Universal Computation with Low-Complexity Wireless Relay Networks

Eric Slottke, Raphael Rolny, and Armin Wittneben
Communication Technology Laboratory, ETH Zurich
{slottke, rolny, wittneben}@nari.ee.ethz.ch

***Abstract*—We propose a method for enabling complex computations in a network of low-complexity wireless devices. By utilizing multihop relaying, such devices can form the wireless equivalent of an artificial neural network (ANN). We provide a method for programming the network functionality in a decentralized fashion and demonstrate the robustness of wireless ANNs against node failures and imperfections. Applications of this scheme exist in low-complexity sensor networks, where elaborate calculations can be carried out in a distributed fashion, or for creating powerful ANNs with very high degrees of interconnectivity realized by the wireless medium.**

I. Introduction

Wireless sensor networks have become an important area of research due to the promising applications they offer, ranging from large-scale seismic monitoring [1] to biomedical sensing with highly miniaturized devices [2]. In such settings, the network typically comprises a large number of low-cost, low-complexity nodes. The extraction and further processing of the data is then usually performed by a central unit with higher complexity, as the computational power of the wireless sensors is very limited. However, employing a central unit may negate some of the key benefits of sensor networks, such as ad-hoc deployment in almost any environment and low maintenance requirements.

Novel networking paradigms, such as node cooperation, applied with the goal of achieving elaborate computational abilities in a distributed manner, are therefore an open research topic. To this end, it has been recognized in literature that the interconnected topology of wireless sensor networks resembles the structure of artificial neural networks (ANNs), a model of computation abstracted from the interaction of biological neurons. Applying the principle of neurocomputation to wireless sensor networks is a natural choice, as the computation in ANNs is distributed, parallel, robust with respect to noisy data, and capable of learning. Consequently, recent research has proposed wireless implementations of artificial neural networks [3]–[5]. These approaches share the concept of implementing an ANN structure by reducing the broadcast nature of the wireless medium to point-to-point connections. In fact, as will be discussed in Sec. II, the ability to control each connection between the nodes in the network separately is essential to achieving the desired computational properties. In state of the art wireless ANN approaches, this control is realized either by orthogonalization of signal transmission, e.g. in frequency, or by abstraction of the ANN functionality using higher communication layers. Both mechanisms, however, have an undesirable scaling of used communication resources and complexity requirements of the nodes as the size of the network increases, rendering them unfeasible for the considered setting of large-scale, low-complexity networks.

In this work, we present an implementation of wireless ANNs that preserves the broadcast properties of the medium based on designating a subset of the nodes in the network as relays. Relay networks have been well studied in literature and are known to improve communications e.g. by enabling orthogonalization of multiple data streams (cf. [6], [7], and the references therein). In an extension of this approach, we use amplify-and-forward relaying to arbitrarily shape the channel matrix, which incorporates the channel as part of the desired computation. This allows for a narrowband and analog implementation of an ANN with low complexity constraints on the nodes even for large networks. We show that the necessary configuration of the relays can be carried out in a distributed form, which further reduces overhead resulting from feedback. Furthermore, the robustness of the computation to error sources typically encountered in wireless networks is demonstrated.

II. Wireless Artificial Neural Networks

We consider wirelessly implementing neural networks of the multilayer feedforward form. In this model, several nodes called neurons are arranged in L sequential layers, as exemplified in Fig. 1. The first layer acts as input to the network, i.e. a data vector $\mathbf{x}^{(1)}$ from this layer is passed further through the network. This process can be described as a concatenation of linear and nonlinear transformations on the input. The linear part represents a weighting and superposition of transmitted

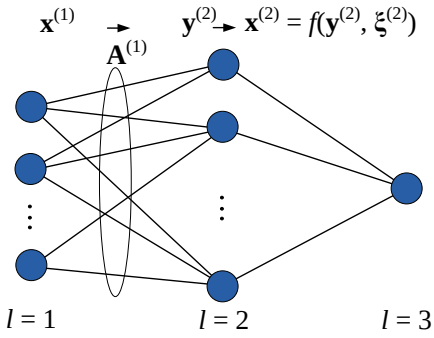

Fig. 1. Multilayer feedforward ANN with $L = 3$ layers of neurons.

