Large Scale Networks for Contextual Inference, Routing and Motor Control

Charles H. Anderson [*] Brian J. Fischer

Department of Anatomy and Neurobiology, Washington University in St. Louis, St. Louis, MO 63110

Department of Electrical and Systems Engineering, Washington University in St. Louis, St. Louis, MO 63110

Abstract

The question of how neurons multiply is an important, but unresolved problem in neuroscience. Motivated by the problems of contextual inference, dynamic routing of information, and nonlinear control of motor systems, we feel the more important question is how networks of large ensembles of neurons multiply many pairs of numbers and sum the result; the computation of bilinear forms. Using the framework described in (Eliasmith and Anderson 2003), we show that bilinear forms can be computed in networks with “hidden layers” of neurobiologically realistic neurons, which can be far more efficient than digital systems and artificial neural networks.

Key words: multiplication, population codes, contextual modulation

1 Introduction

The question of how neurons multiply is an important, but unresolved problem in neuroscience. Most studies have focussed on multiplication at the level of individual neurons [2],[3],[6], whereas we feel the more important question is how networks of large ensembles of neurons multiply many pairs of numbers and sum the result; the computation of bilinear forms. Bilinear forms are required for contextual inference, dynamic routing, and nonlinear control of motor systems[1]. Previous work in this area has utilized dynamic modulation of the synaptic weights [8], nonlinearities in the dendrites [5],[4], or the use of

[*] Corresponding Author
Email addresses: `cha@wustl.edu` (Charles H. Anderson), `fischerb@pcg.wustl.edu` (Brian J. Fischer).

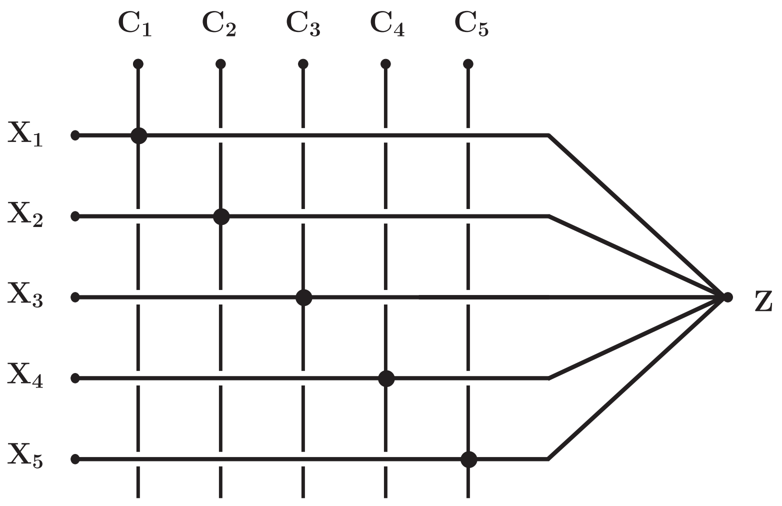

Fig. 1. Simple switch/contextual modulation circuit

"gain fields" [9],[7]. Much of the work on the role of synchrony and oscillations in neuronal circuits also addresses these issues, but in an intrinsically weaker computational manner. By formalizing the computation of bilinear forms using the framework described in Eliasmith and Anderson [1], we show that: (1) "gain fields" can be understood as a form of "hidden layer" computation, (2) the usual combinatoric explosion of hidden layer units does not happen when implemented using very noisy biologically realistic neurons, and (3) implementing these circuits as going directly from hidden layer to hidden layer makes them more efficient, while making the response properties of individual neurons broadly tuned along many dimensions, much as in cortical circuits.

2 Bilinear forms

Multiplicative interactions arise in the fundamental, but seemingly diverse, problems of contextual inference, dynamic routing of information, and nonlinear control of dynamical systems. These points are illustrated by the simple circuit shown figure (1), which implements the computation

$$Z = \sum_n C_n X_n. \tag{1}$$

When the values of the control parameters are limited to the values of 0 or 1, the circuit acts as switch that selects which input, $X_1, X_2, \ldots, X_5$ is allowed to flow into the output node indicated by Z. Such a circuit could be used to dynamically control the flow of information and perform coordinate transformations. On the other hand, if one lets the values of the control parameters take on arbitrary values, then the output of the circuit is a weighted sum of the inputs, where the weights C_n can be set by contextual information derived from bottom up clues, local lateral flow of information, or top down context. As diagramed in figure (1), the suggestion is that the modulation should take place either at the synapses [8] or within the dendrites of the neurons [4],[5]. However, when one generalizes the problem to computing bilinear forms in high dimensional spaces it is more natural to approach the problem using a set of hidden layer neurons.

