Integrated Computer-Aided Engineering 9 (2002) 37–56
IOS Press

A synergetic neural network-genetic scheme for optimal transformer construction

Nikolaos D. Doulamis[a,*], Anastasios D. Doulamis[a], Pavlos S. Georgilakis[b], Stefanos D. Kollias[a] and Nikos D. Hatziargyriou[c]
[a]*Digital Signal Processing Laboratory, Department of Electrical and Computer Engineering, National Technical University of Athens, Greece*
[b]*R&D Department, Elvim Plant, Schneider Electric, Greece*
[c]*Electric Energy Systems Laboratory, Department of Electrical and Computer Engineering, National Technical University of Athens, Greece*

Abstract: In this paper, a combined neural network and an evolutionary programming scheme is proposed to improve the quality of wound core distribution transformers in an industrial environment by exploiting information derived from both the construction and transformer design phase. In particular, the neural network architecture is responsible for predicting transformer iron losses prior to their assembly, based on several actual core measurements, transformer design parameters and the specific core assembling. A genetic algorithm is applied to estimate the optimal core arrangement, (i.e. the way of core assembling) that yields a set of three-phase transformers of minimal iron losses. The minimization is performed by exploiting information derived from the neural network model resulting in a synergetic neural network-genetic algorithm scheme. After the transformer construction, the prediction accuracy of the neural network model is evaluated. If accuracy is poor, a weight adaptation algorithm is applied to improve the prediction performance. For the weight updating, both the current and the previous network knowledge are taken into account. Application of the proposed neural network-genetic algorithm scheme to our industrial environment indicates a significant reduction in the variation between the actual and the designed transformer iron losses. This further leads to a reduction of the production cost since a smaller safety margin can be used for the transformer design.

1. Introduction

Construction of three-phase distribution transformers of high quality at a minimum possible cost is an important key for any transformer manufacturing industry to face today's market competition [28]. As one measure of transformer quality, the iron losses (core losses) are used. In particular, the higher the transformer quality is, the less the transformer losses become [5,20]. Usually, the electric power suppliers, (i.e. the customers), determine the transformer quality by specifying an upper iron loss limit. In case that the transformer iron losses violate this limit, the manufacturer is forced to pay loss penalties. This, however, apart from reducing the manufacturer's profit, it also ruins the good reputation of the industry. In order to satisfy the customers' requirements, the transformers are first designed and several parameters are selected so that the transformer iron losses are equal to or slightly smaller than those specified by the customers (guaranteed losses).

Core losses consist of eddy current losses and hysteresis losses [1]. For a given volume of core, the eddy current losses are nearly proportional to the square of the product of power frequency, flux density and lamination thickness. Regarding the hysteresis losses, they are associated with the magnetization and demagnetization cycle of the core [22]. Thus, eddy current losses can be reduced using, for example, thin laminations of high-resistivity material, while reduction of hysteresis

[*]Corresponding author : Nikolaos Doulamis, Department of Electrical and Computer Engineering Computer Science Division National Technical University of Athens, Greece, 9, Heroon Polytechniou Str., 157 73, Zografou, Athens, Greece. Tel.: +30 1 772 30 39; Fax: +30 1 772 24 92; E-mail: ndoulam@cs.ntua.gr.

losses is achieved using a magnetic material of smaller hysteresis loop [22]. Although, in theory, this is a sufficient condition to meet customer's requirements, in practice, additional parameters, such as the quality of cores that assemble the transformer, are involved during the construction process, which deviate the actual losses from the designed ones. Consequently, it is possible for the transformer iron losses to exceed the guaranteed losses.

Most works, reported in the literature, have been concentrated on the estimation of transformer losses in a theoretical basis. These techniques are mainly focused on the analysis of the core electromagnetic field. In particular, in [18] the 3D-leakage fields are estimated, while in [11] 3D magnetic-field calculations are used to evaluate the performance of several parameters on power transformers. Other approaches are based on the method of finite elements either to compute the core loss currents [6] or to estimate the loss distribution in case of stacked transformers [21]. Apart from the above mentioned works, which are mainly based on the arithmetic analysis of the core electromagnetic field, modeling of three-phase transformers has been also proposed based on the analysis of their magnetic circuits [26].

Other approaches used to improve the calculation accuracy of transformer iron losses are based on experimental observations. In particular, in [24,27] linear or simple non-linear models have been proposed to relate transformer iron losses with electromagnetic and geometric properties. Furthermore, in [14] estimation of transformer losses is performed based on experimental curves. However, these approaches present satisfactory performance only for data (transformers) on which the model parameters have been estimated. The model accuracy deteriorates in case of data that are not included in the "training set". This is due to the fact that there is not a simple and analytical relationship expressing the impact of the aforementioned parameters on the transformer performance, as the previous models assume.

Regardless of the method adopted, theoretical or experimental, and despite the model accuracy, in all the aforementioned approaches the reduction of iron losses is achieved through better transformer design. For this reason, all the above-mentioned models provide one prediction value for the iron losses of all the transformers of the same production batch. Unfortunately, this is never verified in an actual industrial environment due to constructional defects. More specifically, it has been found that the maximum divergence between the theoretical and the actual iron losses could be up to $\pm$ 10% for a given production batch [7,9].

In this paper, the improvement of transformer quality is addressed through an original point of view; the improvement is achieved during the transformer production phase and not during the transformer design phase. Our work is concentrated on wound core distribution transformers. In particular, the technique used aims at estimating the optimal core arrangement for a given design so that transformers of best quality (minimal losses) are constructed. In fact, the proposed method compensates the effect of constructional defects on transformer iron losses by appropriately combining the individual cores. Such an approach reduces the deviation between the actual and the theoretical iron losses and thus yields losses closer to the designed ones. For this reason, transformers can be designed using a higher safety margin since it is less possible for their iron losses to exceed the guaranteed losses, which leads to a reduction of the total production cost.

The optimal core arrangement is estimated in this paper through a combined neural network-genetic algorithm scheme. The neural network architecture is used to accurately estimate the transformer losses, while the genetic algorithm exploits the neural network output to determine that core arrangement of individual cores, which produces a set of three-phase transformers of optimal quality, (i.e. of minimal losses). Finally, the prediction accuracy of the neural network model is evaluated, after the transformer construction. In case that the prediction accuracy is poor, a weight adaptation is activated to improve the network performance. For the weight updating, a novel algorithm is proposed in this paper so that (i) the network output is approximately equal to the current iron losses and (ii) a minimal degradation over the previous network knowledge is provided. The proposed scheme has been implemented to a transformer manufacturing industry for increasing the quality of the produced transformers.

This paper is organized as follows: Section 2 presents the problem formulation and describes the techniques used in the current practice for transformer design as well as core assembling. The neural network architecture used to predict transformer iron losses is presented in Section 3, while Section 4 explains the weight adaptation mechanism. The genetic algorithm-based grouping method is discussed in Section 5 to estimate the optimal core arrangement that generates transformers of minimal losses. Application of the proposed neural network-genetic algorithm scheme to our industrial environment is described in Section 6, while Section 7 concludes the paper.

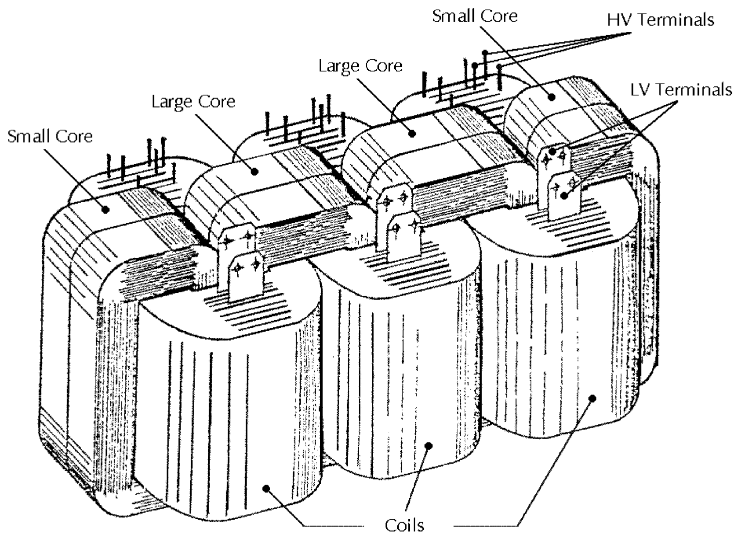


Fig. 1. A graphical representation of a wound core transformer.

2. Problem description

A three-phase wound core distribution transformer consists of two small and two large individual cores. The difference between small and large cores is the width. In general, the width of a small core is the half of the width of a large core. The order that these four cores are arranged to construct a three-phase transformer is shown in Fig. 1. Particularly, the order is; a small core, followed by two large cores, followed by another small core [2]. Given a fixed number of possible small and large cores, we can determine the total number of the transformers that may be constructed. In particular, given $2 * M$ small cores and $2 * M$ large cores, we can construct exactly M different transformers at the same time. Let us denote as t_i the ith transformer among all M possible ($i = 1, 2, \dots, M$). Let us also denote as s_i, $i = 1, 2, \dots, 2 * M$, the ith small individual core, while as $l_i, i = 1, 2, \dots, 2 * M$, the ith large core. Assuming that the transformer t_i, consists of the s_p and the s_q small core (with p /q, $p, q \in \{1, 2, \dots, M\}$) and the l_m and the l_n large core (with $m \neq n,\ n, m \in \{1, 2, \dots, M\}$), can be written in a vector form as follows

$$t_i = [s_p l_m l_n s_q]^T \tag{1}$$

where in the previous equation we have taken into account the pre-determined core arrangement as depicted in Fig. 1.

2.1. Transformer design

The transformers are designed so that the theoretical iron losses, say $P^t_{t_i}$, satisfy the following equation,

$$P^t_{t_i} \approx (1 - \gamma) P_0 \tag{2}$$

where P_0 are the iron losses specified by the customers, while $0 < \gamma < 1$ a safety margin. A typical value for safety margin γ is 0.05. It should be mentioned that all transformers of the same design present the same theoretical iron losses.

Based on Eq. (2) and using the transformer geometry and the typical loss curve, the theoretical iron losses of the individual cores as well as the losses of the assembled transformer are estimated [9,12]. However, the actual core losses deviate from the theoretical ones due to constructional defects, producing cores of lower or higher quality than the designed one. Therefore if a random core arrangement is adopted, it is probable to assemble four low-quality cores together, generating transformers of low quality. To avoid this, we use a core grouping process, so that cores of low-quality are combined with cores of high-quality to produce three-phase transformers of iron losses close to the designed ones.

