frontiers in Computational Neuroscience

ORIGINAL RESEARCH
published: 30 June 2020
doi: 10.3389/fncom.2020.00055

Neurons as Canonical Correlation Analyzers

Cengiz Pehlevan[1]*, *Xinyuan Zhao*[2], *Anirvan M. Sengupta*[3] and *Dmitri Chklovskii*[4,5]

[1] John A. Paulson School of Engineering and Applied Sciences and Center for Brain Science, Harvard University, Cambridge, MA, United States, [2] Center for Neural Science, New York University, New York, NY, United States, [3] Rutgers, The State University of New Jersey, New Brunswick, NJ, United States, [4] Center for Computational Biology, Flatiron Institute, New York, NY, United States, [5] Langone Medical Center, New York University, New York, NY, United States

Normative models of neural computation offer simplified yet lucid mathematical descriptions of murky biological phenomena. Previously, online Principal Component Analysis (PCA) was used to model a network of single-compartment neurons accounting for weighted summation of upstream neural activity in the soma and Hebbian/anti-Hebbian synaptic learning rules. However, synaptic plasticity in biological neurons often depends on the integration of synaptic currents over a dendritic compartment rather than total current in the soma. Motivated by this observation, we model a pyramidal neuronal network using online Canonical Correlation Analysis (CCA). Given two related datasets represented by distal and proximal dendritic inputs, CCA projects them onto the subspace which maximizes the correlation between their projections. First, adopting a normative approach and starting from a single-channel CCA objective function, we derive an online gradient-based optimization algorithm whose steps can be interpreted as the operation of a pyramidal neuron. To model networks of pyramidal neurons, we introduce a novel multi-channel CCA objective function, and derive from it an online gradient-based optimization algorithm whose steps can be interpreted as the operation of a pyramidal neuron network including its architecture, dynamics, and synaptic learning rules. Next, we model a neuron with more than two dendritic compartments by deriving its operation from a known objective function for multi-view CCA. Finally, we confirm the functionality of our networks via numerical simulations. Overall, our work presents a simplified but informative abstraction of learning in a pyramidal neuron network, and demonstrates how such networks can integrate multiple sources of inputs.

Keywords: neural networks, Canonical Correlation Analysis (CCA), Hebbian plasticity, pyramidal neuron, biologically plausible learning

OPEN ACCESS

Edited by:
Paul Miller,
Brandeis University, United States

Reviewed by:
Guosheng Yi,
Duke University, United States
Paul R. Adams,
Stony Brook University, United States

***Correspondence:**
Cengiz Pehlevan
cpehlevan@seas.harvard.edu

Received: *22 December 2019*
Accepted: *18 May 2020*
Published: *30 June 2020*

Citation:
Pehlevan C, Zhao X, Sengupta AM and Chklovskii D (2020) Neurons as Canonical Correlation Analyzers. Front. Comput. Neurosci. 14:55. doi: 10.3389/fncom.2020.00055

1. INTRODUCTION

As neural networks evolved for competitive behaviorally-relevant tasks, it is natural to model them using a normative approach, where one starts from a principled objective function and derives an online optimization algorithm that models an operation of a neural system. Such approach often leads to simplified and interpretable models for complex biological phenomena. Perhaps, the most famous example of this is modeling a neuron as an online PCA algorithm (Oja, 1982). This model accounts for the weighted summation of synaptic inputs in a single, linear, point

(single-compartment) neuron and Hebbian synaptic plasticity (synaptic weight is proportional to the correlation of pre- and post-synaptic activity). Recently, Oja's model of a single neuron has been extended to a network of neurons by deriving it from a multi-channel-PCA objective function (Pehlevan et al., 2015). In addition to the phenomena explained by the Oja model, the extension (Pehlevan et al., 2015) accounted for the anti-Hebbian learning rules of the lateral synaptic connections (synaptic weight is proportional to the negative of the correlation between pre- and post-synaptic activity). Because of their analytical tractability, the output of such network models can be predicted for any input.

However, there is mounting experimental evidence that a point neuron is an extreme oversimplification. Many neurons contain multiple segregated dendritic compartments which integrate synaptic inputs separately and can each have a membrane potential different from that of the soma (Poirazi and Mel, 2001; Polsky et al., 2004). Neuronal firing often requires coincident input onto distal and proximal dendrites (Larkum et al., 2009; Larkum, 2013). Moreover, synaptic plasticity is the function of the neuronal output (spiking) and membrane potential in the corresponding compartment (Bittner et al., 2015).

Multiple influential, biophysically grounded models have been proposed to describe the computational role of dendritic compartmentalization and nonlinearities (Poirazi and Mel, 2001; Poirazi et al., 2003; Polsky et al., 2004; Jadi et al., 2014; Urbanczik and Senn, 2014; Alemi et al., 2017; Guerguiev et al., 2017; Haga and Fukai, 2017). These models provide mechanistic and computational descriptions of the active membrane processes that account for experimental observations.

In this paper, we start from computational principles first and apply a normative approach to networks of multi-compartment neurons to derive algorithmic descriptions of their function. Specifically, we propose to model information processing in pyramidal neurons as online CCA algorithms (Hotelling, 1992; Yang et al., 2019). Because we derive these models from principled objective functions, we can predict the output of the network for any input analytically.

CCA is a natural choice for extending the normative modeling of point-neuron networks as PCA algorithms (Oja, 1982, 1992; Földiak, 1989; Rubner and Tavan, 1989; Sanger, 1989; Pehlevan et al., 2015) to multi-compartment neuron networks. In such neurons different dendritic compartments receive inputs from different sources, for example, top-down and bottom-up inputs, or inputs from multiple modalities. Information from these multiple sources can be integrated using CCA by linearly projecting each of the datasets onto low dimensional subspaces with maximal cross-correlation (Parise and Ernst, 2016). For Gaussian data, CCA can be optimal for various objectives such as, for example, mutual information (Chechik et al., 2005), but its utility extends to real-world applications as well (Painsky and Tishby, 2017).

Previous work on neural network implementations of CCA include (Lai and Fyfe, 1999; Pezeshki et al., 2003; Vía et al., 2007; Haga and Fukai, 2017), all of which use point neurons and non-local learning rules. In biology, synaptic learning rules must be local i.e., depend only on the information available in the corresponding pre- and post-synaptic neurons. In Lai and Fyfe (1999), there are two or three output neurons, and the synapses that innervate one of them need to have access to the activity of another neuron to update their strength. In Pezeshki et al. (2003), an estimate for the covariance matrix of the data needs to be stored and be globally available to all neurons. In Vía et al. (2007), update rules again involve numerous nonlocal computations.

We make the following contributions:

1. We interpret an existing online algorithm for single-channel CCA, as the operation of a two-compartment pyramidal neuron.
2. We derive a pyramidal neuron network from a novel objective function for multi-channel CCA.
3. We derive a model of a neuron with more than two dendritic compartments from an objective function for a single-channel multi-view CCA.

The rest of the paper is organized as follows. In section 2, we derive an online algorithm from the single-channel CCA objective function and map it onto the operation of a pyramidal neuron. In section 3, we explain why the standard objective function for multi-channel CCA yields a biologically implausible network of pyramidal neurons. In section 4, we propose a novel objective function for multi-channel CCA from which we derive a biologically plausible network of pyramidal neurons. In section 5, we derive a model of a neuron with more than two dendritic compartments from a single-channel multi-view CCA objective function. In section 6, we provide numerical simulation results for these neuronal algorithms.

2. A PYRAMIDAL NEURON AS AN ONLINE SINGLE-CHANNEL CCA ALGORITHM

Given two datasets with the same number of data points but in different subspaces, a single-channel CCA projects them onto a common line so that the two unit-variance projections are maximally correlated. Let us represent each pair of data points as $\mathbf{x}_t \in \mathbb{R}^n$ and $\mathbf{y}_t \in \mathbb{R}^m$, $t = 1, \ldots, T$. The CCA objective function has the following form:

$$\max_{\mathbf{a}\in\mathbb{R}^n, \mathbf{b}\in\mathbb{R}^m} \frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{a}^\top \mathbf{x}_t\right)\left(\mathbf{b}^\top \mathbf{y}_t\right), \quad \text{s.t. } \frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{a}^\top \mathbf{x}_t\right)^2 = 1, \frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{b}^\top \mathbf{y}_t\right)^2 = 1. \quad (1)$$

In **Supplementary Information A**, we provide CCA's analytical solutions and their various properties for completeness.

