International Scholarly Research Network
ISRN Artificial Intelligence
Volume 2012, Article ID 628496, 10 pages
doi:10.5402/2012/628496

Research Article

Prediction of Ultimate Bearing Capacity of Cohesionless Soils Using Soft Computing Techniques

S. Adarsh,[1] R. Dhanya,[2] G. Krishna,[3] R. Merlin,[2] and J. Tina[4]

[1] *Department of Civil Engineering, TKM College of Engineering, Kerala, Kollam 691005, India*
[2] *Department of Civil Engineering, National Institute of Technology, Kerala, Calicut 673601, India*
[3] *Department of Civil Engineering, National Institute of Technology, Karnataka, Surathkal, Mangalore 575025, India*
[4] *Department of Civil Engineering, College of Engineering, Thiruvananthapuram 695016, India*

Correspondence should be addressed to S. Adarsh, adarsh_lce@yahoo.co.in

Received 31 July 2011; Accepted 7 September 2011

Academic Editor: M. Abbod

This study examines the potential of two soft computing techniques, namely, support vector machines (SVMs) and genetic programming (GP), to predict ultimate bearing capacity of cohesionless soils beneath shallow foundations. The width of footing (B), depth of footing (D), the length-to-width ratio (L/B) of footings, density of soil (γ or γ'), angle of internal friction (Φ), and so forth were used as model input parameters to predict ultimate bearing capacity (q_u). The results of present models were compared with those obtained by three theoretical approaches, artificial neural networks (ANNs), and fuzzy inference system (FIS) reported in the literature. The statistical evaluation of results shows that the presently applied paradigms are better than the theoretical approaches and are competing well with the other soft computing techniques. The performance evaluation of GP model results based on multiple error criteria confirms that GP is very efficient in accurate prediction of ultimate bearing capacity cohesionless soils when compared with other models considered in this study.

1. Introduction

Design of foundations is performed based on two criteria: ultimate bearing capacity and limiting settlement. The ultimate bearing capacity is governed by shear strength of the soil and is estimated by theories proposed by Terzaghi [1], Meyerhof [2], Hansen [3], Vesic [4], and others. However, the different bearing capacity formulae shows wide degree of variability while estimating bearing capacity of dense sand on cohesionless soils. Also the bearing capacities are validated through laboratory studies performed on small-scale models. Due to the "scale effect" for the large-scale foundations on dense sand, shearing strain show that considerable variation along the slip line and the average mobilized angle of shearing resistance along the slip line are smaller than the maximum value (Φ_{max}) obtained by plane shear tests [5]. Thus, the use of Φ_{max} may lead to an overestimated bearing capacity value for the calculations based on different formulae [1–4].

In the recent past, the use of soft computing techniques has attracted many researchers and applied quite successfully for solving many complex geotechnical engineering problems. Artificial neural networks (ANNs) may probably be the most popular among these tools, applied for prediction of bearing capacity of cohesionless soils [5], bearing capacity of piles, settlement predictions, liquefaction, and slope stability problems [6]. Support vector machines (SVMs) are recent addition to the soft computing family that uses statistical learning theory as the working principle. SVM and its variants are applied for geotechnical problems such as prediction of pile load capacity [7], settlement of foundations [8], slope stability [9], and liquefaction potential [10].

The evolutionary computational techniques may be a better alternative for solving regression problems as they follow an optimization strategy with progressive improvement towards the global optima. They start with possible trial solutions within a decision space, and the search is guided by genetic operators and the principle of "survival of the fittest" [11]. Genetic Algorithm (GA) is one of the most popular and powerful evolutionary optimization technique [11] explored by [12], but it cannot be used to evolve complex models

such as equations. This limitation is overcome by Genetic Programming (GP) introduced by Koza [13], which works on the principle of GA. GP writes expressions or computer programs instead of strings in GA. In this paper, SVM and GP are used as alternate paradigms to predict bearing capacity of cohesionless soils under shallow foundations.

2. Support Vector Machine

Support vector machine (SVM) is a relatively recent addition to the family of soft computing techniques evolved from the concept of statistical learning theory explored by Boser et al. [14]. SVM performs the regression by using a set of nonlinear functions that are defined in a high-dimensional space. SVM has been used to solve nonlinear regression problems by the principle of structural risk minimization (SRM), where the risk is measured using Vapnik's accuracy intensive loss function (ε) [15]. SVM uses a risk function consisting of the empirical error and a regularization term. More details on SRM can be found in Cortes and Vapnik [16]. Considering a set of input-output pairs as training dataset, $[(x_1, y_1), (x_2, y_2) \cdots (x_l, y_l)] x \in R^N,\ y \in r$, where x is the input, y is the output, R^N is the N-dimensional vector space, and r is the one-dimensional vector space. In this problem, the width of footing (B), depth of footing (D), the length-to-width ratio (L/B) of footings, density of soil (γ or γ') angle of internal friction (Φ), and so forth were used as model input parameters to predict ultimate bearing capacity (q_u). Hence, for this problem, $x = [B,\ D,\ (L/B),\ \gamma,\ \Phi]$ and $y = [q_u]$.

The intension of SVM is to fit a function that can approximately predict the value of output on supplying a new set of predictors (input variables).

The ε-intensive loss function can be described as follows:

$$L_\varepsilon(y) = 0 \quad \text{for } |f(x) - y| \le \varepsilon, \tag{1}$$

otherwise,

$$L_\varepsilon(y) = |f(x) - y| - \varepsilon. \tag{2}$$

This defines an ε-tube so that if the predicted value is within the tube, the loss is zero; otherwise the loss is equal to the absolute value of the deviation minus ε. This concept is depicted in Figure 1.

SVM attempts to find that a function $f(x)$ that gives the deviation of "ε" from the actual output is as flat as possible.

