidden nonlinear layer, the presented networks perform clustering in the nonlinear feature space. Shown is the NN of [32], where the hidden layer is formed of random Fourier features (RFFs) [33] to obtain a low-rank approximation to a Gaussian kernel. The two-layer NN operates as an online kernel clustering algorithm. (d) The two-layer NN performs on par to other state-of-the-art kernel clustering algorithms [32]. Shown is performance on forest cover-type data set. The figure is modified from [32]. NMI: normalized mutual information.

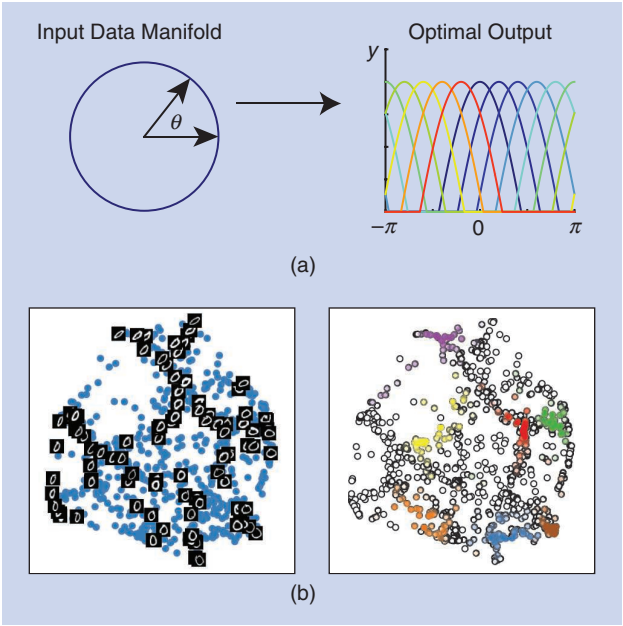

FIGURE 5. Analytical and numerical manifold-tiling solutions of (17) for representative data sets provide accurate and useful representations. (a) A circular manifold (left) is tiled by overlapping localized receptive fields (right). In the continuum limit ($k \to \infty$), receptive fields are truncated cosines of the polar angle, θ [31]. Similar analytical and numerical results are obtained for a spherical 3D manifold and SO(3) (see [31]). (b) The learning of the manifold of the zero digit from the MNIST data set by tiling the manifold with overlapping localized receptive fields. On the left is 2D linear embedding (PCA) of the outputs. The data are organized according to different visual characteristics of the handwritten digit (e.g., orientation and elongation). On the right are sample receptive fields over the low-dimensional embedding.

We can prove this result analytically by taking advantage of the convex relaxation in the limit of an infinite number of output dimensions (i.e., $k \to \infty$). Indeed, if we introduce Gramians $\mathbf{D}$, such that $D_{tt'} = \mathbf{x}_t^\top \mathbf{x}_{t'}$, and $\mathbf{Q}$, such that $Q_{tt'} = \mathbf{y}_t^\top \mathbf{y}_{t'}$, and do not specify the dimensionality of $\mathbf{y}$ by leaving the rank of $\mathbf{Q}$ open, we can rewrite (17) as

$$\min_{\substack{\mathbf{Q} \in \mathcal{CP}^T \\ \operatorname{diag}\mathbf{Q} \le 1}} -\operatorname{Tr}((\mathbf{D} - \alpha \mathbf{E})\mathbf{Q}), \tag{18}$$

where $\mathbf{E}$ is a matrix whose elements are all ones, and the cone of completely positive $T \times T$ matrices (i.e., matrices $\mathbf{Q} \equiv \mathbf{Y}^\top \mathbf{Y}$ with $\mathbf{Y} \ge 0$) is denoted by $\mathcal{CP}^T$ [34]. Redefining the variables makes the optimization problem convex. For arbitrary data sets, optimization problems in $\mathcal{CP}^T$ are often intractable for large T [34] despite the convexity. However, for symmetric data sets, such as circle, two sphere and SO(3), we can optimize (18) by analyzing the Karush–Kuhn–Tucker conditions [31], as seen in Figure 5(a).