Formally, we begin by describing the neural implementation of a class of mappings $F : (\mathbf{X}, \mathbf{Y}) \rightarrow \mathbf{Z}$ defined by the bilinear forms

$$Z_l = \sum_{nm} W_{lnm} X_n Y_m = \mathbf{X}^T W_l \mathbf{Y} \tag{2}$$

where $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \in R^D$, $W_l \in R^{D \times D}$, and $l \in \{1, \ldots, D\}$. We assume that separate populations of N neurons represent $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$. As described in [1], multiplication of two independently represented variables begins by the introduction of 'hidden layer' neurons that encode the two input spaces in an additive fashion. Having a sufficient numbers of neurons with a wide diversity of nonlinear response properties allows one to decode a large array of nonlinear transforms including functions like equation (2). Suppose that $\mathbf{X}$ is represented by a population of neurons with activities denoted by $\{a_i(\mathbf{X})\}$. The representation is summarized by the encoding rule

$$a_i(\mathbf{X}) = G[< \tilde{\phi}_i, \mathbf{X} >] \tag{3}$$

and the linear decoding rule

$$\hat{\mathbf{X}} = \sum_i \phi_i^x a_i(\mathbf{X}). \tag{4}$$

The ϕ_i^x have the same dimension as $\mathbf{X}$ and are found by minimizing a mean square error [1]. Similarly, $\mathbf{Y}$ is represented by a population of neurons with activities denoted by $\{b_j(\mathbf{Y})\}$. The representation is summarized by the encoding rule

$$b_j(\mathbf{Y}) = \mathcal{G}[< \tilde{\phi}_j, \mathbf{Y} >] \tag{5}$$

and the linear decoding rule

$$\hat{\mathbf{Y}} = \sum_j \phi_j^y b_j(\mathbf{Y}). \tag{6}$$

The hidden layer population of neurons represents the $2D$ dimensional space of vectors $(\mathbf{X}, \mathbf{Y})$ in the product space $R^D \times R^D$ with activities denoted by $d_l(\mathbf{X}, \mathbf{Y})$. The encoding rule for the hidden layer is

$$\begin{aligned} d_l(\mathbf{X},\mathbf{Y}) &= \mathcal{G}[<\tilde{\phi}^x_l, \mathbf{X}> + <\tilde{\phi}^y_l, \mathbf{Y}>] \\ &= \mathcal{G}[<\tilde{\phi}^x_l, \sum_i \phi^x_i a_i(\mathbf{X})> + <\tilde{\phi}^y_l, \sum_j \phi^y_j b_j(\mathbf{Y})>] \\ &= \mathcal{G}[\sum_i <\tilde{\phi}^x_l, \phi^x_i> a_i(\mathbf{X}) + \sum_j <\tilde{\phi}^y_l, \phi^y_j> b_j(\mathbf{Y})] \\ &= \mathcal{G}[\sum_i \omega^x_{li} a_i(\mathbf{X}) + \sum_j \omega^y_{lj} b_j(\mathbf{Y})]. \end{aligned} \tag{7}$$

The vector $\mathbf{Z} \in R^D$ with components given by the bilinear form of equation (2) can be decoded from the hidden layer activities with the linear decoding rule

$$\hat{\mathbf{Z}} = \sum_l \phi_l d_l(\mathbf{X},\mathbf{Y}). \tag{8}$$

Fundamentally, this problem is identical to the simpler coordinate transformation problem Zipser and Andersen [9] addressed many years ago. While they used back propagation to find the coupling weights, the above formulation allows one to compute the weights explicitly in this more general case and provides insight into why 'gain fields' arise in neurobiological circuits.

The difficulty with using hidden layers is that the number of units can grow exponentially with the complexity of the problem. For the case of equation (2), the dimension of each input space (D) may be on the order of 100 for some neurobiological systems such as cerebral cortex. The required number of $X_n - Y_m$ pairs that need to be multiplied is D^2, or $10,000$, and the number of units in the hidden layer would normally be scaled to the same degree. Surprisingly, the problem is not as bad for a neurobiological system built from large numbers of noisy realistic neurons as it is for a digital implementation, or even a standard artificial neural network using high signal to noise nonlinear devices. This unexpected result is based on the following signal to noise analysis.