We would like to rewrite (1) in a form amenable to a biologically plausible online algorithm. One way of doing so is

by completing a square with constant terms (see constraints):

$$\begin{aligned}
&\arg\max_{\mathbf{a},\mathbf{b}} \frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{a}^\top\mathbf{x}_t\right)\left(\mathbf{b}^\top\mathbf{y}_t\right)\\
&=\arg\max_{\mathbf{a},\mathbf{b}} \frac{1}{T}\sum_{t=1}^{T}\left[\left(\mathbf{a}^\top\mathbf{x}_t\right)\left(\mathbf{b}^\top\mathbf{y}_t\right)+\frac{1}{2}\left(\mathbf{a}^\top\mathbf{x}_t\right)^2+\frac{1}{2}\left(\mathbf{b}^\top\mathbf{y}_t\right)^2\right]\\
&=\arg\max_{\mathbf{a},\mathbf{b}} \frac{1}{2T}\sum_{t=1}^{T}(\mathbf{a}^\top\mathbf{x}_t+\mathbf{b}^\top\mathbf{y}_t)^2\\
&\text{s.t.}\quad \frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{a}^\top\mathbf{x}_t\right)^2=1,\quad \frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{b}^\top\mathbf{y}_t\right)^2=1. \qquad (2)
\end{aligned}$$

Next, we introduce the constraints into the objective using the method of Lagrange multipliers and regroup the terms:

$$\begin{aligned}
&\max_{\mathbf{a},\mathbf{b}}\min_{\alpha,\beta} \frac{1}{2T}\sum_{t=1}^{T}(\mathbf{a}^\top\mathbf{x}_t+\mathbf{b}^\top\mathbf{y}_t)^2-\frac{\alpha}{2}\left(\frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{a}^\top\mathbf{x}_t\right)^2-1\right)\\
&-\frac{\beta}{2}\left(\frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{b}^\top\mathbf{y}_t\right)^2-1\right)\\
&=\max_{\mathbf{a},\mathbf{b}}\min_{\alpha,\beta} \frac{1}{2T}\sum_{t=1}^{T}\left[(\mathbf{a}^\top\mathbf{x}_t+\mathbf{b}^\top\mathbf{y}_t)^2-\alpha\left(\left(\mathbf{a}^\top\mathbf{x}_t\right)^2-1\right)\right.\\
&\left.-\beta\left(\left(\mathbf{b}^\top\mathbf{y}_t\right)^2-1\right)\right]. \qquad (3)
\end{aligned}$$

To simplify the notation below we define the following variables:

$$c_t^a := \mathbf{a}^\top\mathbf{x}_t,\qquad c_t^b := \mathbf{b}^\top\mathbf{y}_t,\qquad c_t := c_t^a + c_t^b. \qquad (4)$$

We solve (3) using stochastic gradient ascent/descent with respect to $\mathbf{a}$, $\mathbf{b}$, α, and β.

$$\begin{aligned}
&\mathbf{a}_{t+1}=\mathbf{a}_t+\eta_a(c_t\mathbf{x}_t-\alpha_t c_t^a\mathbf{x}_t),\quad \mathbf{b}_{t+1}=\mathbf{b}_t+\eta_b(c_t\mathbf{y}_t-\beta_t c_t^b\mathbf{y}_t),\\
&\alpha_{t+1}=\alpha_t+\frac{\eta_\alpha}{2}\left(\left(c_t^a\right)^2-1\right),\quad \beta_{t+1}=\beta_t+\frac{\eta_\beta}{2}\left(\left(c_t^b\right)^2-1\right). \qquad (5)
\end{aligned}$$

We can interpret this algorithm as the operation of a pyramidal neuron. Here, c_t^b and c_t^a are proximal and distal dendritic currents, c_t is the output of the pyramidal neuron, $\mathbf{b}_t$ is a vector of proximal synaptic weights, $\mathbf{a}_t$ is a vector of distal synaptic weights, and α_t and β_t are scalar variables confined to proximal and distal dendrites respectively, and ηs are learning rates (**Figure 1**). The fixed point of this Algorithm corresponds to the solution of Equation (1) (see **Supplementary Information B**).

To see how these learning rules can be implemented biologically, we note that they can be expressed solely in terms of total output, c, signaled by backpropagating spikes (Stuart et al., 1997) and the distal dendritic current, c^a, signaled by the calcium plateau potentials. The key to this is the relationship $c = c^a + c^b$ in Equation (4) which allows expressing the learning rules only in terms of the two available signals. Remarkably, such non-Hebbian plasticity rules have been reported experimentally (Golding et al., 2002; Bittner et al., 2017; Magee and Grienberger, 2020).

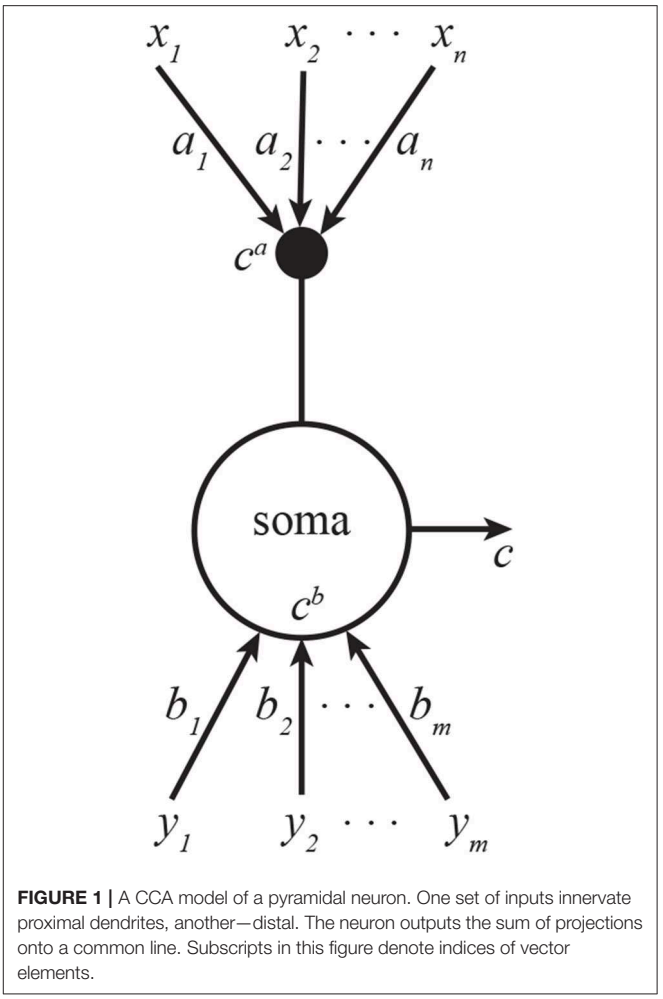

FIGURE 1 | A CCA model of a pyramidal neuron. One set of inputs innervate proximal dendrites, another—distal. The neuron outputs the sum of projections onto a common line. Subscripts in this figure denote indices of vector elements.

Although algorithm (5) can be found in Lai and Fyfe (1999) (up to a redefinition of variables α_t and β_t) it was interpreted there as the learning dynamics of a network with two point neurons and non-local learning rules. Our interpretation of the algorithm as operation of a pyramidal neuron will allow us to generalize it to multi-channel CCA implemented by a network of neurons.

3. THE STANDARD MULTI-CHANNEL CCA REQUIRES BIOLOGICALLY IMPLAUSIBLE INTERACTIONS

Given two datasets with the same number of data points but in different vector spaces, a multi-channel CCA projects them onto a common vector space so that the different components of the same projection are uncorrelated but the corresponding components of the two projections are maximally correlated. Given the success of our approach in the previous section, one

may want to derive a network of neurons from the standard multi-channel CCA cost function:

$$\begin{aligned}&\max_{\substack{\mathbf{a}_1,\ldots,\mathbf{a}_d,\\ \mathbf{b}_1,\ldots,\mathbf{b}_d}} \frac{1}{T}\sum_{i=1}^{d}\sum_{t=1}^{T}(\mathbf{a}_i^\top \mathbf{x}_t)(\mathbf{b}_i^\top \mathbf{y}_t),\\ &\text{s.t.}\ \ \frac{1}{T}\sum_{t=1}^{T}(\mathbf{a}_i^\top \mathbf{x}_t)(\mathbf{a}_j^\top \mathbf{x}_t)=\delta_{ij},\\ &\quad\frac{1}{T}\sum_{t=1}^{T}(\mathbf{b}_i^\top \mathbf{y}_t)(\mathbf{b}_j^\top \mathbf{y}_t)=\delta_{ij},\quad i,j=1,\ldots,d. \end{aligned}\tag{6}$$

Following the same derivation procedure as in section 2, we complete the squares and use the method of Lagrange multipliers to arrive at:

$$\begin{aligned}&\max_{\substack{\mathbf{a}_1,\ldots,\mathbf{a}_d,\\ \mathbf{b}_1,\ldots,\mathbf{b}_d}}\ \min_{\substack{\alpha_1,\ldots,\alpha_d,\\ \beta_1,\ldots,\beta_d}}\ \min_{\substack{A_{ij},B_{ij},\\ i,j=1,\ldots,d,\ i\neq j}} \frac{1}{2T}\sum_{i=1}^{d}\sum_{t=1}^{T}\Bigg[\left(\mathbf{a}_i^\top \mathbf{x}_t+\mathbf{b}_i^\top \mathbf{y}_t\right)^2\\ &-\alpha_i\left(\left(\mathbf{a}_i^\top \mathbf{x}_t\right)^2-1\right)-\beta_i\left(\left(\mathbf{b}_i^\top \mathbf{y}_t\right)^2-1\right)\Bigg]\\ &-\frac{1}{2T}\sum_{i=1}^{d}\sum_{\substack{j=1,\\ j\neq i}}^{d}A_{ij}\sum_{t=1}^{T}(\mathbf{a}_i^\top \mathbf{x}_t)(\mathbf{a}_j^\top \mathbf{x}_t)\\ &-\frac{1}{T}\sum_{i=1}^{d}\sum_{\substack{j=1,\\ j\neq i}}^{d}B_{ij}\sum_{t=1}^{T}(\mathbf{b}_i^\top \mathbf{y}_t)(\mathbf{b}_j^\top \mathbf{y}_t). \end{aligned}\tag{7}$$

The first two lines of (7) has d copies of the Lagrangian formulation for a single-channel CCA (3) suggesting that its online optimization can correspond to the operation of d pyramidal neurons. However, the interactions between these neurons given by $d(d-1)$ constraints in the third and fourth lines of (7) lead to a biologically implausible algorithm (see **Supplementary Information C**). Indeed, the decorrelation of proximal (as well as distal) currents among different neurons requires neurons to communicate information about such currents to each other. Yet, biological neurons do not output proximal or distal currents separately, only their sum.

