PAPER *Special Issue on ZZZ*

A Functional Analytic Approach to Incremental Learning in Optimally Generalizing Neural Networks

Sethu VIJAYAKUMAR† *and* **Hidemitsu OGAWA** † ,*Members*

SUMMARY For a given set of training data, a method of learning for optimally generalizing neural networks using functional analytic approach already exists. Here, we consider the case when additional training data is made available at a later stage. We devise a method of carrying out optimal learning with respect to the entire set of training data (including the newly added one) using the results of the previously learned stage. This ensures that the learning operator and the learned function can both be computed incrementally, leading to a reduced computational cost. Finally, we also provide a simplified relationship between the newly learned function and the previous function, opening avenues for work into selection of optimal training set.
key words: incremental learning, generalization, feedforward neural networks, Wiener learning, inverse problem

1. Introduction

Multi-layer feedforward neural networks are essential approximators, mapping from a finite dimensional input space to an output space. This paper discusses the learning problem in these kinds of networks as an inverse problem from the functional analytic point of view. Functional analytic approach to learning in Neural Networks(NNs) has been proved to improve generalizing abilty because of their focus on the original function space rather than the sampled space. In earlier research work on this approach [9], [10], [11], learning has been discussed under the framework of obtaining an optimal learning operator from a given set of training data. However, there is a need to look at cases where we have an optimally trained NN and there are additions to the training data to be incorporated at a later stage. Re-learning using the entire training set is a possible but not very efficient solution. These kind of situations also arise when we are dealing with 'active learning' in which the set of training data can be provided only one after the another and also uses the intermediate learned results for deciding on the next best sample. Recently, a lot of work on these lines are being carried out, some of them being the papers by Zhang[16] and Jutten[2]. However, most of them, though being *incremental* in the nature of addition of training data or increase in the number of hidden units etc., almost always involves re-learning or re-training with the addition of information or units.

Manuscript received xxx, 1994
Manuscript revised yyy, 1994
†The authors are with the Tokyo Institute of Technology

In this paper, the concept of 'truly' incremental learning with additive training data is formalized. Based on this, an efficient method to incorporate new data incrementally, without re-learning from scratch, is formulated. This incremental method is shown to reduce the computational overhead while computing the new learning operator.

2. Neural Network learning as an inverse problem

In this section, we will present a brief review of the basic formalization necessary for discussing the learning problem in NNs from an analytic point of view. Let us begin by considering a three-layer feedforward neural network whose number of input, hidden, and output units are L, N, and 1, respectively as shown in Fig.1. Let $\xi_1, \xi_2, ..., \xi_L$ be the inputs to the input layer neurons. Let these inputs constitute the elements of an L-dimensional vector x, referred to as the 'input vector'. An input-output relation of the n-th hidden unit is denoted by $u_n(x)$ and called the basis function. In conventional NNs, the following basis function is used:

$$u_n(x) = \sigma(\sum_{l=1}^{L} w_{nl}\xi_l), \tag{1}$$

where $\sigma(.)$ is a sigmoidal function. However, here we use more general basis functions of L-variables. Let w_n be the corresponding weight connecting the n-th hidden unit and the output unit. In this case, the network can be considered as a real valued function $f_0(x)$ of L variables and expressed as follows:

$$f_0(x) = \sum_{n=1}^{N} w_n u_n(x). \tag{2}$$

The *learning problem* is to construct a neural network by using a training set so that the NN expresses the best approximation $f_0(x)$ to a desired function $f(x)$ under some given learning criterion. Next, we define some of the notations used here.

$\{x_m\}_{m=1}^{M}$: A *training set* given as a set of M input vectors.

$\{y_m\}_{m=1}^{M}$: The corresponding desired output values, where $y_m = f(x_m)$.