2.2. The current core grouping methods

Two different approaches are used in the current practice to perform the core grouping. In the first approach, the cores are classified into "quality classes" according to the deviation of their actual iron losses from the theoretical ones (quality class method). For example, let us assume that three quality classes are available, named as A, B and C respectively. The class A contains all small or large cores whose the quality (iron losses) is around the designed one. On the contrary, cores of lower quality than the cores of the class A are classified into class B, while cores of higher quality into class C. Then, a transformer is constructed by assembling either a) one small core and one large core of class B and one small core and one large core of class C or b) four cores of class A.

However, such an approach assigns the same quality grades to all cores belonging to the same quality class. For this reason, another grouping process is also used in the current practice based on the minimal deviation of the core losses (minimal deviation method) [8]. In both approaches as a measure of the transformer quality the sum of the losses of the four individual cores is used, that is

$$F^a_{t_i} = P^a_{s_p} + P^a_{s_q} + P^a_{l_m} + P^a_{l_n} \quad \text{and} \tag{3a}$$

$$\begin{aligned} F^t_{t_i} &= P^t_{s_p} + P^t_{s_q} + P^t_{l_m} + P^t_{l_n} \\ &= 2 * (P^t_s + P^t_l) \end{aligned} \tag{3b}$$

where Eq. (3a) refers to the actual losses, while Eq. (3b) to the theoretical ones. In Eq. (3), $P^a_{s_p}$ and $P^a_{s_q}$ ($P^t_{s_p}$ and $P^t_{s_q}$) denotes the actual (theoretical) losses of the small cores s_p and s_q of transformer t_i [see Eq. (1)]. Similarly, $P^a_{l_m}$ and $P^a_{l_n}$ ($P^t_{l_m}$ and P_{l_n}) are the actual (theoretical) losses of the large cores l_m of t_i.

However, the transformer losses, both the actual and the theoretical ones, $P^a_{t_i}$ and $P^t_{t_i}$ respectively, diverge from $f^a_{t_i}$ and $F^t_{t_i}$, since additional losses in general appear during the core assembling [2]. In particular, $P^a_{t_i} > F^a_{t_i}$, and $P^t_{t_i} > F^t_{t_i}$. For example, reordering the two small or the two large cores of a transformer, results in different actual iron losses, (i.e. $P^a_{t_i}$), though the average losses of the four cores, (i.e. $F^a_{t_i}$), remain the same. For this reason, the current grouping methods do not provide the optimal arrangement of all available $2 * M$ small and $2 * M$ large cores so that transformers of minimal losses are constructed. To overcome the aforementioned difficulties, a novel synergetic neural network-genetic algorithm scheme is adopted in this paper.

2.3. *The proposed scheme*

The block diagram of the proposed scheme is illustrated in Fig. 2. Initially, the transformers are designed to satisfy, at least theoretically, the customers' requirements (with perhaps a safety margin) and then, the individual cores are constructed and several measurements are taken for each core to determine its quality. These measurements and several transformer design parameters along with the specified core arrangement are fed as inputs in a neural network structure, responsible for predicting the actual transformer losses. In the following, the optimal core arrangement is estimated, which produces transformers of the best quality using a genetic algorithm grouping scheme. After the transformer construction, the prediction accuracy of the network model is monitored (evaluation phase). If the prediction accuracy is poor, a weight adaptation algorithm is activated to estimate new network weights which are then used to predict future samples (transformers) of the process. Otherwise, the same weights are used as shown in Fig. 2.

3. Transformer iron loss modeling

The neural network architecture used for predicting the transformer iron losses is described in this section. Let us gather in the vector $x(t_i)$ all network features of transformer t_i. Let us also denote as $y_w(x(t_i))$ the network output in case that $x(t_i)$ is the network input. The subscript w indicates the dependence of the network output on its weights and biases. In our case, we consider that the network approximates the actual specific iron losses of the transformer t_i, (i.e. the normalized losses per weight unit). Particularly, it is held that

$$S^a_{t_i} = P^a_{t_i}/K_{t_i} \tag{4}$$

where $S^a_{t_i}$ corresponds to the actual specific iron losses of the transformer t_i, and K_{t_i} to its actual core weight.

The neural network models the non-linear relationship, say $g(\cdot)$, between the input features $x(t_i)$ and the specific iron losses $S^a_{t_i}$,

$$S^a_{t_i} = g(x(t_i)) \tag{5}$$

However, under different production conditions, (i.e. different thickness, grade and/or supplier of the magnetic material), different non-linear relationship exists. This is due to the fact that different type of magnetic material has a different effect on the transformer iron

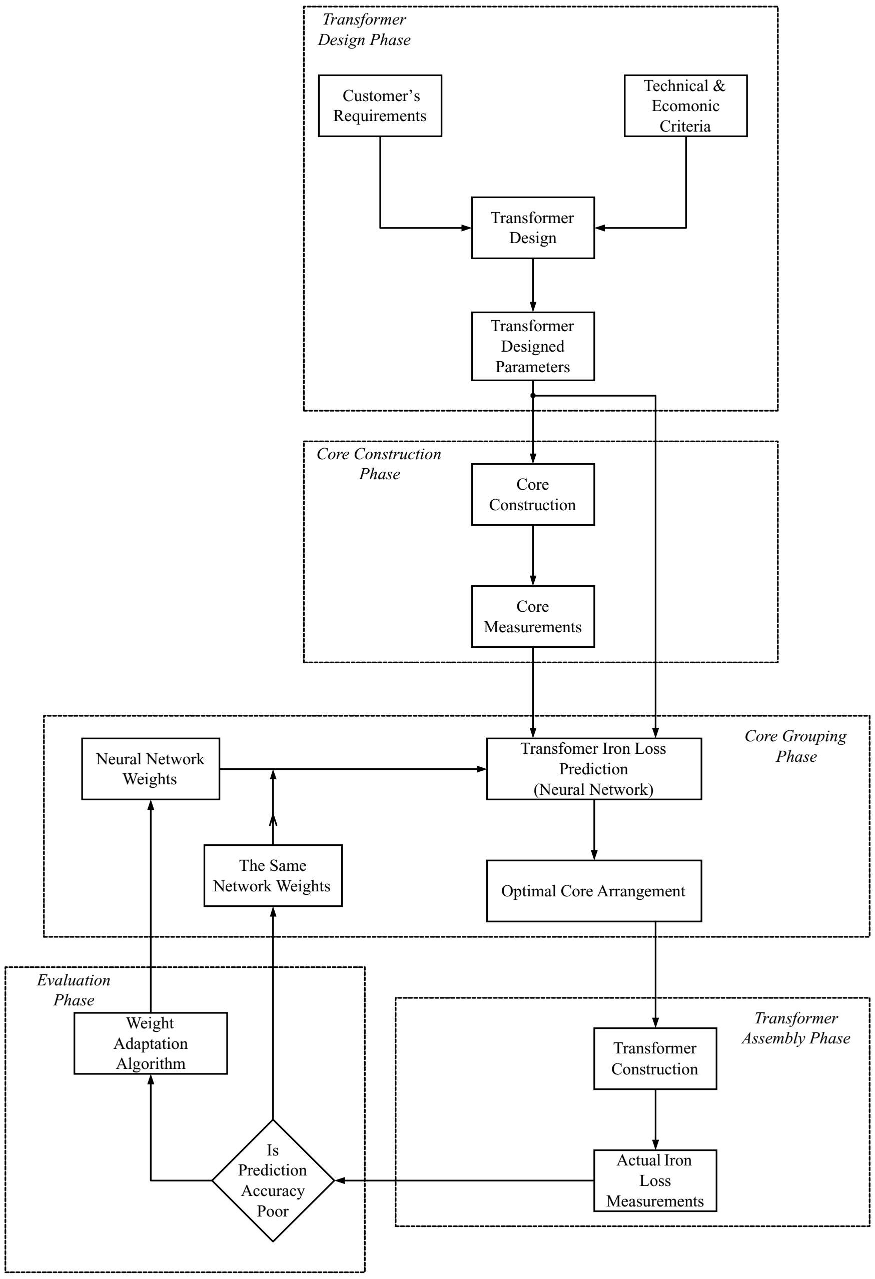


Fig. 2. Block diagram of the proposed architecture.

losses, while different suppliers usually follow a specific technology for producing the magnetic material. Assuming that L different production conditions are available, then, L different neural networks are used, each of which corresponds to a specific condition. Selection of the most appropriate condition is performed during the design phase, where both the type of the magnetic material and the respective supplier are determined. In the following analysis, the iron loss modeling is performed for a given production condition, since similar conclusions can be drawn for the other conditions too.

A feedforward neural network architecture is adopted in this paper to approximate the unknown function $g(\cdot)$ [13]. Figure 3 illustrates such a network, consisting of one hidden layer of l neurons, one output neuron and

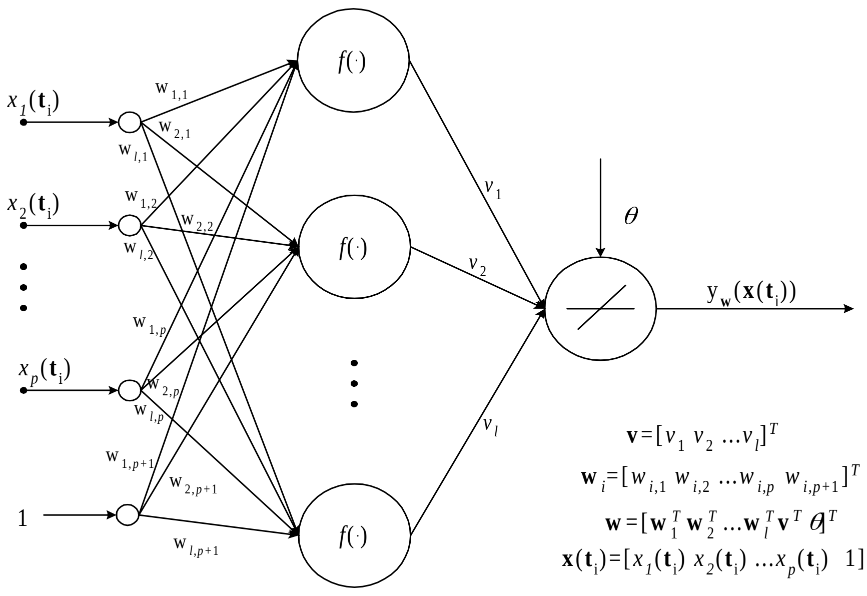

$x_1(\mathbf{t}_i)$
$x_2(\mathbf{t}_i)$
$x_p(\mathbf{t}_i)$
1
$w_{1,1}$
$w_{2,1}$
$w_{l,1}$
$w_{1,2}$
$w_{2,2}$
$w_{l,2}$
$w_{1,p}$
$w_{2,p}$
$w_{l,p}$
$w_{1,p+1}$
$w_{2,p+1}$
$w_{l,p+1}$
$f(\cdot)$
v_1
v_2
v_l
θ
$y_w(\mathbf{x}(\mathbf{t}_i))$
$\mathbf{v}=[v_1\ v_2\ ...v_l]^T$
$\mathbf{w}_i=[w_{i,1}\ w_{i,2}\ ...w_{i,p}\ w_{i,p+1}]^T$
$\mathbf{w}=[\mathbf{w}_1^T\ \mathbf{w}_2^T ...\mathbf{w}_l^T\mathbf{v}^T\ \theta]^T$
$\mathbf{x}(\mathbf{t}_i)=[x_1(\mathbf{t}_i)\ x_2(\mathbf{t}_i)\ ...x_p(\mathbf{t}_i)\ \ 1]^T$

Fig. 3. The feedforward neural network used to predict transformer losses.

p input elements. Let us denote by $w_{i,k}$, $i = 1, \dots, l$, $k = 1, \dots, p$ the weights, which connect the ith hidden neuron to the kth input element and as $w_{i,p+1}$ the bias of the ith hidden neuron. These weights and biases are also illustrated in Fig. 3 for clarity. Then, we form the $(p+1) \times 1$ vectors $w_i[w_{i,1} \dots w_{i,p+1}]^T$, $i = 1, 2, \dots, l$ containing all the weights and biases of the ith hidden neuron. In this case, the output of all hidden neurons can be written in a matrix form as follows

$$u(x(t_i)) = \begin{bmatrix} u_1(x(t_i)) \\ \vdots \\ u_l(x(t_i)) \end{bmatrix} = \begin{bmatrix} f(w_1^T \cdot x(t_i)) \\ \vdots \\ f(w_l^T \cdot x(t_i)) \end{bmatrix} = f(W^T \cdot x(t_i)) \quad (6)$$

where $u(x(t_i)) = [u_1(x(t_i)) \dots u_l(x(t_i))]^T$ is a vector containing the outputs of all hidden neurons, $u_k(x(t_i)), k = 1, \dots, l$, in case that $x(t_i)$ is the network input. In Eq. (6), W denotes a $(p+1) \times l$ matrix, the columns of which correspond to the weight vector w_i that is $W = [w_1 w_2 \dots w_l]$, while $f(\cdot)$ is a vector-valued function, the elements of which correspond to the activation function $f(\cdot)$, of the hidden neurons. In our case, the sigmoid function has been selected as the activation function of all hidden neurons of the network.

Let us also define by $v = [v_1 v_2 \dots v_l]^T$ an $l \times 1$ vector, which contains the network weights, say v_i, connecting the ith hidden neuron to the output neuron. Let us also define as θ the bias of the output neuron. Then, vector $w = [w_1^T w_2^T \dots w_l^T v^T \theta]^T$ represents all network weights and biases. The network output, which provides a non-linear approximation, say $\hat{g}(\cdot)$ of function $g(\cdot)$ and thus an estimate, $\hat{S}^a_{t_i}$ of the specific iron losses $S^a_{t_i}$, is given by the following equation

$$y_w(x(t_i)) \equiv \hat{S}^a_{t_i} = v^T \cdot u(x(t_i)) + \theta \quad (7)$$

where, in this case, a linear activation function has been used for the output neuron, since the network output approximates a continuous valued signal, (i.e. the specific iron losses of a transformer).

A training set of N samples (transformers), say T, is used to estimate the network weights w. The training set, $T = \left\{ \left(x(t_1), S^a_{t_1}\right), \dots, \left(x(t_N), S^a_{t_N}\right) \right\}$, contains the actual specific iron losses of N transformers, which have been constructed under the same production condition, along with the respective network features. The network is trained to minimize the mean squared value of the error over all samples of set T

$$C = \frac{1}{2} \sum_{i=1}^{N} \left\{ S^a_{t_i} - y_w)(x(t_i)) \right\}^2 \quad (8)$$

In our approach, a second order method has been selected to train the network based on a modification of the Marquardt-Levenberg algorithm, as presented in [17]. This algorithm has been selected due to its efficiency and fast convergence. Furthermore, a cross validation method has been incorporated in the proposed scheme to improve the generalization performance of the network [13].

Table 1
Features selected by the PCA, which are used as inputs to the neural network architecture

Symbol	Expression
x_1	B
x_2	$(U_{s_p} + U_{l_m} + U_{l_n} + U_{s_q})/4$
x_3	$(V_{s_p} + V_{l_m} + V_{l_n} + V_{s_q})/4$
x_4	$K^a_{t_i}/K^t_{t_i}$
x_5	$F^a_{t_i}/F^t_{t_i}$
x_6	$\left(S^a_{s_p} + S^a_{l_m}\right) / \left(S^t_{s_p} + S^t_{l_m}\right)$
x_7	$\left(S^a_{l_m} + S^a_{l_n}\right) / \left(S^t_{l_m} + S^t_{l_n}\right)$
x_8	$\left(S^a_{l_n} + S^a_{s_q}\right) / \left(S^t_{l_n} + S^t_{s_q}\right)$

A principal component analysis (PCA) has been also included to appropriately select the input features. Since most of the measurements obtained are highly correlated, it is useful for improving the network training and generalization performance to discard the redundant information. Principal component analysis eliminates those input components, which contribute the least to the network output, while retaining the most important (principal) ones. To implement the PCA, a neural network architecture similar to that proposed in [25] has been adopted in this paper. The number of the principal components extracted is estimated through the amount of energy that those components can retain. In particular, in our case, the components, which correspond to 85% of the total energy (8 in our case), are selected as the most significant.

In Table 1, we summarize the features, which are closer to that specified by the PCA to provide a physical explanation of the most important features used. More specifically, feature x_1 is the rated magnetic induction, which is also used in order to calculate iron losses at the design phase by using the loss curve (design parameters). This is also denoted as B in this table. Features x_2 and x_3 express the average specific losses (W/Kg at 15000 Gauss, and 17000 Gauss, respectively) of magnetic material of the four individual cores used for transformer construction. These parameters are also denoted as U and V respectively in the table. The subscripts of these variables refer to one of the four cores that assemble the transformer t_i similarly to Eq. (1). For example, U_{s_p} denotes average specific losses of magnetic material at 1500 Gauss for the left small core (s_p), while U_{l_n} refers to the second (right) large core (l_n). Feature x_4 is the ratio of actual $(K^a_{t_i})$ over theoretical weight $(K^t_{t_i})$ of the four individual cores that assemble t_i. Feature x_5 is equal to the ratio of actual $(F^a_{t_i})$ over theoretical $(F^t_{t_i})$ iron losses of the four individual cores of t_i. The significance of the feature x_5 is that the iron losses of the three-phase transformer depend on the iron losses of its individual cores. In the industrial environment considered, it is observed that the arrangement of cores influences the assembled transformer core losses. This is reflected through the selection of features x_6, x_7, and x_8 (see Table 1). In all cases, the notation for the four cores is similar to Eq. (1). For instance, $S^a_{s_p}$ refers to the actual specific iron loss of the s_p small core (i.e. the first one), while $S^t_{s_p}$ to the theoretical ones.

4. Network weight adaptation

The weight adaptation training algorithm modifies the network weights so that the network output appropriately responses to the new data, while simultaneously providing a minimal degradation over the old information [4]. Particularly, let us denote by w_b the network weights before the weight adaptation. Let us assume that these weights have been estimated using data of set T. Let us also assume that during the evaluation phase, m transformers have been examined and that their actual specific iron losses significantly deviate from the predicted ones. These transformers are denoted as z_i, $i = 1, \ldots, m$, where vector z has the same form as vector t (Eq. (1)). In this case, a new training set, $T_c = \left\{\left(x(z_1), S^a_{z_1}\right), \ldots, \left(x(z_m), S^a_{z_m}\right)\right\}$,, is created consisting of pairs of the features $x(z_i)$ for the transformers z_i along with the respective actual specific iron losses $S^a_{z_i}$. Then, the new network weights, say w_a, are estimated by minimizing the following equation,

$$\begin{aligned} w_a &= \arg\min_{w} \frac{1}{2} \sum_{i=1}^{N} \left(y_w(t_i) - S^a_{t_i}\right)^2 \\ &= \frac{1}{2} \sum_{i=1}^{N} D_i \end{aligned} \tag{9a}$$

subject to

$$y_{w_a}(x(z_i)) \approx S^a_{z_i} \quad \text{for } i = 1, \ldots, m \tag{9b}$$

where $D_i = \left(y_w(t_i) - S^a_{t_i}\right)^2$. Equation (9a) indicates that the network weights are adapted in such a way that a minimal degradation over the existing old information, as that expressed by the set T, is accomplished. The minimization constraint [Eq. (9b)] means that the network output, after the weight adaptation, is approximately equal to the actual specific iron losses for all transformers z_i of T_c.

Assuming that a small weight perturbation is sufficient to perform the weight modification, we can relate the weights before and after the adaptation as follows

$$w_a = w_b + \Delta w \tag{10}$$

where Δw represents small increments of the network weights.

The effect of Δw in Eq. (9) can be expressed by estimating the sensitivity of errors D_i with respect to the network weights and by linearizing Eq. (9b) using a first order Taylor series expansion. In this case, Eq. (9) yields to the following constraint minimization

$$\min \frac{1}{2}\Delta w^T \cdot J^T \cdot J \cdot \Delta w \tag{11a}$$

subject to

$$b = A \cdot \Delta w \tag{11b}$$

where matrix J corresponds to the Jacobian matrix of the errors D_i, while vector b and matrix A are expressed in terms of the previous weights, w_b. More details about the form of J, A and b can be found in the appendices A and B.

