Adaptive Restructuring of Radial Basis Functions Using Integrate-and-Fire Neurons

Jeremy A. Marvel, *Member, IEEE*

***Abstract*— This paper proposes a neurobiology-based extension of integrate-and-fire models of Radial Basis Function Neural Networks (RBFNN) that adapts to novel stimuli by means of dynamic restructuring of the network's structural parameters. The new architecture automatically balances synapses modulation, re-centers hidden Radial Basis Functions (RBFs), and stochastically shifts parameter-space decision planes to maintain homeostasis. Example results are provided throughout the paper to illustrate the effects of changes to the RBFNN model.**

Keywords—radial basis functions; neural networks; feed-forward networks; machine learning

I. Introduction

A LIMITATION of many implementations of Radial Basis Function Neural Networks (RBFNNs) is the requirement for knowing the upper and lower bounds of input vector elements. During the construction of RBFNNs, archetype inputs are used to center the various Radial Basis Functions (RBFs) in the RBFNN's hidden layers. The result is a pre-processed normalization of the input space, and makes the RBFNN incapable of accommodating input data points beyond the numerical bounds of their initial training sets. Subsequent, post-training inputs are expected to fall somewhere within these multidimensional boundaries lest they fall into a "dead zone" in the network coverage. This limits their applicability within many unbounded domain problems (e.g., characterization and optimization of manufacturing and research robotics).

In contrast, the more traditional Feed-Forward Artificial Neural Network (FFANN) model trained using backwards propagation, which does not indelibly center its hidden-layer neurons based solely on single input vectors, is typically more capable of extrapolating and interpreting inputs beyond the scope of the initial training set.

The system introduced in this study represents an alternative training mechanism of the integrate-and-fire RBFNN model in [1]. Taking cues from neurobiology, the proposed RBFNN extension is capable of automatically adapting to novel stimuli and pruning its internal architecture to prevent homeostasis. Section II provides a brief overview of the base integrate-and-fire RBFNN model. Section III describes the biological basis of the adaptive retraining and restructuring of neural networks. Section IV discusses the process of dynamically adjusting the RBF weight exploration range for accelerated convergence. Section V describes a mechanism for identifying and accommodating novel stimuli. Section VI concludes with a description of a process for adaptively restructuring decision planes to optimize relevancy.

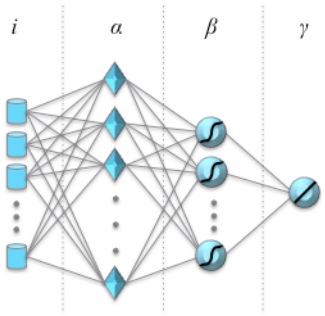

Fig. 1: RBFNN topology from [1] with two hidden layers. The first hidden layer (α) acts as an arbitrarily large number of decision planes to divide the input space (i), while the second (β) is used to recognize regions of the subdivided space for the output layer (γ).

II. Integrate-and-Fire RBFNN

Nonlinear patterns are more likely to be linearly separable in high dimensional spaces than they are in lower dimensions [2]. RBFNNs exploit this phenomenon by using nonlinear activation functions to decimate a parameter space. The individual RBFs are then trained to respond strongly for a finite region of input parameter space, and weakly for all others.

One can approximate the functionality of RBFNNs by using a simple, two-hidden-layer network topology (Fig. 1) consisting of simple integrate-and-fire neurons (e.g., [1, 3]). In Fig. 1, I input i-layer neurons feed into J α-layer decision plane neurons, which then feed into K β-layer basis neurons, which then funnel into a single output γ-layer linear perceptron.

The RBFs are centered in parameter space by using the α layer as a series of decision planes. Such decision planes are effectively switching bipolar neurons, defined as

$$g_j^{\alpha}\left(u_j\right) = g_j^{\alpha}\left(\sum_{i=0}^{j} x_i w_{j,i}\right) = \begin{cases} +1 \; \forall u_j \geq 0 \\ -1 \; \forall u_j < 0 \end{cases}, \qquad (1)$$

where the output of the j^{th} RBF, g, subdivides the parameter space based on the function inputs, x, and their associated weights, w. Here the α neurons have their input weights $w_{j,i}$ set to random values (ex. in range of -1 to +1). Assuming that the inputs are scaled to be in range of $x_i \in [-1, 1]$, the bias weights $w_{j,0}$ can be chosen randomly in the range of

$$w_{j,0} \in \left[-\sum_{i=1}^{j}\left|w_{j,i}\right|, \sum_{i=1}^{j}\left|w_{j,i}\right|\right]. \qquad (2)$$

Selecting bias weights in this range prevents the decision planes from either never responding or always responding.