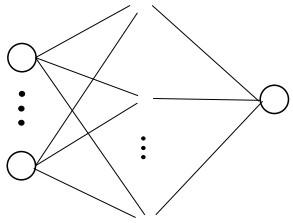

Fig. 1 NN as a real valued function

y: Vector consisting of elements $\{y_m\}_{m=1}^M$.

$\{x_m, y_m\}_{m=1}^M$: Set of training data.

The learning problem of neural networks can now be discussed from the point of view of function approximation. In other words, the learning problem can be regarded as the same problem as that of the general sampling theorem [7] and image restoration [6], [8]. In this context, x_m and y_m can be called the m-th *sample point* and the corresponding *sample value* of the desired function f, respectively.

Let H be the set of all functions f to be aproximated by the neural networks. Assume that H is a Hilbert space with a *reproducing kernel* $K(x, x')$. Let D be the domain of the basis functions, which is a subset of the L-dimensional Euclidean space R^L. The reproducing kernel $K(x, x')$ is a bivariate function defined on $D \times D$ which satisfies the following two conditions:

1. For any fixed x' in D , $K(x, x')$ is a function in H.
2. For any function f in H and for any x' in D, it holds that

$$(f(x), K(x, x')) = f(x'), \tag{3}$$

where the left hand side of eq.(3) denotes the inner product in H.

In the theory of Hilbert space, arguments are developed by regarding a function as a point in that space. Thus, things such as 'value of a function at a point' cannot be discussed under the general framework of Hilbert space. However, if the Hilbert space has a reproducing kernel, then it is possible to deal with the value of a function at a point. Indeed, if we define functions $\psi_m(x)$ as

$$\psi_m(x) = K(x, x_m) : 1 \leqq m \leqq M, \tag{4}$$

then, the value of f at a sample point x_m is expressed in Hilbert space language as the inner product of f and ψ_m as

$$f(x_m) = (f, \psi_m). \tag{5}$$

Once the training set $\{x_m\}_{m=1}^M$ is fixed, the vector y is uniquely determined from f. So, we can introduce an operator A which transforms f to y:

$$y = Af. \tag{6}$$

The operator A, called the *sampling operator*, becomes a linear operator even when we are concerned with nonlinear neural networks. It is expressed by using the Schatten product as

$$A = \sum\nolimits_{m=1}^{M} e_m \otimes \overline{\psi_m}, \tag{7}$$

where $\{e_m\}_{m=1}^M$ is the so-called natural basis † in R^M. The Schatten product denoted by $(. \otimes .)$ is defined by

$$(e_m \otimes \overline{\psi_m})f = (f, \psi_m)e_m. \tag{8}$$

Hence, the learning problem is the problem of obtaining an estimate , say f_0, to f from y in the model (See Fig. 2). This can be considered as an *inverse problem* [10], [11] equivalent to obtaining an operator X which provides f_0 from y:

$$f_0 = Xy. \tag{9}$$

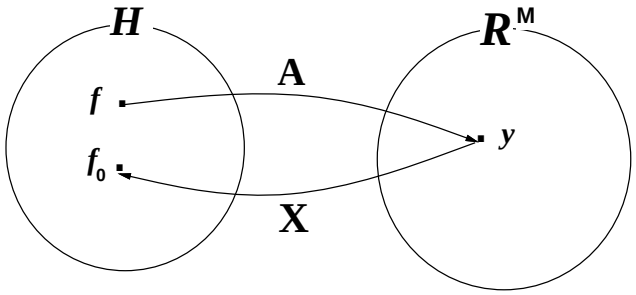

Fig. 2 NN learning as an inverse problem

The operator X is called the *learning operator*. It is also referred to elsewhere in other fields by different names like *restoration operator, reconstruction operator, approximation operator*, etc. This operator X can be optimized based on different criteria. These criteria, known as the *learning criteria*, are of different types like the memorization criterion, the Wiener criterion, the projection criterion, etc. [11].

3. Wiener learning

As described in the previous section, the operator X can be optimized based on different criteria. One of them is the *Wiener criterion* J_W, which is described below. The mathematical representation of the criterion for the *noiseless* case is given in the following equation:

†The vector e_m is the M-dimensional vector consisting of zero elements except the element m equal to 1.

$$\text{min. over } X : J_W[X] = E_f \|XAf - f\|^2, \qquad (10)$$

where $\|.\|$ is the norm in H and E_f is the expectation taken over the ensemble $\{f\}$. An operator X satisfying the above criterion is called a Wiener learning operator. The criterion aims at reducing the difference between the original function f and the function f_0 reconstructed by using the learning operator X. The thing to be noted is that this minimization is done in the original function space H and in an *averaged sense* with respect to all the possible functions to be approximated.

The Wiener criterion can be transformed into a more useful form [5]. Let R be the *correlation operator* of all the functions to be approximated and be defined as

$$R = E_f(f \otimes \bar{f}). \qquad (11)$$

A necessary and sufficient condition for the criterion to be satisfied by an operator X is given as

$$XARA^* = RA^*, \qquad (12)$$

where A^* is the adjoint operator of A. A general form of the solution of eq.(12) is given as

$$X = RA^*U^\dagger + W(I - UU^\dagger), \qquad (13)$$

where W is any operator from R^M to H, $U^\dagger$ represents the Moore-Penrose generalized inverse of U [1] and U is defined as

$$U = ARA^*. \qquad (14)$$

Once the training set is fixed, A can be calculated using eqs.(7) and (4). Hence, corresponding to this sampling operator, a learning operator X satisfying the Wiener criterion can be obtained using eqs.(13) and (14).

4. Incremental learning with optimal generalization

So far, the learning problem has been discussed under the basic framework of finding an optimum learning operator X from the given set of training data [9],[10],[11]. In other words, the sampling operator A is fixed based on the training data as shown in eqs.(4) and (7), and then, the corresponding learning operator is optimized based on various criteria. If an additional training data is made available at a later stage, it would involve recalculating the learning operator from scratch to incorporate this new information, which is highly inefficient and computationally expensive.

With a view to solve the above mentioned problem, we would like to introduce the concept of *incremental learning*. Incremental learning, here, refers to successive or progressive learning with the addition of new training data.

Since we will be dealing with a variable number of sample points and hence, a variable dimensional sampled space, we will use lower case letters to represent the number of training data and the dimension of the sampled space from now on. Let $\{e_n^{(m)}\}_{n=1}^m$ represent the natural basis of the sampled space R^m. Let A_m be the sampling operator from H to R^m while considering m sample points, defined as

$$A_m = \sum_{n=1}^{m} (e_n^{(m)} \otimes \overline{\psi_n}). \qquad (15)$$

Let $y^{(m)}$ be the vector consisting of elements $\{y_n\}_{n=1}^m$ and X_m be the corresponding learning operator for the m sample points.

As depicted in Fig.3, with the incorporation of additional training data, the learned function f_m approaches the original function f. In non-incremental approaches, the new learning operator X_{m+1} is computed from $\{x_n, y_n\}_{n=1}^{m+1}$. But, in the incremental approach, the new learning operator X_{m+1} is obtained from the old learning operator X_m, the old sampling operator A_m, and the next sampled location x_{m+1}.

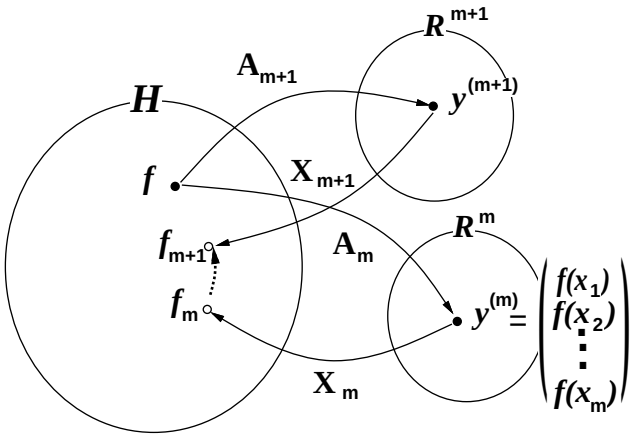

Fig. 3 Incremental learning with additional training data

Summarizing the advantages of the incremental approach using functional analytic approach,