The gradient projection method is used in our case to minimize Eq. (11a) subject to Eq. (11b) [19]. In particular, the algorithm starts from an initial feasible point [i.e. a point which satisfies Eq. (11b)], say $\Delta w(0)$ and then the weight increments are updated as follows

$$\Delta w(n+1) = \Delta w(n) + \gamma(n)h(n) \tag{12}$$

where $\gamma(n)$ is a scalar that determines the convergence rate, while $h(n)$ indicates the negative gradient of Eq. (11a) onto the subspace that is tangent to the surface defined by the constraint at the nth iteration of the algorithm,

$$h(n) = -P \cdot J \cdot \Delta w(n) \tag{13}$$

$$\text{with} \quad P = I - A^T(A \cdot A^T)^{-1}A \tag{14}$$

As initial feasible point $\Delta w(0)$, the minimal distance from the origin to the surface $b - A \cdot \Delta w = 0$ is selected, given by

$$\Delta w(0) = A^T \cdot (A \cdot A^T)^{-1} \cdot b \tag{15}$$

After the weight adaptation, the old information, (i.e. the training set T), is updated by inserting in it all pairs of set T_c. However, in this case, the size of T continuously increases, since more and more pairs are included. For a more efficient implementation, the number of pairs of T can be considered constant and thus each time new pairs get into the set T, the oldest ones are removed from it.

5. Optimal core arrangement

The aforementioned neural network architecture predicts the transformer iron losses, for a given production condition, based on several core measurements, transformer design parameters and taking into account a given core arrangement and core assembling. This means that for different cores and/or different core arrangement, the network provides different predicted iron losses. Exploiting this information, we can estimate, for the given $2 * M$ small and $2 * M$ large available cores, that core arrangement which produces M transformers of the highest quality (minimum losses).

Let us denote as c a vector containing all the M possible constructed transformers

$$c = \text{vec}\{T^T\} \tag{16a}$$

with

$$T = [t_1 t_2 \dots t_M] \tag{16b}$$

where vec$\{T\}$ presents a vector formed by stacking up all rows of matrix T.

As observed from Eq. (16a), the dimension of c is $4M \times 1$ since each transformer t_i is represented by a 4×1 vector as Eq. (1) indicates (four individual cores). Vector c provides a possible arrangement (combination) of all small and large cores, which construct the M three-phase transformers. Therefore, different elements of c correspond to different arrangement of individual cores. Based on the above definition, it is clear that searching for a set of M transformers of the highest quality is equivalent to searching for a vector c that minimizes the following equation

$$c_{\text{opt}} = \arg\min_c E(c) = \arg\min_c \left\{ \sum_{i=1}^{M} P^a_{t_i} \right\} \tag{17}$$

where c_{opt} is the vector corresponding to the optimal core arrangement.

In Eq. (17), the actual transformer iron losses, $P^a_{t_i}$, are involved, which are in fact, unknown before the transformer construction. For this reason, for the minimization of Eq. (17), an approximation of $P^a_{t_i}$ is used based on the aforementioned neural network model. As mentioned in Section 3, the neural network estimates the specific iron losses, $S^a_{t_i}$ instead of $P^a_{t_i}$. Consequently, an estimate of the actual losses, $P^a_{t_i}$, is obtained by multiplying the network output (specific losses) by the actual weights of the four cores that assemble the transformer t_i. Furthermore, to avoid the risk of violating the customers' requirements, a very large value is assigned to the predicted losses that lead to trans-

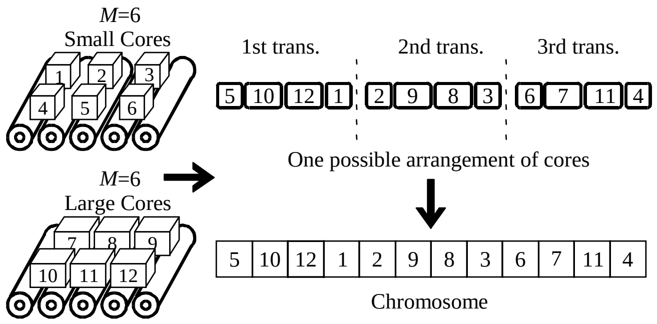


Fig. 4. An example of the genetic algorithm representation.

formers of unacceptable quality, (i.e. their losses are somehow greater than P_0), with perhaps a safety margin estimated based on the relative prediction error of the network.

For a typical number of small/large cores, direct minimization of Eq. (17) is practically infeasible since the computational complexity for an exhaustive search is very high. For example, assuming that 100 small and 100 large individual cores are available, about 5.35*10^{22} combinations of core arrangements should be considered! For this reason, the minimization process is performed, in the following, using a genetic algorithm (GA) [23], resulting in synergetic neural network-genetic algorithm scheme for optimal transformer construction.

5.1. Genetic algorithm

In the GA approach, the vector c is considered as one chromosome, while its elements, (i.e. the serial numbers of individual cores), as genetic material of the respective chromosome. Figure 4 presents an example of such representation in case of six large and six small cores. Initially, a population of L chromosomes is created, say $P(0) = \{c_1(0), \ldots, c_L(0)\}$ where $c_i(0), i = 1, \ldots, L$ corresponds to the ith chromosome of the population $P(0)$. Although, in theory, the initial population can be randomly selected, fast convergence is achieved in case that the genetic material of the initial chromosomes is of somehow good quality. For this reason, in our case the L initial chromosomes are selected as possible solutions of the core grouping method used in the current practice.

The performance of each chromosome, which actually represents a particular core arrangement, is evaluated as the sum of the predicted actual iron losses, (provided by the neural network model) over all transformers that can be constructed by this particular chromosome (vector c). Since this measure is inverse proportional to the transformer quality, the fitness function of the genetic algorithm, which is responsible for measuring the chromosome quality, is given by following equation.

$$F(c_i(n)) = B - E(c_i(n)) \qquad (18)$$

where we recall that $E(c_i(n))$ denotes the sum of the predicted iron losses over all transformers constructed using the core arrangement of the chromosome $c_i(n)$. The constant B in Eq. (18) is selected so that negative values of the fitness function are avoided.

Based on $F(c_i(n))$ for all chromosomes $c_i(n), i = 1, 2, \ldots, L$ of the current population $P(n)$, appropriate "parents" are selected so that a fitter chromosome has a higher chance of survival in next generation. In particular, in our case, a probability is assigned to each chromosome, equal to $F(c_i(n))/\sum_{i=1}^{L} F(c_i(n))$ and then $Q < L$ chromosomes are randomly selected based on their assigned probabilities as candidate parents (roulette wheel selection procedure [10]).

A set of new chromosomes (offspring) is then produced by mating the genetic material of the parents using a crossover operator, which defines how the genes should be exchanged to produce the next generation. Several crossover mechanisms have been reported in the literature. In our approach, a modification of the uniform crossover operator [10,23] has been adopted. This is due to the fact that it is possible for an individual core to appear more than once in the genetic material of each new generated chromosome. This means that one individual core is placed to more than one transformer or to more than one position of the same transformer, which corresponds to an unacceptable core arrangement. For this reason, the following modification of the uniform crossover operator is adopted and explained using a simple example of six small and six large cores (Fig. 5). In this example, it is assumed that the two

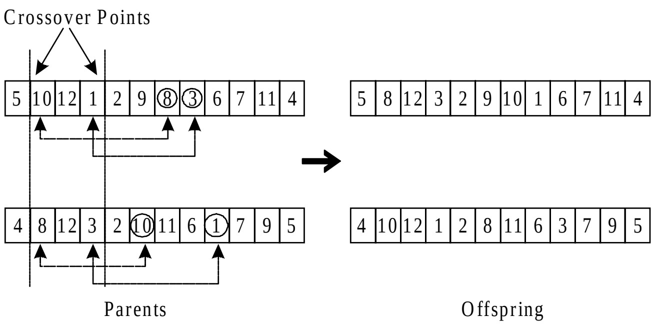


Fig. 5. An example of the crossover operator used.

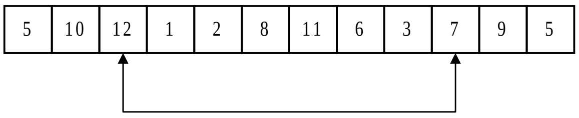


Fig. 6. An example of the mutation mechanism used.

parents exchange their genes between the crossover points 2, 3 and 4. As observed, the genes $\{10,12,1\}$ of the 1st parent are exchanged with the genes $\{8,12,3\}$ of the 2nd parent. By applying this exchange of genes, in the 1st chromosome, the genes 8 and 3 appear twice, while genes 10 and 1 disappear. An equivalent problem occurs in the 2nd chromosome. For this reason, in the 1st chromosome, the genes $\{10,1\}$ are one-by-one exchanged with genes $\{8,3\}$ as Fig. 5 depicts. The same happens for the 2nd chromosome.

The next step of the algorithm is to apply mutation to the newly created chromosomes, introducing random gene variations that are useful for restoring lost genetic material, or for producing new material that corresponds to new search areas. In our case, the genetic material of a chromosome is randomly mutated with a probability p_m. In particular, for each gene, a random number is generated uniformly distributed in the interval [0 1] and if this number is smaller than the mutation rate p_m, then this gene undergoes mutation. To avoid the appearance of an individual core more than once in the genetic string after mutation, (i.e. to preserve valid core arrangement), we swap each mutated gene for other randomly selected of the same category, (i.e. large or small core). This is illustrated in Fig. 6. In this figure, we assume that the large core with label '12', (the third gene) undergoes mutation. Thus, it is swapped for another randomly selected gene of the same category, for example the large core with label '7' in our case.

After that, the next population $P(n+1)$ is created by inserting the new chromosomes and deleting the older ones. Several GA cycles take place by repeating the procedures of fitness evaluation, parent selection, crossover and mutation, until the population converges to an optimal solution.

5.2. Explanation of the GA convergence

As can be seen in the following section of the experimental results, the proposed genetic algorithm scheme sufficiently estimates the optimal core arrangement so that transformers of minimal losses (best quality) are constructed, even in case that a large number of cores (variables) are involved in the minimization process. This is due to the fact that the variables (individual cores) do not take any arbitrary value. Instead, the cores are designed so that Eq. (2) is satisfied and thus they present the same theoretical losses. Divergence from this requirement is due to constructional defects during the core production phase. Assuming that the industry follows specified constructional standards, this divergence is not so large. Consequently, the form of Eq. (17) is not so complex as is expected from the number of variables (large/small cores) involved in the process. Another issue towards this direction is that individual cores are usually produced in groups. Therefore, it is expected that for cores belonging to the same group to present similar characteristics (losses) since they undergo almost the same constructional effects.