To solve this problem, in the next section, we present a new objective function for CCA where the constraints are formulated in terms of neural outputs.

4. A NETWORK OF PYRAMIDAL NEURONS DERIVED FROM A NOVEL MULTI-CHANNEL CCA OBJECTIVE

To derive a biologically plausible multi-channel CCA algorithm we resort to deflation: assuming we know the top $d-1$ canonical variable pairs we find the dth canonical variable pair. We formulate a CCA objective with constraints expressed in terms of neural outputs accessible to other neurons based on the following proposition.

Proposition 1. *Given the top $d-1$ canonical variable pairs, $\mathbf{a}_1,\ldots,\mathbf{a}_{d-1}$ and $\mathbf{b}_1,\ldots,\mathbf{b}_{d-1}$, the solution to the following optimization problem gives the dth pair of canonical variables:*

$$\begin{aligned}&\max_{\mathbf{a}\in\mathbb{R}^n,\mathbf{b}\in\mathbb{R}^m}\frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{a}^\top \mathbf{x}_t\right)\left(\mathbf{b}^\top \mathbf{y}_t\right),\\ &\text{s.t.}\ \ \frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{a}^\top \mathbf{x}_t\right)^2=1,\frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{b}^\top \mathbf{y}_t\right)^2=1,\\ &\frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{a}^\top \mathbf{x}_t+\mathbf{b}^\top \mathbf{y}_t\right)\left(\mathbf{a}_i^\top \mathbf{x}_t+\mathbf{b}_i^\top \mathbf{y}_t\right)=0,\quad i=1,\ldots,d-1. \end{aligned}\tag{8}$$

Proof: See **Supplementary Information D**. □

Starting from the optimization problem (8) and following the steps that lead to (5), one can derive an online algorithm for the dth canonical variable pair. The optimization problem in the Lagrange multiplier formulation has the following form:

$$\begin{aligned}&\max_{\mathbf{a},\mathbf{b}}\min_{\alpha,\beta,m_i}\frac{1}{T}\sum_{t=1}^{T}\Bigg[\frac{1}{2}(\mathbf{a}^\top \mathbf{x}_t+\mathbf{b}^\top \mathbf{y}_t)^2-\frac{\alpha}{2}\left((\mathbf{a}^\top \mathbf{x}_t)^2-1\right)\\ &-\frac{\beta}{2}\left((\mathbf{b}^\top \mathbf{y}_t)^2-1\right)-\sum_{i=1}^{d-1}m_i(\mathbf{a}_i^\top \mathbf{x}_t+\mathbf{b}_i^\top \mathbf{y}_t)(\mathbf{a}^\top \mathbf{x}_t+\mathbf{b}^\top \mathbf{y}_t)\Bigg] \end{aligned}\tag{9}$$

We define the following variables

$$\begin{aligned}&c_{d,t}^a:=\mathbf{a}_t^\top \mathbf{x}_t,\quad c_{d,t}^b:=\mathbf{b}_t^\top \mathbf{y}_t,\\ &c_{i,t}:=\mathbf{a}_{i,t}^\top \mathbf{x}_t+\mathbf{b}_{i,t}^\top \mathbf{y}_t,\quad i=1,\ldots,d-1,\\ &c_{d,t}:=c_{d,t}^a+c_{d,t}^b-\sum_{i=1}^{d-1}m_{i,t}c_{i,t}, \end{aligned}\tag{10}$$

and optimize using stochastic gradient/ascent:

$$\begin{aligned}&\mathbf{a}_{d,t+1}=\mathbf{a}_{d,t}+\eta_a(c_{d,t}\mathbf{x}_t-\alpha_{d,t}c_{d,t}^a\mathbf{x}_t),\\ &\mathbf{b}_{d,t+1}=\mathbf{b}_{d,t}+\eta_b(c_{d,t}\mathbf{y}_t-\beta_{d,t}c_{d,t}^b\mathbf{y}_t),\\ &m_{i,t+1}=m_{i,t}+\eta_m(c_{d,t}^a+c_{d,t}^b)c_{i,t},\quad i=1,\ldots,d-1,\\ &\alpha_{d,t+1}=\alpha_{d,t}+\frac{\eta_\alpha}{2}\left(\left(c_{d,t}^a\right)^2-1\right),\\ &\beta_{t+1}=\beta_t+\frac{\eta_\beta}{2}\left(\left(c_{d,t}^b\right)^2-1\right). \end{aligned}\tag{11}$$

In addition, we replace $c_{d,t}^a+c_{d,t}^b$ with $c_{d,t}$ to make the algorithm biologically plausible. Our modification to learning rule for m_i is:

$$\begin{aligned}&m_{i,t+1}=m_{i,t}+\eta_m(c_{d,t}^a+c_{d,t}^b)c_{i,t}\\ \text{changes to}\quad &m_{i,t+1}=m_{i,t}+\eta_m c_{d,t}c_{i,t}. \end{aligned}\tag{12}$$

To see why the modified rule works, assume that the weight updates converged to a stationary state, i.e. $0=\langle\Delta m_i\rangle=\left\langle(c_d^a+c_d^b)c_i\right\rangle-m_i$, where we used $\langle c_ic_j\rangle=\delta_{ij}$ (**Supplementary Information A**) and the brackets denote an average over the inputs. The desired CCA solution for the d^{th} canonical variable pair, i.e., $m_i=0$ and $\left\langle(c_d^a+c_d^b)c_i\right\rangle=0$ satisfies this condition.

Even though other solutions exist, in simulations (section 6), we see that the algorithm converges to only the desired one.

The resulting algorithm can be implemented by pyramidal neurons. As before, we interpret $c^a_{d,t}$, $c^b_{d,t}$, and $c_{d,t}$ as distal, proximal, and total output currents, respectively, $\mathbf{a}_{d,t}$, $\mathbf{b}_{d,t}$, as distal and proximal synaptic weight vectors, $\alpha_{d,t}$, $\beta_{d,t}$ as distal and proximal dendritic variables. We interpret $c_{i,t}$ as the outputs of neurons that were trained to extract ith ($i < d$) pair of canonical variables and m_{di} are lateral weights from those neurons. Note that these lateral weight updates are anti-Hebbian.

The above derivation assumed that all $i = 1, \cdots, d-1$ canonical components are already extracted successfully, each by a different pyramidal neuron. To compute CCA from scratch, one way is to extract the d components sequentially: ith neuron has to wait for the result of $(i-1)$th neuron. Once the distal and proximal weights of the $(i-1)$th neuron converges, they are frozen and one moves to the next neuron. A subtle point here is about the lateral connections. Algorithm defined by (10) and (11) omits lateral connections between previous neurons in the sequence, suggesting that lateral connections should be removed once a neuron's weights converges. Such removal naturally happens because lateral weights decay to zero at the fixed point of the algorithm.

However, it is also possible, and more biologically plausible, to train all pyramidal neurons simultaneously using an asymmetric network architecture (**Figure 2**). This is akin to the asymmetric lateral connectivity in the Generalized Hebbian Network (Sanger, 1989) and APEX (Diamantaras and Kung, 1996) network for PCA. The resulting algorithm is given in Algorithm 1.

