UNIVERSITY OF MAURITIUS RESEARCH JOURNAL – Volume 15 – 2009
University of Mauritius, Réduit, Mauritius

Research Week 2007

A Neural Network Model for Dynamics Simulation

Ajeevsing Bholoa *
Mauritius
Email: ubholoa@yahoo.com

Paper Accepted on 01 June 2009

Abstract

In this work we discussed the representation ability, the model building process and the applicability of a non-standard feedforward neural network (FNN) approach. Traditional neural network models have a static mapping capability since they are composed of a fixed number of input nodes. However, in a dynamic environment, it is desirable to have a non-standard network that uses a variable set of input data. This approach enables already presented knowledge to adapt to new situations and conditions of the environment and new knowledge to be integrated into the fitting database. We applied successfully the approach to a system of linear chain of silicon (Si) atoms. In this study, a back-propagation algorithm was employed to train a feedforward neural network. The Levenberg-Marquardt (LM) technique was chosen from the various back-propagation training algorithms available for use in this study.

Keywords: Feedforward neural network, Back-propagation algorithm, Levenberg-Marquardt technique

**For correspondences and reprints*

1. INTRODUCTION

At the current stage of knowledge, neural network models are crude approximations of the human brain such that a very important feature of neural networks is their adaptive nature, where “learning by example” replaces “programming” in solving problems[1]. Neural networks must first be trained on samples of the data so that it can learn to recognise patterns in the data. Once trained it can generalise, *i.e.,* make predictions by detecting patterns in future data. This feature makes these computational models very appealing in application domains where one has little or incomplete understanding of the problem to be solved but where training data is readily available.

One such application domain is developing an accurate and efficient method that can be used systematically to fit empirical potential energy surfaces of systems with many degrees of freedom[2]. These potentials are simplified mathematical expressions that attempt to model interatomic forces arising from quantum mechanical interactions of electrons and nuclei. Empirical potentials are generally developed by first choosing an appropriate analytic functional form containing adjustable parameters. The parameters are then optimized, fitting the potential to experimentally determined structures or, to the results of quantum mechanical calculations. The success of the fitting procedure is determined to a large extent by the derived functional form for the potential. At present, there is no definitive functional form that describes all types of multi-atom bonding[3]. In the case of Si, more than 30 empirical potentials have been reported[4].

On the other hand, no assumptions about the functional form of the potential are required to achieve a good representation using neural networks. The neural network models can extract the underlying relationships between the input and output variables directly from the data being modelled. Several efforts to use neural networks to describe potential functions have been reported, in most cases with good results[5-18]. The resulting function is infinitely differentiable and globally defined. The fit is not only merely local, like a spline, but can also reproduce global features, such as symmetry[2]. Thus, neural networks are legitimate candidates substantiated by the fact that FNN's are known to possess the universal approximation property[19].

The basic FNN performs a non-linear transformation of input data in order to approximate output data. This results in a static network structure. In some situations, such as a dynamic environment (*e.g.*, a molecular dynamics (MD) simulation whereby an atom constantly changes its local environment and number of neighbours), knowledge acquisition remains incomplete. Already represented knowledge has to be adapted to new situations and conditions of the environment and new knowledge has to be integrated into the fitting database. At this point, the principle unresolved issue is how to present the information regarding the variability of the local environment of an atom to the input layer of a FNN.

The aim of this paper is to demonstrate the use of a non-standard (dynamic mapping) FNN to deal with the issue of the variability of the atomic local environment in some simple linear chain models of Si of two and three atoms. The insight available from these simple models may be a helpful guide to understanding the limitations of neural networks and improving their performance. The ultimate goal of the project is to systematise the development of accurate and transferable potential energy surfaces using neural networks.

The paper is organized as follows. Section 2 describes the structure of a traditional FNN. In section 3, the non-standard FNN is discussed. The fitting process of the non-standard network using data derived from Si linear chains is examined in section 4. Finally, the results of the training are presented and analysed in section 5.