In addition, as initial population of the genetic algorithm, chromosomes obtained from the current grouping method are used. This selection further improves the GA convergence since the algorithm starts from

Table 2
The neural network training and configurations details

	Production conditions		
Neural network (MLP) training and configuration details	#1	#2	#3
Total number of measurement sets (MS)	2240	2350	1980
Number of MS of the training set (75% of the total MS)	1680	1762	1485
Number of MS of the validation set (10% of the training set)	168	235	198
Number of MS of the test set (25% of the total MS)	560	588	495
Slope of the sigmoid (activation) function	1	1	1
Number of neurons of the input layer	8	8	8
Number of neurons of the hidden layer	5	6	4
Number of neurons of the output layer	1	1	1

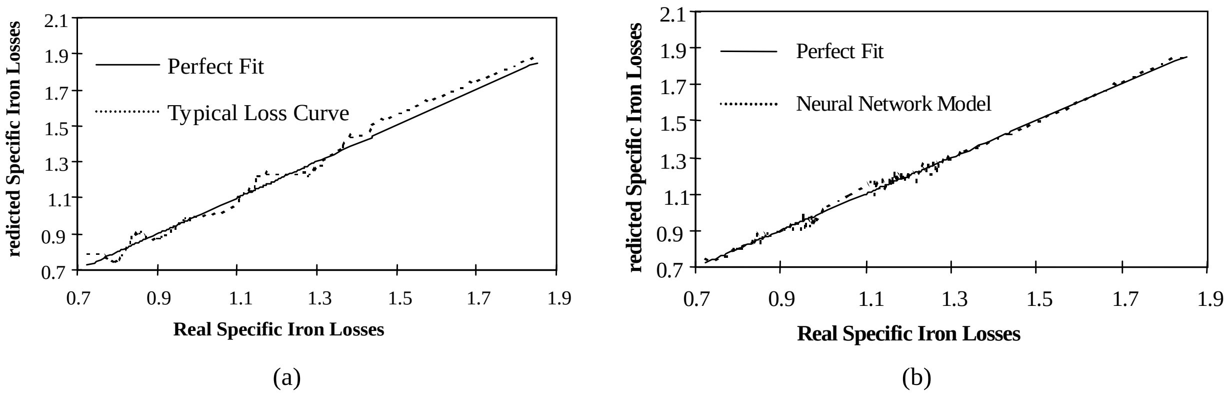


Fig. 7. Prediction performance of transformer specific iron losses for the first production condition, (a) using the typical loss curve (Current Practice), (b) using the proposed neural network model.

chromosomes of somehow good quality. This selection also leads to construction of transformers of better quality (i.e. less losses) than those obtained from the current grouping method. The GA improves the performance of the initial chromosomes, which represent the core arrangements of the current grouping.

6. Experimental results

In our industrial environment, three different production conditions have been examined. The first corresponds to a magnetic steel of grade M3, according to the USA AISI 1983, thickness of 0.23 mm, while the supplier of the material is the Supplier A. For the second condition, a magnetic steel of grade M4, thickness of 0.27 mm and the Supplier B are selected. Finally, in the third condition, the grade is Hi-B, the thickness 0.23 mm and the Supplier A.

6.1. Neural network modeling

A set consisting of $N = 1680$ actual industrial measurements is used to train the network in case of the first production condition. The 10% of the training data are selected as validation set, while 560 test data have been used to evaluate the prediction accuracy of the network. Similar number of data has been also used for the training, testing, and validation sets of the other two conditions. The network configurations, which have been used for predicting the transformer iron losses, are presented in Table 2 for the three aforementioned production conditions.

Figures 7(a) and (b) present the Q-Q (Quantile-Quantile) plots [16] of the specific iron losses, using the typical loss curve (current practice) and the proposed neural network method respectively, for the first production condition. According to this method, the real specific iron losses are plotted versus the predicted ones and as a result, perfect prediction lies on a line of 45° slope. As is observed, the neural network-based prediction scheme provides, on average, more accurate results than the typical loss curve method. In particular, the current method (loss curve) shows a maximum absolute relative error of 9.9%, while the respective error of the neural network scheme is 4.8%. As far as the average error is concerned, the proposed method yields 1.5% error instead of 2.9% of the current prac-

Table 3
Comparison of specific iron loss prediction using the typical loss curve method and the neural network model for three different production conditions

	#1 Production condition		#2 Production condition		#3 Production condition	
	Typical loss curve	Neural network	Typical loss curve	Neural network	Typical loss curve	Neural network
Relative error	2.9	1.5	3.1	1.7	3.3	1.8
Minimum Error (%)	−6.1	−4.5	−7.1	−4.9	−7.4	−5.1
Maximum Error (%)	9.9	4.8	10.6	5.5	10.8	5.8

tice. Similar results are also obtained for the other two production conditions and these results are summarized in Table 3.

6.2. *Weight adaptation*

The weight adaptation algorithm described in Section 4 has been applied to our industrial environment to improve the prediction accuracy of the network in future production batches. This means that the network is able to adapt its performance in slight changes of the production condition. In particular, the network performance is monitored during the evaluation phase, and in case that the prediction error exceeds a predetermined threshold, for a given production batch, the weight adaptation algorithm is activated. Figure 8(a) illustrates the average prediction error for several production batches, all consisting of 50 transformers, along with the specified threshold. In the experiments considered, this threshold is defined to be 10% higher than the average prediction error. As is observed, at the 19th production batch the prediction error exceeds this threshold and the weight adaptation is activated. To avoid large number of equations [Eq. (11b)], only 10% (i.e. 5 transformers) of the 19th batch are included in the set T_c, selected using a principal component analysis. The remaining transformers are used as test set in order to evaluate the network prediction performance. After the weight updating, the prediction accuracy has been improved as depicted in Fig. 8(b), where the prediction error for the batches 20 to 30 is shown.

6.3. *Genetic algorithm performance*

In the following experiments, the genetic algorithm has been applied to group 100 small and 100 large cores of the same production batch of 50 transformers, 100 kVA, 50 Hz of the second production condition.

Figure 9 presents the effect of the initial population selection on the convergence rate of the GA. In this figure, three different approaches have been examined. The first is based on a random selection of the initial population $P(0)$, the second, on the quality class method of the current grouping process, while the third, on the minimal deviation method (see Section 2). For the quality class method L possible solutions can be used as the initial population, while for the minimal deviation method, the L best core arrangements have been selected. In all cases, a mutation rate equal to 5% has been used, the population was 25 chromosomes while at each iteration 30% of the total population (i.e. 8) parents have been adopted. As is observed, the worst performance appears to be the random selection, while the minimal deviation method presents the fastest convergence.

The total transformer losses over all 50 examined transformers versus mutation rate is presented in Fig. 10(a), for the three different approaches used for initial population selection. As can be seen, in case of high mutation rates, the performance of the class quality and the minimal deviation methods are almost similar, while the random initial population selection provides the slowest convergence for all mutation rates. In this case the number Q is equal to 30% of the total population, while 100 GA cycles have been used to terminate the iteration process. Furthermore, it seems that in all cases the minimal losses fall in the range of 4–6% mutation rate. This is due to the fact that small mutation probability may trap the solution to a local minimum. Instead, large probability leads to random search, which deteriorates the GA performance. However, the minimal deviation method outperforms compared to the other two ones.

Figure 10(b) presents the mutation rate versus total transformer losses for different values of Q (10%, 30% and 50% of the total population), in case that the minimal deviation method is used as the initial population selection and for 100 GA cycles. Mutation probabilities around 4–6% provide the best results in this case too. It can be seen, from this figure, that, as the number of Q increases, the algorithm reaches the optimal solution in fewer GA cycles. However, large values of Q also increase the cycle computational load, which may affect the total GA complexity. Table 4 presents the

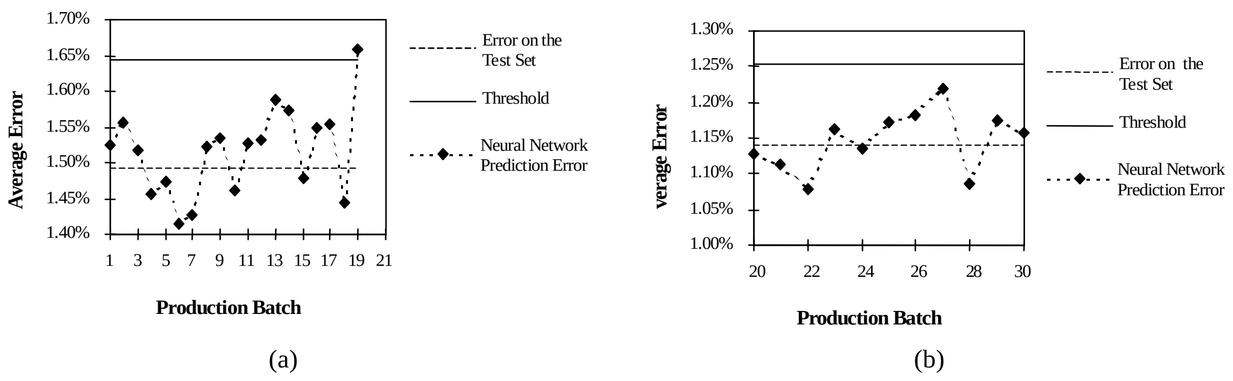


Fig. 8. Evolution of average absolute relative error over various production batches of the first production condition, (a) for the first 19 production batches (before the weight adaptation), (b) for the following 11 production batches (after the weight adaptation).

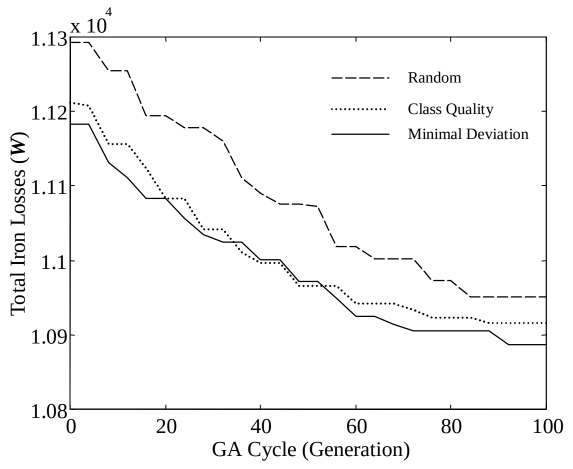


Fig. 9. The genetic algorithm convergence for different initial population selection mechanisms.