Algorithm 1 CCA network

Input: Parameters d, η_a, η_b, η_α, η_β and η_m. Initial $\alpha_1, \ldots, \alpha_d$ and $\beta_1, \ldots, \beta_d$. Initial synaptic weights $\mathbf{A} \in \mathbb{R}^{n\times d}$, $\mathbf{B} \in \mathbb{R}^{m\times d}$, and $\mathbf{M} \in \mathbb{R}^{d\times d}$. $M_{ij} = 0$ for $j \geq i$.

for $t = 1, 2, 3, \ldots, T$ **do**

`// Neural activity`

Receive inputs $\mathbf{x}_t$ and $\mathbf{y}_t$

for $i = 1, \ldots, d$ **do**

Calculate proximal and distal dendritic currents: $c^a_{i,t} = \sum_{j=1}^n A_{ji,t} x_{j,t}$, $\quad c^a_{j,t} = \sum_{j=1}^m B_{ji,t} y_{j,t}$,

Calculate pyramidal neuron outputs: $c_{i,t} = c^a_{i,t} + c^b_{i,t} - \sum_{j=1}^{i-1} M_{ij,t} c_{j,t}$

end for

`// Synaptic and homeostatic plasticity`

Update synaptic weights:

$$A_{ij,t+1} = A_{ij,t} + \eta_a \left(c_{j,t} x_{i,t} - \alpha_{j,t} c^a_{j,t} x_{i,t} \right), \quad i = 1, \ldots, n, \quad j = 1, \ldots, d$$

$$B_{ij,t+1} = B_{ij,t} + \eta_b \left(c_{j,t} y_{i,t} - \beta_{j,t} c^b_{j,t} y_{i,t} \right), \quad i = 1, \ldots, m, \quad j = 1, \ldots, d$$

$$M_{ij,t+1} = M_{ij,t} + \eta_m c_{i,t} c_{j,t}, \quad i = 1, \ldots, d, j < i$$

Update dendritic variables:

$$\alpha_{i,t+1} = \alpha_{i,t} + \tfrac{\eta_\alpha}{2} \left(\left(c^a_{i,t} \right)^2 - 1 \right), \quad i = 1, \ldots, d$$

$$\beta_{i,t+1} = \beta_{i,t} + \tfrac{\eta_\beta}{2} \left(\left(c^b_{i,t} \right)^2 - 1 \right), \quad i = 1, \ldots, d$$

end for

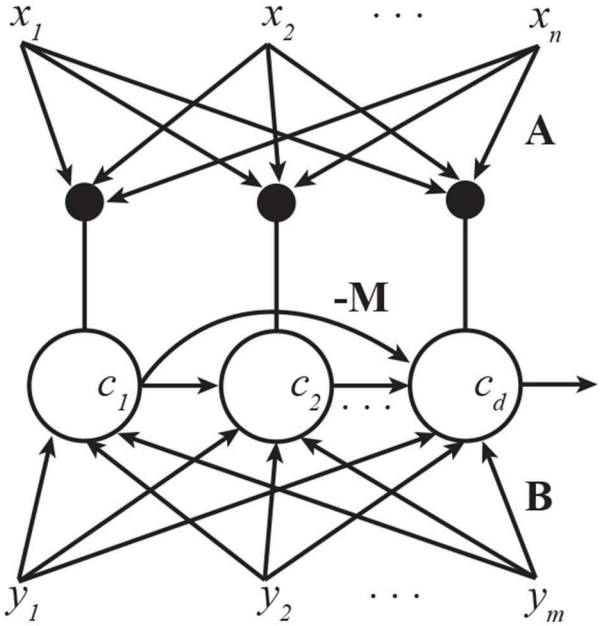

FIGURE 2 | A pyramidal neuron network that implements multi-channel CCA. Lateral connections are anti-Hebbian and asymmetric. Subscripts in this figure denote indices of vector elements.

5. NEURONS WITH MULTIPLE DENDRITIC BRANCHES AS MULTIVIEW CANONICAL CORRELATION ANALYZERS

In this section, we return to the model of a single pyramidal neuron and extend it to neurons with more than two dendritic compartments using a multiview single-channel CCA, i.e., CCA on multiple sets of variables (Kettenring, 1971). There are multiple ways to generalize CCA to multiple sets of variables; the version we are using here is "SUMCOR" (sum of correlations): given a dataset with k views, $\{\mathbf{x}^{(i)}_t \in \mathbb{R}^{n(i)}, t = 1, \ldots, T, i = 1, \ldots, k\}$, the SUMCOR version of multiview CCA projects each dataset on the common line that maximizes the sum of pairwise correlations between them. Formally, we consider the following SUMCOR objective function (Kettenring, 1971):

$$\max_{\mathbf{a}^{(1)}, \mathbf{a}^{(2)}, \cdots, \mathbf{a}^{(k)}} \frac{1}{T} \sum_{i<j} \sum_{t=1}^{T} (\mathbf{a}^{(i)\top} \mathbf{x}^{(i)}_t)(\mathbf{a}^{(j)\top} \mathbf{x}^{(j)}_t),$$

$$\text{s.t.} \ \frac{1}{T} \sum_{t=1}^{T} (\mathbf{a}^{(i)\top} \mathbf{x}^{(i)}_t)^2 = 1, \quad i = 1, 2, \cdots, k \qquad (13)$$

From now on, the SUMCOR version of multiview CCA is simply referred to as multiview CCA.

In order to derive a neural algorithm for multiview CCA, as before we complete the square in the objective and introduce

Lagrange multipliers:

$$\max_{\mathbf{a}^{(1)},\mathbf{a}^{(2)},\cdots,\mathbf{a}^{(k)}} \min_{\alpha^{(1)},\alpha^{(2)},\cdots,\alpha^{(k)}} \frac{1}{2T}\sum_{t=1}^{T}\left(\sum_{i=1}^{k}\mathbf{a}^{(i)\top}\mathbf{x}_t^{(i)}\right)^2 - \sum_{i=1}^{k}\frac{\alpha^{(i)}}{2}\left(\frac{1}{T}\sum_{t=1}^{T}(\mathbf{a}^{(i)\top}\mathbf{x}_t^{(i)})^2 - 1\right) \tag{14}$$

Defining

$$c_t^{(i)} := \mathbf{a}_t^{(i)\top}\mathbf{x}_t^{(i)}, \quad c_t := \sum_{i=1}^{k} c_t^{(i)} \tag{15}$$

We optimize this objective by gradient descent/ascent:

$$\mathbf{a}_{t+1}^{(i)} = \mathbf{a}_t^{(i)} + \eta_a^{(i)}\left(c_t - \alpha_t^{(i)}c_t^{(i)}\right)\mathbf{x}_t^{(i)}, \quad \alpha_{t+1}^{(i)} = \alpha_t^{(i)} + \frac{\eta_\alpha^{(i)}}{2}\left(c_t^{(i)2} - 1\right) \tag{16}$$

This algorithm can be implemented by a neuron with k dendritic compartments. The neural structure is depicted in **Figure 3**, and the pseudocode—in Algorithm 2. As a generalization of the neuron shown in **Figure 1**, the soma is summing all k inputs instead of just two, and each branch of dendrites is carrying out exactly the same learning operations as **Figure 1**.

Algorithm 2 Multiview CCA neuron

Input: Parameters $\eta_a^{(i)}$ and $\eta_\alpha^{(i)}$ for $i = 1, 2, \ldots, k$. Initial dendritic variables $\alpha^{(i)}$ and initial synaptic weights $\mathbf{a}^{(i)} \in \mathbb{R}^{n^{(i)}}$ $i = 1, 2, \ldots, k$.

for $t = 1, 2, 3, \ldots, T$ **do**

`// Neural activity`

Receive inputs $\mathbf{x}_t^{(i)}$ for $i = 1, 2, \ldots, k$

Calculate dendritic currents for all k compartments: $c_t^{(i)} := \mathbf{a}_t^{(i)\top}\mathbf{x}_t^{(i)}, \quad i = 1, 2, \ldots, k$

Calculate neuronal output: $c_t := \sum_{i=1}^{k} c_t^{(i)}$

`// Synaptic and homeostatic plasticity`

Update synaptic weights: $\mathbf{a}_{t+1}^{(i)} = \mathbf{a}_t^{(i)} + \eta_a^{(i)}\left(c_t - \alpha_t^{(i)}c_t^{(i)}\right)\mathbf{x}_t^{(i)}$, $i = 1, 2, \ldots, k$

Update dendritic variables: $\alpha_{t+1}^{(i)} = \alpha_t^{(i)} + \frac{\eta_\alpha^{(i)}}{2}\left(\left(c_t^{(i)}\right)^2 - 1\right)$, $i = 1, 2, \ldots, k$

end for

6. NUMERICAL SIMULATIONS

In this section, we provide numerical simulations of our algorithms.

6.1. Multichannel CCA Algorithm

We simulated our CCA network (Algorithm 1) and several other algorithms on various datasets.

