Prediction of Pile Driving Resistance using a Self-evolving Neural Network (SEANN)

ABDUSSAMAD ISMAIL and DONG-SHENG JENG
Division of Civil Engineering
University of Dundee
DD1 4HN, Dundee
UNITED KINGDOM

Abstract: -A self evolving neural network (SEANN) is used to develop an empirical model to predict the soil resistance to pile driving, considering the variation in the type of soil encountered, the pile type and hammer characteristics. A pile driving record is used to train and validate the network model. Results obtained showed that the proposed model reasonably captures the behavior of a pile subject to hammer driving. The network predictions are reasonably accurate compared with the field data.

Key-Words: - Pile driving Self-evolution Neural networks

1 Introduction

Piles serve as foundations to support important superstructures such as tall buildings, bridges, offshore oil and gas infrastructures and wind turbines. Piles are constructed either by drilling a hole to create a space for installing the pile when the supporting soil is relatively stiff, or by driving the pile into the ground under the impact of hammer blows in the case of relatively soft ground. The depth of penetration in the case of the former is quite predetermined prior to pile placement, while in the case of the latter, the depth of penetration is difficult to estimate with certainty due to the dependence of pile driveability on the operating conditions during installation. Although the some information about factors dictating the maximum depth of pile embedment (refusal depth) such as the hammer features, pile properties and sub-surface behaviour are can be obtained, the complex interaction between the variables is difficult to predict. For decades, the wave equation pile driving analysis implemented numerically by smith [1] is the most popular technique used in assessing the pile driveability for a given subsurface condition, pile and hammer. However, due to the simplistic assumption of soil properties and some difficulty in accurately accounting for the hammer effect, a huge disparity between the prediction of wave equation model and the observed response is often obtained.

In this paper, a neural network model is used to develop an empirical model which estimates the driving resistance in terms of number of hammer blows required to drive a pile an inch further into the ground (Blows/m). The parameters required by the model include the pile stiffness and geometry, current penetration depth, hammer characteristics as well as the standard penetration test (SPT) N-values along the depth of penetration. A pile driving data is used to train and test the proposed neural network model. In order to use a minimum possible network size for the modelling, a self-evolving neural network is used, where both network parameters and structure are simultaneously optimized during the training process.

2 ANN modelling

Neural networks owe their ability of exploring complex relationships between variables to the parallel nature of their architecture, which is inspired by the complex information processing infrastructure of biological nervous system. Although initially developed with the intention to study the function of biological brain, it gained a considerable popularity as a powerful tool for pattern recognition and function approximation sequel to the implementation of back-propagation algorithm in neural network optimization by Rumelhart et al. [2]. In the present
work, self-evolving neural network (SEANN) is used to predict the resistance to pile driving at various stages during the installation operation.

2.1 Self-evolving neural networks

One of the key challenges associated with neural network development is how to optimize the network architecture. The complex nature of the topology space makes it quite a difficult task to optimize network architecture and arrive at the best combination of network parameters simultaneously [3]. The classical topology optimization techniques such as Network pruning and Incremental learning algorithm tend to get the network entrapped in the topology space local minima [4]. To enhance the chance of escaping topology local minima, a variety of bio-inspired evolutionary concepts have been employed to simultaneously optimize network topology and parameters. These include the genetic algorithm (GA) based algorithms [4] and techniques based on Particle swarm optimization [5,6]. Although PSO is computationally simpler than GA-based algorithms, the key disadvantage of the self evolution algorithms developed based PSO is the random generation of network topology, which tends to compromise the computational efficiency of the optimization process.

In this paper, another variant of PSO based self-evolution algorithm is used for optimizing ANN model. Unlike other PSO based approaches, this method focuses on minimizing network size by adopting an incremental learning procedure, in which the network size is increased gradually, starting with a small network with single hidden node. The initial step of the process involves generating a population of neural nets with each having a random set of connection and synaptic parameters. The connection parameters are binary, assuming a value of 1 if there is a connection between two nodes and 0 if otherwise (Figure 2). Subsequent steps involve updating the network parameters until sufficient accuracy is achieved. The binary parameters for each network in the population are updated using a jumping particle swarm optimization (JPSO) procedure. JPSO algorithm, developed by Martınez-Garcıa and Moreno-Pe´rez [7] is discrete optimization algorithm which tends perform better in finding the best solution in discrete space compared to discrete particle swarm optimization algorithm (DPSO) proposed by kennedy and Eberhart [8]. In Jumping PSO (JPSO) algorithm, the particle jumps from its current position to a new position under the influence of particle’s experience, global best position and explorative tendency. The particle’s position is stochastically updated as follows:

\[
x_{t+1} = \lambda_1 \otimes x_t + \lambda_2 \otimes b + \lambda_3 \otimes g
\]

where \(x_t\) and \(x_{t+1}\) are the vectors of current and future particle positions in the discrete search space. The parameters \(\lambda_1\), \(\lambda_2\) and \(\lambda_3\) are probabilities of jumping randomly, towards the best particle position and to the best swarm position respectively. \(b\) and \(g\) are, respectively, the particle best and global best positions. The updated position could be worse than the current one, therefore a random local search around the updated position is carried out to find a better solution. The local search is conducted using few steps of back-propagation algorithm due to the mixed nature of variables involved in simultaneous optimization of network topology, activation functions and network parameters. The proposed JPSO algorithm is represented by the flowchart in Figure 3.