data values, performed by the the independent connections between each transmitting and receiving neuron. In general, the received data vector at the neurons of the $(l+1)$th layer is given by the linear relation

$$\mathbf{y}^{(l+1)} = \mathbf{A}^{(l)}\mathbf{x}^{(l)}, \qquad l \in \{1, \ldots, L-1\}, \tag{1}$$

where $\mathbf{x}^{(l)}$ are the outputs of the neurons in layer l, and the weights of each connection correspond to the elements of the weight matrix $\mathbf{A}^{(l)}$.

After receiving, the neurons perform a nonlinear transformation described by their activation function f; the result of this transformation is passed as output to the subsequent layer. This output $x_i^{(l)}$ of the ith neuron in the lth layer is therefore defined as:

$$x_i^{(l)} = f\left(y_i^{(l)}, \xi_i^{(l)}\right), \qquad i \in \mathcal{M}^{(l)}, l \in \{2, \ldots, L\}, \tag{2}$$

where the input value $y_i^{(l)}$ is the ith element of the receive vector $\mathbf{y}^{(l)}$, and the activation function may additionally depend on a constant parameter $\xi_i^{(l)}$. Furthermore, $\mathcal{M}^{(l)}$ denotes the index set of the neurons in the lth layer.

The result of the overall computation realized by the ANN is given by the output vector of the final layer $\mathbf{x}^{(L)}$, which obviously depends on the weight matrices and activation functions of all layers. It is known that this neural network structure has universal computational ability if the network is sufficiently large and the activation functions and weights are chosen appropriately [8]. It should be noted that the weights implementing a specific computation may either (in simple cases) be found by analysis, or, more practically relevant, by the use of learning algorithms (e.g. [9]).

A. Extension to Wireless Neurons

If we consider the interconnects between the neuron layers to be wireless, the linear input-output relation needs to be modified to describe the transformation of the data by the wireless channel. We assume each layer transmits its output data in an analog broadcast, leading to the following input-output relation:

$$\mathbf{y}^{(l+1)} = \mathbf{H}^{(l)}\mathbf{x}^{(l)} + \mathbf{n}^{(l+1)}, \qquad l \in \{1, \ldots, L-1\}. \tag{3}$$

The entries of the channel matrix $\mathbf{H}^{(l)}$ describe a scaling and phase rotation of the data signals, and the additional term $\mathbf{n}^{(l)}$ is due to thermal noise at the receiving neurons. The network implements a specific computation if the condition

$$\mathbf{H}^{(l)} = \mathbf{A}^{(l)}, \qquad \forall l \in \{1, \ldots, L-1\} \tag{4}$$

is fulfilled. Herein lies the key difference of using wireless interconnects: while conventional, or "wired" ANNs can control each connection weight independently, the weighting of a broadcast transmission will affect all receivers simultaneously. This means wireless neurons can not trivially set the weights of their transmitted data according to the weight matrices required to implement a specific computation.

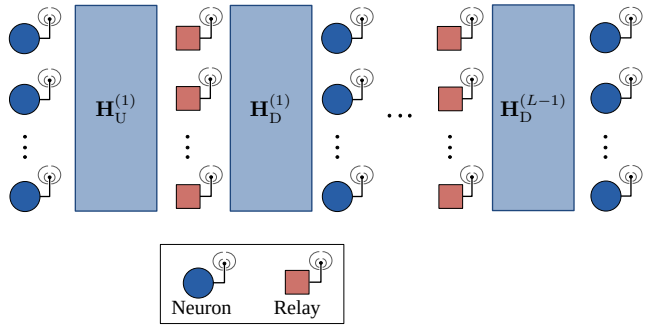

Fig. 2. System model of wireless ANN with L neuron layers. The input data is passed through the network in a multihop fashion.