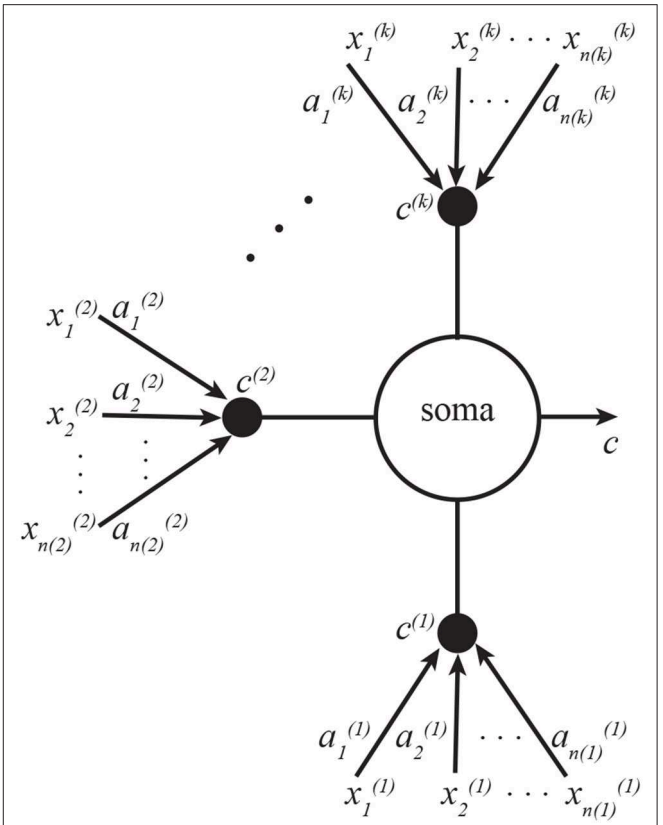

FIGURE 3 | A neuron with k dendritic compartments can implement k-view CCA. Here the soma sums the currents from all k compartments, which are indexed by superscripts. Subscripts in this figure denote indices of vector elements.

6.1.1. Datasets

We use two 5-dimensional inputs, i.e., $\mathbf{x}_t, \mathbf{y}_t \in \mathbb{R}^5$, and three datasets, whose samples are generated according to the following procedures:

1. `gaussian`: We draw $\mathbf{x}_t$ and $\mathbf{y}_t$ jointly from a 10-dimensional Gaussian distribution with random covariance matrices $\mathbf{A}^\top\mathbf{A}/100$, where elements of the square matrix $\mathbf{A}$ are independently drawn from a standard Gaussian distribution.
2. `mnist`: We generated a synthetic dataset from the training portion of the MNIST dataset (LeCun, 1998). We took the 15th row of 28-by-28 MNIST images, and set $\mathbf{x}_t$ to be the 10th to 14th pixels of this row, and $\mathbf{y}_t$ to be the 15th to 19th pixels of this row (**Figure 4**).
3. `mediamill`: We used the Mediamill dataset (Snoek et al., 2006) [1], which contains 101 textual features and 120 visual features of multiple TV frames. We let $\mathbf{x}_t$ be the 5 textual features with highest frequency of occurrence, while $\mathbf{y}_t$ is set to be the first 5 visual features in the dataset.

[1] downloaded from http://isis-data.science.uva.nl/cgmsnoek/mediamill/mediamill-challenge.tar.gz/

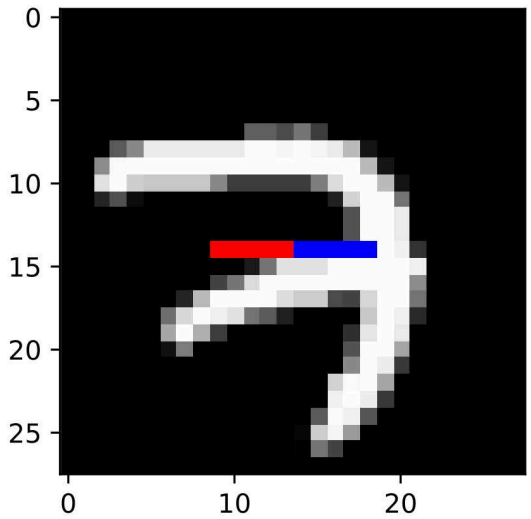

FIGURE 4 | Illustration of `mnist` datasets. $\mathbf{x}_t$ and $\mathbf{y}_t$ corresponds to red and blue pixels, respectively.

All of the above datasets are centered such that the mean of $\mathbf{x}_t \in \mathbb{R}^5$ and $\mathbf{y}_t \in \mathbb{R}^5$ are zero. We generate 10,000 samples in advance, and take $\mathbf{x}_t$ and $\mathbf{y}_t$ from these samples randomly during each training step. We extract the top $d = 3$ canonical components for all datasets.

6.1.2. Metrics of Performance

Next, we define the performance metrics for comparisons between different algorithms.

- **Normalized objective:** The first metric we used is the value of objective function during training divided by the optimal objective $f(\mathbf{A},\mathbf{B})/f(\mathbf{A}_{cca},\mathbf{B}_{cca})$ where f is defined as:

$$f(\mathbf{A},\mathbf{B})=\frac{\frac{1}{T}\sum_{i=1}^{d}\sum_{t=1}^{T}(\mathbf{a}_i^\top\mathbf{x}_t)(\mathbf{b}_i^\top\mathbf{y}_t)}{\left[\frac{1}{T}\sum_{i=1}^{d}\sum_{t=1}^{T}(\mathbf{a}_i^\top\mathbf{x}_t)^2\right]^{1/2}\left[\frac{1}{T}\sum_{i=1}^{d}\sum_{t=1}^{T}(\mathbf{b}_i^\top\mathbf{y}_t)^2\right]^{1/2}} \tag{17}$$

  Here $\mathbf{A}$ and $\mathbf{B}$ are synaptic weights, as in Algorithm 1, and $\mathbf{a}_i$ (or $\mathbf{b}_i$) is the ith column of $\mathbf{A}$ (or $\mathbf{B}$). $\mathbf{A}_{cca}$ and $\mathbf{B}_{cca}$ are the correct solution of CCA; therefore, normalized objective should converge to one if the algorithm is successful. The numerator is the objective of (6), while the denominator corresponds to the constraints. We include the denominator because during the optimization process the constraints are not always fulfilled, potentially causing a misleading normalized objective. Note that this measure can take values above 1 again due to constraints being not satisfied.
- **Angular error:** We used a metric from Ge et al. (2016) defined as:

$$\arccos\left(\frac{\frac{1}{T}\sum_{i=1}^{d}\sum_{t=1}^{T}[(\mathbf{a}_i^\top\mathbf{x}_t)(\mathbf{a}_{cca,i}^\top\mathbf{x}_t)+(\mathbf{b}_i^\top\mathbf{y}_t)(\mathbf{b}_{cca,i}^\top\mathbf{y}_t)]}{\left\{\frac{1}{T}\sum_{i=1}^{d}\sum_{t=1}^{T}[(\mathbf{a}_i^\top\mathbf{x}_t)^2+(\mathbf{b}_i^\top\mathbf{y}_t)^2]\right\}^{1/2}\left\{\frac{1}{T}\sum_{i=1}^{d}\sum_{t=1}^{T}[(\mathbf{a}_{cca,i}^\top\mathbf{x}_t)^2+(\mathbf{b}_{cca,i}^\top\mathbf{y}_t)^2]\right\}^{1/2}}\right) \tag{18}$$

  where subscript, $_{cca}$, denotes correct solution of CCA. Angular error is zero when the algorithm finds the correct solution of CCA. This metric measures the cosine of the angles between $(\mathbf{A},\mathbf{B})$ and $(\mathbf{A}_{cca},\mathbf{B}_{cca})$ based on the following inner product:

$$\langle(\mathbf{A},\mathbf{B}),(\mathbf{A}_{cca},\mathbf{B}_{cca})\rangle = \sum_{i=1}^{d}\left[\mathbf{a}_i^\top\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{x}_t\mathbf{x}_t^\top\right)\mathbf{a}_{cca,i} + \mathbf{b}_i^\top\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{y}_t\mathbf{y}_t^\top\right)\mathbf{b}_{cca,i}\right], \tag{19}$$

  which gives higher weight to synapses whose input has higher variance.

6.1.3. Simulated Algorithms

We simulated four algorithms:

1. `proposed`: CCA network (Algorithm 1) with linearly decaying learning rates. We set $\eta = \eta_a = \eta_b = \eta_\alpha = \eta_\beta = \eta_m$, and $\eta(t) = 0.02 \times \max(1-\alpha t, 0.1)$, where $\alpha = 5\times10^{-6}$. We also tried other learning rate decay schemes such as $\eta(t) \propto 1/t$ or $1/\sqrt{t}$, but found them to be performing worse. We initialized all weights randomly by drawing each of their elements independently from a standard gaussian distribution.
2. `nondecay`: CCA network(Algorithm 1) with constant learning rate. We used the same parametrization as above but with $\alpha = 0$.
3. `nonlocal`: The nonlocal (biologically implausible) algorithm (Algorithm 3). The initialization of weights and initial learning rates are exactly the same as Algorithm 1, with constant learning rate during training.
4. `MSG-CCA`: A non-neural online algorithm "Matrix Stochastic Gradient for CCA" described in Arora et al. (2017). As suggested by Arora et al. (2017), we used learning rate decay scheme $\eta_t = \frac{0.1}{\sqrt{t}}$. This algorithm requires an auxiliary training data, i.e., it has to take some samples in advance and learn on them in an offline manner. We chose this sample size to be 100.

6.1.4. Results

The performances of all four algorithms [`pyramidal, nonlocal, nondecay, MSG-CCA`] on the three datasets are shown in **Figure 5**. The angular error of `MSG-CCA` algorithm is not calculated because the algorithm does not produce a deterministic and explicit estimate of the canonical variables (Arora et al., 2012), however calculating normalized objective is possible. All algorithms have similar performance except the `nonlocal` algorithm which does not converge at all. Note that the normalized objective can go over 1 because the constraint

of CCA may not be satisfied. The `proposed` and `nondecay` algorithms only differ in their learning rate scheme, and we can see that the `proposed` algorithm with learning rate decay has a better performance.