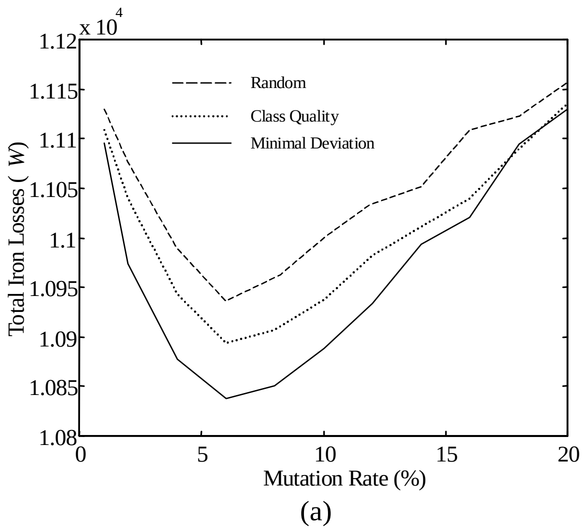
average computational load per GA cycle for different values of Q along with the average number of iterations (cycles) required to achieve total iron loss less than 10900 W. In this case, the results have been obtained on a P-II 350 MHz PC and a mutation rate equal to 5% has been used. It can be seen that $Q = 30\%$ of the total population provides the fastest convergence although it requires greater number of cycles than other Q. The effect of the mutation probability on the average computational load is also shown in Table 5. The best number of parents, (i.e. $Q = 30\%$), has been used in this case. This table indicates the average computational load per GA cycle, along with the GA cycles required to achieve a minimum loss less than 10900 W. It is observed that mutation probability around 6% yields the best results as far as the computational cost is concerned.

As a result, the values of the GA parameters, which are involved in the process and optimize the GA convergence, are the minimal deviation grouping method as the initial population, mutation rate around 6%, number of parents undergone crossover 30% of the total population, and constant population size equal to 25 chromosomes. These values are used in the following results.

Figure 11(a) evaluates the performance of the genetic algorithm by comparing the predicted with the actual iron losses (measured after the transformer construction) for the same batch of 50 transformers of 100 kVA, 50 Hz of the second production condition. Similarly, Figure 11(b) evaluates the performance of the proposed scheme for another batch of 50 transformers of 250 kVA, 50 Hz, second production condition.

The computational complexity of the proposed neural network-genetic algorithm scheme depends on a) the cost required for the GA convergence and b) on the testing time that the neural network takes for predicting the transformer actual losses for a give core arrangement. For the GA, 8 different runs have been conducted and the core arrangement which yields the minimal losses over all runs is selected as the most appropriate. At each run, the GA terminates when the best core arrangement remains constant for 30 number of generations, indicating that further optimization is unlikely. Using this termination criterion, the average number of GA cycles over all 8 runs is 168. Since the cost of each cycle is 517 ms (see Table 5 for $Q = 30\%$ and mutation rate 6%), the total complexity of the GA over all cycles and runs is 8*168*517 ms $=$ 11.58 min.

At each GA cycle, the neural network is activated to predict the actual transformer losses. Since in our case, 50 transformers are constructed (100 large/small cores) and the GA population consists of 25 chromosomes, the neural network is applied $50 * 25 = 1250$ times for every GA cycle. The average time that the neural



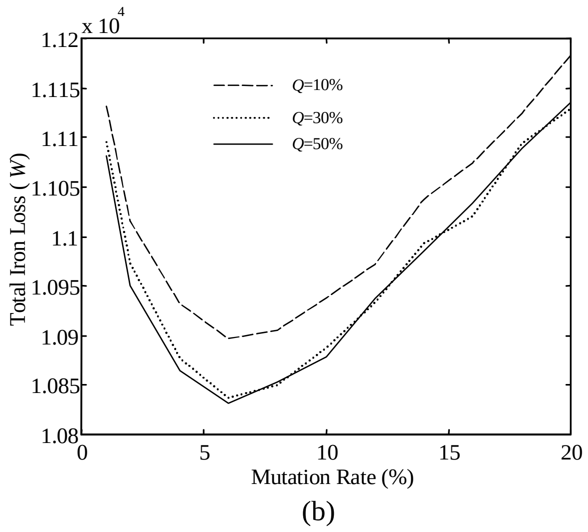


Fig. 10. Total transformer losses versus mutation probability for 100 GA cycles. (a) Different initial population selection. (b) Different number of selected parents.

Table 4
Average execution of the genetic algorithm for different values of Q

Number of parents (Q)	Cost/GA cycle (msec)	Average iterations (sec)	Total cost
10%	295	344	101.48
20%	381	142	54.02
30%	498	77	38.35
40%	604	71	42.88
50%	700	67	46.90
60%	820	64	52.48

Table 5
Average execution of the genetic algorithm for different mutation probabilities

Mutation rate	Cost/GA cycle (msec)	Average iterations	Total cost (sec)
1%	396	803	317.98
2%	446	476	212.30
4%	479	86	41.19
6%	517	62	32.05
8%	562	75	42.15
10%	618	95	58.71

network requires to predict the actual losses of all 50 transformers and 25 chromosomes has been measured to be 358 ms. As a result, the total cost over the average 168 GA cycles and 8 runs is 8.02 min. Thus, the total cost of the proposed neural-genetic algorithm grouping scheme is $11.58 + 8.02 = 19.6$ min.

In this cost, it should be added the time that the network requires to adapt its weights (network weight adaptation). However, this is performed only in case that the network prediction accuracy is considered poor. In our experiments, where 30 different production batches have been examined, the weight adaptation mechanism has been activated only once and the total cost for this weight perturbation was 1 s if $m = 8$. For the above results, a P-II (~ 350 MHz) PC has been used with a C implementation of the proposed neural-genetic grouping scheme.

It should be mentioned that the execution time of the proposed algorithm is small compared to other times involved in the transformer construction process. For example, only the annealing of the individual cores takes about 11.5 h to be completed. Furthermore, the

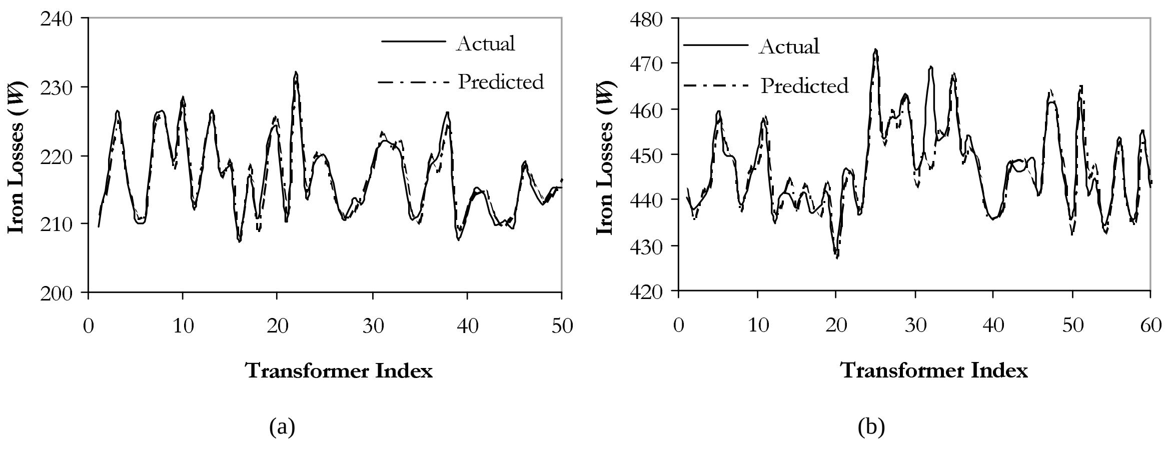


Fig. 11. Evaluation of the genetic algorithm for different production conditions: (a) production batch of 50 transformers, 100 kVA, 50 Hz of the second production condition, (b) production batch of 50 transformers, 250 kVA, 50 Hz, second production condition.

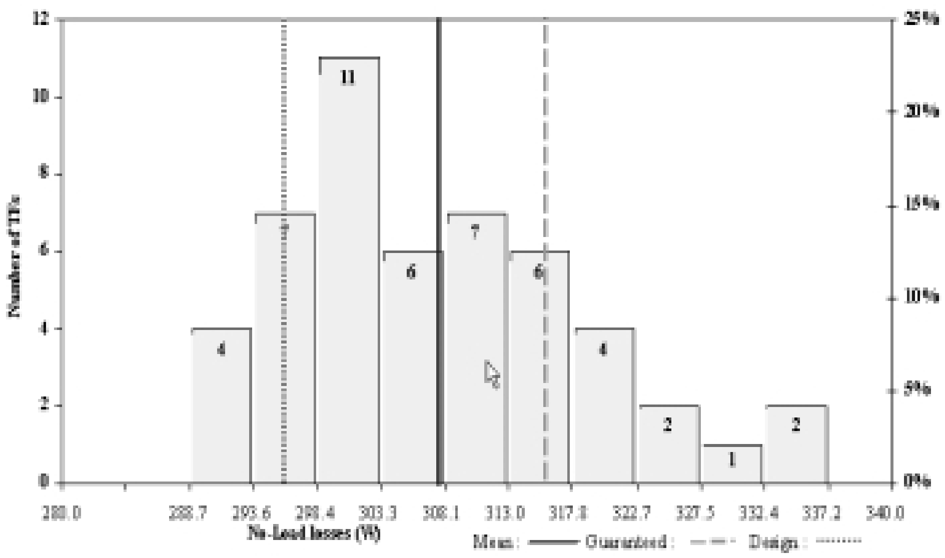


Fig. 12. Iron loss distribution of 50 transformers, 160 kVA (second production condition) using the quality class method as grouping process.

proposed scheme can be applied "in parallel" with the transformer construction. For example, it can be conducted, while the individual core-department produces the cores of the following batch and the transformer-department constructs the transformers of the previous batch.

6.4. Iron loss reduction

In this subsection, the proposed neural network-genetic algorithm scheme is compared with the minimal deviation grouping process and the quality class method, both used in the current practice for a production batch of 50 transformers of 160 kVA.