It is well known that the average of a large number of quantities that are represented at a low signal to noise ratio (SNR) produces a result whose SNR is increased by the square root of the number of items in the sum. If one assumes the output SNR of the Z_l to be the same as that of the inputs X_m, Y_n, which are proportional to the square root of the number of neurons in the representations (N) divided the by the dimension D, then the SNR required to represent the individual $X_n - Y_m$ multiplicative pairs that are summed to form the outputs Z_l can be much lower.
Let $\mathbf{SNR}_z$, $\mathbf{SNR}_x$, $\mathbf{SNR}_y$, and $\mathbf{SNR}_{xy}$ be the signal to noise ratios of the Z_l, X_n, Y_m, and X_nY_m variables, respectively. Assume that $\mathbf{SNR}_z = \mathbf{SNR}_x = \mathbf{SNR}_y \propto \sqrt{N/D}$ where $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ are each represented by N neurons. Given that there are D^2 $X_n - Y_m$ pairs we have that $\mathbf{SNR}_{xy} \propto \sqrt{N_{xy}/D^2}$. If we assume that each Z_l variable is the result of summing D $X_n - Y_m$ pairs

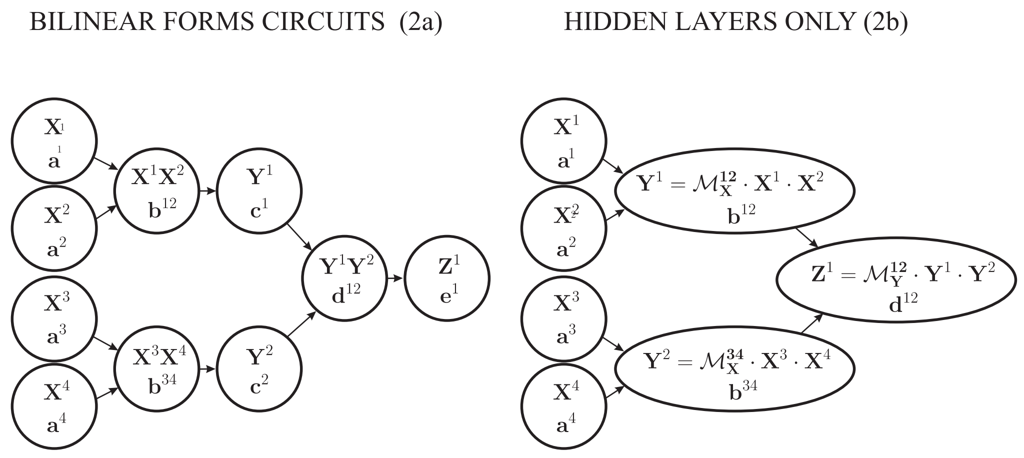
BILINEAR FORMS CIRCUITS (2a)
HIDDEN LAYERS ONLY (2b)
$\mathbf{X}^1$ $\mathbf{a}^1$
$\mathbf{X}^2$ $\mathbf{a}^2$
$\mathbf{X}^3$ $\mathbf{a}^3$
$\mathbf{X}^4$ $\mathbf{a}^4$
$\mathbf{X}^1\mathbf{X}^2$ $\mathbf{b}^{12}$
$\mathbf{X}^3\mathbf{X}^4$ $\mathbf{b}^{34}$
$\mathbf{Y}^1$ $\mathbf{c}^1$
$\mathbf{Y}^2$ $\mathbf{c}^2$
$\mathbf{Y}^1\mathbf{Y}^2$ $\mathbf{d}^{12}$
$\mathbf{Z}^1$ $\mathbf{e}^1$
$\mathbf{Y}^1 = \mathcal{M}_{\mathbf{X}}^{\mathbf{12}} \cdot \mathbf{X}^1 \cdot \mathbf{X}^2$ $\mathbf{b}^{12}$
$\mathbf{Z}^1 = \mathcal{M}_{\mathbf{Y}}^{\mathbf{12}} \cdot \mathbf{Y}^1 \cdot \mathbf{Y}^2$ $\mathbf{d}^{12}$
$\mathbf{Y}^2 = \mathcal{M}_{\mathbf{X}}^{\mathbf{34}} \cdot \mathbf{X}^3 \cdot \mathbf{X}^4$ $\mathbf{b}^{34}$

Fig. 2. Bilinear forms computed using hidden layer circuits

then we have $\mathbf{SNR}_z = \sqrt{D}\mathbf{SNR}_{xy}$ or, equivalently, $\sqrt{N/D} = \sqrt{D}\sqrt{N_{xy}/D^2} = \sqrt{N_{xy}/D}$. The net result is the number of neurons required to represent the $X_n - Y_m$ products in the hidden layer is roughly the same as the number required to represent the input and output spaces. It should be noted that in digital computers, most artificial neural networks, and in our simulations of these circuits, one does not generally have the freedom to reduce the precision of representation, hence one cannot achieve this reduction in resources.