Consider a linear function of the form,

$$f(x) = (w \cdot x) + b, \quad w \in R^N,\ b \in r, \tag{3}$$

where w is an adjustable weight vector and b is the scalar threshold. Fitness means the search for a small value of "w". It can be represented as a minimization problem with an objective function comprising the Euclidian norm as follows:

$$\begin{aligned} &\text{Minimize :} && \frac{1}{2}\|w\|^2 \\ &\text{Subject to} && y_i - [(w \cdot x_i) + b] \le \varepsilon, \quad i = 1, 2, 3, \ldots, l, \\ & && [(w \cdot x_i) + b] - y_i \le \varepsilon, \quad i = 1, 2, 3, \ldots, l. \end{aligned} \tag{4}$$

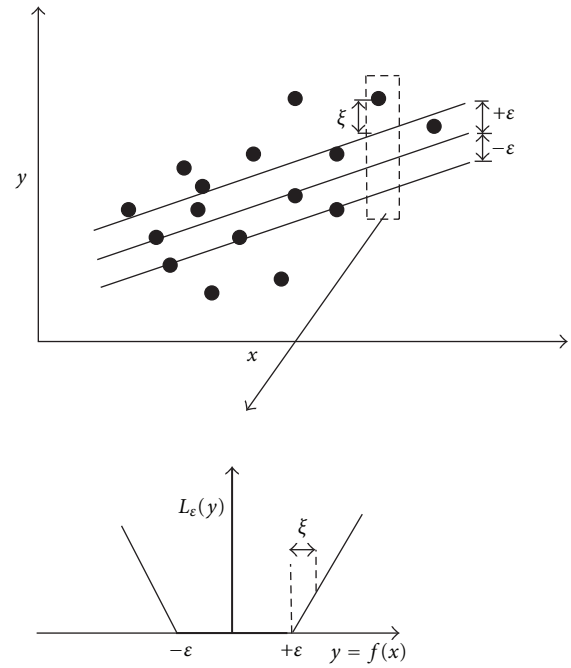

Figure 1: The ε-tube and slack variable (ξ) in SVM.

Some allowance for errors (ε) may also be introduced. Two slack parameters ξ and ξ^* have been introduced to penalize the samples with error more than "ε". Thus the infeasible constraints of the optimization problem are eliminated. The modified formulation takes the following form:

$$\begin{aligned} &\text{Minimize :} && \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\left(\xi_i + \xi_i^*\right) \\ &\text{Subject to} && y_i - [(w \cdot x_i) + b] \le \varepsilon + \xi_i, \quad i = 1, 2, 3, \ldots, l, \\ & && [(w \cdot x_i) + b] - y_i \le \varepsilon + \xi_i^*, \quad i = 1, 2, 3, \ldots, l. \\ & && \xi_i \ge 0, \quad \xi_i^* \ge 0, \quad i = 1, 2, 3, \ldots, l. \end{aligned} \tag{5}$$

The constant $0 < C < \infty$ determines the tradeoff between the flatness of $f(x)$ and the amount up to which the deviations larger than "ε" are tolerated [17]. The above optimization problem is solved by Vapnik [15] using Lagrange multiplier method. The solution is given by

$$\begin{aligned} f(x) &= \sum_{i=1}^{M}(\alpha_i - \alpha_i^*)(x_i \cdot x) + b, \\ \text{where} \quad b &= -\left(\frac{1}{2}\right) w \cdot (x_r + x_s), \end{aligned} \tag{6}$$

where x_s and x_r are known as support vectors and M is the number of support vectors.

Some Lagrange multipliers ($\alpha_i,\ \alpha_i^*$) will be zero, which implies that these training solutions are irrelevant to the final solution (known as sparseness of the solution). The training objects with nonzero Lagrange multipliers are called support

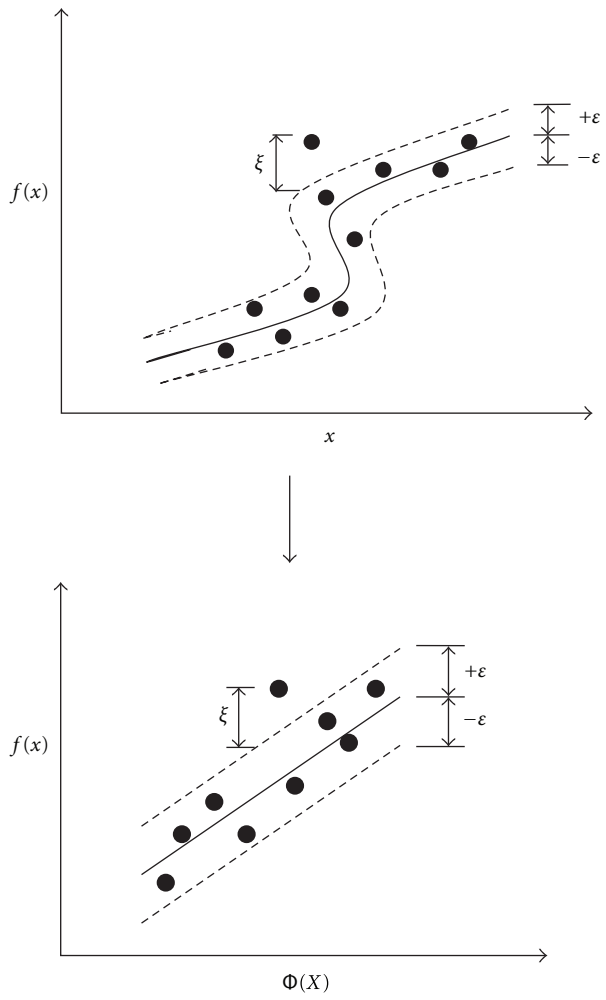

Figure 2: Concept of nonlinear regression using SVM.

vectors. When linear regression is not appropriate, input data have to be mapped into a high-dimensional feature space through nonlinear mapping and the linear regression needs to be performed in the high-dimensional feature space [14]. Kernel function K is used to transform nonlinear data from the input to the feature space in linear form. Then linear fitting in new space will be equal to nonlinear fitting in original space:

$$K\left(x_i, x_j\right) = \phi(x_i) \cdot \phi\left(x_j\right), \tag{7}$$

where K is the kernel function, x_i and x_j are inputs, and $\phi(x_i) \cdot \phi(x_j)$ is the dot product in the high-dimensional space.