B. Channel Shaping by Coherent Relaying

To enable implementation of the desired weights between subsequent layers, our approach employs an additional $n_{\mathrm{R}}^{(l)}$ nodes as coherent relays between the lth and $(l+1)$th neuron layer, leading to the network structure given in Fig. 2. The relays apply an amplify-and-forward operation, meaning the output the relays is given by their respective input, scaled by a complex gain coefficient ${g_k}^{(l)}$, and with an additional noise term ${m_k}^{(l)}$. Here, k is the relay index: $k \in \{1, \ldots, n_{\mathrm{R}}^{(l)}\}$. Introducing the relays splits the original channel matrix $\mathbf{H}^{(l)}$ into an uplink channel from the lth neuron layer to the subsequent relays, denoted as $\mathbf{H}_{\mathrm{U}}^{(l)}$, and a downlink channel between the relays and the $(l+1)$th neuron layer, $\mathbf{H}_{\mathrm{D}}^{(l)}$. With this model, the input-output relation given (3) is replaced by

$$\mathbf{y}^{(l+1)} = \mathbf{H}_{\mathrm{D}}^{(l)}\mathbf{G}^{(l)}\mathbf{H}_{\mathrm{U}}^{(l)}\mathbf{x}^{(l)} + \mathbf{H}_{\mathrm{D}}^{(l)}\mathbf{G}^{(l)}\mathbf{m}^{(l)} + \mathbf{n}^{(l+1)}, \tag{5}$$

where $l \in \{1, \ldots, L-1\}$, the matrix $\mathbf{G}^{(l)} = \mathrm{diag}\left\{\mathbf{g}^{(l)}\right\}$ contains the gains of the lth relay layer, and $\mathbf{m}^{(l)}$ is the corresponding relay noise vector.

The desired computation is implemented with this model if the choice of relay gains fulfills the following condition for all effective channels:

$$\mathbf{H}_{\mathrm{D}}\mathbf{G}\mathbf{H}_{\mathrm{U}} = \mathbf{A}. \tag{6}$$

We have dropped the layer index l here for convenience of notation. The relay gains satisfying (6) can be found analytically by solving the linear problem

$$\tilde{\mathbf{H}} \cdot \mathbf{g} = \mathrm{vec}\left\{\mathbf{A}^{\mathrm{T}}\right\}, \tag{7}$$

with the matrix $\tilde{\mathbf{H}}$ being given by

$$\tilde{\mathbf{H}} = \left[\mathbf{J}_{n_{\mathrm{RX}}} \otimes \mathbf{H}_{\mathrm{U}}{}^{\mathrm{T}}\right] \odot \left[\mathbf{P}_{n_{\mathrm{TX}}}\left(\mathbf{J}_{n_{\mathrm{TX}}} \otimes \mathbf{H}_{\mathrm{D}}\right)\right]. \tag{8}$$

Herein, $\mathbf{J}_n$ is the n-dimensional all ones vector, $n_{\mathrm{TX}} = \left|\mathcal{M}^{(l)}\right|$ is the number of transmitting neurons and $n_{\mathrm{RX}} = \left|\mathcal{M}^{(l+1)}\right|$ the number of receiving neurons. The operators $\otimes$ and $\odot$ further denote the Kronecker and Hadamard product respectively, and $\mathbf{P}$ is a permutation matrix which performs a sorting of rows in the right hand term.

Equation (7) has a unique solution for each relay layer if the number of used relays is $n_{\mathrm{R}} = n_{\mathrm{TX}} \cdot n_{\mathrm{RX}}$. As there may be an additional number of n_{ex} excess relays in the layer, we will choose the solution vector $\hat{\mathbf{g}}$ with minimal length, obtained by

$$\hat{\mathbf{g}} = \tilde{\mathbf{H}}^{+} \mathrm{vec}\left\{\mathbf{A}^{\mathrm{T}}\right\}, \tag{9}$$

where $(\cdot)^{+}$ denotes the pseudoinverse operation.