2. BASIC FEEDFORWARD NEURAL NETWORK

The architecture of a FNN is defined by a directed, acyclic graph and a choice of neuron activation functions[20]. The FNN can be arranged into (at least) three layers: one input layer, one (or more) hidden layer(s) and one output layer. The interconnections within the network are such that every neuron in each layer is connected to every neuron in the adjacent layers. Each interconnection has associated with it a scalar weight which is adjusted during the training phase. The hidden and output layer nodes are computational nodes typically having biases and sigmoid activation functions φ given by equation (1).

$$\varphi(x) = \frac{1}{1+e^{-x}}, x \in \mathbb{R} \tag{1}$$

Figure 1 illustrates a three-hidden-layered FNN with P input nodes, Q, R and S nodes in the first, second and third hidden layers respectively, and T output nodes where $P, Q, R, S, T \in \mathbb{N}$.

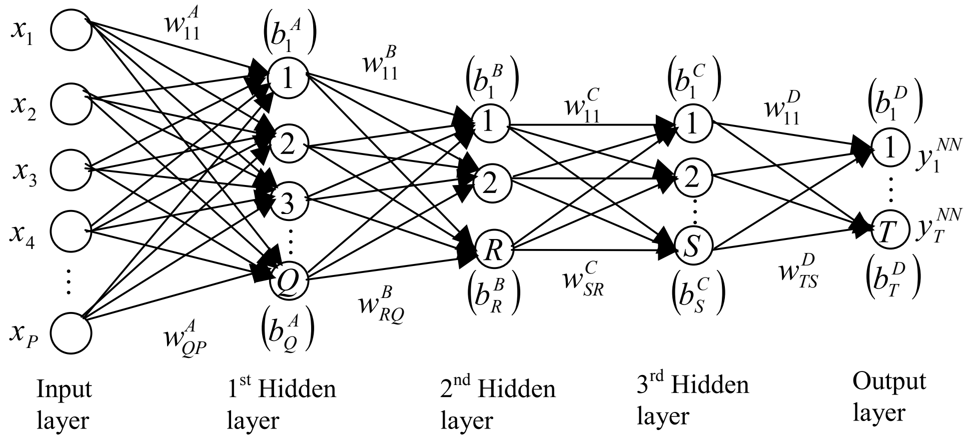


Figure 1: A feedforward neural network model consisting of P inputs, Q, R and S nodes in the first, second and third hidden layers respectively, and T output nodes. The x's are the input variables to the network. The biases and the weights are indicated by the b's and w's respectively. The output of the neurons are denoted by the y^{NN}'s.

The number of input and output nodes is determined by the nature of the modelling problem being tackled, the input data representation and the form of the network output required. Thus, the network in figure 1 has P input variables $x_1, x_2, ..., x_P$ and T output variables $y_1^{NN}, y_2^{NN}, ..., y_T^{NN}$. Such a network can be interpreted as a composition of function from $\mathbb{R}^P$ to $\mathbb{R}^T$ given by equation (2).

$$y_t^{NN} = \varphi\left(\sum_{s=1}^{S} \varphi\left(\sum_{r=1}^{R} \varphi\left(\sum_{q=1}^{Q} \varphi\left(\sum_{p=1}^{P} x_p w_{qp}^A + b_q^A\right) w_{rq}^B + b_r^B\right) w_{sr}^C + b_s^C\right) w_{ts}^D + b_t^D\right) \quad (2)$$

for $t = 1,2,...,T$ whereby $w_{\alpha\beta}^{\gamma}$ represents the γ^{th} level of weights between nodes α and β and b_{α}^{γ} represents the γ^{th} level of biases associated with node α.

The number of hidden layer nodes is related to the complexity of the system being modelled. A number of papers[21,22] have shown that a two-hidden-layered FNN has the ability to approximate any non-linear continuous function to an arbitrary degree of exactness, provided that the hidden layer contains sufficient nodes. More than two hidden layers have proved to be useful in certain applications[18,23-25] as well.