Figure 2a schematizes a hierarchy of these multiplicative circuits. The system variables are denoted by the capital letters within each circle and the lower case letters denote the activities of the neurons that represent them. The inputs $\mathbf{X}^1$ and $\mathbf{X}^2$, represented by the neuronal ensembles a^1 and a^2, to the 'hidden layer' neurons b^{12} are additive. Linear decoding rules are then used to compute the outputs $\mathbf{Y}^1$ according to a bilinear form such as equation (2), which are stored in the neurons c^1. A parallel circuit computes the output space $\mathbf{Y}^2$, and then $\mathbf{Y}^1$ and $\mathbf{Y}^2$ are combined to form the output $\mathbf{Z}^1$. Figure 2b illustrates that there is no need to allocate resources (i.e. the neuronal ensembles c^1, c^2 and, e^1) to explicitly decode the output vector spaces $\mathbf{Y}^1$, $\mathbf{Y}^2$ and $\mathbf{Z}^1$. Instead one can simply jump from 'hidden layer' to 'hidden layer' and do the decoding of the output variables implicitly rather than explicitly. A consequence of this formulation is that the response properties of each neuron will be relatively diffuse even though the population encodes very precise information implicitly.

3 Summary

In summary, we have described a network solution to the contextual inference - routing problem that provides new insights as to how real neurobiological circuits can differ from convention digital computers and artificial neural network circuits using high signal to noise nonlinear units. (1) These circuits do not require dynamic synapses [8], nor nonlinear interactions in the dendrites [4]. While the latter may be used to increase the computational richness, they suffer from the property that the nonlinear interactions computed in dendritic trees cannot be shared across multiple output neurons. (2) The circuits build upon and formalize the intuitions about "gain field" response properties. (3) A signal to noise argument shows the number of hidden layer units does not grow as much as the required number of intermediate results would indicate. Indeed, these circuits depend on huge numbers of noisy neurons, with a wide diversity of response properties and tuning along many dimensions, i.e. very similar to real cortical neurons. (4) Finally, these circuits have been simulatationed using several thousand leaky integrate and fire spiking neurons.

Acknowledgments: The Mathers Foundation and The McDonnell Center for Higher Brain Function

References

[1] Eliasmith, C., Anderson, C.H., Neural Engineering, MIT press, 2003.

[2] Hatsopoulos, N., Gabbiani, F., Laurent, G., Elementary computation of object approach by a wide-field visual neuron., Science, 270: 1000-1003, 1995.

[3] Koch, C., Poggio, T., Multiplying with synapses and neurons. In Single Neuron Computation, McKenna, T., Davis, J.L., and Zornetzer, S.F., eds., Cambridge, MA, Academic Press, 315-345, 1992.

[4] Mel, B., Why have dendrites? A computational perspective., In Dendrites, Stuart, G., Spruston, N., and Hausser, M. eds., Oxford University Press, 271-289, 1999.

[5] Olshausen, B., Anderson, C.H., and Van Essen, D.C., A neurobiological model of visual attention and invariant pattern recognition based on dynamical routing of information., J. Neurosci., 13: 4700-4719, 1993.

[6] Peña, J.L., Konishi, M., Auditory receptive fields created by multiplication., Science, 292: 249-252, 2001.

[7] Salinas, E., Abbott, L.F., A model of multiplicative neural responses in parietal cortex., Proc. Natl. Acad. Sci. USA, 93: 11956-11961, 1996.

[8] Von der Malsburg, C. and Bienenstock, E., 1986, Statistical coding and short-term synaptic plasticity: A scheme for knowledge representation in the brain, in *Disordered Systems and Biological Organization* (E. Bienenstock et al., Eds.), NATO ASI Series, vol F20, Berlin: Springer-Verlag.

[9] Zipser, D., Anderson, R.A., A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons. Nature, 331: 679-684, 1988.