6.2. Multiview CCA Algorithm

Next, we simulated our Algorithm 2 for multiview CCA. We set constant learning rates $\eta_a^{(i)} = \eta_\alpha^{(i)} = 0.005$ for all $i = 1, 2, \cdots, k$, and initialize all network weights according to standard gaussian distribution.

6.2.1. Datasets

Since multiview CCA does not have an analytical solution (Kettenring, 1971), and is computationally intractable (Rupnik and Shawe-Taylor, 2010), we choose datasets to which we know or approximately know the solution. The following two datasets are used:

1. `mnist4X`: Here we set $\mathbf{x}_t^{(1)}, \mathbf{x}_t^{(2)}, \mathbf{x}_t^{(3)}, \mathbf{x}_t^{(4)}$ to be four quarters of the t'th MNIST image (**Figure 6A**). The closer two pixels are, the higher the correlation between them is, so our generalized CCA would extract pixels closest to each other. In this case, it would approximately be the four pixels at the center of the whole image. Like before, instead of drawing a sample at every training step, we take 10000 MNIST images and generate from them 10,000 samples of $\mathbf{x}_t^{(1)}, \mathbf{x}_t^{(2)}, \mathbf{x}_t^{(3)}, \mathbf{x}_t^{(4)}$ in advance, and randomly draw a sample of these at each training step.
2. `svd-max`: We used singular value decomposition (SVD) to construct a dataset such that all pairs of dendritic currents (that is $c^{(i)}, i = 1, 2, \cdots, k$) have a correlation coefficient of one, i.e. maximally correlated. Specifically, we let $(\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, \cdots, \mathbf{x}_T^{(i)}) = \mathbf{X}^{(i)} = \mathbf{U}^{(i)}\mathbf{S}^{(i)}\mathbf{V}^{(i)\top}$, where columns of $\mathbf{U}^{(i)} \in \mathbb{R}^{n^{(i)} \times n^{(i)}}$ (and also $\mathbf{V}^{(i)} \in \mathbb{R}^{T \times n^{(i)}}$) are orthonormal to each other, and $\mathbf{S}^{(i)} \in \mathbb{R}^{n^{(i)} \times n^{(i)}}$ is diagonal. For all $i = 1, 2, \cdots, k$, diagonal elements of $\mathbf{S}^{(i)}$ are independently drawn from *Uniform*([0.1, 1]), and then sorted in descending order. For all $i = 1, 2, \cdots, k$, $\mathbf{U}^{(i)}$ is constructed by perfroming SVD on a standard gaussian distributed matrix of the same size. The first column of $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, \cdots, \mathbf{V}^{(k)}$ are set to be the same, and all the other columns of all these k matrices are set to be orthonormal to each other: this is done by first constructing a $T \times \left(\sum_{i=1}^k n^{(i)} - k + 1\right)$ matrix whose columns are orthonormal to each other, and then slicing this matrix to yield $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, \cdots, \mathbf{V}^{(k)}$. The weights that optimize objective function (13) is $\mathbf{a}^{(i)} = \frac{1}{s_1^{(i)}}\mathbf{u}_1^{(i)}$, where $s_1^{(i)}$ is the first element of $\mathbf{S}^{(i)}$, and $\mathbf{u}_1^{(i)}$ is the first column of $\mathbf{U}^{(i)}$. For the simulation below, we set $k = 3, n^{(1)} = 4, n^{(2)} = 5, n^{(3)} = 6, T = 10,000$.

6.2.2. Metrics of Performance

As before, we have to define the metrics for performance of multiview CCA algorithm.

- **Absolute objective:**

$$\frac{2}{k(k-1)} \sum_{i=1}^{k} \sum_{j=i+1}^{k} \frac{\frac{1}{T}\sum_{t=1}^{T} (\mathbf{a}^{(i)\top}\mathbf{x}_t)(\mathbf{a}^{(j)\top}\mathbf{x}_t)}{\left[\frac{1}{T}\sum_{t=1}^{T} (\mathbf{a}^{(i)\top}\mathbf{x}_t)^2\right]^{1/2} \left[\frac{1}{T}\sum_{t=1}^{T} (\mathbf{a}^{(j)\top}\mathbf{x}_t)^2\right]^{1/2}}, \tag{20}$$

  which is strictly between -1 and 1.
- **Angular error:**

$$\arccos\left(\frac{\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{k} (\mathbf{a}^{(i)\top}\mathbf{x}_t)(\mathbf{a}_{cca}^{(i)\top}\mathbf{x}_t)}{\left[\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{k} (\mathbf{a}^{(i)\top}\mathbf{x}_t)^2\right]^{1/2} \left[\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{d} (\mathbf{a}_{cca}^{(i)\top}\mathbf{x}_t)^2\right]^{1/2}}\right), \tag{21}$$

  where $_{cca}$ denotes correct solution to CCA.

6.2.3. Results

The performance of the algorithm on the `mnist4X` dataset is shown in **Figures 6A–D**. We visualize the learned weights on the `mnist4X` dataset (**Figure 6B**) and observe that all the weights cluster in the center of the MNIST image. Absolute objective and angular error are shown in **Figures 6C,D**. To calculate a groundtruth value of optimal weights, we ran the offline version of algorithm 2 on `mnist4X` dataset. We used the offline weights to approximate the groundtruth weights, and calculated an approximated angular error (**Figure 6D**). Training results on `svd-max` dataset is shown in **Figures 6E,F**. Note that for this dataset the analytical solution is available, so we could calculate angular error precisely.

7. CONCLUSION

In this paper, we propose mathematically tractable multicompartment neuron models that capture more biological features than previous such models. A multichannel CCA algorithm is implemented by an asymmetric network of two-compartment pyramidal neurons. A neuron with more than two dendritic compartments may implement an online multiview CCA algorithm.

Naturally, our model is a drastic simplification of a real biological pyramidal neuron. We assumed that the activities of neurons are linear, continuous and deterministic, instead of nonlinear, spiking and stochastic. The two-compartment structure is the most prominent feature of pyramidal neuron's dendritic structure, which is well-captured by our model, but this is also a simplification of actual dendritic integration, which depends on more complex dendritic morphology (Spruston, 2008). In our model, distal and proximal compartments have symmetric status in driving neuronal activity and learning rules, since the two input vectors have symmetric status in CCA. However, in a biological pyramidal neuron, these two compartments are asymmetric: their sizes are different, and, for example, their excitability might also be different (Spruston, 2008). Dendritic and synaptic nonlinearities which we ignored can endow a neuron with a rich computational repertoire (Poirazi and Mel, 2001; Poirazi et al., 2003; Polsky et al., 2004; Larkum, 2013; Jadi et al., 2014; Haga and Fukai, 2017). Temporal dynamics and synaptic activity delays are other aspects not captured by our model.