The learning in a FNN is done in a supervised manner which assumes the availability of a set of training data made up of N_{Train} input-output examples $\{(\mathbf{x}_l, \mathbf{d}_l)\}_{l=1}^{N_{Train}}$ where $\mathbf{x}_l$ is the input vector of the *l*th example and $\mathbf{d}_l$ is the desired (target) response of the *l*th example. Given the training sample, the network parameters (weights and biases) are adjusted in such a way that the difference between the actual output of the neural network $\mathbf{y}_l^{\mathbf{NN}}$ due to $\mathbf{x}_l$ and the desired output $\mathbf{d}_l$ is minimised for all *l*. Usually this is done by minimising the sum-squared error (*SSE*) function (equation (3)).

$$SSE = \frac{1}{2}\sum_{t=1}^{T}\sum_{l=1}^{N_{Train}} (d_{l,t} - y_{l,t}^{NN})^2 \quad (3)$$

The problem of determining the network parameters is essentially a non-linear optimisation task. The back-propagation method[26,27], which is a distributed gradient descent technique, is the most popular training algorithm. However, a major limitation of the algorithm is that it does not always converge and can be excruciatingly slow, particularly when dealing with a difficult learning task that requires the use of a large network[28]. More efficient methods include the Conjugate Gradient[29], Quasi-Newton[30] and the LM[29,31] algorithms. In particular, the LM algorithm was designed specifically for minimising *SSE* with a single output variable (*i.e.,* $T = 1$)[32].

In general, to reach the best generalisation, the dataset should be split into three parts: a training set is used to train a neural net and the error of this dataset is minimised during training; a validation set is used to determine the performance of the neural network on patterns that are not trained during learning; and a test set for finally checking the overall performance of the neural net. The learning should be stopped at the minimum of the validation set error.

3. FEEDFORWARD NEURAL NETWORK FOR DYNAMICS SIMULATION

Reconciling the architecture of a FNN with the requirements of an interatomic potential energy function for use in a MD simulation is non-trivial as the local environment (namely the geometry and the number of neighbours) of an atom will be constantly varying throughout the course of the simulation. In addition, the network must be able to provide a continuous mapping as atoms move in and out of the neighbour distance. The network potential must also be well behaved in the sense that spurious potential minima do not occur and the model must be invariant to the ordering of the input vectors for a given atom.

The architecture of the neural network adopted by Hobday *et al.*[14] provides a useful means to deal with the issue of the variability of the atomic local environment. However, they did not try to systematise the approach to produce a general potential energy surface. In that respect, the neural network model in this work is a significant improvement over the work of Hobday *et al.* and represents a reasonable compromise between the predicting capabilities of the network within the fitting database and its transferability outside the fitting database.

The architecture of the non-standard neural network in this work is shown in figure 2. This dynamic mapping model has a variable set of input nodes in the input layer that represents the variable set $\left\{\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}, \ldots, \mathbf{x}_i^{(N)}\right\}$ of N input vectors for each atom i that could be formed in a MD simulation. Each set of input nodes corresponding to the vector $\mathbf{x}_i^{(n)}\left(n = 1,2,\ldots,N\right)$ has a one-to-one correspondence with a set of neurons in the first hidden layer containing a fixed number, Q, of neurons. To meet the requirement of the potential to be invariant to any ordering of input data and to provide a continuous mapping, the following conditions are applied to the model:

(1) The first level A of weights and biases between the corresponding input nodes and the first hidden nodes are identical. Thus, we have

$$w_{qp}^{A(1)} = w_{qp}^{A(2)} = \ldots = w_{qp}^{A(N)} = w_{qp}^{A} \qquad (4)$$

for $p = 1,2,\ldots,P$ and $q = 1,2,\ldots,Q$.

Similarly, for the biases, we have

$$b_q^{A(1)} = b_q^{A(2)} = \ldots = b_q^{A(N)} = b_q^{A} \qquad (5)$$

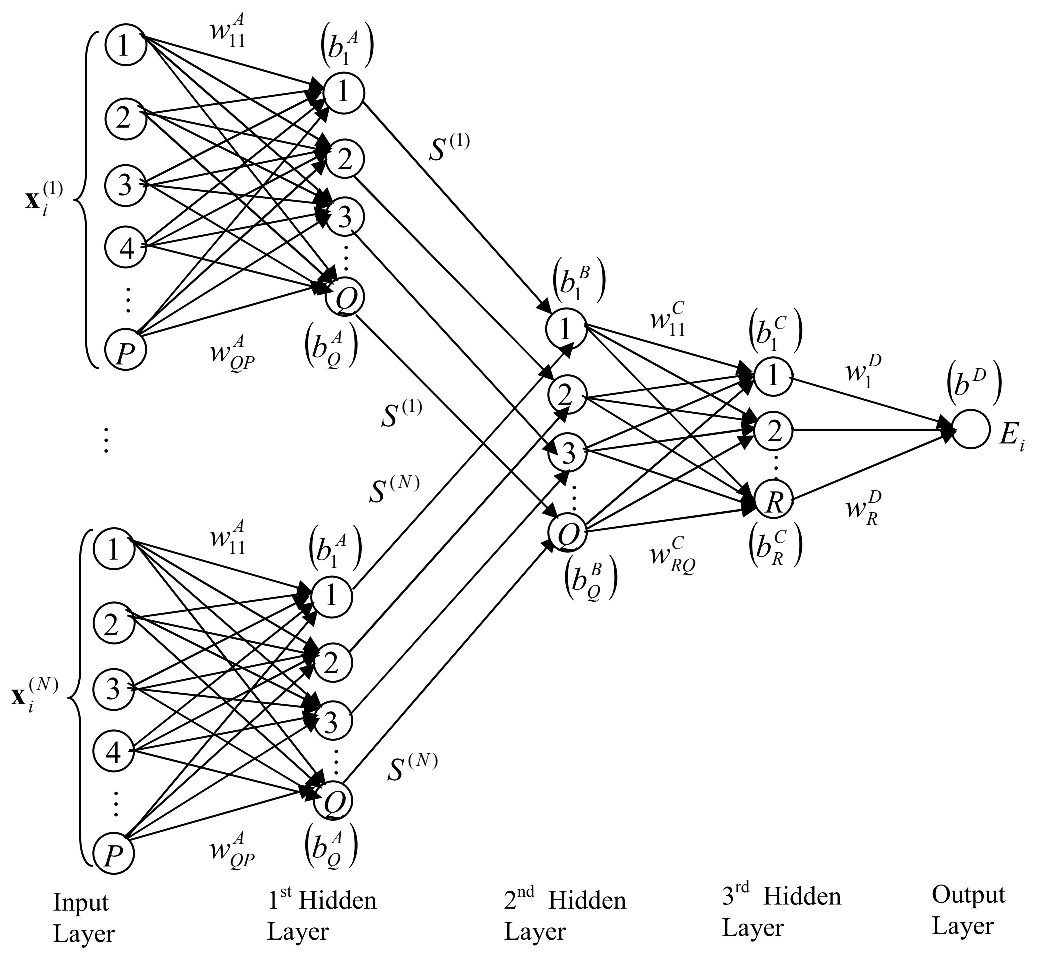


Figure 2: The neural network model for dynamics simulation. The $\mathbf{x}_i$'s are the input vectors due to atom i; the weights are denoted by the w 's; E_i represents the energy of atom i. Each neuron has a bias represented by the b 's. The connections between the first and the second hidden layers are measured by the screening factors $S^{(n)}$.

(2)A feedforward connection $(\rightarrow)$between a neuron, $neuron_{q_1}^{(n)}$, in the n^{th} set of the first hidden layer and a neuron, $neuron_{q_2}$, in the second hidden layer is made only when the following condition is met:

$$neuron_{q_1}^{(n)} \rightarrow neuron_{q_2} \Leftrightarrow q_1 = q_2 \tag{6}$$

where $q_1, q_2 = 1,2,...,Q$.

(3) The strength of the connections is equal for the corresponding neurons in the first and second hidden layers which satisfy the relation (6). The strength of the connections is measured in terms of a screening factor $S^{(n)} = S_{ij}^{(n)}$, which measures the extent of screening in the bond $i - j$ between atom i and atom j due to neighbouring atoms. The full details of the screening procedure[18] is based on the idea put forward by Baskes[33]. Hence we have

$$S_1^{(n)} = S_2^{(n)} = ... = S_Q^{(n)} = S^{(n)} \text{ for } n = 1,2,...,N \qquad (7)$$

From the second hidden layer to the output E_i, the energy per atom i, the neural network is connected in a traditional feedforward manner. Therefore, for connections between the output and the second hidden layer, the application of the back-propagation algorithm is identical to that of a standard network. For the adjustment of weights and biases between the first hidden layer and the input layer during the training process, the back-propagation is slightly modified by the special form of the input layer[18].