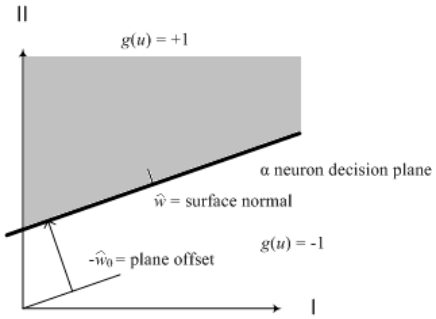

Fig. 2: A single α-layer neuron can be used to subdivide the parameter space into two regions.

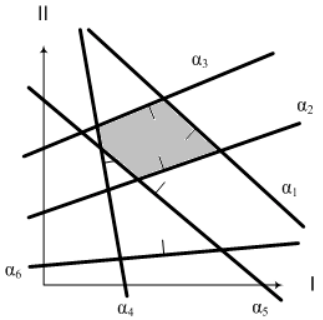

Fig. 3: The logical combination of multiple decision planes can be used to specify a particular region in space.

Neither of these extremes offers useful information to the network. The bias term defines the offset of the plane relative to the origin of the input space (Fig. 2). When multiple decision planes are thus defined, the input space is divided into sub-regions (Fig. 3). Specific sub-regions can be indexed in terms of the α-layer neuron activations. For instance, given the α-layer outputs, $\{\alpha_1, \alpha_2, \ldots, \alpha_6\}$, the shaded region in Fig. 3 is defined as $\{1, 1, 1, 1, 1, 1\}$.

The k^{th} β-layer neuron is "centered" to respond strongly to specific shaded regions by assigning the synaptic weight between the j^{th} α-layer neuron to

$$w_{k,j} = \frac{\operatorname{sgn}\left(g_j\left(u_j\right)\right)}{J}, \tag{3}$$

and the weight from the bias input to the k^{th} β-layer neuron is

$$w_{k,0} = -1. \tag{4}$$

With these weights, the input of the k^{th} β-layer neuron will be:

$$u_k\left(x_c\right) = w_{k,0} + \sum_{j=1}^{J} g_j^{\alpha}\left(x_c\right) w_{k,j} \ . \tag{5}$$

The k^{th} β-layer neuron will have a net input of 0 for all inputs x_c within the shaded region. Any inputs outside of the shaded region will have a negative net input.

Each β node is tuned to recognize specific network inputs, and is associated with a specific target output t_p. When presented with its trained stimulus, β-node k responds strongly with 1, while all other β nodes will respond less strongly, if at all. The weight $w_{l,k}$ from β-node k to the output γ node, l, is initialized as $w_{l,k} = t_i$. Thus the input to the linear γ node is

$$u_i = \sum_{k=0}^{K} w_{l,k} g_k^{\beta}\left(x_i\right) = t_i g_k^{\beta}\left(x_i\right) = t_i. \tag{6}$$

The α-layer decision planes can be set randomly and permanently to divide up the input parameter space. However, the placement of β nodes requires some forethought based on the training sets. Selecting random input sequences for centering is somewhat effective provided those random sequences are representative of the input population as a whole (e.g., if the inputs are themselves random with a known distribution). If the distribution of the input sequences is not known, however, applying simple clustering algorithms (e.g., K-Means [4]) to the training set may be more effective in producing useful responsiveness.

While the k^{th} β node dominates the output, $w_{l,k} = t_i$ is adequate for the input-output mapping approximation provided that 1) the number of β nodes equals the number of training inputs, and 2) the β node activation functions have narrow receptive fields. Otherwise, the β-layer activation receptive fields should partially overlap. If this is true, then the weight assignments of $w_{l,k} = t_i$ results in poor function fits. However, these weights may be optimized algorithmically.

Let $\mathbf{F}_K$ be the $P \times K$ matrix that describes the outputs of the K β-layer neurons for the P training patterns, and let $\mathbf{t}$ be the associated vector of P training pattern target outputs:

$$\mathbf{F}_K = \begin{bmatrix} g_1^{\beta}\left(x_1\right) & g_2^{\beta}\left(x_1\right) & \cdots & g_K^{\beta}\left(x_1\right) \\ g_1^{\beta}\left(x_2\right) & g_2^{\beta}\left(x_2\right) & \cdots & g_K^{\beta}\left(x_2\right) \\ \vdots & \vdots & \ddots & \vdots \\ g_1^{\beta}\left(x_P\right) & g_2^{\beta}\left(x_P\right) & \cdots & g_K^{\beta}\left(x_P\right) \end{bmatrix}, \quad \mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_P \end{bmatrix}.$$

The vector of β-γ weights $\mathbf{w}$ is computed by solving for the pseudo-inverse:

$$\mathbf{w} = \left(\mathbf{F}_K^{\mathrm{T}} \mathbf{F}_K\right)^{-1} \mathbf{F}_K^{\mathrm{T}} \mathbf{t}. \tag{7}$$

Alternatively, the weight vector solution can be approximated using a stochastic gradient descent search.

The error function, $e(\mathbf{w})$, describes the distance from the output of the RBFNN to the target output vector $\mathbf{t}$:

$$e\left(\mathbf{w}\right) = \left\| \mathbf{F}_K \mathbf{w} - \mathbf{t} \right\| \tag{8}$$

The optimum set of weights, $\mathbf{w}_{\text{opt}}$, minimizes this error expression. The solution for this is known to be equal to the pseudo inverse computed in Eq. 7, and has a known minimum error $e_{\min} = \left\| \mathbf{F}_K \mathbf{w}_{\text{opt}} - \mathbf{t} \right\|$.

For the n^{th} trial iteration, the β-γ synaptic weights are $\mathbf{w}_n$, and have an associated error e_n. For iteration n+1, a weight vector of random perturbations, $\delta\mathbf{w}$, is added such that $\mathbf{w}_{n+1} = \mathbf{w}_n + \delta\mathbf{w}$, and $e_n = \left\| \mathbf{F}_K \mathbf{w}_{n+1} - \mathbf{t} \right\|$. If $e_{n+1} < e_n$, then the network performance is improved, and the new synaptic weights are

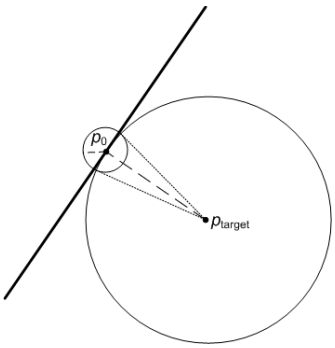

Fig. 4: The radius of the circle centered at the optimum β-γ weight vector, p_{target} is equal to the Euclidean distance from the optimum weights to the current synaptic weight estimate, p_0.

kept. Otherwise the changes are rejected and the weights are reverted such that $\mathbf{w}_{n+1} = \mathbf{w}_n$. This rejection mechanism ensures that the convergence of $e_n \rightarrow e_{\min}$ is monotonic.

III. Neurobiological Basis for Learning

The biological central nervous system, on which artificial neural networks are based, can adapt to, learn from, and remember novel experiences. Unlike artificial neural networks, which converge on solutions by complex algorithms, biological neural systems adapt through long-term potentiation (LTP) to modulate communication through the synapses. The combination of LTP and long-term depression (LTD) creates, destroys, strengthens, and weakens synapses as the fundamental component of biological learning.

In 1961, Hebb introduced the model of synaptic modulation as the underlying basis for memory and learning [5]. The model of "Hebbian learning" is typically simplified as, "cells that fire together, wire together." Taken by itself, however, the Hebbian algorithm of synapse modulation is not sufficient for explaining the process of learning. For instance, it focuses on synaptic strengthening, and does not consider the reversed action of synaptic weakening (e.g., [6]). Regardless, it is a convenient basis for modeling LTP.

The neurotransmitter Dopamine is the leading agonist for the positive reinforcement of Hebbian learning within cerebral tissues. Moreover, background levels of Dopamine indicate the feedback response states. Increased Dopamine concentrations signal positive learning results, while decreased levels mark depression. Consistent concentrations of Dopamine indicate status quo, and spikes in the Dopamine levels signal short-term reinforcement.

While Dopamine acts as the catalyst for increases in synaptic effectiveness, the LTP trigger involves the postsynaptic activity of the N-Methyl D-Aspartate (NMDA) neurotransmitter receptor. The NMDA receptor is activated by Glutamate, but is inhibited by an extracellular magnesium ion blocks. This magnesium ion precludes the NMDA receptor from contributing to the signal propagation process [7].

The induction of LTP requires the activation of the postsynaptic NMDA receptors during the postsynaptic action potential. This is induced by an increase in sensor input frequency [8]. Following high-frequency stimulation and postsynaptic action potentials during the Dopamine-assisted strengthening, the magnesium ion is ejected from the NMDA receptor. Glutamate is permitted to bind, which results in more channels in the neuron's walls opening. The pre- and post-synaptic action combination results in an increase in calcium concentration within the cell. This concentration increase serves as a coincidence detector that associates the signal propagation with a given reward (i.e., Dopamine). The result is an increased number of neurotransmitter receptors on the synapses linking those neurons to the cell, and more effective signal transmission.

The process of LTP can be further influenced by other neurological phenomena. For instance, neurons may fire either spontaneously or as a result of errant signals. Both pre- and postsynaptic spikes can induce LTP and LTD [9], allowing even optimized networks to recognize and adapt to novel stimuli. Another observed reaction is that of homeostasis, where synaptic signals are turned down when it is determined they are not providing useful feedback to the system. Although considered to be separate from Hebbian learning and typically associated with entire systems of neurons, homeostatic responses have been observed at localized regions in the hippocampus [10].

The traditional RBFNN construction has been shown to be effective for predefined training sets. However, the RBFNN's inability to interpret or adapt to new, significantly different inputs outside of the initial training set prevents it from being as versatile as FFANNs trained through backpropagation. The RBFNN implementation must thus be extended to accommodate dynamic data sets. The biological system described previously provides a plausible basis for accomplishing this.

IV. Dynamic *B*-*Γ* Weight Mutation Ranges

The background concentration of Dopamine can be simulated within in the stochastic gradient descent search of the β-γ weight space. Positive reinforcement results in improved performance (i.e., reduced error). The Dopaminergic reinforcement can be used to accelerate and fine-tune the convergence of the weights to their optimum values.

Fig. 4 illustrates a two-dimensional (2D) weight space representation of the current and target β-γ synaptic weights. The coordinate p_0 is the current value of the β-γ weights, and p_{target} is the globally optimal weight vector. The radius of the circle centered at p_0 is the random perturbation range $\delta\mathbf{w}$. The radius of the circle centered at p_{target} is the Euclidean weight-space distance separating p_0 and p_{target}, $\overline{p_0 p_{target}}$. The tangent bisecting the circle centered at p_0 is a hypothetical probability line. For sufficiently small enough $\delta\mathbf{w}$, the probability that the perturbation will result in a positive motion toward p_{target} is roughly 50 %. As $\delta\mathbf{w}$ increases relative to the radius $\overline{p_0 p_{target}}$, however, this probability decreases. Because it is assumed that p_{target} is not known, the optimum value of $\delta\mathbf{w}$ is also not known. Using the Dopaminergic reinforcement model, however, the value can be approximated dynamically.

The algorithm for this Dopaminergic reinforcement model is simple: if the application of a weight perturbation results in an improved fit, increase the value of $\delta\mathbf{w}$, otherwise

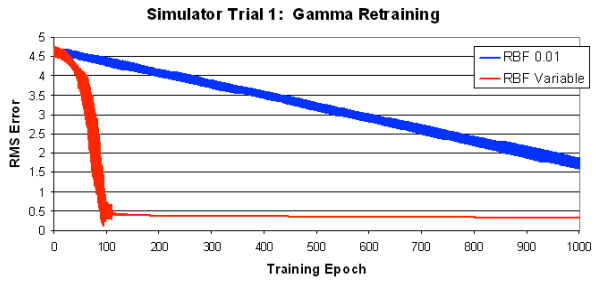

Fig. 5: The performance for the stochastic weight search is monotonically decreasing for both the fixed and the variable weight ranges, but the variable model approaches a minimum error at a significantly increased rate.