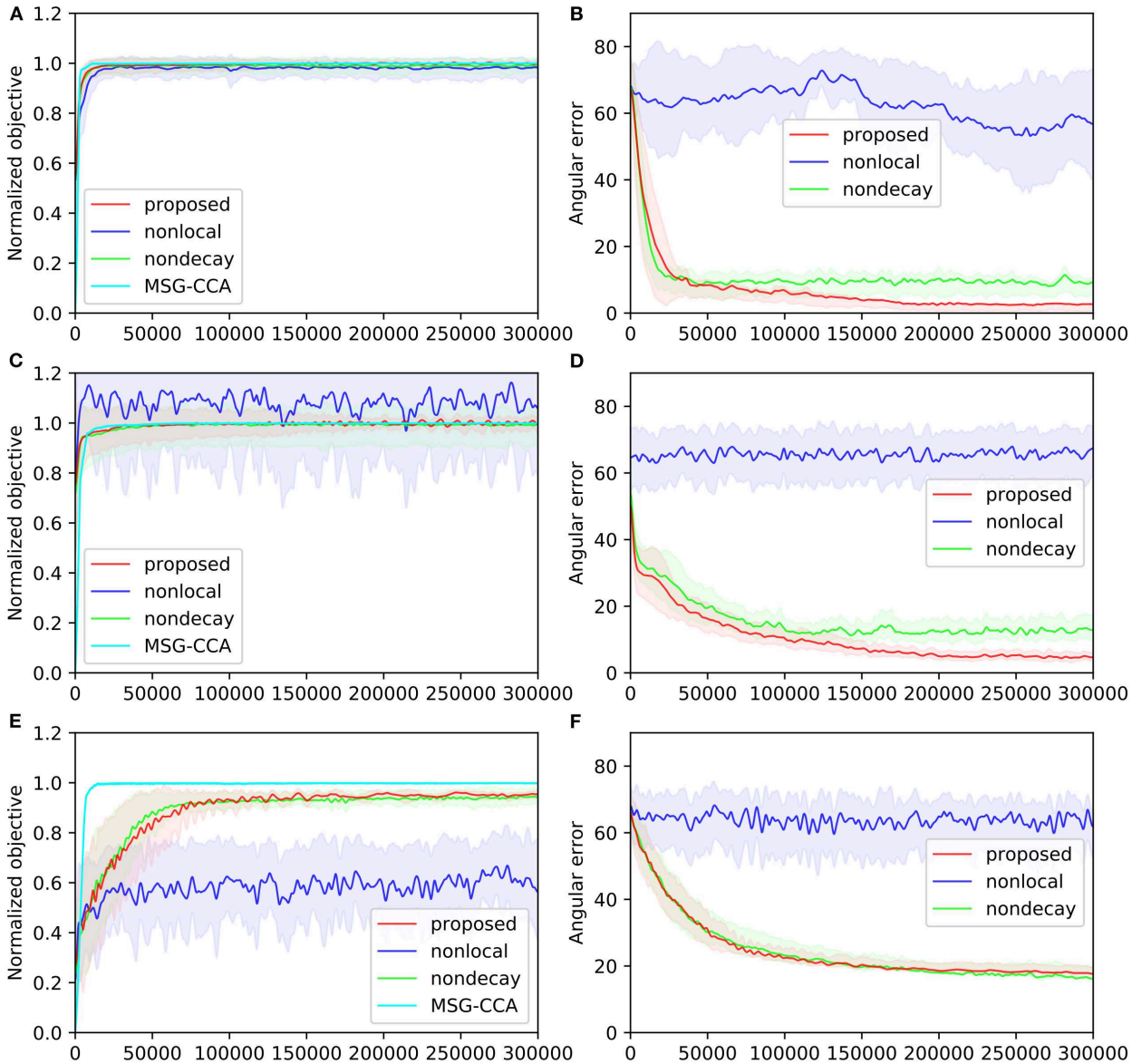

FIGURE 5 | Four algorithms [`pyramidal, nonlocal, nondecay, MSG-CCA`] run on three datasets [`gaussian`**(A,B)**, `mnist`**(C,D)**, `mediamill`**(E,F)**]. Performance is measured by normalized objective **(A,C,E)** and angular error **(B,D,F)** All algorithms are run 10 times except `MSG-CCA` because of computational constraints. Solid lines show mean value over 10 simulations. Shaded regions of the same color show standard deviation over 10 simulations. Both mean and standard deviation are smoothed along the time course of training.

Pyramidal neurons have been proposed to integrate feedback stimulation in the distal dendrite and feedforward information in the proximal dendrite (Spratling and Johnson, 2004). Since our model of pyramidal neurons performs CCA on distal and proximal inputs, it provides a new interpretation of the computation performed on feedforward and feedback inputs. Some recent work (Guerguiev et al., 2017; Sacramento et al., 2018) proposes that top-down dendritic input may be instructively gating plasticity in feedforward synapses, implementing approximately the backpropagation algorithm. Our model may provide an alternative mechanism for credit assignment, where a correlation between error related feedback signals and feedforward signals are learned.

It has been proposed that multisensory integration requires correlation detection among different sensory modalities (Parise and Ernst, 2016), and that such integration could happen on a single neuron (Stein and Stanford, 2008). The multi-compartmental version of our algorithm that performs multiview CCA can be useful for modeling such a neuron, which takes signals from different modalities as inputs and extracts a component from each input that are maximally correlated with

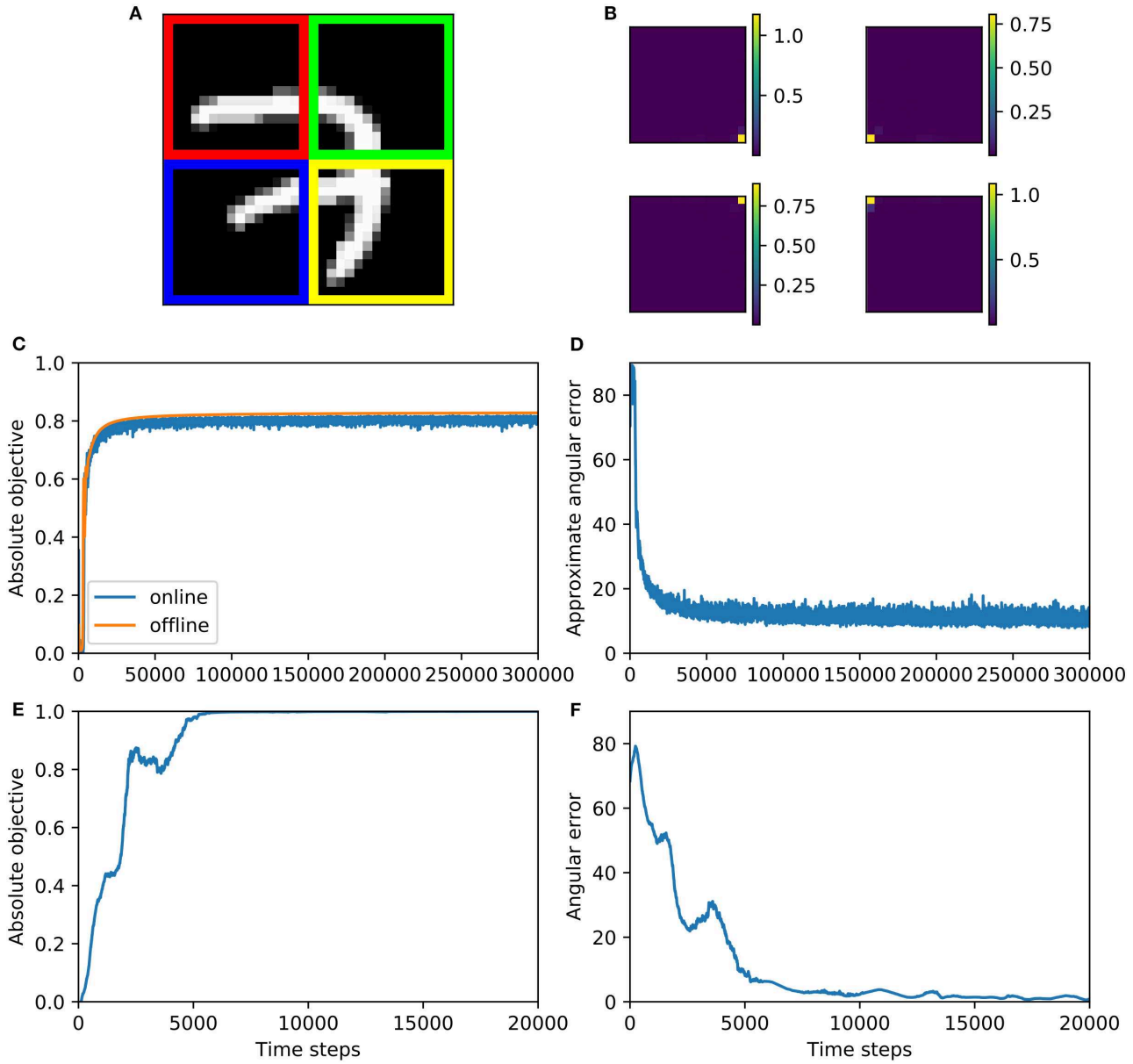

FIGURE 6 | Multiview CCA algorithm 2 is trained on `mnist4X` dataset **(A–D)** and `svd-max` dataset **(E,F)**. **(A)** This figure shows how MNIST dataset is split, where red, green, blue, and yellow parts of the image correspond to $\mathbf{x}_t^{(1)}$, $\mathbf{x}_t^{(2)}$, $\mathbf{x}_t^{(3)}$, and $\mathbf{x}_t^{(4)}$, respectively. **(B)** The synaptic weights after training, which are measured by variance contribution, match our expectation for groundtruth. The variance contribution is calculated by $u_i = \frac{1}{T}\sum_{t=1}^{T}(a_i x_{i,t})^2$ where i is the index of image pixel. In this visualization, these u_i are positioned according to the positions of their corresponding pixel and their values are indicated by the colorbars. **(C)** Absolute objective for both online and offline version of Algorithm 2 increases during training and converge to near 0.8. **(D)** Approximate angular error decreases and then fluctuate above zero during online training. **(E)** Objective function increases during training and converge to one—the largest possible correlation value. **(F)** Angular error decreases during training and eventually fluctuate around zero.

each other. One experimental observation that may fit such interpretation is that correlated sensory and motor information has been shown to be detected by pyramidal neurons through dendritic integration (Xu et al., 2012).

DATA AVAILABILITY STATEMENT

The datasets generated for this study are available on request to the corresponding author.

AUTHOR CONTRIBUTIONS

All authors designed the study. CP, XZ, and DC contributed to the analytical results and wrote the manuscript. XZ performed the numerical simulations.

FUNDING

CP received funding from the Intel Corporation for this study. The funder was not involved in the study design, collection, analysis, interpretation of data, the writing of this article or the decision to submit it for publication.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fncom.2020.00055/full#supplementary-material

REFERENCES

Alemi, A., Machens, C., Denéve, S., and Slotine, J.-J. (2017). Learning arbitrary dynamics in efficient, balanced spiking networks using local plasticity rules. *arXiv preprint arXiv:1705.08026*.