4. FITTING PROCESS

To illustrate the applicability of the network, simple Si systems of dimers (figure 3(a)) and linear trimers (figure 3(b)) are considered.

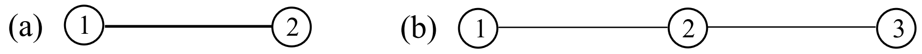


Figure 3: Si (a) dimer and (b) linear trimer considered in the process to fit the non-standard neural network.

We used the neural network model to predict the energy E_i, for each atom $i \; (i = 1,2,3)$, such that

$$E_i(\mathbf{R}) = Y(\mathbf{X}_i(\mathbf{R}), \mathbf{w}) \qquad (8)$$

where $\mathbf{R}$ is the set of atomic Cartesian coordinates of all the atoms in a given system, $\mathbf{X}_i(\mathbf{R}) = \{\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}, ..., \mathbf{x}_i^{(N)}\}$ is the set of input variables associated with atom i and is derived from $\mathbf{R}$, $\mathbf{w}$ is the vector of the network weights and biases, and Y is the composition of activation functions representing the model. The response E_i can be expressed as follows:

$$E_i = \varphi\left(\sum_{r=1}^{R} w_r^D \varphi\left(\sum_{q=1}^{Q} w_{rq}^C \varphi\left(\sum_{n=1}^{N} S^{(n)} \varphi\left(\sum_{p=1}^{P} w_{qp}^A x_p^{(n)} + b_q^A\right) + b_q^B\right) + b_r^C\right) + b^D\right) \qquad (9)$$

where $x_p^{(n)} \; (p = 1,2,...,P)$ are elements of the input vector $\mathbf{x}_i^{(n)}$. The number of parameters, $N_{parameter}$, of the network is given by equation (10).

$$N_{parameter} = Q(P+2) + R(Q+2) + 1 \qquad (10)$$

We have 6 input variables (*i.e., P* = 6), based on the geometry, to describe the structural properties of the systems. These variables are listed in table 1. Thus, for the neural network model shown in figure 2, the input vector $\mathbf{x}_i^{(n)}$, is defined as

$$\mathbf{x}_i^{(n)} = \left(r_{ij}^{(n)}, r_{jk}^{(n)}, \cos\theta_{ijk}^{(n)}, N_i, N_j, N_{inputs}\right)' \qquad (11)$$

where $i \neq j \neq k$. The variables N_i, N_j and N_{inputs} are given in terms of the screening factors. For the simple linear chain systems, $S_{ij} = S_{jk} = 1$. To model the absent bond $j-k$ in the dimer, an imaginary interaction at a distance r_{max} was used and the associated missing bond angle was taken to be 180°. The input vectors per atom for the dimer (figure 3(a)) and linear trimer (figure 3(b)) generated using equation (11) are given in table 2 and table 3 respectively. In particular, atom $i = 2$ of the linear trimer has $N = 2$ input vectors.

The training data for the neural network potential was generated using the Frauenheim non-orthogonal tight binding (FTB) method[34] from PLATO[35]. Furthermore,

Variable	**Description**
r_{ij}	length of bond $i-j$
r_{jk}	length of bond $j-k$
$\cos\theta_{ijk}$	cosine of angle between bonds $i-j$ and $j-k$
$N_i = \sum_{j\neq i} S_{ij}$	sum of screening present over every bond $i-j$ formed per atom i
$N_j = \sum_{\substack{k\neq i\\k\neq j}} S_{jk}$	sum of screening present over every bond $j-k$ formed per atom j
$N_{inputs} = \sum_{j\neq i} S_{ij}(1+\sum_{\substack{k\neq i\\k\neq j}} S_{jk})$	sum of screening present over every chain $i-j-k$