1. It is not necessary to recalculate the entire learning operator when a new training data is made available. Instead, results of the previous stage can be used for the computation, leading to a saving in the computational cost.

2. This approach can also be used in 'active learning' where training data is decided one after the other. Intermediate reconstruction results, as in f_m, are available. This information may be used to improve sampling efficiency by adaptive optimization criteria [3],[4].

5. Incremental computation of learning operator and learned function

In this section, we will present a method of efficiently computing the new learning operator and the newly learned function. As an example of the learning criterion, we consider the Wiener criterion; the reason behind this choice being the fact that the Wiener learning reduces to the projection criterion and partial projection criterion when R (defined in eq.(11)) is set to the identity operator and the projection operator P_1, respectively [15].

5.1 General result for generalized inverses

We shall now present a general result for the recursive computation of the generalized inverse which will be used in our incremental computation of the learning operator.

Let V_m be an $m \times m$ *non-negative definite matrix.* Let V_{m+1} be represented in the form

$$V_{m+1} = \begin{pmatrix} V_m & s_m \\ s_m^T & \sigma \end{pmatrix}, \tag{16}$$

where s_m and σ represent an m-dimensional vector and scalar, respectively. We assume that $V_0 = 0$, $s_0 = 0$. Let t_m and α be defined as

$$t_m = V_m^\dagger s_m, \tag{17}$$
$$\alpha = \sigma - (t_m, s_m). \tag{18}$$

Lemma 1: [1] If V_{m+1} is non-negative definite, then, $\alpha \geqq 0$ and the Moore-Penrose generalized inverse of V_{m+1} for $\alpha > 0$ is given as

$$V_{m+1}^\dagger = \begin{bmatrix} V_m^\dagger + \frac{1}{\alpha}(t_m \otimes \overline{t_m}) & -\frac{1}{\alpha} t_m \\ -\frac{1}{\alpha} t_m^T & \frac{1}{\alpha} \end{bmatrix}. \tag{19}$$

The result for the condition $\alpha = 0$ has been omitted because this condition works out to be a trivial case in context to the learning problem, as will be proved later.

5.2 Incremental computation of X_{m+1}

We shall apply Lemma 1 to our incremental learning operator computation. Considering a particular solution for the Wiener learning, i.e., taking $W = 0$ in eq.(13), we have

$$X_{m+1} = R A_{m+1}^* U_{m+1}^\dagger, \tag{20}$$

where

$$U_{m+1} = A_{m+1} R A_{m+1}^*. \tag{21}$$

In order to express the new sampling operator A_{m+1} in terms of the old sampling operator A_m, we introduce an operator T defined as

$$T = \sum_{n=1}^{m} (e_n^{(m+1)} \otimes \overline{e_n^{(m)}}). \tag{22}$$

Lemma 2: Let $z = (z_1 z_2 ... z_m)$ be an m-dimensional vector. Then, $Tz \in R^{m+1}$ is of the form $Tz = (z_1 z_2 ... z_m 0)^T$.