Arora, R., Cotter, A., Livescu, K., and Srebro, N. (2012). "Stochastic optimization for PCA and PLS," in *Allerton Conference on Communication, Control, and Computing*, 861–868. doi: 10.1109/Allerton.2012.6483308

Arora, R., Marinov, T. V., Mianjy, P., and Srebro, N. (2017). "Stochastic approximation for canonical correlation analysis," in *Advances in Neural Information Processing Systems*, 4775–4784.

Bittner, K. C., Grienberger, C., Vaidya, S. P., Milstein, A. D., Macklin, J. J., Suh, J., et al. (2015). Conjunctive input processing drives feature selectivity in hippocampal CA1 neurons. *Nat. Neurosci.* 18:1133. doi: 10.1038/nn.4062

Bittner, K. C., Milstein, A. D., Grienberger, C., Romani, S., and Magee, J. C. (2017). Behavioral time scale synaptic plasticity underlies ca1 place fields. *Science* 357, 1033–1036. doi: 10.1126/science.aan3846

Chechik, G., Globerson, A., Tishby, N., and Weiss, Y. (2005). Information bottleneck for Gaussian variables. *J. Mach. Learn. Res.* 6, 165–188.

Diamantaras, K., and Kung, S. (1996). *Principal Component Neural Networks: Theory and Applications*. New York, NY: John Wiley & Sons, Inc.

Földiak, P. (1989). "Adaptive network for optimal linear feature extraction," in *International Joint Conference on Neural Networks*, 401–405. doi: 10.1109/IJCNN.1989.118615

Ge, R., Jin, C., Netrapalli, P., Sidford, A., et al. (2016). "Efficient algorithms for large-scale generalized eigenvector computation and canonical correlation analysis," in *International Conference on Machine Learning*, 2741–2750.

Golding, N. L., Staff, N. P., and Spruston, N. (2002). Dendritic spikes as a mechanism for cooperative long-term potentiation. *Nature* 418, 326–331. doi: 10.1038/nature00854

Guerguiev, J., Lillicrap, T. P., and Richards, B. A. (2017). Towards deep learning with segregated dendrites. *eLife* 6:e22901. doi: 10.7554/eLife.22901.027

Haga, T., and Fukai, T. (2017). Dendritic processing of spontaneous neuronal sequences for one-shot learning. *bioRxiv* 165613. doi: 10.1101/165613

Horn, R. A., and Johnson, C. R. (1991). *Topics in Matrix Analysis*. Cambridge: Cambridge University Press. doi: 10.1017/CBO9780511840371

Hotelling, H. (1992). "Relations between two sets of variates," in *Breakthroughs in Statistics. Springer Series in Statistics (Perspectives in Statistics)*, eds S. Kotz and N. L. Johnson (New York, NY: Springer), 162–190. doi: 10.1007/978-1-4612-4380-9_14

Jadi, M. P., Behabadi, B. F., Poleg-Polsky, A., Schiller, J., and Mel, B. W. (2014). An augmented two-layer model captures nonlinear analog spatial integration effects in pyramidal neuron dendrites. *Proc. IEEE* 102, 782–798. doi: 10.1109/JPROC.2014.2312671

Kettenring, J. R. (1971). Canonical analysis of several sets of variables. *Biometrika* 58, 433–451. doi: 10.1093/biomet/58.3.433

Kushner, H. J., and Clark, D. S. (1978). *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. New York, NY: Springer-Verlag. doi: 10.1007/978-1-4684-9352-8

Lai, P. L. and Fyfe, C. (1999). A neural implementation of canonical correlation analysis. *Neural Netw.* 12, 1391–1397. doi: 10.1016/S0893-6080(99)00075-1

Larkum, M. (2013). A cellular mechanism for cortical associations: an organizing principle for the cerebral cortex. *Trends Neurosci.* 36, 141–151. doi: 10.1016/j.tins.2012.11.006

Larkum, M. E., Nevian, T., Sandler, M., Polsky, A., and Schiller, J. (2009). Synaptic integration in tuft dendrites of layer 5 pyramidal neurons: a new unifying principle. *Science* 325, 756–760. doi: 10.1126/science.1171958

LeCun, Y. (1998). *The MNIST Database of Handwritten Digits*. Available online at: http://yann.lecun.com/exdb/mnist/

Magee, J. C., and Grienberger, C. (2020). Synaptic plasticity forms and functions. *Annu. Rev. Neurosci.* 43, 95–117. doi: 10.1146/annurev-neuro-090919-022842

Oja, E. (1982). Simplified neuron model as a principal component analyzer. *J. Math. Biol.* 15, 267–273. doi: 10.1007/BF00275687

Oja, E. (1992). Principal components, minor components, and linear neural networks. *Neural Netw.* 5, 927–935. doi: 10.1016/S0893-6080(05)80089-9

Painsky, A., and Tishby, N. (2017). Gaussian lower bound for the information bottleneck limit. *J. Mach. Learn. Res.* 18, 7908–7936.

Parise, C. V., and Ernst, M. O. (2016). Correlation detection as a general mechanism for multisensory integration. *Nat. Commun.* 7:11543. doi: 10.1038/ncomms11543

Pehlevan, C., Hu, T., and Chklovskii, D. B. (2015). A hebbian/anti-hebbian neural network for linear subspace learning: a derivation from multidimensional scaling of streaming data. *Neural Comput.* 27, 1461–1495. doi: 10.1162/NECO_a_00745

Pezeshki, A., Azimi-Sadjadi, M. R., and Scharf, L. L. (2003). A network for recursive extraction of canonical coordinates. *Neural Netw.* 16, 801–808. doi: 10.1016/S0893-6080(03)00112-6

Poirazi, P., Brannon, T., and Mel, B. W. (2003). Pyramidal neuron as two-layer neural network. *Neuron* 37, 989–999. doi: 10.1016/S0896-6273(03)00149-1

Poirazi, P., and Mel, B. W. (2001). Impact of active dendrites and structural plasticity on the memory capacity of neural tissue. *Neuron* 29, 779–796. doi: 10.1016/S0896-6273(01)00252-5

Polsky, A., Mel, B. W., and Schiller, J. (2004). Computational subunits in thin dendrites of pyramidal cells. *Nat. Neurosci.* 7:621. doi: 10.1038/nn1253

Rubner, J., and Tavan, P. (1989). A self-organizing network for principal-component analysis. *EPL* 10:693. doi: 10.1209/0295-5075/10/7/015

Rupnik, J., and Shawe-Taylor, J. (2010). "Multi-view canonical correlation analysis," in *Conference on Data Mining and Data Warehouses (SiKDD 2010)*, 1–4.

Sacramento, J., Costa, R. P., Bengio, Y., and Senn, W. (2018). "Dendritic cortical microcircuits approximate the backpropagation algorithm," in *Advances in Neural Information Processing Systems*, 8721–8732.

Sanger, T. (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Netw.* 2, 459–473. doi: 10.1016/0893-6080(89)90044-0

Snoek, C. G., Worring, M., Van Gemert, J. C., Geusebroek, J.-M., and Smeulders, A. W. (2006). "The challenge problem for automated detection of 101 semantic concepts in multimedia," in *Proceedings of the 14th ACM International Conference on Multimedia* (ACM), 421–430. doi: 10.1145/1180639.1180727

Spratling, M. W., and Johnson, M. H. (2004). A feedback model of visual attention. *J. Cogn. Neurosci.* 16, 219–237. doi: 10.1162/089892904322984526

Spruston, N. (2008). Pyramidal neurons: dendritic structure and synaptic integration. *Nat. Rev. Neurosci.* 9:206. doi: 10.1038/nrn2286

Stein, B. E., and Stanford, T. R. (2008). Multisensory integration: current issues from the perspective of the single neuron. *Nat. Rev. Neurosci.* 9:255. doi: 10.1038/nrn2331

Stuart, G., Spruston, N., Sakmann, B., and Häusser, M. (1997). Action potential initiation and backpropagation in neurons of the mammalian CNS. *Trends Neurosci.* 20, 125–131. doi: 10.1016/S0166-2236(96)10075-8

Urbanczik, R., and Senn, W. (2014). Learning by the dendritic prediction of somatic spiking. *Neuron* 81, 521–528. doi: 10.1016/j.neuron.2013.11.030

Vía, J., Santamaría, I., and Pérez, J. (2007). A learning algorithm for adaptive canonical correlation analysis of several data sets. *Neural Netw.* 20, 139–152. doi: 10.1016/j.neunet.2006.09.011

Xu, N.-l., Harnett, M. T., Williams, S. R., Huber, D., O'Connor, D. H., Svoboda, K., et al. (2012). Nonlinear dendritic integration of sensory and motor input during an active sensing task. *Nature* 492:247. doi: 10.1038/nature11601

Yang, X., Weifeng, L., Liu, W., and Tao, D. (2019). A survey on canonical correlation analysis. *IEEE Trans. Knowl. Data Eng.* doi: 10.1109/TKDE.2019.2958342

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.