Table 1: List of input variables used in the neural network training of the dimer and linear trimer systems.

i	N	$\mathbf{x}_i^{(n)}$
1	1	$\mathbf{x}_1^{(1)} = \left(r_{12}, r_{max}, \cos 180^\circ, 1, 0, 1\right)'$
2	1	$\mathbf{x}_2^{(1)} = \left(r_{21}, r_{max}, \cos 180^\circ, 1, 0, 1\right)'$

Table 2: The input vectors for the atoms in the dimer structure.

i	N	$\mathbf{x}_i^{(n)}$
1	1	$\mathbf{x}_1^{(1)} = \left(r_{12}, r_{23}, \cos 180^\circ, 1,1,1\right)'$
2	2	$\mathbf{x}_2^{(1)} = \left(r_{21}, r_{max}, \cos 180^\circ, 1,0,2\right)'$ $\mathbf{x}_2^{(2)} = \left(r_{23}, r_{max}, \cos 180^\circ, 1,0,2\right)'$
3	1	$\mathbf{x}_3^{(1)} = \left(r_{31}, r_{32}, \cos 180^\circ, 1,1,1\right)'$

Table 3: The input vectors for the atoms in the linear trimer structure.

if two or more atoms had the same local geometry, then the information for only one of those atoms was considered. Calculations for E_i (in eV/atom) for dimmers and linear trimers were cumulatively performed and used for training purposes. We found that E_i for 2400 dimers and 2600 linear trimers were the least amount of data required to accurately train the neural network potential.

Thus we had 2400 distinct data points for E_i for the dimers and 2600 distinct data points were randomly extracted from the linear trimer data set. The training data set contained 4550 randomly drawn data from the 5000 dimers and linear trimer data points. The validation set was then similarly generated but with 100 data points from the remaining 450 data points. The remaining 350 data points form the pre-defined test data set. The variables in the data sets were linearly transformed in the interval [0.1,0.9] using their respective minimum and maximum values recorded. The constant parameters $\cos\theta_{ijk}$ and r_{max} were pre-processed to 0.1 and 0.9 respectively.

To find the optimal parameters of the network the cost function SSE (equation (3)) was minimised (with $T = 1$, since E_i was the only output) and the parameters were updated during learning by the LM based back-propagation algorithm. The LM parameter μ was initially taken to be 0.1 while the fixed LM constants μ^- and μ^+ were taken to 0.5 and 2.0 respectively. The values of Q and R were determined by pruning the least effective hidden nodes. In this study, Q and R were varied until a network configuration was obtained that resulted in the least increase of the SSE in the network.

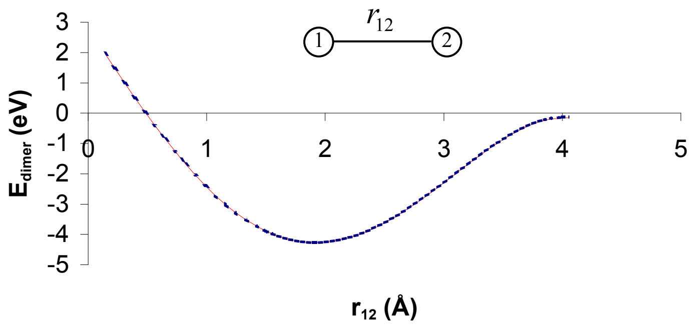
5. RESULTS AND DISCUSSIONS

15 training processes were carried out and the final network chosen corresponded to the lowest mean sum-squared errors (MSE) recorded on the test data set. The relevant statistics of the final neural network model developed is given in table 4.

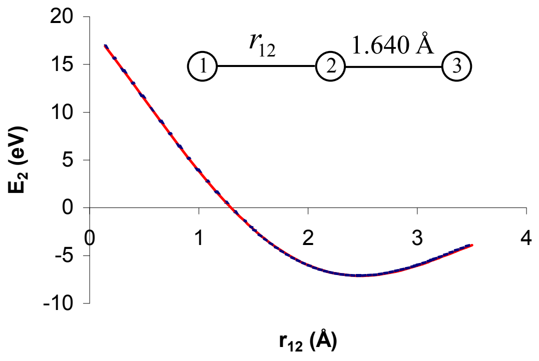
Number of epochs	322
MSE of the transformed training set	1.559×10^{-5}
MSE of the transformed test set	2.053×10^{-5}
Number of hidden nodes, Q	11
Number of hidden nodes, R	11

Table 4: Properties of the neural network model which has best fitted E_i for the system of dimers and linear trimers.