Thus, (6) can be replaced by

$$f(x) = \sum_{i=1}^{M} (\alpha_i - \alpha_i^*) K\left(x_i \cdot x_j\right) + b. \tag{8}$$

The concept of nonlinear mapping is depicted in Figure 2.

The functions which satisfy Mercer's theorem can be used for fitting the data [14]. Polynomial functions, radial basis function (RBF), and splines are the most commonly used kernel functions for data fitting using SVM. The mathematical forms of some popular kernel functions can be found in [18].

3. Genetic Programming

Genetic Programming (GP) is an automatic programming technique for evolving computer programs to solve, or approximately solve, problems introduced by Koza [13]. GP is basically an optimization paradigm that can also be effectively applied to the genetic symbolic regression (GSR). GSR involves finding a mathematical expression in symbolic form relating finite values of set of independent variables (x_i) and a set of dependent variables (y_i) [19]. GP works on Darwin's natural selection theory in evolution. Here, a population is progressively improved by selectively discarding the not-so-fit population and breeding new children to form better populations. Like other evolutionary algorithms, the solution is started with a random population of individuals (equations or computer programs). Each possible solution set can be visualized as a "parse tree" comprising the terminal set (input variables) and functions (general operators such as +, −, ∗, /, logarithmic or trigonometric). The "fitness" is a measure of how closely a trial solution solves the problem. The objective function—the minimization of error between estimated and observed values—is the fitness function. The solution set in a population associated with the "best fit" individuals will be reproduced more often than the less fit solution sets. It iteratively transforms a population of computer programs into a new generation of programs by applying analogs to naturally occurring genetic operators like reproduction, mutation, and crossover. The different genetic operations can be found in detail in [13]. The basic procedure of GP is presented as a flow chart in Figure 3.

In the recent past, GP is effectively applied to solve a wide range of geotechnical engineering problems [20–22]. GP can evolve an explicit equation or equivalent computer program relating the input and output variables which is a more understandable depiction of the cause-effect process. Some literature suggests that the program-based GP approach (i.e., the GP algorithms which give a computer program which helps for estimating the predictant value for a given set of predictors) can perform equally well with an equation-based approach and other soft computing tools like ANN [23–26]. A program-based GP approach is adopted for the present study.

4. Model Development and Results

The primary step in model development for the estimation of bearing capacity of cohesionless soils underneath shallow foundations is identification of parameters that affect the bearing capacity. The basic form of equation for bearing capacity of cohesionless soil is [5]

$$q_u = \gamma D N q S q d q + 0.5 B \gamma N_\gamma S_\gamma d_\gamma, \tag{9}$$

where B is the width of foundation, D is the depth of foundation, γ is the unit weight of sand, N_q, N_γ is the bearing

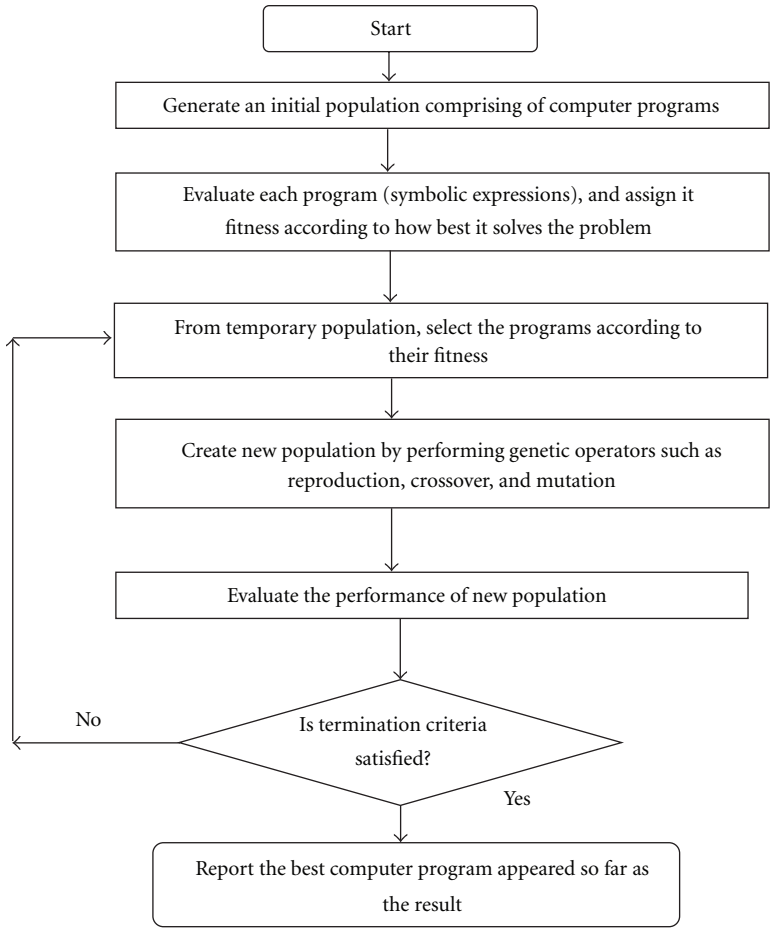

Figure 3: Flow chart of Genetic Programming.

capacity factors, S_q, S_γ is the shape factors, and d_q, d_γ the depth factors. These factors primarily depend on the angle of shearing resistance, unit weight of the sand, and the geometry of the foundation.

The main factors affecting the bearing capacity are its width (least lateral dimension, B), length of footing (L), shape (square, rectangular, and circular), and depth of embedment (D). The depth of foundation has the greatest effect on the bearing capacity of all the physical properties of the foundation. There are some other factors such as compressibility and thickness of the soil layer beneath the foundation that contribute to a lesser degree [5]. The effect of compressibility is small, except for loose densities, and is generally less important in bearing capacity computation [5]. Moreover, there are insufficient data to consider compressibility as well as thickness of soil stratum. Therefore, they are not considered in this study.