Initially, the 100 small and 100 large individual cores (producing 50 transformers) have been grouped using the quality class method (see Section 2), and the distribution of the transformer iron losses is depicted in Fig. 12. In this experiment, the desired (guaranteed) losses, which are related to P_0, are 315 W, while the designed losses are 296 W, (i.e. about 6% lower than the guaranteed losses). Similarly, in Fig. 13 the distribution of iron losses is depicted if the grouping method of the minimal deviation (presented in [8]) is used for the

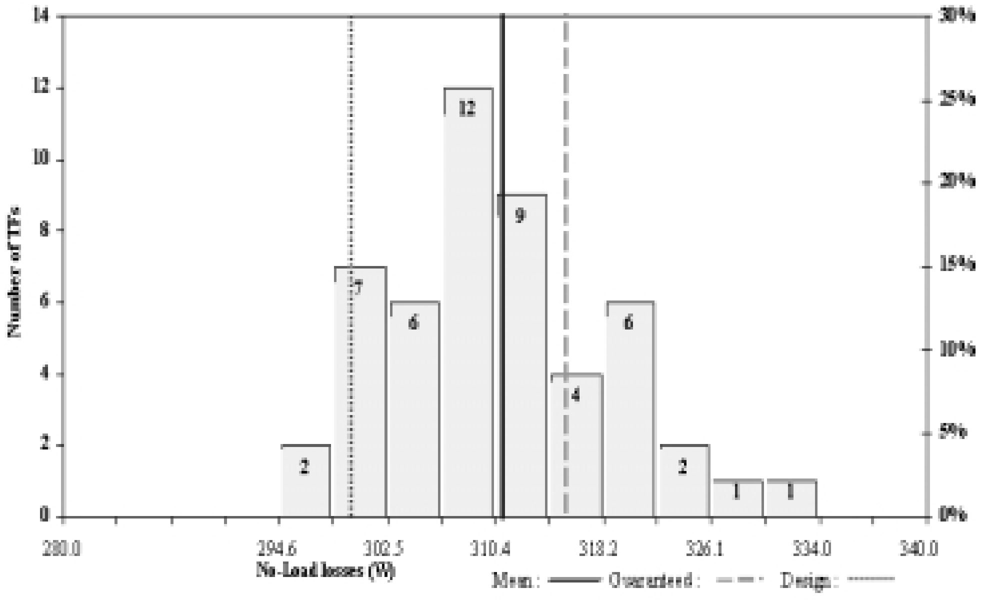


Fig. 13. Iron loss distribution of 50 transformers, 160 kVA (second production condition) using the minimal deviation method as grouping process.

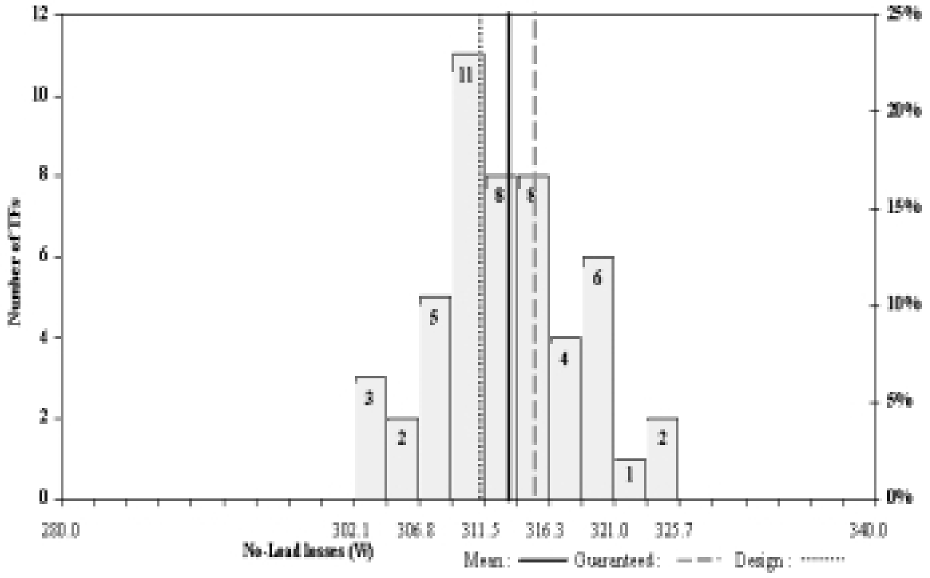


Fig. 14. Iron loss distribution of 50 transformers, 160 kVA (second production condition) using the proposed neural network-genetic algorithm scheme.

core arrangement. In this experiment, the same guaranteed losses are used, while the designed iron losses are 4.8% lower the guaranteed ones, since it is expected that this method provides better results than the quality class approach. As is observed, in Fig. 12, the average losses are 307.31 W, (i.e. 3.82% higher than the designed ones), while a loss fluctuation of 48.5 W is encountered. On the contrary, in Fig. 13, the average losses are 310.6 W, (i.e. 3.53% higher than the designed losses), while the loss fluctuation has been restricted to 39.4 W.

Finally, Figure 14 presents the iron loss distribution

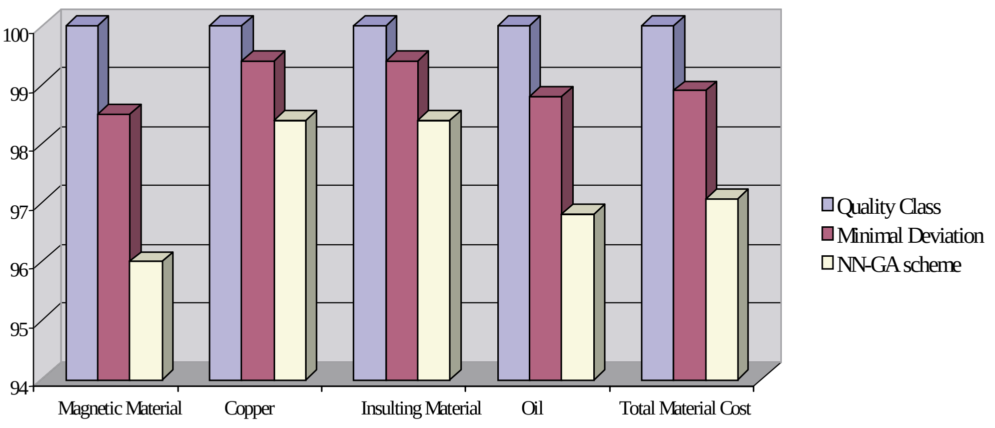


Fig. 15. Cost reduction for transformers of 50 kVA (first condition).

when the proposed neural network-genetic algorithm scheme is used for transformer construction. In this case, a much smaller safety margin has been selected and the designed losses are 311 W, (i.e. only 1.3% lower than the guaranteed ones). However, as can be seen from Fig. 14, the average losses are very close to the designed ones (313.15 W or 0.69% higher than the designed losses), while the smallest loss fluctuation is encountered (23.6 W). Furthermore, the maximum (minimum) losses in this case is lower (higher) than those obtained from the other two cases.

Usually, transformers, whose iron losses are about 10% higher than the guaranteed losses (in our case 346 W), are considered as unacceptable by the customers [3,15]. Consequently, as depicted in Figs 12–14, all grouping methods yield transformers of acceptable quality. However, in the proposed neural-genetic scheme, the width of loss distribution was narrower than the other two grouping methods. Thus, it is less probable to generate transformers whose iron losses violates the upper limit of 346 W. Furthermore, the proposed scheme also yields a significant reduction of the production cost. This is due to the fact that smaller safety margin is used in this case which saves magnetic material. The latter also leads to transformers of smaller dimensions and further results in a reduction of the weight of the material of the windings (copper), insulating materials and transformer oil or equivalently to an overall reduction of the required cost. The reduction of the material cost is presented in Fig. 15 in case of 50 transformers of 50 kVA (first condition). In this figure, all costs have been depicted with respect to the cost of the quality class method, the cost of which is assumed to be 100.

7. Conclusion

In this paper, a synergetic neural network and genetic algorithm scheme has been proposed to reduce transformer iron losses by exploiting information derived from both the design and the transformer construction phase. More specifically, for a given number of small and large individual cores, our target is to estimate the optimal core arrangement so that transformers of minimal iron losses are assembled. This is accomplished by minimizing a cost function, which expresses the aggregate iron losses over all possible constructed transformers, using an evolutionary programming scheme. Since, however, the actual iron losses of a three-transformer are in fact unknown before the transformer construction, a neural network architecture is adopted in this paper to provide an accurate estimate of the transformer losses prior to their assembly. Several actual core measurements, transformer design parameters along with the way of core arrangement are used as inputs to the network. Furthermore, the prediction accuracy of the neural network model is monitored after the transformer construction (evaluation phase). In case that a poor prediction accuracy is encountered, a weight adaptation mechanism is activated to estimate new network weights taking into account both the current and the previous network knowledge.

The proposed scheme has been applied in our industrial environment and has been compared with the current grouping techniques as they are discussed in Section 2. In particular, it has been observed that the neural network-genetic algorithm approach yields a significant reduction between the deviation of the designed and the actual iron losses. With such a reduction, the transformers can be designed with a smaller safety margin, which saves magnetic material and reduces the total production cost.

Appendix A

Let us define by $w_{i,b}, w_{i,a}$ the $(p+1)\times 1$ vectors containing all network weights and biases, which connect the ith hidden neuron to the input layer before and after the weight adaptation respectively. Then, matrices $W_a,\ W_b$ can be formed as follows

$$\begin{aligned} &W_a = [w_{1,a}\ w_{2,a}\ \ldots\ w_{l,a}] \quad and \\ &W_b = [w_{1,b}\ w_{2,b}\ \ldots\ w_{l,b}] \end{aligned} \tag{A1}$$

similarly to the matrix W of Section 3. Let us also define by $v_b,\ v_a$ the $l\times 1$ vectors which contain the network weights connecting the ith hidden neuron to the output neuron before and after the weight adaptation respectively. Similarly, θ_a, θ_b correspond to the biases of the output neuron. Since Eq. (10) is valid for all network weights, it can be derived that

$$\begin{aligned} &W_a = W_b + \Delta w,\ v_a = v_b + \Delta v, \\ &\theta_a = \theta_b + \Delta\theta \end{aligned} \tag{A2}$$

where ΔW, Δv, $\Delta\theta$ are small increments of the respective network weights.

Thus, Eq. (6) can be written as for a given transformer z_i

$$\begin{aligned} &u_a(x(z_i)) \\ &= f\left(W_b^T\cdot x(z_i) + \Delta W^T\cdot x(z_i)\right) \end{aligned} \tag{A3}$$

where subscript b and a refer to before and after the weight adaptation respectively.

Application of a first order Taylor series expansion to Eq. (A3) yields to

$$\begin{aligned} &u_a(x(z_i)) \\ &= f\left(W_b^T\cdot x(z_i) + Q\cdot\Delta W^T\cdot x(z_i)\right) \end{aligned} \tag{A4}$$

where Q is the gradient of $f(\cdot)$ and can be expressed by the following diagonal matrix,

$$Q = \text{diag}\{\delta_{1,b}(x(z_i)),\ldots,\delta_{l,b}(x(z_i))\} \tag{A5}$$

where

$$\begin{aligned} &\delta_{i,b}(x(z_i)) \\ &= u_{i,b}(x(z_i))\cdot[1-u_{i,b}(x(z_i))] \end{aligned} \tag{A6}$$

indicate the gradient of the hidden neuron output, assuming that the activation function of the hidden neurons is the sigmoid.