![Figure 1: topology of self-evolving network](image)

The synaptic weights of individual networks in the population are updated using a combination of PSO and BP algorithm. For the PSO part, the network parameters are updated using the equations proposed by Clerc and Kennedy [9], which are given as follows:

\[
v_{i,j,t+1} = \chi[v_{i,j,t} + c_1 r_1 (b_i - x_{i,j,t}) + c_2 r_2 (g_i - x_{i,j,t})]
\]

\[
x_{i,j,t+1} = x_{i,j,t} + v_{i,j,t+1}
\]

in which \(v_{i,j,t+1}\) and \(v_{i,j,t}\) are the updated and current velocity vectors of particle I respectively. \(x_{i,j,t+1}\) and \(x_{i,j,t}\) are, respectively, the updated and current co-ordinates of particle I, whereas \(c_1\) and \(c_2\) stand for acceleration constants. \(r_1\) and \(r_2\) are uniformly distributed between 0 and 1.
distributed random numbers from 0 -1, while \( b \) and \( g \) represent the best particle’s position so far and the best swarm experience respectively. \( \chi \) is the constriction factor.

\[
\chi \text{ is the constriction factor.}
\]

Randomly generate \( N \) particles

Evaluate particles’ fitness \( f \) and determine \( b \) and \( g \)

For all particles

\[
\lambda + \tilde{\lambda} < r \leq 1
\]

Generate random value \( r[0,1] \)

\[
x_{i} = x^*g
\]

\[
c_i < r \leq \lambda + \lambda
\]

\[
x_{i} = x^*p[0,1]
\]

\[
x_{i} = x^*b
\]

\[
f(x_{i}) > f(x_{j})\]

Accept \( x_{i} \)

\[
f(x_{i}) < f(x_{j})\]

Reject \( x_{i} \)

Yes

Perform local search

\[
f(x_{i}) > f(x_{j})\]

Accept \( x_{i} \)

\[
f(x_{i}) < f(x_{j})\]

Reject \( x_{i} \)

No

Update \( b \) and \( g \)

No

Stopping criteria satisfied?

Yes

End

Yes

Figure 2: Flowchart describing JPSO algorithm

The advantage of combining the two techniques is to benefit from the global search capability of the PSO and the ability of the BP algorithm to efficiently perform a local search. The algorithm involves initially training the network parameters using PSO for a certain number of iterations, then training some selected (best performing) particles among the swarm population using BP algorithm for few number of iterations. The results of the local search by BP algorithm are then used to update the positions of relevant particles and the PSO takes over again. The cycle is repeated until a sufficiently accurate is obtained.

When no further improvement is observed, the complexity of the network is increased by adding more nodes, one node at a time. To prevent the destruction of the so far acquired knowledge, the previous best particle positions (both topology and synaptic weights) are retained while adding one more node to the members of the swarm population.

In this way, the computational burden of dealing with unnecessarily large networks is avoided as in the case of the algorithms proposed by Kiranyaz et al. [5] and Xian-Lun et al. [6], while at the same time avoiding the risk of getting stuck in the local minima of topology space.

### 2.2 Activation function

The choice of suitable activation function plays a central role in successful development of neural networks. Sigmoid function has been the most widely used model for ANN development due to its stability. However, despite the popularity it enjoys, it doesn’t necessarily yield efficient networks [15,16]. In this research, a combination of several functions is used as a processing function. The processing function used is expressed in the following equation as:

\[
f(x) = \sum_{i=1}^{n} k_i c_i \phi_i(x)
\]

where \( n \) is the number of sub-functions \( \phi \). \( c_i \) is an adaptive coefficient, while \( k_i \) is a binary number; \( x \) is the vector of inputs to the node. The binary number assumes a value of 1 if the associated functions are switched on and 0 if the functions are excluded. The adaptive coefficients are updated are trained alongside the synaptic weights, while the binary parameters are updated together with connection parameters using JPSO algorithm. In this manner, the topology, the synaptic weights and the activation functions are simultaneously optimized. The sub-functions considered in this paper include Linear, wavelet, sinusoid and sigmoid.

### 3 Pile driving records

The data used to in training and testing the model consists of 122 pile driving records, with each record consisting of number of hammer blows versus penetration, making a total of 2824 data points. The pile data is sourced from FHWA deep foundations load test database. The data is also available online at www.geotech.com. The data consists of cylindrical and square concrete, H-steel piles and closed end steel pipe piles. Various types of hammer used in installing the pipes can be broadly grouped into grouped into open end single acting, closed end double acting and internal combusting hammers. The soil test results in the database consist of uncorrected SPT N-values as...
well as standard soil classification. The soil classification is used to give additional information to help characterize the subsurface more accurately. The soil types in the database range from coarse soils (sand) to fine soils (silt and clay) with varying degree of plasticity.