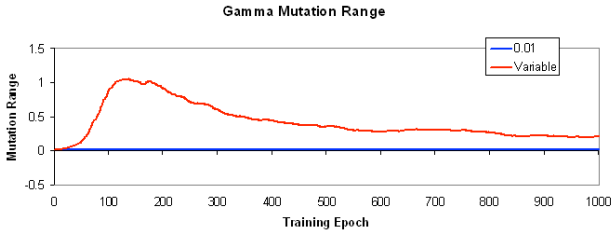

Fig. 6: In the Gaussian model example, the variable weight range implementation rapidly increase in the value of δw until it nears an optimum value, after which the mutation range narrows for fine-tuning.

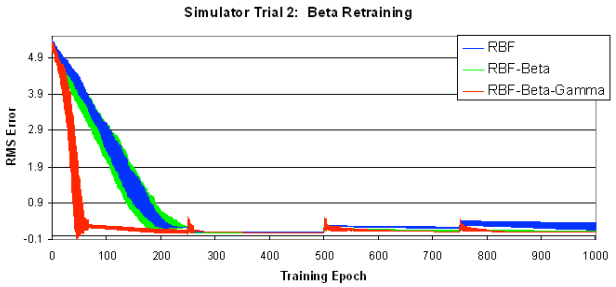

Fig. 7: The performance of a trained RBFNN is impacted by the introduction of new data outside of the network's known parameter space. Without assimilating new data into the network the RMS error steadily increases.

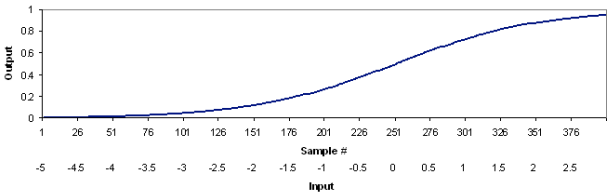

Fig. 8: Sub-sampled sigmoid training data for the β-layer retraining trials.

reduce the value of δ**w**. One possible implementation of the algorithm is as follows:

```
if e_{n+1} < e_n then
  Fail = 0
  ∀k, δw_k *= δ_inc, where δ_inc > 1.0
else
  Fail = Fail + 1
  if Fail > Fail_max then
    ∀k, δw_k *= δ_dec, where δ_dec < 1.0
```

The performance error, e_{n+1}, is compared to the previous step's error. The values of δ_{inc}, δ_{dec}, and $Fail_{max}$ are arbitrary and thus user-defined, but may be empirically derived. The results of applying this model to a simple example are shown in Fig. 5 and Fig. 6. Fig. 5 shows the root mean square (RMS) error for an example RBF application using a fixed δ**w** of 0.01 as compared with the RMS error of the same RBF using a variable-value δ**w**. Each line represents the average of 10 runs, and the thickness of the plots represent 1σ standard deviation. For the Dopaminergic reinforcement model, $\delta_{inc} = 1.1$, $\delta_{dec} = 0.99$, and $Fail_{max} = 10$. The network's training patterns consist of 200 randomly sampled points on the surface of a symmetric 2D Gaussian curve defined by

$$z = e^{\frac{(x-c_X)^2}{2\sigma_X^2}} + e^{\frac{(y-c_Y)^2}{2\sigma_Y^2}}, \qquad (9)$$

where $c_X = c_Y = 3$, and $\sigma_X = \sigma_Y = 1$.

The network is composed of 500 randomly placed α-layer linear neurons, and 25 β-layer sigmoidal neurons centered using K-means clustering. The results of training the network ten times were averaged to provide an expected performance for this example. These results are plotted in Fig. 5, where the line widths represent 1 standard deviation of the RMS error at each time step. The constant δ**w** model has a slow, monotonically decreasing error. In contrast, the new variable-value δ**w** converges much faster. Fig. 6 illustrates example values of δ**w** as the number of training iterations increases in one test evaluation. A notable increase in δ**w** is observed with the variable weight model as the algorithm begins its exploration. As the values of **w** approach $\mathbf{w}_{opt}$, the value of δ**w** steadily decreases.