4.1. Database. The data used in the present study has been adopted from Padmini et al. [5]. The five input parameters used for the model development in this study are width of footing (B), depth of footing (D), footing geometry (L/B), unit weight of sand (γ), and angle of shearing resistance (Φ). Ultimate bearing capacity (q_u) is the single output. The data thus compiled comprises a total of 97 data sets, which consists of results of load test data of square, rectangular, and strip footings of different sizes tested in sand beds of various densities. Out of the total 97 sets of data, 78 are used for training and 19 are used for validation in all the experiments considered in this study. The data division is done in such a way that the same 19 sets of data used by Padmini et al. [5] are kept as the validation dataset to enable a comparison of results of the present study with those obtained by ANN and FIS by Padmini et al. [5].

4.2. Development of SVM Model. The data mining software WEKA 3.6.1 proposed by Witten and Frank [27] is used for developing SVM model. In this study an ε-variant of SVM (ε-SVM) is used for support vector regression, and the loss function (ε) is fixed as 0.001. Initially, a polynomial kernel of degree (d) 2 is used to fit a nonlinear model. The selection of regularization parameter C and kernel-specific parameters (d and σ for polynomial and RBF kernel, resp.) may influence the results. A large value of C indicates that the objective function is only to minimize the empirical risk, which makes the learning machine more complex. On the other hand, a smaller C may cause learning errors with poor approximation [28].

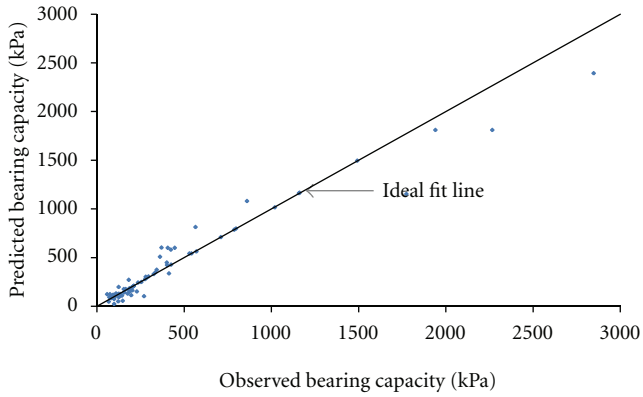

Figure 4: Scatter plot of SVM model with polykernel for training dataset.

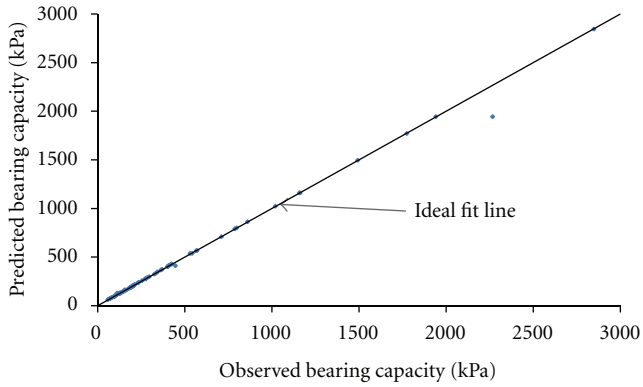

Figure 5: Scatter plot of SVM model with RBF Kernel for training dataset.

A trial and error approach is followed to find the optimal value of C for model with polynomial kernel. The C parameter of 100 is found to be quite successful in giving satisfactory performance. Then a radial basis function (RBF) kernel is used to fit a nonlinear model in the present study to build an SVM model. The combination of control parameters such as $C = 250$ and $\sigma = 3$ gives very good training performance. The plot between observed and predicted values of training dataset is shown in Figure 4 (polykernel) and Figure 5 (RBF kernel). These plots indicate that the model is well trained.

4.3. Development of GP Model. The genetic programming software DISCIPULUS [29] is used for developing GP model. The models are created in the form of "evolved" computer programs as GP uses Darwinian natural selection to create them. Using this model, the output of statistically similar input data can be predicted with very much accuracy. The initial control parameters used for the problem are population size: 500, crossover probability: 0.95, and mutation probability: 0.5. The basic arithmetical functions (such as addition, multiplication, subtraction, and division ($+$, $*$, $-$, $/$)) constitute the function set. The fitness function is selected as the root-mean-square error between the measure and predicted values of ultimate bearing capacity. The best program generated by GP software for predicting the UBC of cohesionless soils is given in the appendix. The plot between observed and predicted values of training dataset is shown in Figure 6. This plot indicates that the model is well trained.

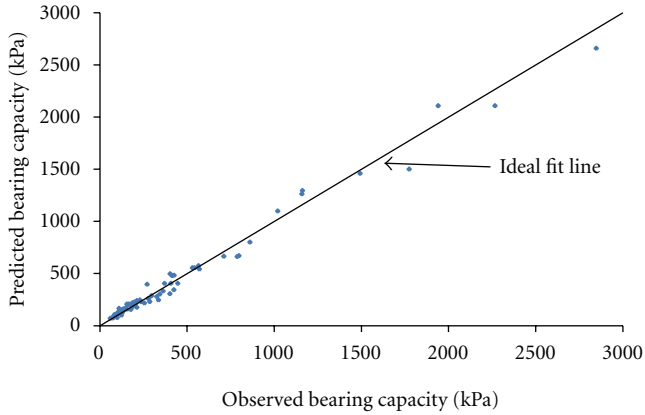

Figure 6: Scatter plot of GP model for training dataset.