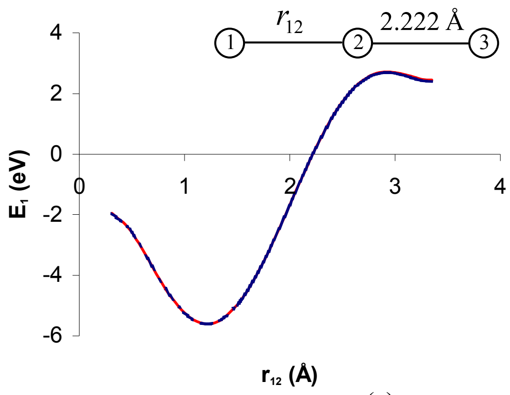
Figure 4 shows the network predictions (the broken curves) of E_i for the selected dimers (figure 4(a)) and linear trimers (figure 4(b)-(e)). The predictions were excellent as, in each case, the root mean sum-of-squares error (RMSE) were very small (< 0.05 eV) while the R^2 value, the coefficient of variation, were > 0.99.



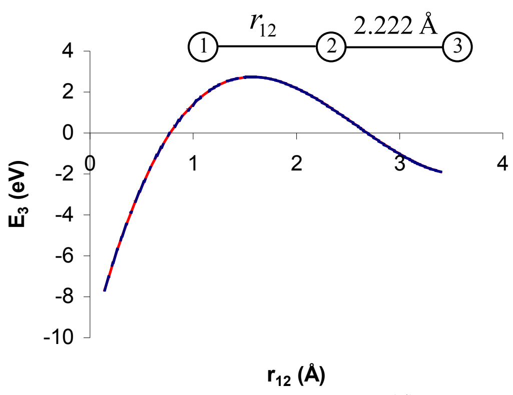
(a)



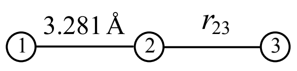
(b)



(c)



(d)



(e)

Figure 4: Predictions of E_i for the system of dimers and linear trimers. The continuous curves represent the target values from the FTB method while the dashed curves represent the neural network predictions.

6. CONCLUSIONS

The approach outlined in this work is novel in terms of the form of the neural network used. The network employed was non-standard as it used a variable set of input data that mapped the environment of the atoms. The method accurately predicted the trends in the energy per atom of the Si dimers and linear trimers. The insight available from these small systems indicates a robust and consistent methodology for fitting empirical potentials which can be applied to a wide range of systems including both small clusters and bulk structures.

7. ACKNOWLEDGEMENT

I would like to thank Dr A. K. Maulloo for his support.

8. REFERENCES

1. M. T. HASSOUN, *Fundamentals of Artificial Neural Networks*, MIT Press (1995).

2. J. B. WITKOSKIE AND D. J. DOREN, *J. Chem. Theory Comput.* **1**, 14-23 (2005).

3. R. SMITH, *Atomic and Ion Collisions in Solids and at Surfaces: Theory, Simulation and Applications*, Cambridge University Press (1997).