V. *B*-Layer Retraining

Biological systems exhibit reactions to repeated, novel stimuli by means of spontaneous firing, cell death, and the growth of new inter-neuron connections. They are also capable of single-instance imprinting of knowledge [11]. This imprinting behavior is inherent in the RBFNN model described in Section II in the form of the initial positioning of the β-layer neurons. However, the RBFNN is not capable of adjusting to new stimuli, nor is it able to allow for the focusing in on regions of interest.

The RBFNN is not able to capture new data points if they are far from the centers of the defined β-layer neurons. While the old memories are still important, especially in preventing network regression, their training requires dedicated β-layer neurons that are not typically generalizable. To accommodate new information, the β-layer neurons may need periodic retraining. Fig. 7 shows the effects of periodic retraining when presented with new data. Each line represents the average of 10 runs ±1 standard deviation. The data fed into the RBFNN is gathered by sequentially evaluating a 1D sigmoid function (Fig. 8) defined as

$$y = \frac{1}{1+e^{-x}}, \qquad (10)$$

which is sampled at regular, increasing intervals for values of x between -5.0 and 3.0.

Three RBFNNs were created for testing: a basic integrate-and-fire RBFNN ("RBF"), an RBFNN with β-layer neuron retraining ("RBF-Beta"), and an RBFNN with both β-layer retraining and dynamic β-γ weight modulation ("RBF-Beta-Gamma"). Each RBFNN consisted of 400 randomly placed α-layer decision planes, and 25 β-layer neurons. The initial training set for the network consisted of the first 25 data points in the range [-5.0, -4.52]. Because the number of β-layer neurons is equal to the number of data points, the RBFNN is over-determined for the initial training set. The second training set consisted of the next 175 sequential data points. The third and fourth training sets are composed of 100 points each, for a total of 400 training sample points. This division of data points was arbitrarily chosen to illustrate the training on nonlinear data.

Each subsequent training set is introduced every 250 training iterations to allow the networks to converge to some optimum state. All training inputs are scaled to be in the range of [-1, 1], and all training outputs are scaled to be in the range of [0, 1]. This pre-scaling served to avoid conflicts with the α-layer decision planes.

Fig. 7 shows the average RMS errors for the three RBFNNs. The three RBFNNs all experience spikes in RMS error whenever a new training set is introduce. The RMS error then slowly declines as the networks adapt to the new information. As the input space becomes more complex with additional data sets, the basic RBFNN is less able to accommodate new information. As a result, the minimum RMS error steadily increases. In contrast, the two RBFNNs with β-layer neuron retraining fare considerably better with noticeably smaller minimum RMS errors.

VI. RBFNN Homeostasis

To prevent the system from going unstable, biological systems utilize means of homeostasis to adjust the synaptic sensitivity between two adjacent neurons [11]. Constant stimulation can wear out the nervous system, so tolerance to stimuli is developed. However, the sensitivity never fully disappears. Similarly, in the absence of stimulation, biological systems have been observed increasing the signal strength to accommodate signal degradation. In instances of sensory or activation deprivation, synapses have increased in size and the number of receptors [12].

At the α-layer, homeostasis is essentially the movement of the decision planes to where most temporally relevant query values are focused. If the activity of a given α-layer neuron is either too frequent or too infrequent, it is moved to a new random location within the parameter space. The signal activity of a given neuron, j, is defined as the activation frequency, p_j^α:

$$p_j^\alpha = \frac{1}{T_j}\sum_{n=1}^{T_j}\left(\left|g_j^\alpha\left(\sum_{i=0}^{I} x_{i_n} w_{j,i}\right)\right| > g_{\min}^\alpha\right). \qquad (11)$$

The "lifespan" of the α-layer neuron, T_j, represents the number of evaluation iterations since the last time the neuron was retrained. The signal threshold, $g_{\min}^\alpha$, is user-defined, but is set at 50 % for evaluative trials.

The neuron is said to be in a homeostatic state if the inequality

$$0 \le p_{\min}^\alpha < p_j^\alpha < p_{\max}^\alpha \le 1, \qquad (12)$$

is true, where $p_{\min}^\alpha$ and $p_{\max}^\alpha$ are user-defined bounding variables. In the unmodified RBFNN implementation, $p_{\min}^\alpha$ and $p_{\max}^\alpha$ are 0 and 1, respectively. If the value of p_j^α is outside of this range, the α-layer decision plane neuron is moved. The optimal placement of decision planes can be achieved via the same stochastic approach used to discover the β-γ weights discussed in Section II. To provide a reasonable evaluation of the performance of each neuron, a minimum lifespan must be achieved before the decision to retrain an α-layer neuron can be made. For trials, the firing frequency is not evaluated until $T_j > 500$ evaluations.