5. Results and Discussions

The efficiency of the developed models is analyzed by different statistical performance evaluation criteria such as correlation coefficient (R), coefficient of efficiency (E), root-mean-square error (RMSE), mean bias error (MBE), and mean absolute relative error (MARE). The equations of different performance evaluation measures were presented in Table 1, in which y_o stands for the observed output value, y_c represents the computed output value, $\overline{y^o}$ is the mean of observed values, $\overline{y^c}$ represents the mean of computed values, and n represents the number of data points. The different performance evaluation criteria estimated for training dataset are presented in Table 2. The predictions for testing dataset using different models are presented in Table 3, and the performance evaluation for these predictions is presented in Table 4. However, it is to be noted that the ANN and FIS results presented in Table 4 are deduced based on the relative error (RE) values reported by Padmini et al. [5]. From Table 4 it can be inferred that the correlation coefficient and coefficient of efficiency are the highest (0.997 and 0.996) and the error criteria such as RMSE, MBE, and MARE are the least (44.967, 4.01, and 7.69) for the GP-based modeling.

Further the scatter plots between observed and predicted values of UBC for SVM models are presented in Figure 7 (polykernel) and Figure 8 (RBF kernel). The 5% error bar lines are plotted along with these scatter plots. Such a plot can be used to indicate the range of standard deviation and to determine whether the differences are statistically significant [8]. A perusal of plots shows that, for GP-based predictions, all points lie within the specified confidence interval of 95%. Thus, it can be inferred that all the soft computing methods outperform the theoretical approaches in the prediction of bearing capacity. Similar plot for predictions with GP model is presented in Figure 9. Also from Table 4, it is seen that the R value and E value are closer to unity and different error

Table 1: Performance evaluation criteria.

Evaluation criteria	Equation
Coefficient of correlation (R)	$R = \frac{\sum_{i=1}^{n}(y_i^o - \overline{y^o})(y_i^c - \overline{y^c})}{\sqrt{\sum_{i=1}^{n}(y_i^o - \overline{y^o})^2}\sqrt{\sum_{i=1}^{n}(y_i^c - \overline{y^c})^2}}$
Coefficient of efficiency (E)	$E = 1 - \frac{\sum_{i=1}^{n}(y_i^o - y_i^c)^2}{\sum_{i=1}^{n}(y_i^o - \overline{y^o})^2}$
Root-mean-square error (RMSE)	$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(y_i^o - y_i^c)^2}{n}}$
Mean bias error (MBE)	$\text{MBE} = \frac{1}{n}\sum_{i=1}^{n}(y_i^c - y_i^o)$
Percentage relative error (RE)	$\text{RE} = \frac{(y_o - y_c)}{y_o} * 100$
Percentage mean absolute relative error (MARE)	$\text{MARE} = \frac{1}{n}\sum_{i=1}^{n}\lvert\text{RE}\rvert$

Table 2: Performance evaluation of different models for training dataset.

Performance index	Meyerhof [2]	Hansen [3]	Vesic [4]	SVM*		GP*
				Polykernel ($C = 100$; $d = 2$)	RBF kernel ($C = 250$; $\sigma = 3$)	
R	0.9307	0.9295	0.9408	0.9742	0.9977	0.9923
E	0.7387	0.6908	0.7208	0.9408	0.9948	0.9900
RMSE (kPa)	260.0015	282.8295	268.7726	123.7923	36.8280	65.0650
MBE (kPa)	−78.3587	−95.8701	−103.2620	−9.5919	−3.5807	−0.8618
MARE	19.6545	20.9268	25.6619	19.3968	1.8036	12.9030

*Present study.

Table 3: Predicted bearing capacity of different models for testing dataset.

Sl no.	Observed bearing capacity (kPa)	Meyerhof [2] (kPa)	Hansen [3] (kPa)	Vesic [4] (kPa)	ANN (kPa)	FIS (kPa)	SVM (kPa)		GP (kPa)
							RBF Kernel ($C = 250$; $\sigma = 3$)	Polykernel ($C = 100$; $d = 2$)	
1	1760	1174.892	1230.649	1194.571	1753.048	1888.48	2137.096	2005.19	1794.292
2	214	163.54	164.0827	117.9378	221.1134	195.1252	257.952	165.809	211.9048
3	681	353.1485	322.5613	372.5071	579.7285	651.3084	999.874	676.838	639.0338
4	137	114.1407	124.0721	94.67636	161.6463	100.6128	125.435	143.271	152.3006
5	322	372.0038	365.65	329.434	226.6236	150.2774	304.729	240.232	311.4622
6	2033	1757.721	1641.436	1510.498	2047.19	2164.129	1863.654	2012.747	2077.698
7	464	560.7228	530.9029	542.3884	475.0664	657.024	540.622	816.999	572.9832
8	461	270	295.4085	214.7597	348.0043	274.1106	409.256	368.095	484.8436
9	1140	865.35	841.3887	780.5128	1064.874	1097.022	1023.901	1059.188	1118.019
10	630	516.9667	565.6161	411.1984	512.1459	541.044	667.666	632.805	498.9244
11	1540	697.3765	608.3824	664.3541	1626.086	1683.99	1691.924	1271.259	1558.777
12	180.5	139.4636	134.7036	115.906	269.2338	241.4368	188.525	175.087	217.0617
13	91.5	101.6777	88.88982	80.81764	99.8448	107.0916	93.131	77.511	88.75574
14	244.6	290.428	279.7286	249.5037	233.3973	230.3643	242.907	250.901	250.7569
15	143.3	138.7347	138.3666	119.3544	108.4208	151.7547	139.657	138.799	127.4984
16	131.5	144.9232	135.2796	162.6324	130.6124	156.5113	129.475	109.495	153.612
17	253.6	283.7822	270.4981	333.1746	226.1098	275.0038	251.614	255.153	266.6882
18	135.2	125.613	121.5267	136.3814	128.8321	178.5992	135.021	131.044	133.1664
19	264.5	329.4678	314.9475	383.6036	198.7189	347.2885	272.425	386.591	245.1108

Table 4: Performance evaluation of different models for testing dataset.

Performance index	Meyerhof [2]	Hansen [3]	Vesic [4]	ANN [5]	FIS [5]	SVM*		GP*
						Polykernel ($C = 100$; $d = 2$)	RBF Kernel ($C = 250$; $\sigma = 3$)	
R	0.9410	0.9366	0.9456	0.9951	0.9899	0.9775	0.9806	0.9972
E	0.7863	0.7583	0.7321	0.9942	0.9858	0.9541	0.9504	0.9965
RMSE (kPa)	269.947	287.099	302.269	62.620	98.002	125.087	130.102	44.967
MBE (kPa)	−127.724	−139.611	−158.552	−21.89	13.92	34.11	4.75	4.019
MARE	22.0679	20.0152	28.515	13.314	19.456	14.7278	9.4528	7.6817

*Present study.

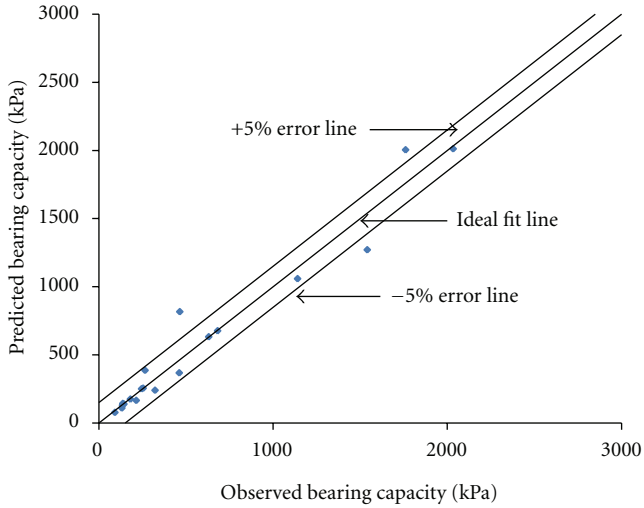

Figure 7: Scatter plot and 5% error bar lines for SVM model with polykernel (testing dataset).

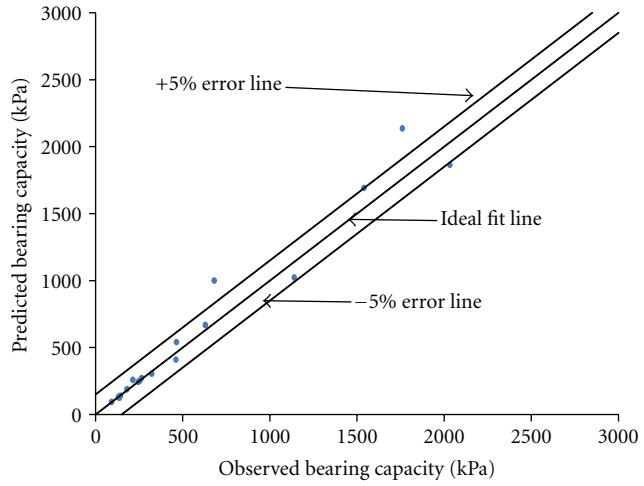

Figure 8: Scatter plot and 5% error bar lines for SVM model with RBF Kernel (testing dataset).

criteria are much lesser for any of the applied soft computing tools when compared with theoretical models.

A statistical evaluation of the predictions by the different soft computing models for the testing dataset is performed and presented in Table 5. The standard deviation, average deviation, and coefficient of variation values of GP model results (607.91, 454.59, and 1.059) show close agreement with that of observed values (600.02, 459.48, and 1.052) followed by that of SVM (RBF) model. This confirms the robustness of the newly applied paradigms.

The different performance evaluation measures of SVM-based modelling (in Tables 4 and 5) show that the performance of RBF-based SVM is competent with ANN and FIS results. Also SVM involves only lesser number of control parameters (such as C and σ), and ANN involves large number of such parameters and their optimal combination is a tedious process. Thus, the SVM approach is quite simple to implement. Further, the performance evaluation of GP-based results (Tables 4 and 5) shows that the R, E and different error criteria are better for the GP model when compared with the theoretical methods, the SVM, and

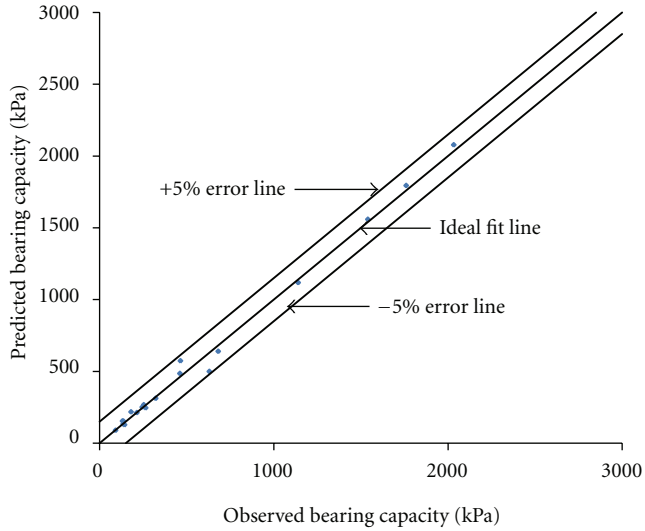

Figure 9: Scatter plot and 5% error bar lines for GP model (testing dataset).

Table 5: Statistical properties of values predicted by different models.

Statistical Measure	Observed	ANN	FIS	SVM (Poly)*	SVM (RBF)*	GP*
Maximum	2033	2047.19	2164.13	2137.1	2077.7	2077.7
Minimum	91.5	99.84	100.61	93.13	77.51	88.75
Average	569.83	547.93	583.7	603.9	574.58	573.84
Average deviation	459.48	455.92	488.4	501.1	468.73	454.59
Standard deviation	600.02	609.3	645.8	641.0	608.60	607.91
Coefficient of variation	1.052	1.112	1.106	1.059	1.059	1.059

*Present study.

interpreted results of ANN and FIS. Thus, GP is proven to be a reliable alternative soft computing technique for prediction of ultimate bearing capacity of shallow foundation on cohesionless soil.

6. Conclusions

In this paper the application of two relatively recent soft computing techniques—SVM and GP—is investigated for the prediction of ultimate bearing capacity of cohesionless soils beneath shallow foundations. SVM results are competent and demand the optimal selection of only a few number of control parameters when compared with ANN. Performance evaluation based on multiple error criteria shows that error is the least and correlation coefficient (R) and coefficient of efficiency (E) are the highest for the GP-based modeling than SVMs, ANN, FIS, and the different theoretical models considered in this study. The GP-based modeling is found to be superior in terms of quality, and it gives the output in the form of computer programs which enables the user to apply for a new set of input data to predict the ultimate bearing capacity. Thus, GP can be recommended as a robust soft computing paradigm to predict the ultimate bearing capacity of soil.

Appendices

A. Note

The C++ Program to predict the ultimate bearing capacity of cohesionless soils is given here. V[0] to V[4] represent the input parameters width of footing (B), the depth of footing (D), the length-to-width ratio (L/B), the field density (γ), the angle of shearing resistance (Φ). f[0], f[1], and so forth, are the temporary computation variables that the programs GP software creates. The output of these programs is the value remaining in f[0] after the program executes. This program needs to be run in the DISCIPULUS software environment to get the predictant value for a new set of predictors.

B. Best Program

```
#define TRUNC(x)(((x)>=0) ? floor(x): ceil(x))
#define C_FPREM (_finite(f[0]/f[1]) ?
f[0]-(TRUNC(f[0]/f[1])
  *f[1]): f[0]/f[1])
#define C_F2XM1 (((fabs(f[0])<=1) &&
  (!_isnan(f[0]))) ? (pow(2,f[0])-1):
  ((!_finite(f[0]) && !_isnan(f[0]) &&
  (f[0]<0)) ? -1: f[0]))

float DiscipulusCFunction(float v[])
{
long double f[8];

long double tmp = 0;

int cflag = 0;
f[0]=f[1]=f[2]=f[3]=f[4]=f[5]=f[6]=f[7]=0;
L0: f[0]/=-1.364008665084839f;
L1: f[0]+=f[1];
L2: f[0]=-f[0];
L3: f[0]-=v[0];
L4: f[0]+=v[4];
L5: f[0]+=v[4];
L6: f[0]=cos(f[0]);
L7: f[0]+=f[0];
L8: f[0]+=f[0];
L9: f[0]+=f[0];
L10: f[0]*=v[1];
L11: f[0]+=v[4];
L12: f[0]-=1.252994060516357f;
L13: f[0]*=pow(2,TRUNC(f[1]));
L14: cflag=(f[0] < f[1]);
L15: f[0]=sqrt(f[0]);
L16: f[0]*=0.2877938747406006f;
L17: tmp=f[1]; f[1]=f[0]; f[0]=tmp;
L18: f[0]-=v[3];
L19: f[0]*=-0.494312047958374f;
```

```
L20: f[0]*=0.7790718078613281f;
L21: f[0]*=f[1];
L22: f[0]=fabs(f[0]);
L23: f[0]=cos(f[0]);
L24: f[0]=−f[0];
L25: f[0]=sqrt(f[0]);
L26: if (cflag) f[0] = f[1];
L27: f[0]+=v[4];
L28: f[0]*=0.9955191612243652f;
L29: f[0]*=0.4281637668609619f;
L30: f[0]−=f[0];
L31: f[0]−=v[3];
L32: f[0]/=v[0];
L33: f[0]−=0.9955191612243652f;
L34: f[0]+=v[4];
L35: cflag=(f[0] < f[1]);
L36: f[0]−=f[1];
L37: f[0]/=f[0];
L38: f[0]*=pow(2,TRUNC(f[1]));
L39: f[0]−=f[1];
L40: f[0]=−f[0];
L41: f[0]=fabs(f[0]);
L42: f[0]=−f[0];
L43: f[0]*=0.4281637668609619f;
L44: f[0]*=f[0];
L45: f[0]*=f[0];
L46: f[0]/=f[0];
L47: f[0]*=−0.7297487258911133f;
L48: f[0]/=1.084159851074219f;
L49: f[0]+=v[4];
L50: tmp=f[0]; f[0]=f[0]; f[0]=tmp;
L51: f[0]/=f[1];
L52: f[0]+=0.7790718078613281f;
L53: f[0]+=v[4];
L54: f[0]−=f[1];
L55: f[0]*=f[1];
L56: f[0]*=0.4281637668609619f;
L57: f[0]−=v[3];
L58: f[0]−=−0.9486191272735596f;
L59: if (cflag) f[0] = f[1];
L60: f[0]/=v[2];
L61: f[0]+=v[4];
L62: f[0]−=v[2];
L63: f[0]−=v[2];
L64: f[0]−=v[2];
L65: f[0]*=v[1];
L66: f[0]+=v[4];
L67: f[0]*=f[1];
L68: f[0]*=0.4281637668609619f;
L69: f[0]−=v[3];
L70: f[1]*=f[0];
L71: f[0]*=f[0];
L72: f[0]−=f[1];
L73: f[1]+=f[0];
L74: if (!cflag) f[0] = f[1];
L75: f[0]−=1.987620830535889f;
L76: f[0]−=v[4];
L77: f[0]−=1.987620830535889f;
L78: f[0]−=v[4];
L79: f[0]−=v[4];
L80: f[0]−=0.7361507415771484f;
L81: f[0]−=1.987620830535889f;
L82: f[0]−=v[4];
L83: f[0]−=1.987620830535889f;
L84: f[0]−=v[4];
L85: f[0]−=1.501374244689941f;
L86: f[0]+=v[2];
L87: f[0]−=v[4];
L88: f[0]−=1.530829906463623f;
L89: if (!cflag) f[0] = f[1];
L90: f[0]+=v[3];
L91: f[0]−=v[4];
L92: f[0]+=−1.907608032226563f;
L93: if (!cflag) f[0] = f[1];
L94: f[0]+=v[3];
L95: f[0]+=v[3];
L96: f[0]+=v[3];
L97: f[0]+=v[3];
L98: f[0]+=v[3];
L99: f[0]+=v[3];
L100: f[0]+=v[3];
L101: f[0]+=v[3];
```

```
L102: f[0]+=v[3];
L103: f[0]+=v[3];
L104: f[0]+=v[3];
L105: f[0]+=v[3];
L106: f[0]+=v[3];
L107: f[0]+=v[3];
L108: f[0]+=v[2];
L109: f[0]+=v[3];
L110:
if (!_finite(f[0])) f[0]=0;
return f[0];
}.
```

Acknowledgments

This paper is a part of a research work carried out at the Department of Civil Engineering, TKM College of Engineering Kollam, Kerala, India, in 2010. The authors thank the Department of Civil Engineering, TKM College of Engineering Kollam for providing all necessary help. They also thank the anonymous reviewer/s who helped to improve the quality of the paper.

References

[1] K. Terzaghi, *Theoretical Soil Mechanics*, John Wiley & Sons, New York, NY, USA, 1943.

[2] G. G. Meyerhof, "Some recent research on the bearing capacity of foundations," *Canadian Geotechnical Journal*, vol. 1, no. 1, pp. 16–26, 1963.

[3] J. B. Hansen, "A general formula for bearing capacity," *Danish Geotechnical Institute Bulletin*, vol. 11, 1961.

[4] A. S. Vesic, "Analysis of ultimate loads of shallow foundations," *Journal of Soil Mechanics and Foundation Division*, vol. 99, no. 1, pp. 45–73, 1973.

[5] D. Padmini, K. Ilamparuthi, and K. P. Sudheer, "Ultimate bearing capacity prediction of shallow foundations on cohesionless soils using neurofuzzy models," *Computers and Geotechnics*, vol. 35, no. 1, pp. 33–46, 2008.

[6] M. A. Shahin, H. R. Maier, and M. B. Jaksa, "Artificial neural network applications in Geotechnical Eng," *Australian Geomechanics*, vol. 36, no. 1, pp. 49–62, 2001.

[7] P. Samui, "Prediction of friction capacity of driven piles in clay using the support vector machine," *Canadian Geotechnical Journal*, vol. 45, no. 2, pp. 288–295, 2008.

[8] P. Samui and T. G. Sitharam, "Least-square support vector machine applied to settlement of shallow foundations on cohesionless soils," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 32, no. 17, pp. 2033–2043, 2008.

[9] P. Samui, "Slope stability analysis: a support vector machine approach," *Environmental Geology*, vol. 56, no. 2, pp. 255–267, 2008.

[10] M. Pal, "Support vector machines-based modelling of seismic liquefaction potential," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 30, no. 10, pp. 983–996, 2006.

[11] J. H. Holland, *Adaptation in Natural and Artificial System*, Ann Arbour Science Press, Ann Arbor, Mich, USA, 1975.

[12] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

[13] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, Mass, USA, 1992.

[14] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *5th Annual ACM Workshop on COLT*, D. Haussler, Ed., pp. 144–152, ACM Press, Pittsburgh, Pa, USA, 1992.

[15] V. N. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, New York, NY, USA, 1998.

[16] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[17] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[18] S. Rajasekaran, S. Gayathri, and T. L. Lee, "Support vector regression methodology for storm surge predictions," *Ocean Engineering*, vol. 35, no. 16, pp. 1578–1587, 2008.

[19] S. T. Khu, S. Y. Liong, V. Babovic, H. Madsen, and N. Muttil, "Genetic programming and its application in real-time runof f forecasting," *Journal of the American Water Resources Association*, vol. 37, no. 2, pp. 439–451, 2001.

[20] A. A. Javadi, M. Rezania, and M. M. Nezhad, "Evaluation of liquefaction induced lateral displacements using genetic programming," *Computers and Geotechnics*, vol. 33, no. 4-5, pp. 222–233, 2006.

[21] A. Johari, G. Habibagahi, and A. Ghahramani, "Prediction of soil-water characteristic curve using genetic programming," *Journal of Geotechnical and Geoenvironmental Engineering*, vol. 132, no. 5, pp. 661–665, 2006.

[22] B. S. Narendra, P. V. Sivapullaiah, S. Suresh, and S. N. Omkar, "Prediction of unconfined compressive strength of soft grounds using computational intelligence techniques: a comparative study," *Computers and Geotechnics*, vol. 33, no. 3, pp. 196–208, 2006.

[23] S. B. Charhate, M. C. Deo, and S. N. Londhe, "Inverse modeling to derive wind parameters from wave measurements," *Applied Ocean Research*, vol. 30, no. 2, pp. 120–129, 2008.

[24] S. Gaur and M. C. Deo, "Real-time wave forecasting using genetic programming," *Ocean Engineering*, vol. 35, no. 11-12, pp. 1166–1172, 2008.

[25] K. Ustoorikar and M. C. Deo, "Filling up gaps in wave data with genetic programming," *Marine Structures*, vol. 21, no. 2-3, pp. 177–195, 2008.

[26] S. S. Kashid, S. Ghosh, and R. Maity, "Streamflow prediction using multi-site rainfall obtained from hydroclimatic teleconnection," *Journal of Hydrology*, vol. 395, no. 1-2, pp. 23–38, 2010.

[27] I. H. Witten and E. Frank, *Data Mining*, Morgan Kaufmann, San Francisco, Calif, USA, 2000.

[28] P. S. Yu, S. T. Chen, and I. F. Chang, "Support vector regression for real-time flood stage forecasting," *Journal of Hydrology*, vol. 328, no. 3-4, pp. 704–716, 2006.

[29] F. D. Francone, *Discipulus Owner's Manual Version 3.0 DRAFT*, Machine Learning Technologies, Littleton, Colo, USA, 1998.