4. M. Z. BAZANT, E. KAXIRAS AND J. F. JUSTO, *Phys. Rev.* B **56**, 8542 (1997).

5. T. B. BLANK, S. D. BROWN, A. W. CALHOUN AND D. J. DOREN, *J. Chem. Phys.* **103**, 4129 (1995).

6. A. J. SKINNER AND J. Q. BROUGHTON, Modell. Simul. Mater. Sci. Eng. **3**, 371 (1995).

7. D. F. R. BROWN, M. N. GIBBS AND D. C. CLARY, *J. Chem. Phys*. **105**, 7597 (1996).

8. K. T. NO, B. H. CHANG, S. Y. KIM, M. S. JHON AND H. A. SCHERAGA, *Chem. Phys. Lett*. **271**, 153 (1997).

9. F. V. PRUDENTE AND J. J. SOARES NETO, *Chem. Phys. Lett*. **287**, 585 (1998).

10. F. V. PRUDENTE, P. H. ACIOLI AND J. J. SOARES NETO, *J. Chem. Phys*. **109**, 8801 (1998).

11. H. GASSNER, M. PROBST, A. LAUENSTEIN AND K. HERMANSSON, *J. Phys. Chem.* A **102**, 4596 (1998).

12. S. LORENZ, A. GROSS AND M. SCHEFFLER, *Chem. Phys. Lett.* **395**, 210 (2004).

13. C. MUÑOZ AND A. NIÑO, *Comput. Chem*. **22**, 355 (1998).

14. S. HOBDAY, R. SMITH AND J. BELBRUNO, Modell. Simul. Mater. Sci. Eng. **7**, 397 (1999).

15. K. W. CHO, K. T. NO AND H. A. SCHERAGA, J. MOL. Struct. **641**, 77 (2002).

16. T. M. ROCHA FILHO, Z. T. OLIVERA JR., L. A. C. MALBOUISSON, R. GARGANO AND J. J. SOARES NETO, *Int. J. Quantum Chem*. **95**, 281 (2003).

17. A. C. P. BITTENCOURT, F. V. PRUDENTE AND J. D. M. VIANNA, *Chem. Phys.* **297**, 153 (2004).

18. A. BHOLOA, *Potential Energy Surfaces Using Neural Networks*, PhD Thesis, Loughborough University (2006).

19. K. HORNIK, M. STINCHCOMBE AND H. WHITE, *Neural Networks Vol. 2*, Issue **5**, 359 (1989).

20. T. L. FINE, *Feedforward Neural Network Methodology*, Springer (1999).

21. E. HARTMAN AND J. D. KEELER, *Neural Comput. 3*, 566 (1991).

22. L. LÖNNBLAD, C. PETERSON AND T. RÖGNVALDSSON, *Phys. Lett. **B*** **278**, 181 (1992).

23. S. E. FAHLMAN AND C. LEBIERE, NIPS **2**, 524-532 (1990).

24. K. J. LANG AND M. J. WITBROCK, *Learning to tell two spirals apart*, in D. Touretzky, G. Hinton and T. Sejnowski, eds., Proceedings of the 1988 Connectionist Models Summer School, San Mateo (1988).

25. Y. LE CUN, B. BOSER, J.S. DENKER, D. HENDERSON, R. E. HOWARD, W. HUBBARD AND L. D. JACKEL, *Neural Comput.* ***1***, 541-551 (1989).

26. D. E. RUMELHART, G. E. HINTON, AND R. J. WILLIAMS, *Learning internal representations by Back-propagating errors*, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. **1**, ed. D. E. Rumelhart and J. L. McClelland, MIT Press, Cambridge, MA (1986).

27. P. J. WERBOS, *The roots of Backpropagation*, Wiley, New York (1994).

28. S. Haykin, *Neural Networks: A comprehensive Foundation,* MacMillan Publishing Company (1994).

29. M. T. HAGAN, H. B. DEMUTH AND M. H. BEALE, *Neural Network Design*, Boston, MA: PWS Publishing (1996).

30. J. E. DENNIS AND R. B. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Englewood Cliffs, NJ: Prentice-Hall (1983).

31. M. T. HAGAN AND M. MENHAJ, IEEE Transactions on Neural Networks, Vol. **5**, No. **6**, 1999, 989-993 (1994).

32. C. M. BISHOP, *Neural Networks for pattern recognition*, Oxford University Press (1995).

33. M. I. BASKES, Mater. Sci. Eng. A **261**, 165 (1999).

34. TH. FRAUENHEIM, F. WEICH, TH. KÖHLER AND S. UHLMANN, Phys. Rev. B **52**, 11492 (1995).

35. S. D. KENNY, A. P. HORSFIELD AND H. FUJITANI, Phys. Rev. B **62**, 4899 (2000).