Making such changes at the α-layer necessitates retraining the entire RBFNN, which will impact the short-term network performance. As a result, the convergence toward optimality is no longer monotonic. However, optimizing the placement of α-layer decision planes based on the observed network activity is expected to provide better long-term network performance.

In contrast, the β-layer equivalent to homeostasis adjusts the receptive fields of individual neurons to maintain an optimized activation rate. Moreover, the β-layer neurons that are rarely queried are moved to locations where they are more likely to actively participate in the network's application. For such processes to be possible, each β-layer requires a level of system-wide knowledge that has no biological equivalent and is computationally inefficient. For example, each β-layer neuron needs to know both the distinction between a training query and a testing query, and the density and locations of other β-layer neurons to avoid over-determination. Further, the parameter influences of the β-layer neurons are not well understood for such modification to be implemented.

To evaluate the performance of α-layer homeostasis, a sample data set was created consisting of random points taken from the surface of an asymmetric 2D Gaussian (Fig. 9) using Eq. 9. For this surface, $c_X = 1$, $c_Y = -1$, $\sigma_X = 1$, and $\sigma_Y = 0.5$. The surface points are divided into two data sets. The first data set consisted of 100 data points sampled for values of x in the range of (-3, -1) and values of y in the range of (-1, 3). The second data set consisted of 300 data points sampled for values of x in the range of (-1, 1), and values of y in the range of (-1, 3). Unlike the evaluation performed in Section V, the inputs and outputs of the training sets are not pre-scaled. Instead, the RBFNNs were modified to accommodate the raw training inputs and outputs.

The network consisted of 200 α-layer decision planes and

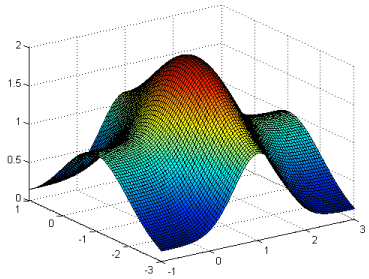

Fig. 9: 2D Gaussian surface used for evaluating the homeostatic placement of α-layer neurons.

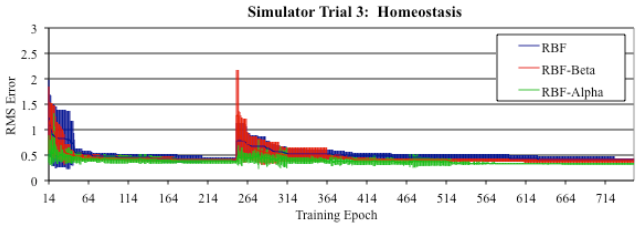

Fig. 10: Without the β-γ weight modulation, the three networks approach optimal solutions at roughly the same rate. The network with α-layer homeostasis, however, generally performs more consistently.

20 β-layer neurons. The RBFNN with α-layer homeostasis, "RBF-Alpha," is compared with a standard integrate-and-fire RBFNN ("RBF") and an RBFNN with β-layer retraining ("RBF-Beta") to accommodate the second data set. With the α-layer homeostasis, $p^{\alpha}_{\min} = 0.1$ and $p^{\alpha}_{\max} = 0.8$. Each trial for the three RBFNNs consisted of 250 training epochs for the first data set, followed by 500 training epochs for the second. The mean results are illustrated in Fig. 10 and Fig. 11, where the thickness of each trend line represents 1σ of the RMS error at each time step.

Fig. 10 omits the first 14 training epochs to minimize visualization issues due to scaling. All three networks experience comparatively high RMS errors when new data points are introduced. Because α-layer homeostasis necessitates β-layer retraining, it is expected RBF-Alpha will perform at least as well as RBF-Beta. However, when given sufficient time to converge on a more-optimal set of decision planes, the RBFNN with α-layer homeostasis performs more consistently in that the standard deviation of RMS error remains comparatively small. However, as is evidenced in Fig. 11, the convergence toward an optimum network topology is not monotonic, and the network is prone to minor performance regressions as α-layer decision planes are moved randomly.