C. Proof of Concept

To assess the feasibility of applying this approach to implement a wireless ANN, we carry out a simulative analysis for an example ANN performing character recognition. The input data to this network are the pixel values of 7 by 4 bitmaps representing the digits zero through nine. Additionally, pixels in the input pattern are randomly flipped with a probability of approximately 0.16. The output generated by the network is an estimate of the digit corresponding to the input pattern. For the simulation of this setup, we assume an i.i.d. Gaussian distribution for the channels $\mathbf{H}_{\mathrm{D}}, \mathbf{H}_{\mathrm{U}} \sim \mathcal{CN}(\mathbf{0}, \mathbf{I})$ and noise vectors $\mathbf{m}, \mathbf{n} \sim \mathcal{CN}\left(0, \sigma_{\mathrm{N}}^2\right)$.

In Fig. 3, the simulated probability of the wireless ANN choosing an incorrect digit is shown in blue as a function of the signal-to-noise ratio (SNR) due to the thermal noise at the relays and neurons. The transmit power has been normalized here to let the SNR be equal to $1/\sigma_{\mathrm{N}}^2$. For comparison, the black line shows the performance of the equivalent wired implementation of the same network, which is assumed to be noiseless. It can be seen that the error probability of the wireless case well matches that of the conventional implementation if the SNR reaches sufficiently high levels of about $35\,\mathrm{dB}$. The error floor seen for high SNR is due to the random bit errors in the input pattern. For the case of error free input patterns, the performance of the wireless network is given by the red curve, which shows an additional improvement in error performance and no saturation for high SNR.

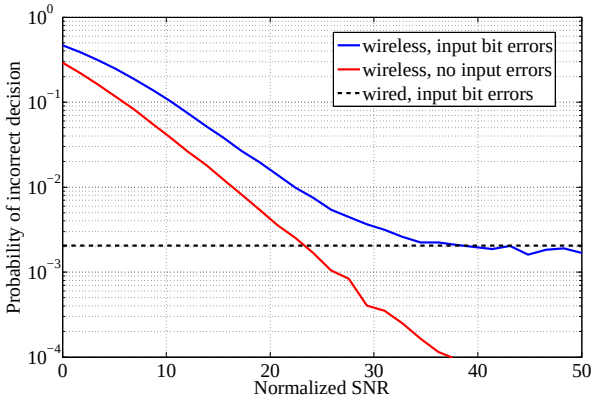

Fig. 3. Performance of wireless and wired implementations of an ANN performing character recognition.

III. Iterative Gain Allocation with Reduced Feedback

Finding the appropriate relay gains to implement a computation using (9) requires full knowledge of $\mathbf{H}_{\mathrm{U}}^{(l)}$ and $\mathbf{H}_{\mathrm{D}}^{(l)}$. The overhead generated by disseminating this channel state information (CSI) scales with the number of relays and neurons. To alleviate this potential drawback, we propose an iterative method for the relays to individually choose their gains based solely on their local CSI and feedback independent of the number of relays. Such a decentralized approach may prove particularly beneficial in large networks. To this end, we express the gain allocation for each layer as the minimization problem

$$\hat{\mathbf{g}} = \arg\min_{\mathbf{g}} \left\|\mathbf{H}_{\mathrm{D}}\mathbf{G}\mathbf{H}_{\mathrm{U}} - \mathbf{A}\right\|_{\mathrm{F}}^2, \tag{10}$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm. We obtain a solution by applying a modified version of the gradient based algorithm presented in [10], where decentralized relay gain allocation has been employed in a communications context. We can accordingly calculate the complex gradient of the norm in (10), which consists of the elements

$$\frac{\partial \left\|\mathbf{H}_{\mathrm{D}}\mathbf{G}\mathbf{H}_{\mathrm{U}} - \mathbf{A}\right\|_{\mathrm{F}}^2}{\partial {g_k}^*} = \mathrm{tr}\left\{\mathbf{H}_{\mathrm{D}}\mathbf{G}\mathbf{H}_{\mathrm{U}} \cdot \mathbf{H}_{\mathrm{U}}^{\mathrm{H}}\mathbf{E}_k^{\mathrm{H}}\mathbf{H}_{\mathrm{D}}^{\mathrm{H}} - \mathbf{A} \cdot \mathbf{H}_{\mathrm{U}}^{\mathrm{H}}\mathbf{E}_k^{\mathrm{H}}\mathbf{H}_{\mathrm{D}}^{\mathrm{H}}\right\}. \tag{11}$$

Here, $(\cdot)^{*}$ denotes the complex conjugate, $(\cdot)^{\mathrm{H}}$ is the Hermitian transposition, and $\mathbf{E}_k$ is a matrix with all zeros except for a one on the kth diagonal position. The desired weight matrix $\mathbf{A}$ can be assumed to be known a priori. Furthermore, the term $\mathbf{H}_{\mathrm{U}}^{\mathrm{H}}\mathbf{E}_k^{\mathrm{H}}\mathbf{H}_{\mathrm{D}}^{\mathrm{H}}$ corresponds to the local CSI at the kth relay, and the term $\mathbf{H}_{\mathrm{D}}\mathbf{G}\mathbf{H}_{\mathrm{U}}$ is the effective channel between the neuron layers, which can be fed back in a broadcast fashion by the receiving neurons. As the dimensions of these terms do not scale with the number of relays, the overhead required for CSI dissemination grows slower with n_{ex} than for methods requiring global CSI, such as the pseudoinverse based approach, making the gradient search well suited for large networks containing many relays.

However, the optimization cost function in (10) does not consider the amplification of noise received by the relays, which may have an impact on the error performance of the overall computation. We will therefore call the previously introduced scheme *gradient A*, and also investigate an alternative problem formulation denoted *gradient B*, which instead attempts to minimize the mean squared error in the signal received by the neurons. The new cost function is then

$$\hat{\mathbf{g}} = \arg\min_{\mathbf{g}} \mathsf{E}\left[\left\|(\mathbf{H}_{\mathrm{D}}\mathbf{G}\mathbf{H}_{\mathrm{U}} - \mathbf{A})\mathbf{x} + \mathbf{H}_{\mathrm{D}}\mathbf{G}\mathbf{m} + \mathbf{n}\right\|_{\mathrm{F}}^2\right], \tag{12}$$

where we use the operator $\mathsf{E}[\cdot]$ to denote expectation with respect to $\mathbf{x}$, $\mathbf{m}$, and $\mathbf{n}$. We can again calculate the complex

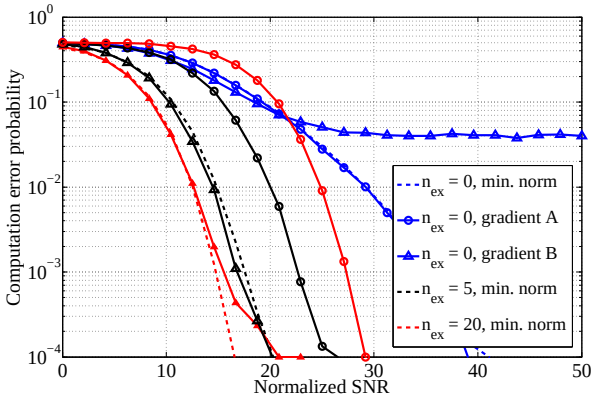

Fig. 4. Error probability of the XOR network versus SNR for different relay gain allocation schemes.