Other similarity-based NNs for clustering and manifold-tiling

A related problem to the objective in (17) is the previously studied convex semidefinite programming relaxation of community detection in graphs [35], which is closely related to clustering. The semidefinite program is related to (18) by requiring the nonnegativity of $\mathbf{Q}$ instead of the nonnegativity of $\mathbf{Y}$

$$\min_{\mathbf{Q} \ge 0,\, \mathbf{Q} \ge 0,\, \operatorname{diag}\mathbf{Q} \le 1} -\operatorname{Tr}((\mathbf{D} - \alpha \mathbf{E})\mathbf{Q}). \tag{19}$$

Although we chose to present our similarity-based NN approach to clustering and manifold tiling through the cost

function in (17), similar results can be obtained for other versions of similarity-based clustering objective functions. The NSM cost function in (15) and the NN derived from it [Figure 2(a)] can be used for clustering and manifold learning as well [21], [30], [31]. The *k*-means cost function can be cast into a similarity-based form, and an NN [Figure 4(b)] can be derived for its online implementation [36]. We introduced a soft *k*-means cost, also a relaxation of another semidefinite program for clustering [37], and an associated NN, as seen in Figure 4(b) [36], and showed that they can perform manifold tiling [38].

The algorithms we discussed operate with the dot product as a measure of similarity in the inputs. By augmenting the presented NNs by an initial random, nonlinear projection layer [Figure 4(c)], it is possible to implement nonlinear similarity measures associated with certain kernels [32]. A clustering algorithm using this idea is shown to perform on par with other online kernel clustering algorithms [32], as seen in Figure 4(d).

Discussion

To overcome the nonlocality of the learning rule in NNs derived from the reconstruction error minimization, we proposed a new class of cost functions called *similarity based*. To summarize, the first term in the similarity-based cost functions,

$$\min_{\forall t, \mathbf{y}_t \in \Omega} \left[-\sum_{t=1}^{T} \sum_{t'=1}^{T} \mathbf{y}_t^\top \mathbf{y}_{t'} \mathbf{x}_t^\top \mathbf{x}_{t'} + f(\mathbf{y}_1, ..., \mathbf{y}_T) \right], \tag{20}$$

is the covariance of the similarity of the outputs and the similarity of the inputs. Hence, the name *similarity-based cost functions*. Previously, such objectives were used in linear kernel alignment [39]. Our key observation is that optimization of objective functions containing such a term in the online setting gives rise to local synaptic learning rules, as in (7) [16].

To derive biologically plausible NNs from (20), one must choose not just the first term but also the function, f, and the optimization constraints, Ω, so that the online optimization algorithm is implementable by biological mechanisms. We and others have identified a whole family of such functions and constraints (see Table 2), some of which were reviewed in this article. As a result, we can relate many features of biological NNs to different terms and constraints in similarity-based cost functions and, hence, give them computational interpretations.

Our framework provides a systematic procedure to design novel NN algorithms by formulating a learning task using similarity-based cost functions. As evidenced by the high-performing algorithms discussed in this article, our procedure of incorporating biological constraints does not impede but rather facilitates the design process by limiting the algorithm search to a useful part of the NN algorithm space. The locality of learning rules in similarity-based NNs makes them naturally suitable for implementation on adaptive neuromorphic systems, which have already been explored in custom analog arrays [3]. For broader use in the rapidly growing world of low-power, spike-based hardware with on-chip learning [2], similarity-based NNs were missing a key ingredient: spiking neurons. Very recent work [26] developed a spiking version of the NSM and took a step toward neuromorphic applications.

Table 2. The current list of objectives, regularizers, and constraints that define a similarity-based optimization problem and are solvable by an NN with local learning.