### 3.1 Input parameters

The resistance to pile penetration is a function of hammer properties such as rated energy, ram and hammer type, pile stiffness and geometry as well as the behaviour of soil along the pile shaft and around the base. In order to provide sufficient information about the component resistance to penetration along the pile shaft, the pile is divided into six segments. Each segment is represented by a neural net as shown in Figure 3. The base component of resistance is also represented by a neural network as shown in the figure. The inputs to the networks representing shaft and base appear on the right-hand side of equations (5) and (6) respectively.

\[
ANN_{s_i} = f\left(A_{s_i}, N_{s_i}, \sigma, Pl, E, H, P_m, k_p, d\right)
\]

\[
ANN_b = f\left(A_b, N_b, \sigma, Pl, E, H, P_m, d\right)
\]

Where \(N_{s_i}\) and \(N_b\) are the average N-value along the shaft segment \(i\), and around the base respectively. \(A_{s_i}\) and \(A_b\) are, respectively, the surface area of shaft segment \(i\) and the area of base. \(\sigma\) is the percentage of fines; \(Pl\) the plasticity parameter (0 for low or no plasticity and 1 for high plasticity); \(E\) the hammer energy and \(H\) the type of hammer. The parameters \(P_m, k_p\) and \(d\) represent the type of pile material, pile cross-sectional stiffness (EA), the current depth respectively. The penetration resistance in terms of number of blows per m is expressed by the following equation:

\[
r_p = \sum_{i=1}^{5} ANN_{s_i} + ANN_b
\]

It should noted that the training data does not contain target outputs for individual shaft and base neural networks, therefore, the networks are simultaneously optimized as part of the objective function described by equation 5. The inputs to the networks along the shaft include the product of the segment area and uncorrected N-value (NA), clay and silt contents, hammer energy, type of hammer (i.e. whether single or double acting), type of pile material (i.e whether concrete pipe, of H-pile) and current depth of penetration. In the case of the network representing the base, the same inputs are used but the area of base is used when multiplying the area with N-value.

### 3.3 Networks Training and validation

The database was partitioned into training and testing (validation) sets. A total of 1845 data sets were used for training, while 979 sets were used for testing. The coefficient of determination (\(R^2\)) was used in assessing the prediction quality in the case of both training and testing.

![Figure 3: representation of shaft and base resistance with neural networks](image)

### 3.3 Performance assessment

The results of ANN predictions are plotted against the training and testing data in the scattergrams shown in Figures 4(a) and 4(b) respectively. As can be seen from the figures, the agreement between the proposed network predictions and both training and testing data is quite reasonable (\(R^2 = 0.8642\) for training and \(R^2 = 0.8432\) for testing).

To further examine the accuracy of the proposed network, two sets of field measurements are compared with the model estimates in figures 5(b) and 6(b) respectively. The pile and hammer properties for the two cases are shown in Table 1. It can be seen from the figures that proposed SEANN model reasonably predicts the driving resistance in both cases.

Table 1: pile and hammer properties
<table>
<thead>
<tr>
<th></th>
<th>Case 1</th>
<th>Case 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>pile type</td>
<td>H-steel pile (HP360 x 132 / HP 14 x 89)</td>
<td>Steel pipe (closed ended)</td>
</tr>
<tr>
<td>perimeter (m)</td>
<td>0.01690</td>
<td>1.01787</td>
</tr>
<tr>
<td>base area (m²)</td>
<td>0.01690</td>
<td>0.08245</td>
</tr>
<tr>
<td>length (m)</td>
<td>35.05</td>
<td>21.48</td>
</tr>
<tr>
<td>Elastic modulus (kN/m²)</td>
<td>2.00E+08</td>
<td>2.00E+08</td>
</tr>
<tr>
<td>Hammer energy (kJ)</td>
<td>69878</td>
<td>81015</td>
</tr>
<tr>
<td>Hammer type</td>
<td>OED</td>
<td>OED</td>
</tr>
</tbody>
</table>

Figure 4(a): Comparison of actual resistance to driving (training data) and self-evolving model predictions.

Figure 5(a): Soil profile (case 1)

Figure 5(b): Driving resistance versus depth (case 1)
4 Conclusion

The ability to predict the maximum depth of pile penetration during driving is quite important to both contractor and design engineer. This paper proposes neural network model to estimate the resistance to pile driving, the reciprocal of which gives an idea about how deep a pile penetrates when subjected to a blow of certain type of hammer. The information required by the model include the type of hammer, the pile type and the sub-surface characteristics of the. The model was developed based on the BP/PSO hybrid parameter optimization algorithm and JPSO based topology optimization technique. Based on its performance, the proposed model was found to agree well with an independent test data which was not included in the training.

References: