Computational Intelligence Techniques for Modelling the Critical Flashover Voltage of Insulators: From Accuracy to Comprehensibility

Evangelos Karampotsis[1(✉)], Konstantinos Boulas[1], Alexandros Tzanetos[1], Vasilios P. Androvitsaneas[2], Ioannis F. Gonos[2], Georgios Dounias[1], and Ioannis A. Stathopulos[2]

[1] Management and Decision Engineering Laboratory (MDE-Lab), Department of Financial and Management Engineering, University of the Aegean, 41 Kountouriotou Street, 82100 Chios, Greece
karabotsis-evangelos@hotmail.com, g.dounias@aegean.gr

[2] High Voltage Laboratory, School of Electrical and Computer Engineering, National Technical University of Athens, 9 Iroon Politechniou Street, Zografou Campus, 15780 Athens, Greece
v.andro@mail.ntua.gr

Abstract. This paper copes with the problem of flashover voltage on polluted insulators, being one of the most important components of electric power systems. A number of appropriately selected computational intelligence techniques are developed and applied for the modelling of the problem. Some of the applied techniques work as black-box models, but they are capable of achieving highly accurate results (artificial neural networks and gravitational search algorithms). Other techniques, on the contrary, obtain results somewhat less accurate, but highly comprehensible (genetic programming and inductive decision trees). However, all the applied techniques outperform standard data analysis approaches, such as regression models. The variables used in the analyses are the insulator's maximum diameter, height, creepage distance, insulator's manufacturing constant, and also the insulator's pollution. In this research work the critical flashover voltage on a polluted insulator is expressed as a function of the aforementioned variables. The used database consists of 168 different cases of polluted insulators, created through both actual and simulated values. Results are encouraging, with room for further study, aiming towards the development of models for the proper inspection and maintenance of insulators.

Keywords: Insulators · Critical flashover voltage · Computational intelligence · Artificial neural networks · Inductive decision trees · Genetic programming · Gravitational search algorithm

S. Benferhat et al. (Eds.): IEA/AIE 2017, Part I, LNAI 10350, pp. 295–301, 2017.
DOI: 10.1007/978-3-319-60042-0_35

1 Introduction

One big problem, appearing on cap & pin insulators (Fig. 1) with surface pollution, is the flashover phenomenon. Briefly, the flashover phenomenon, has different occurrence process (depending on the material of the insulator), is presented at a particular operating voltage, called critical flashover voltage and can cause the partial or total evacuation of an insulator until the collapse of a power line [1].

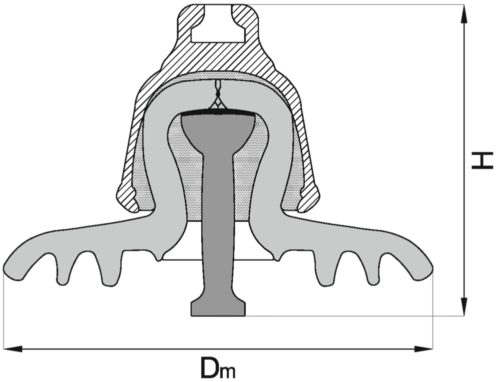

Fig. 1. Basic design parameters of Cap & Pin Insulator

This paper tries to balance between accurate and comprehensible results, through the modelling effort of the flashover phenomenon using Artificial Neural Networks (ANNs), Genetic Programming (GP) and Inductive Decision Trees (IDT) with reference point, a standard data analysis approach (Multiple Linear Regression-MLR). Finally, using an evolutionary algorithm (Gravitational Search Algorithm-GSA) we try to optimize the proposed model given from genetic programming.

The database was common to all four techniques applied to the problem and consists of 168 cases (i.e. series of related measurements) represented by six (6) numeric variables, namely the insulator's maximum diameter (D_m), height (H), creepage distance (L), insulator's manufacturing constant (F), and the insulator's pollution (C) and the critical flashover voltage (U_c). The critical flashover voltage of a polluted insulator is expressed in relation to these variables. A part of these application data (140 cases) are simulated data derived from a specialized model corresponding to incidents of flashover voltage on polluted Cap & Pin insulators, while the rest of the data (28 cases) consist of real experimental observations [1].

The main database was divided into two parts. The first part, which is called Training-Validation Set (130 cases of simulated data and 20 cases of actual data), was used for the development of the predictive models and the second part, which is called Test Set (10 cases of actual data and 8 cases of simulated data), was used to test the final predictive models.

2 Modelling Approaches

The **MLR model (MLR)** for the estimation of flashover voltage is described by Eq. 1 and its performance through different measures is given in Table 1.

$$U_C = 1.0368 \cdot D_m + 0.468 \cdot H + 1.8342 \cdot L + 0.4292 \cdot F - 6.8969 \cdot C + 13.0605 \quad (1)$$

Table 1. Aggregation and comparative scoreboard of MLR, ANN-2 GP-1, GP-2 and GSA-1

	r	MAE (kV)	RMSE (kV)	RAE (%)	RRSE (%)
Model: MLR					
Training	0.8878	2.2314	2.6763	43.659	45.9127
Test	0.7942	2.5908	2.8413	–	–
Model: ANN-2					
Training	0.9991	0.146	0.2522	2.8568	4.3264
Test	0.9904	0.731	0.8256	–	–
Model: GP-1					
Training	0.9979685	0.169555	0.2499	1.2901	–
Validation	0.999247	0.171671	0.2177	1.2332	–
Test	0.993614	0.276758	0.4538	3.5585	–
Model: GP-2					
Training	0.99506	0.32137	0.41889	2.5808	–
Test	0.99564	0.26586	0.29275	2.2182	–
Validation	0.98985	0.50008	0.62641	3.4887	–
Model: GSA-1					
Training	0.99728	0.33031	0.19303	–	–
Test	0.99958	0.03388	0.01541	–	–
Validation	0.99027	0.46951	0.38056	–	–

Its performance is clearly lower than all other intelligent approaches applied, as it can also be seen in Table 1.

The **ANN model (ANN-2)**, presented in Fig. 2, is a Multilayer Perceptron (MLP) with 1 hidden layer (20 nodes) and activation function of hidden nodes was the logistic function. ANN-2 training was supervised, into batch mode and was performed using the back propagation algorithm (BP). Moreover, the validation was held together with the process of training using the 10-fold cross validation method based on the same dataset (Training-Validation Set). The performances are depicted in Table 1 [2, 3].

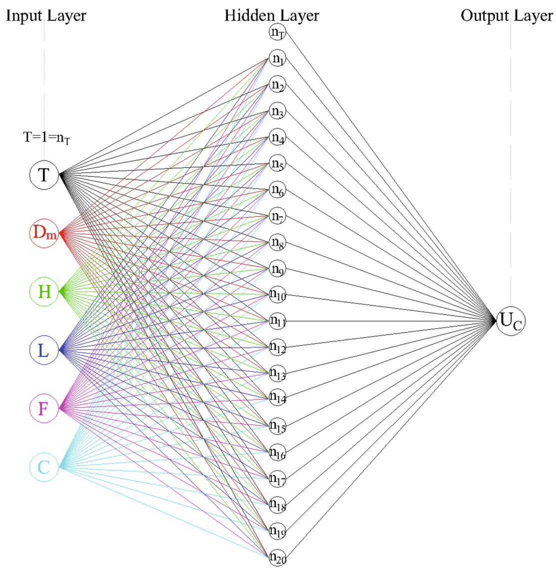

Fig. 2. The ANN-2 model

The **GP models (donate as GP-1 and GP-2)**, emerged using the Training-validation Set and following the procedure of 10-fold cross validation. Specifically, for each fold, 30 independent runs were made within the same modulation. All runs trained their solution with k-1 subsets and all were compared in the rest testing data. So, ten (10) candidate models resulted, which were compared in the second set of 18 cases, and the best model in terms of correlation coefficient was chosen. The GP-1 and GP-2 are presented, respectively in Eqs. 2 and 7. Their performances are depicted in Table 1 [4, 5].

$$U_C = A \cdot (B \cdot K + J) + 3.254 \quad (2)$$

where:

$$\begin{aligned} A =& 2.159 \cdot \frac{10^{-6}}{0.998 \cdot F + \log(1.908 \cdot C) + 7.182} \cdot (\log(0.531 \cdot L) - 13.224) \cdot \\ & (-0.741 \cdot F - \log(0.106 \cdot L) + 11.943) \end{aligned} \quad (3)$$

$$\begin{aligned} B =& (-0.604 \cdot D_m \cdot F + 0.016 \cdot H \cdot (2.267 \cdot L + 277.2) \\ & - 24.76 \cdot L \cdot (-\log(0.106 \cdot L) + \frac{\log(1.022 \cdot C)}{-0.741 \cdot F - \log(0.352 \cdot F) + 11.943}) \end{aligned} \quad (4)$$

$$K = 0.604 \cdot D_m \cdot F - 21.253 \cdot \log(1.908 \cdot C) - 21.253 \quad (5)$$

$$J = -151.67 \cdot L \cdot \left(0.604 \cdot D_m \cdot F - 21.253 \cdot \log(1.908 \cdot C) + 15.311\right) + \log(1.908 \cdot C) + \frac{\log(1.908 \cdot C)}{0.741 \cdot F - 24.76 \cdot L - 4.761} - \frac{6.813 \cdot 10^{-5}}{1.145 \cdot D_m + \frac{\log(1.908 \cdot C)}{0.083 \cdot C + 3020702}} \quad (6)$$

$$U_c = (\log(0.054012 \cdot C))^2 \cdot \left(1.298 \cdot L + 1.4616 \cdot D_m\right) \cdot 0.0053419 + 1.9647 \quad (7)$$

The **GSA model (GSA-1)** arose during the optimization effort of the performances of GP models. The GSA-1 (Eq. 8) uses Harmony Search for initialization, is based on GP-2 and its performance are depicted in Table 1 [6, 7].

$$U_c = E \cdot Z + G \quad (8)$$

where:

$$E = \log^2(0.0544783402089791 \cdot C) \quad (9)$$

$$Z = 0.0053419 \cdot \left(1.29462312655721 \cdot L + 1.46957461951861 \cdot D_m\right) \quad (10)$$

$$G = 1.95794336098125 \quad (11)$$

Finally, the **IDT model (IDT-1)** presented in this work, emerged after of several tests, which were related to the number of the data cases, the classification of used data cases (according to the values of U_c and with the modify of key operating parameters, such as the training-validation process. Specifically, for comparison reasons with the other AI approaches used in this paper, a 10-fold cross validation and no pruning was selected for the entire data set (i.e. 150 cases for training and 18 cases for testing the produced model in new, unseen cases) according of the classification of 5 kV of U_c (i.e. six classes ranging within [5,10) kV, [10,15) kV, [15,20) kV, [20,25) kV, [25,30) kV, [30,35] kV) [8–11].

The resulting decision tree contains 32 internal nodes and is translated into 30 different rules (the detailed outcome is not given in the paper due to space limitations). Seven of the rules produced, are handy and accurate as they cover a considerable number of cases from the entire data set and no negative classifications and the probability of correct classification of new unseen cases in the future is higher than 95%, i.e. statistical tests can take place for the verification of the acquired rule-based knowledge. In total, 13 rules are interesting ("cover" means how many cases from the dataset verify the produced rule and the number in brackets following each rule corresponds to the probability of correct classification of new cases in the future using the specific rule):

- Rule 1: (cover 41) IF L ≤ 40.6 AND C > 0.28 THEN class U_c =5–10 kV [0.977]
- Rule 2: (cover 32) IF F ≤ 0.9 AND C > 0.34 THEN class U_c =5–10 kV [0.971]
- Rule 3: (cover 30) IF H ≤ 17 AND C > 0.37 THEN class U_c =5–10 kV [0.969]
- Rule 4: (cover 25) IF L ≤ 33 AND C > 0.23 THEN class Uc =5–10 kV [0.963]
- Rule 5: (cover 23) IF D_m ≤ 25.4 AND L ≤ 33 AND C > 0.1 THEN class U_c = 5–10 kV [0.960]
- Rule 6: (cover 23) IF L > 33 AND L≤ 43.2 AND C > 0.1 AND C ≤ 0.28 THEN class U_c = 10–15 kV [0.960]
- Rule 7: (cover 21) IF H ≤ 17 AND L > 33 AND C > 0.13 AND C ≤ 0.28 THEN class U_c =10–15 kV [0.957]
- Rule 8: (cover 16) IF L> 33 AND C > 0.16 AND C ≤ 0.28 THEN class Uc =10–15 kV [0.944]
- Rule 9: (cover 10) IF L≤ 33 AND C > 0.06 AND C ≤ 0.16 THEN class Uc =10–15 kV [0.917]
- Rule 14: (cover 10) IF H≤17 AND C > 0.05 AND C ≤ 0.06 THEN class Uc =15–20 kV [0.917]
- Rule 15: (cover 10) IF H≤ 14.6 AND C > 0.02 AND C ≤ 0.06 THEN class Uc =15–20 kV [0.917]
- Rule 16: (cover 29) IF Dm≤ 28 AND C > 0.03 AND C ≤ 0.06 THEN class Uc =15–20 kV [0.903]
- Rule 20: (cover 8) IF H> 14.6 AND L<= 40 AND C ≤ 0.03 THEN class Uc =20–25 kV [0.900]

3 Comparative Results and Conclusions

In this paper four different intelligent techniques were employed to estimate the value of critical flashover voltage for polluted insulators. Results obtained cannot be compared always to exactly the same datasets and experimental conditions, but can be comparable in a somewhat fair manner, in order to draw general conclusions.

The results show that methods like ANNs and GP, which have inner procedures, not fully understandable by humans, produce accurate models. ANNs prove the strongest approach in terms of accuracy. Furthermore, GP may be programmed in such a way that can aim to shorter equations without any discount on accuracy. Its comprehensibility sometimes can be considered higher than that of the ANNs' structured, as standard mathematical formulas are some times more common to people. In addition, under proper encoding, grammar guided GP approaches can also produce fully comprehensible rule-based systems (IF-THEN rules or even Fuzzy Rule Based Systems). Models derived by GP-approaches can be optimized further by Nature Inspired Evolutionary Algorithms to give more accurate results, as it has been shown in this work. Inductive machine learning techniques presuppose the formation of reasonable decision classes when the target variable is of numerical nature (selecting a proper discretization). Its performance in terms of accuracy is lower, while there is need for larger collections of experimental data in order to be able to draw firm conclusions on the value of the technique. Nevertheless, experts find the acquired rules comprehensible and easy to use in the inspection process, while, as observed in the indicative results given in this paper, their comprehensibility is indeed high, in terms of measuring the number of resulting rules acquired and the conditions contained within these rules.

Further work includes experimentation to other competitive hybrid intelligent schemes based on nature inspired optimization approaches. The enrichment of the dataset is also a priority for the research team, as results seem encouraging.

References

1. Topalis, F.V., Gonos, I.F., Stathopulos, I.A.: Dielectric behaviour of polluted porcelain insulators. IEE Proc.-Gener. Transm. Distrib. **148**(4), 269–274 (2001)
2. Witten, I.H., Frank, E., Hall, M.A.: Data Mining: Practical Machine Learning Tools and Techniques, 3rd edn., pp. 233–244. Elsevier Inc. (2001)
3. Refaeilzadeh, P., Tang, L., Liu, H.: Cross-Validation. Arizona State University (2008)
4. Crane, E.F., McPhee, N.F.: The effects of size and depth limits on tree based genetic programming. In: Yu, T., Riolo, R., Worzel, B. (eds.) Genetic Programming Theory and Practice III. Genetic Programming, vol. 9, pp. 223–240. Springer, Boston (2006). doi: 10.1007/0-387-28111-8_15
5. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge (1992)
6. Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S.: GSA: a gravitational search algorithm. Inf. Sci. **179**(13), 2232–2248 (2009)
7. Geem, Z.W., Kim, J.H., Loganathan, G.V.: A new heuristic optimization algorithm: harmony search. Simulation **76**(2), 60–68 (2001)
8. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1994)
9. Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**(1), 81–106 (1986)
10. Quinlan, J.R.: Learning logical definitions from relations. Mach. Learn. **5**, 239–266 (1990)
11. Mitchell, T.M.: Machine Learning, pp. 55–58. McGraw-Hill (1997)