Optimization Feature	Biological Feature
Similarity (anti)alignment	(Anti-)Hebbian plasticity [16], [17]
Nonnegativity constraint	Rectifying neuron activation function [18], [21]
Sparsity regularizer	Adaptive neural threshold [40]
Constrained output correlation matrix	Adaptive lateral weights [7], [18]
Constrained PSD output Gramian	Anti-Hebbian interneurons [7]
Copositive output Gramian	Anti-Hebbian inhibitory neurons [31]
Constrained activity l_1-norm	Giant interneuron [36]

Despite the successes of similarity-based NNs, many interesting challenges remain. First, whereas numerical experiments indicate that our online algorithms perform well, most of them lack global convergence proofs. Even for PCA networks, we can prove linear stability of the desired solution only in the stochastic approximation setting. Second, motivated by biological learning, which is mostly unsupervised, we focused on unsupervised learning. However, supervision, or reinforcement, does take place in the brain. Therefore, it is desirable to extend our framework to supervised, semisupervised, and reinforcement learning settings. Such extensions may be valuable as general purpose machine-learning algorithms.

Third, whereas most sensory stimuli are correlated time series, we assumed that data points at different times are independent. How are temporal correlations analyzed by NNs? Solving this problem is important both for modeling brain function and developing general-purpose machine-learning algorithms. Fourth, another challenge is stacking similarity-based NNs. A heuristic approach to stacking yields promising results [24]. However, except for the NICA problem introduced in the "Nonnegative Similarity-Matching Objective and Nonnegative Independent Component Analysis" section, we do not have a theoretical understanding of how and why to stack similarity-based NNs. Finally, neurons in biological NNs signal each other using all-or-none spikes, or action potentials, as opposed to real-valued signals we considered. Is there an optimization theory accounting for spiking in biological NNs?

Acknowledgments

We thank our collaborators Anirvan Sengupta, Mariano Tepper, Andrea Giovannucci, Alex Genkin, Victor Minden, Sreyas Mohan, and Yanis Bahroun for their contributions; Andrea Soltoggio for discussions; and Siavash Golkar and Alper Erdogan for commenting on the manuscript. This work was supported in part by a gift from the Intel Corporation.

Authors

Cengiz Pehlevan (cpehlevan@seas.harvard.edu) received his undergraduate degrees in physics and electrical engineering from Bogazici University, Istanbul, Turkey, in 2004 and his

doctorate in physics from Brown University, Providence, Rhode Island, in 2011. He was a Swartz Fellow at Harvard University, Cambridge, Massachusetts, a postdoctoral associate at Janelia Research Campus, Ashburn, Virginia, and a research scientist in the neuroscience group at the Flatiron Institute, New York. He is an assistant professor of applied mathematics at the Harvard John A. Paulson School of Engineering and Applied Sciences. His research interests include theoretical neuroscience and neural computation.

Dmitri B. Chklovskii (dchklovskii@flatironinstitute.org) received his Ph.D. degree in theoretical physics from the Massachusetts Institute of Technology, Cambridge. From 1994 to 1997, he was a junior fellow at the Harvard Society of Fellows. He transitioned to neuroscience at the Salk Institute for Biological Studies, San Diego, California. From 1999 to 2007, he was an assistant/associate professor at Cold Spring Harbor Laboratory, New York. Then, as a group leader at Janelia Research Campus, Ashburn, Virginia, he led the team that assembled the largest-at-the-time connectome. He is a group leader for neuroscience at the Flatiron Institute, New York, and a research associate professor at New York University Medical Center. Informed by the function and structure of the brain, his group develops online-learning algorithms for big data.

References

[1] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychol. Rev.*, vol. 65, no. 6, pp. 386–408, 1958. doi: 10.1037/h0042519.

[2] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi et al., "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018. doi: 10.1109/MM.2018.112130359.

[3] J. H. Poikonen and M. Laiho, "A mixed-mode array computing architecture for online dictionary learning," in *Proc. 2017 IEEE Int. Symp. Circuits and Systems*, pp. 1–4.