Thus, Eq. (11b) can be written as for a given transformer z_i

$$\begin{aligned} S_{z_i}^a &\approx y_{w_a}(x(z_i)) \\ &= v_b^T\cdot u_a(x(z_i)) + \Delta v^T\cdot u_a(x(z_i)) \\ &\quad +\theta_b + \Delta\theta \end{aligned} \tag{A7}$$

Combining, Eqs (A7) and (A4), and ignoring the second order terms we can find that

$$\begin{aligned} &S_{z_i}^a - v_b^T\cdot u_b(x(z_i)) - \theta_b \\ &= v_b^T\cdot Q\cdot\Delta W^T\cdot x(z_i) \\ &\quad +\Delta v^T\cdot u_b(x(z_i)) + \Delta\theta \end{aligned} \tag{A8}$$

Equation (A8) can be rewritten as

$$b(z_i) = a(z_i)^T\cdot\Delta w \tag{A9}$$

where

$$b(z_i) \equiv S_{z_i}^a - v_b^T\cdot u_b(x(z_i)) - \theta_b \tag{A10}$$

is the prediction error before weight adaptation, while vector $a(z_i)$ is produced by reordering the right term of Eq. (A8) for all network weights. In the previous equations, we have added the dependence on the transformer z_i.

$$\begin{aligned} a(z_i)^T\cdot\Delta w = v_b^T\cdot Q\cdot\Delta W^T\cdot x(z_i) \\ +\Delta v^T\cdot u_b(x(z_i)) + \Delta\theta \end{aligned} \tag{A11}$$

Equation (A11) is a linear equation with respect to the weights increment Δw and vector $a(z_i)$ can be estimated by simply identifying the terms of the right and left hand of Eq. (A10). In particular,

$$a(z_i)[\text{vec}\{r\cdot x(z_i)^T\}u_b(x(z_i))\ 1]^T \tag{A12}$$

with $r = v_b^T\cdot Q$ and vec$\{r\cdot x(z_i)^T\}$ denoting a vector formed by stacking up all rows of matrix $r\cdot x(z_i)^T$.

Equation A(8) for all transformers z_i can be written as follows

$$b = A\cdot\Delta w \tag{A13}$$

where

$$b = [b(z_1)b(z_2)\ldots b(z_m)]^T \tag{A14}$$

and

$$A^T = [a(z_1)a(z_2)\ldots a(z_m)] \tag{A15}$$

Appendix B

Let us first define g_i the difference between the target output and the network output for the ith element of the training set T (old information) in case that the old weights are used.

$$g_i = \left(S^a_{t_i} - y_{w_b}(x(t_i))\right) \tag{B1}$$

Let us also recall that

$$\delta_i(t_j) = u_{i,b}(x(t_j)) \cdot (1 - u_{i,b}(x(t_j))) \tag{B2}$$

is the derivative of the ith hidden neuron output when the transformer t_i is fed as input to the network using the old weights, w_b.

Differentiating Eq. (11a) with respect to the network weights $w_{j,k}$, we have

$$\frac{\partial D_{i,b}}{\partial w_{j,k}} = -g_i \cdot v_j \cdot \delta_k(x(t_i)) \cdot x_k(t_i) \tag{B3}$$

where $x_k(t_i)$ is the kth element of the feature vector $x(t_i)$ of the transformer t_i. We recall that D_i is the squared error of the ith element of T, that is $D_i = \left(S^a_{t_i} - y_{w_b}(x(t_i))\right)^2$ while $w_{j,k}$ refers to the network weight that connects the jth hidden neuron with the kth input element. The $w_{j,p+1}$ is the bias of the jth hidden neuron and thus in this case $x_{p+1}(t_i) = 1$ for all t_i. Differentiating Eq. (11a) with respect to the network weights v_k and θ we find that

$$\frac{\partial D_i}{\partial v_k} = -g_i \cdot u_k(x(t_i)) \tag{B4a}$$

$$\frac{\partial D_i}{\partial v_k} = -g_i \tag{B4b}$$

Consequently, the sensitivity of the error D_i to all network weights can be expressed as

$$\Delta D_i = \sum_{j,k} \frac{\partial D_i}{\partial w_{j,k}} \Delta w_{j,k} + \sum_k \frac{\partial D_i}{\partial v_k} \Delta v_k + \frac{\partial D_i}{\partial \theta} \Delta\theta \tag{B5}$$

Therefore, for all elements in T. That is for all $i = 2, \ldots, L$ the previous equation can be written in a matrix form

$$\Delta D = J \cdot \Delta w \tag{B6}$$

where

$$\Delta D = [\Delta D_1 \ \Delta D_2 \ \Delta D_L]^T \tag{B7}$$

J is the Jacobian matrix given by

$$J = \begin{bmatrix} \cdots \frac{\partial D_1}{\partial w_{j,k}} & \cdots \frac{\partial D_1}{\partial v_k} & \cdots \frac{\partial D_1}{\partial \phi} \\ \cdots \frac{\partial D_2}{\partial w_{j,k}} & \cdots \frac{\partial D_2}{\partial v_k} & \cdots \frac{\partial D_2}{\partial \phi} \\ \vdots \quad \vdots & \vdots & \vdots \\ \cdots \frac{\partial D_L}{\partial w_{j,k}} & \cdots \frac{\partial D_L}{\partial v_k} & \cdots \frac{\partial D_L}{\partial \phi} \end{bmatrix} \tag{B8}$$

The effect of perturbation Δw in Eq. (11a) can be modeled by $\sum_{i=1}^{L} \Delta D_i^2$ [4]. Thus, minimization of Eq. (11a) is equivalent to minimization of

$$\min \frac{1}{2} \Delta w^T \cdot J^T \cdot J \cdot \Delta w \tag{B9}$$

References

[1] S.A. Stigant and A.C. Franklin, *The J&P Transformer Book,* Newnes-Butterworths, 1973.

[2] R.L. Bean, N. Chackan, H.R. Moore and E.C. Wentz, *Transformers for the Electric Power Industry,* Westinghouse Electric Corporation, 1959.

[3] BS EN 60076-1, Power Transformers – Part 1: General, 1997.

[4] A. Doulamis, N. Doulamis and S. Kollias, On Line Retrainable Neural Networks: Improving the Performance of Neural Networks in Image Analysis Problems, *IEEE Trans. on Neural Networks* **11** (2000), 137–155.

[5] A.E. Fitzerald and C. Kingsley, *Electric Machinery,* MacGraw-Hill, 1961.

[6] E.F. Fuchs, M.A.S. Masoum and D.J. Roesler, Large Signal Nonlinear Model of Anisotropic Transformers for Nonsinusoidal Operation; Part II: Magnetizing and Core-loss Currents, *IEEE Transactions on Power Delivery* **6** (1991), 1509–1516.

[7] P.S. Georgilakis, N.D. Hatziargyriou, N.D. Doulamis, A.D. Doulamis and S.D. Kollias, Prediction of Iron Losses of Wound Core Distribution Transformers Based on Artificial Neural Networks, *Neurocomputing* **23** (1998), 15–29.

[8] P.S. Georgilakis, N.D. Hatziargyriou, N.D. Doulamis, A.D. Doulamis and S.D. Kollias, A Neural Network Framework for Predicting Transformer Core Losses, IEEE Int. Conference on Power Industry Computer Applications (PICA), San Francisco, USA, 1999.

[9] P.S. Georgilakis, Contribution of artificial intelligence techniques in the reduction of distribution transformer iron losses, Ph.D. Dissertation, National Technical University of Athens, 2000.

[10] D.E. Goldberg, *Genetic Algorithm in Search, Optimization and Machine Learning,* Addison Wesley, 1989.

[11] R.S. Girgis, D.A. Yannucci and J.B. Templeton, Performance Parameters on Power Transformers using 3D Magnetic Field Calculations, *IEEE Transactions on PAS* **103** (1984), 2708–2713.

[12] N.D. Hatziargyriou, P.S. Georgilakis, D.S. Spiliopoulos and J.A. Bakopoulos, Quality Improvement of individual Cores of Distribution Transformers using Decision Trees, *Int. Journal of Engineering Intelligent Systems* **6** (1998), 141–146.

[13] S. Haykin, *Neural Networks: A Comprehensive Foundation,* Macmillan, 1994.

[14] B. Hochart, *Power Transformer Handbook,* Butterworths, 1987.

[15] IEC 76-1, Power Transformers – Part 1: General, 1993.

[16] H. Kobayashi, *Modeling and Analysis*, Addison-Welsey, 1981.

[17] S. Kollias and D. Anastassiou, An Adaptive Least Squares Algorithm for the Efficient Training of Artificial Neural Networks, *IEEE Trans. on Circuits and Systems* **36** (1989), 1092–1101.

[18] G. Lian, Y. Ruoping and C. Pizhang, An Equivalent Magnetization Surface Current Approach of Calculation 3D-leakage Fields of a Transformer, *IEEE Transactions on Power Delivery* **2** (1987), 817–822.

[19] D.J. Luenberger, *Linear and non Linear Programming,* Addison-Wesley, 1984.

[20] B.W. McConnell, Increasing Distribution Transformer Efficiency: Potential for Energy Savings, *IEEE Power Engineering Review* **18** (1998), 8–10.

[21] G.F. Mechler and R.S. Girgis, Calculation of Spatial Loss Distribution in Stacked Power and Distribution Transformer Cores, *IEEE Transactions on Power Delivery* **13** (1998), 532–537.

[22] Members of the Staff of the Department of Electrical Engineering of M.I.T., Magnetic Circuits and transformers, John Wiley, 1947.

[23] Z. Michalewicz, *Genetic Algorithms* + *Data Structures* = *Evolution Programs*, Springer Verlag, 1994.

[24] K. Milodakis, *Quality Control of Transformer Cores,* Technical Report University of Crete, 1993.

[25] E. Oja, Neural Networks, Principal Components and Subspaces, *Journal of Neural Systems* **1** (1989), 1–68.

[26] B.C. Papadias, N.D. Hatziargyriou, J.A. Bakopoulos and J.M. Prousalidis, Three Phase Transformer Modelling for Fast Electromagnetic Transients, *IEEE Transactions on Power Delivery* **9** (1994), 1151–1159.

[27] D. Paparigas, D. Spiliopoulos, S. Elefsiniotis and J. Bakopoulos, Estimation of Magnetic Material for Individual Transformers Cores, Proc. of European Conference on Technological and Economic Advances of the Use of Distribution Transformers, Greece, May 1994.

[28] Proceedings of the International Conference on Electric Utility Deregulation and Restructuring and Power Technologies 2000, Loi Lei Lai (Editor), City University, London, IEEE Catalog N00EX382, April 2000.