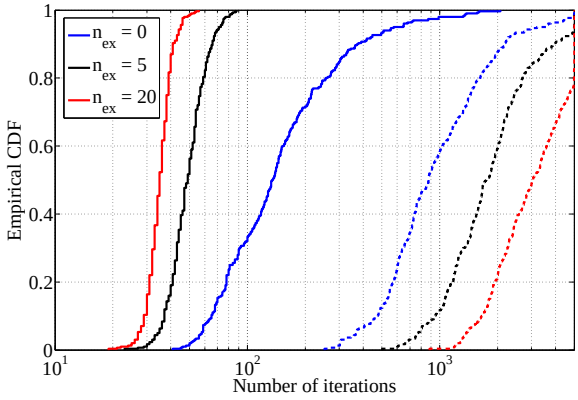

Fig. 5. Convergence behavior of gradient based relay gain allocation schemes. (Solid) gradient scheme A. (Dashed) gradient scheme B.

gradient, which now has the elements

$$\frac{\partial \mathsf{E}\left[\|(\mathbf{H}_\mathrm{D}\mathbf{G}\mathbf{H}_\mathrm{U}-\mathbf{A})\mathbf{x}+\mathbf{H}_\mathrm{D}\mathbf{G}\mathbf{m}+\mathbf{n}\|_\mathrm{F}^2\right]}{\partial g_k{}^*} = \mathrm{tr}\left\{\mathbf{H}_\mathrm{D}\mathbf{G}\mathbf{H}_\mathrm{U}\cdot\mathbf{K}_\mathbf{x}\cdot\mathbf{H}_\mathrm{U}^\mathrm{H}\mathbf{E}_k^\mathrm{H}\mathbf{H}_\mathrm{D}^\mathrm{H}-\mathbf{A}\mathbf{K}_\mathbf{x}\cdot\mathbf{H}_\mathrm{U}^\mathrm{H}\mathbf{E}_k^\mathrm{H}\mathbf{H}_\mathrm{D}^\mathrm{H} + \sigma_\mathrm{N}^2\mathbf{H}_\mathrm{D}\mathbf{G}\mathbf{E}_k^\mathrm{H}\mathbf{H}_\mathrm{D}^\mathrm{H}\right\}. \quad (13)$$

This gradient is similar to (11), but now considers the covariance matrix of the input signal, $\mathbf{K}_\mathbf{x}$, and the noise contribution term $\sigma_\mathrm{N}^2\mathbf{H}_\mathrm{D}\mathbf{G}\mathbf{E}_k^\mathrm{H}\mathbf{H}_\mathrm{D}^\mathrm{H}$.

To evaluate the suitability of the two presented decentralized gain allocation schemes, we apply them to a simple network to find the relay gains achieving a Boolean XOR operation on random input bits, which is a canonical and well understood example in the theory of neural networks. A possible implementation of the XOR operation is given by a network with $L = 3$ neuron layers using activation functions chosen as a shifted unit step function $f\left(y_i^{(l)}, \xi_i^{(l)}\right) = u(y_i^{(l)} - 0.5)$, and weight matrices chosen according to

$$\mathbf{A}^{(1)} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \text{ and } \mathbf{A}^{(2)} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}. \quad (14)$$

The key parameters used in the analysis of this network are the number of excess relays n_ex, and the noise variance σ_N^2, which have been chosen identically for all layers. Figure 4 shows the error probability of the XOR operation as a function of the SNR for n_ex equal to 0 (shown in blue), 5 (green), and 20 (red) relays. The dashed lines represent relay gain allocation using the closed form minimum norm approach in (9), whereas the triangle and circle markers represent gradient schemes A and B, respectively. It can be seen that the error probability of the minimum norm scheme improves with the number of excess relays. For $n_\mathrm{ex} = 0$, the performance of gradient scheme A is equivalent to the closed form allocation, but as the number of excess relays is increased, it suffers from converging to suboptimal solutions. Gradient scheme B, however, does not converge to a sensible solution for the minimum relay configuration, but closely matches the performance of the minimum norm approach if the number of excess relays is increased.