[4] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, pp. 607–609, 1996. doi: 10.1038/381607a0.

[5] E. Oja, "Simplified neuron model as a principal component analyzer," *J. Math. Biology*, vol. 15, no. 3, pp. 267–273, 1982. doi: 10.1007/BF00275687.

[6] M. Udell, C. Horn, R. Zadeh, and S. Boyd, "Generalized low rank models," *Foundations and Trends in Machine Learning*, vol. 9, no. 1, pp. 1–118, 2016. doi: 10.1561/2200000055.

[7] C. Pehlevan and D. B. Chklovskii, "A normative theory of adaptive dimensionality reduction in neural networks," in *Proc. 28th Int. Conf. Neural Information Processing Systems*, 2015, pp. 2269–2277.

[8] R. Arora, A. Cotter, K. Livescu, and N. Srebro, "Stochastic optimization for PCA and PLS," in *Proc. 2012 50th Annu. Allerton Conf. Communication, Control, and Computing*, pp. 861–868.

[9] J. Weng, Y. Zhang, and W.-S. Hwang, "Candid covariance-free incremental principal component analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 8, pp. 1034–1040, 2003.

[10] A. Giovannucci, V. Minden, C. Pehlevan, and D. B. Chklovskii, "Efficient principal subspace projection of streaming data through fast similarity matching," in *Proc. 2018 IEEE Int. Conf. Big Data (Big Data)*, pp. 1015–1022.

[11] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999. doi: 10.1038/44565.

[12] P. Földiak, "Adaptive network for optimal linear feature extraction," in *Proc. Int. 1989 Joint Conf. Neural Networks*, pp. 401–405.

[13] K. I. Diamantaras and S. Y. Kung, *Principal Component Neural Networks: Theory and Applications*. New York: Wiley, 1996.

[14] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vision Res.*, vol. 37, no. 23, pp. 3311–3325, 1997. doi: 10.1016/S0042-6989(97)00169-7.

[15] R. Ge, J. D. Lee, and T. Ma, "Matrix completion has no spurious local minimum," in *Proc. 30th Int. Conf. Neural Information Processing Systems*, 2016, pp. 2973–2981.

[16] C. Pehlevan, A. M. Sengupta, and D. B. Chklovskii, "Why do similarity matching objectives lead to Hebbian/anti-Hebbian networks?" *Neural Comput.*, vol. 30, no. 1, pp. 84–124, 2018. doi: 10.1162/neco_a_01018.

[17] C. Pehlevan, T. Hu, and D. Chklovskii, "A Hebbian/anti-Hebbian neural network for linear subspace learning: A derivation from multidimensional scaling of streaming data," *Neural Comput.*, vol. 27, no. 7, pp. 1461–1495, 2015. doi: 10.1162/NECO_a_00745.

[18] H. S. Seung and J. Zung, A correlation game for unsupervised learning yields computational interpretations of Hebbian excitation, anti-Hebbian inhibition, and synapse elimination. 2017. [Online]. Available: https://arxiv.org/abs/1704.00646

[19] C. Pehlevan and D. B. Chklovskii, "Optimization theory of Hebbian/anti-Hebbian networks for PCA and whitening," in *Proc. 2015 53rd Annu. Allerton Conf. Communication, Control, and Computing*, pp. 1458–1465.

[20] V. Minden, C. Pehlevan, and D. B. Chklovskii, "Biologically plausible online principal component analysis without recurrent dynamics," in *Proc. 2018 52nd Asilomar Conf. Signals, Systems, and Computers*, pp. 104–111.

[21] C. Pehlevan and D. B. Chklovskii, "A Hebbian/anti-Hebbian network derived from online non-negative matrix factorization can cluster and discover sparse features," in *Proc. 2014 48th Conf. Signals, Systems and Computers*, pp. 769–775.

[22] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. 14th Int. Conf. Artificial Intelligence and Statistics*, 2011, pp. 215–223.