VII. Conclusions

This paper introduces extensions to the integrate-and-fire RBFNN model that enables the neural network to adapt to novel stimuli and self-optimize the network's topology. These extensions are based on neurologically observable phenomenon, and improve the performance of integrate-and-fire RBFNNs. Much like the observed effects of Dopamine

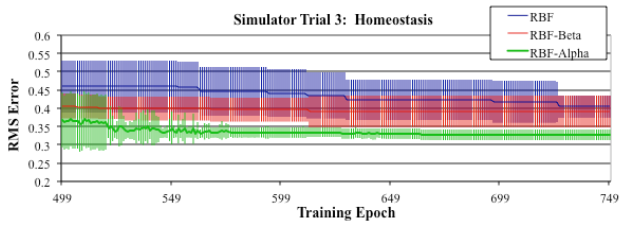

Fig. 11: The RMS error of the RBFNN with α-layer homeostasis, while not monotonically improving, results in improved network performance when given sufficiently long training times.

on LTP, modulating the β-γ weight mutation rate results in accelerated convergence on synapse optima. By periodically retraining the β-layer neurons as new training sets are introduced, a RBFNN can adapt to novel stimuli and generalize the network to accommodate the entire spectrum of inputs beyond the original training set. Implementing α-layer homeostasis on a RBFNN can stochastically optimize the placement of decision planes such that the α-layer activations remain relevant for the entire spectrum of active inputs, and can adapt as some inputs lose relevancy. This dynamic restructuring allows the RBFNN to be used in online learning scenarios, recognizing and adapting quickly to changes in the environment, expanding their use to a myriad of new application domains such as manufacturing and collaborative robotics (e.g., dynamic parameter and control optimization, tolerance band shifting, dynamic retraining).

References

[1] R. Hudson, J. Marvel, and W. Newman, "Modeling and training radial basis functions with integrate-and-fire neurons," *Proc. 9th Int. Conf. Mach. Learn. Appl.*, 2010, pp. 217-222.

[2] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Trans. Electron. Comput.,* vol. EC-14, pp. 326-334, 1965.

[3] G.-B. Huang, L. Chen, and C.-K. Siew, "Universal Approximation Using Incremental Constructive Feedforward Networks with Random Hidden Nodes," *IEEE Trans. Neural Networks,* vol. 17, pp. 879-892, 2006.

[4] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proc. 5th Berkeley Symp. Math. Stat. Probab.*, 1967, pp. 281-297.

[5] D. O. Hebb, *Organization of Behavior: A Neuropsychological Theory*. New York: John Wiley & Sons, Inc., 1961.

[6] C. Koch, *Biophysics of Computation: Information Processing in Single Neurons*. New York: Oxford University Press, 1999.

[7] G. M. J. Ramakers, I. J. A. Urban, P. N. E. deGraan, and W. H. Gispen, "Hebbian plasticity in the hippocampus: Involvement of protein phosphorylation and protein Kinase C," in *Neurobehavioral Plasticity: Learning, Development, and Response to Brain Insults*, N. E. Spear, L. P. Spear, and M. L. Woodruff, Eds., ed: Lawrence Erlbaum Associates, 1995.

[8] R. C. Malenka, "Synaptic plasticity in the hippocampus: LTP and LTD," *Cell,* vol. 78, pp. 535-538, 1994.

[9] L. F. Abbott and S. B. Nelson, "Synaptic plasticity: Taming the beast," *Nat. Neurosci.,* vol. 3, pp. 1178-1183, 2000.

[10] N. Vitureira and Y. Goda, "The interplay between Hebbian and homeostatic synaptic plasticity," *J. Cell Biol.,* vol. 203, pp. 175-186, 2013.

[11] E. R. Kandel, J. H. Schwartz, and T. M. Jessell, *Principles of Neural Science*, 4 ed.: McGraw-Hill Medical, 2000.

[12] G. O. Turrigiano and S. B. Nelson, "Homeostatic plasticity in the developing nervous system," *Nat. Reviews: Neurosci.,* vol. 5, pp. 97-107, 2004.