We furthermore investigate the convergence behavior of both gradient schemes. Depicted in Fig. 5 is the empirical CDF of the number of iterations the schemes perform before their solution meets the defined termination criteria and terminates. The maximum number of permissible iterations, after which the schemes are forced to terminate, was set to 5000 for this analysis. Clearly, gradient scheme B (dashed curves) takes comparatively long to converge, and while it gains from an increased number of excess relays in terms of error probability, increasing n_ex negatively impacts its convergence speed. On the other hand, gradient scheme A (solid curves) is not only faster in convergence, but even gains from an increased number of relays. Both schemes therefore have their benefits and drawbacks, and a choice between them would depend on the requirements of the implemented system.

IV. Robustness of Computation

ANNs are known to be robust with respect to errors in the input data. In this section, we will investigate the sensitivity of wireless ANNs to additional imperfections. Low-complexity wireless ANNs exhibit error sources beyond receiver noise that are typically not encountered in their wired counterparts. One such issue is the fact that the coherent relaying approach used in this work assumes perfect phase synchronization of the relays in each stage. Especially considering the low-complexity setting, the performance of a real network may suffer from synchronization inaccuracies. To investigate the impact of this, we model the imperfect synchronization as random phase offset for each relay gain, drawn from a zero mean Gaussian distribution with variance σ_P^2.

Figure 6 shows the error probability of the previously introduced XOR network as a function of the phase noise variance σ_P^2. Here, the relay weights have been chosen using the closed form solution, and the thermal noise variance in the network has been set to a constant value of $\sigma_\mathrm{N}^2 = 10^{-3}$. Clearly the computation performance degenerates badly with phase noise present in the system, especially for the minimum

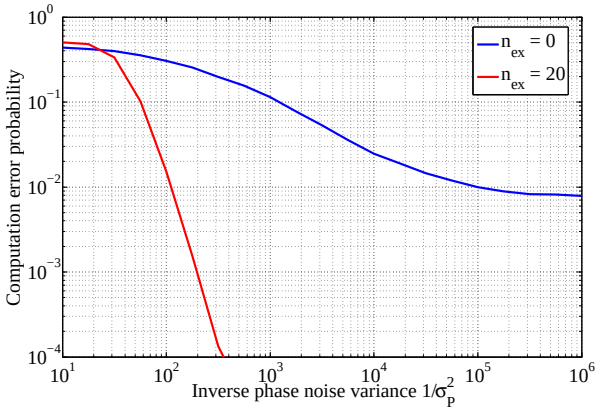

Fig. 6. Error probability of XOR wireless ANN versus phase noise at relays. Relay weights initialized with closed form solution.

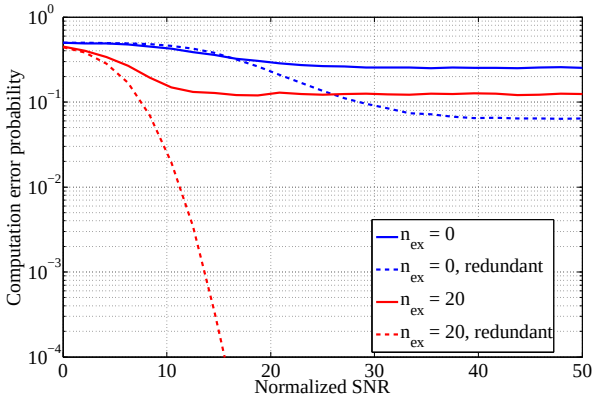

Fig. 7. Error probability of reference wireless ANN versus SNR with single random node failure. Relay weights initialized with closed form solution.

relay configuration shown in blue. However, increasing the number of excess relays can improve the error probability enough to still achieve acceptable error probabilities even for considerable phase noise variance.