[23] A. Krizhevsky and G. Hinton, "Convolutional deep belief networks on CIFAR-10," unpublished.

[24] Y. Bahroun and A. Soltoggio, "Online representation learning with single and multi-layer Hebbian networks for image classification," in *Proc. Int. Conf. Artificial Neural Networks*, 2017, pp. 354–363.

[25] C. Pehlevan, S. Mohan, and D. B. Chklovskii, "Blind nonnegative source separation using biological neural networks," *Neural Comput.*, vol. 29, no. 11, pp. 2925–2954, 2017. doi: 10.1162/neco_a_01007.

[26] C. Pehlevan, "A spiking neural network with local learning rules derived from nonnegative similarity matching," in *Proc. 2019 IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pp. 7958–7962.

[27] Y. Bahroun, E. Hunsicker, and A. Soltoggio, "Building efficient deep Hebbian networks for image classification tasks," in *Proc. Int. Conf. Artificial Neural Networks*, 2017, pp. 364–372.

[28] A. J. Bell and T. J. Sejnowski, "The "independent components" of natural scenes are edge filters," *Vision Res.*, vol. 37, no. 23, pp. 3327–3338, 1997. doi: 10.1016/S0042-6989(97)00121-1.

[29] M. Plumbley, "Conditions for nonnegative independent component analysis," *IEEE Signal Process. Lett.*, vol. 9, no. 6, pp. 177–180, 2002. doi: 10.1109/LSP.2002.800502.

[30] D. Kuang, C. Ding, and H. Park, "Symmetric nonnegative matrix factorization for graph clustering," in *Proc. 2012 SIAM Int. Conf. Data Mining*, pp. 106–117.

[31] A. Sengupta, C. Pehlevan, M. Tepper, A. Genkin, and D. Chklovskii, "Manifold -tiling localized receptive fields are optimal in similarity-preserving neural networks," in *Proc. 2018 IEEE Int. Conf. Big Data (Big Data), Advances in Neural Information Processing Systems,* pp. 7080–7090.

[32] Y. Bahroun, E. Hunsicker, and A. Soltoggio, "Neural networks for efficient nonlinear online clustering," in *Proc. Int. Conf. Neural Information Processing*, 2017, pp. 316–324.

[33] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Proc. 20th Int. Conf. Neural Information Processing Systems*, 2008, pp. 1177–1184.

[34] A. Berman and N. Shaked-Monderer, *Completely Positive Matrices*. Singapore: World Scientific, 2003.

[35] T. T. Cai and X. Li, "Robust and computationally feasible community detection in the presence of arbitrary outlier nodes," *Ann. Statist.*, vol. 43, no. 3, pp. 1027–1059, 2015. doi: 10.1214/14-AOS1290.

[36] C. Pehlevan, A. Genkin, and D. B. Chklovskii, "A clustering neural network model of insect olfaction," in *Proc. 2017 51st Asilomar Conf. Signals, Systems, and Computers*, pp. 593–600.

[37] B. Kulis, S. Basu, I. Dhillon, and R. Mooney, "Semi-supervised graph clustering: A kernel approach," *Mach. Learn.*, vol. 74, no. 1, pp. 1–22, 2009. doi: 10.1007/s10994-008-5084-4.

[38] M. Tepper, A. M. Sengupta, and D. Chklovskii, "Clustering is semidefinitely not that hard: Nonnegative SDP for manifold disentangling," *J. Mach. Learning Res.*, vol. 19, no. 1, pp. 3208–3237, 2018.

[39] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. S. Kandola, "On kernel-target alignment," in *Proc. 14th Int. Conf. Neural Information Processing Systems: Natural and Synthetic*, 2002, pp. 367–373.

[40] T. Hu, C. Pehlevan, and D. B. Chklovskii, "A Hebbian/anti-Hebbian network for online sparse dictionary learning derived from symmetric matrix factorization," in *Proc. 2014 48th Asilomar Conf. Signals, Systems and Computers*, pp. 613–619.

SP