Finally, sensor networks are known to be prone to node failures due to the low level of maintenance and low complexity of the devices. It is therefore thinkable that one of the nodes comprising a wireless ANN may cease to function. In case of a relay failure, the network may restore its computational functionality by reacquiring an updated choice of gains calculated over the remaining subset of relays, given the remaining number of relays still satisfies the minimum relay configuration. If on the other hand a neuron fails, the network will also need to define a new set of weight matrices, e.g. by learning.

However, the parallel structure of ANNs suggests that computation might be robust to random node failures even without reconfiguring the network, and that this robustness can be increased by adding additional neurons and relays to perform the computation, thus adding redundancy. We therefore compare the reference implementation of the XOR network to a redundant version, consisting of three parallel reference XOR networks and a final stage of majority decision. For this simulation, one node is randomly selected in both networks. The failure of this node is modeled by setting $g_k^{(l)} \equiv 0$ in case of the selected node being a relay, and $x_i^{(l)} \equiv 0$ if the node is a neuron. Figure 7 shows the average error probability of both networks after a random failure vs. normalized SNR. If there is no margin for loss of relays, both the reference and the redundant network are strongly affected by a random node failure. However, if sufficiently many excess relays are present and the network structure is redundant, the network preserves its computational ability even if it is not reconfigured immediately. If this condition can be fulfilled, wireless ANNs may therefore provide a reliable means of computation even in an unreliable environment.

V. Conclusion

In this work, we have presented an approach to implement artificial neural networks with wireless nodes, based on the use of multihop coherent relaying. In contrast to previous work on wireless ANNs, this method allows for a resource efficient, low-complexity analog implementation of the neural network. We show that the choice of relay gains necessary to implement the ANN functionality can be carried out in a decentralized fashion with a limited feedback being required at the relays, which can be beneficial in large networks with many relays. Together with the fact that a large number of relays proves to enhance both the computational error performance as well as the robustness of the network to imperfections, the formation of ANNs is a promising computation paradigm for wireless sensor networks.

References

[1] G. Werner-Allen *et al.*, "Monitoring volcanic eruptions with a wireless sensor network," in *Wireless Sensor Networks, 2005. Proceeedings of the Second European Workshop on*, 2005, pp. 108 – 120.

[2] I. F. Akyildiz and J. M. Jornet, "Electromagnetic wireless nanosensor networks," *Nano Communication Networks*, vol. 1, no. 1, pp. 3–19, 2010.

[3] A. Kulakov, D. Davcev, and G. Trajkovski, "Application of wavelet neural-networks in wireless sensor networks," in *SNPD/SAWN 2005.*, 2005, pp. 262 – 267.

[4] F. Oldewurtel and P. Mahonen, "Neural wireless sensor networks," in *Systems and Networks Communications, 2006. ICSNC'06. International Conference on*. IEEE, 2006, pp. 28–28.

[5] M. Eshaghian-Wilner, A. Friesz, A. Khitun, S. Navab, A. Parker, K. Wang, and C. Zhou, "Emulation of neural networks on a nanoscale architecture," in *Journal of Physics: Conference Series*, vol. 61. IOP Publishing, 2007, p. 288.

[6] A. Wittneben and B. Rankov, "Distributed antenna systems and linear relaying for gigabit MIMO wireless," in *VTC-Fall 2004, IEEE*, vol. 5. IEEE, 2004, pp. 3624–3630.

[7] H. Bolcskei, R. Nabar, O. Oyman, and A. Paulraj, "Capacity scaling laws in mimo relay networks," *Wireless Communications, IEEE Transactions on*, vol. 5, no. 6, pp. 1433–1444, 2006.

[8] W. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Math. Biology*, vol. 5, pp. 115–133, 1943.

[9] D. Rumelhart, G. Hintont, and R. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[10] R. Rolny, J. Wagner, C. Esli, and A. Wittneben, "Distributed gain matrix optimization in non-regenerative MIMO relay networks," in *Signals, Systems and Computers (ASILOMAR), 2009 Forty Third Asilomar Conference on*. IEEE, 2009, pp. 1503–1507.