Now, we can write the new sampling operator A_{m+1} as

$$A_{m+1} = T A_m + (e_{m+1}^{(m+1)} \otimes \overline{\psi_{m+1}}) \tag{23}$$

because of eq.(15) and Lemma 2.
Let

$$s_m = A_m R \psi_{m+1}, \tag{24}$$
$$\sigma = (R\psi_{m+1})(x_{m+1}), \tag{25}$$
$$t_m = U_m^\dagger s_m, \tag{26}$$
$$\alpha = \sigma - (t_m, s_m). \tag{27}$$

It follows from eqs.(21) and (23) that

$$\begin{aligned} U_{m+1} = {} & T U_m T^* + (T s_m \otimes \overline{e_{m+1}^{(m+1)}}) \\ & + (e_{m+1}^{(m+1)} \otimes \overline{T s_m}) \\ & + \sigma (e_{m+1}^{(m+1)} \otimes \overline{\psi_{m+1}}). \end{aligned} \tag{28}$$

Looking at different terms of the expansion, it is easy to show that the matrix U_{m+1} can be written in the form as shown in Fig.4, which also corresponds to the structure of eq.(16). It can also be proved that the matrix represented by the terms U_{m+1} is *non-negative definite* because of eq.(21). Therefore, Lemma 1 can be applied to calculate the term $U_{m+1}^\dagger$ recursively for the condition $\alpha > 0$. However, we still have to consider the case when $\alpha = 0$. We will now present a few lemmas to be used in the analysis.

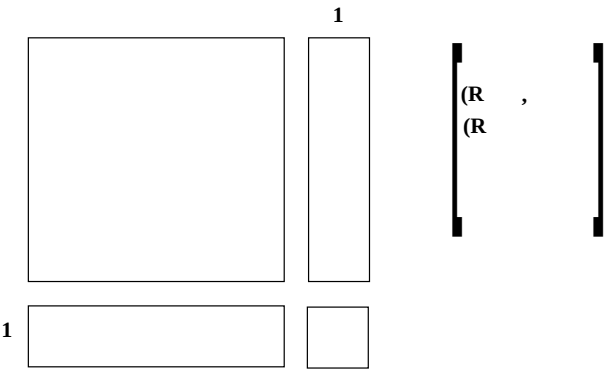

Fig. 4 Structure of the matrix U_{m+1} (See eq.(28))

Let S_m and S_{mc} be defined as

$$S_m = \mathcal{R}(R A_m^*), \tag{29}$$
$$S_{mc} = \mathcal{R}(R) \cap \mathcal{N}(A_m), \tag{30}$$

where $\mathcal{R}(.)$ and $\mathcal{N}(.)$ refer to the range and the null space of an operator, respectively. Then, $\mathcal{R}(R)$ can be decomposed using the direct sum decomposition as follows.

Lemma 3: [14]

$$\begin{aligned}\mathcal{R}(R) &= \mathcal{R}(RA^*) \dot{+} \mathcal{R}(R) \cap \mathcal{N}(A) \\ &= S_m \dot{+} S_{mc}. \end{aligned} \tag{31}$$

Lemma 4: [13] For Wiener learning, the learned function $f_m = X_m A_m f$, for every $f \in \mathcal{R}(R)$, is an oblique projection of f onto S_m along S_{mc}, that is, †

$$f_m = X_m A_m f = P_{S_m, S_{mc}} f. \tag{32}$$

In general, $S_{m+1} \supseteq S_m$. However, based on Lemma 3 and Lemma 4, it is easy to prove the following lemma.

Lemma 5: [13] If $S_{m+1} = S_m$, then the newly learned function does not differ from the previously learned function, i.e., $f_{m+1} = f_m$. On the other hand, if $S_{m+1} \supset S_m$, then $f_{m+1} \neq f_m$, in general.

Lemma 6: It holds that

$$rank(U_{m+1}) = rank(U_m) + sgn(\alpha). \tag{33}$$

The proof of the above lemma is provided in Appendix A.

Using the Lemma 6, the non-negative definiteness of R and the definition of U_{m+1} in eq.(21), we can derive the following lemma.

Lemma 7:

$$dim(S_{m+1}) = dim(S_m) + sgn(\alpha). \tag{34}$$

Since Lemma 7 deals with finite dimensional subspaces, the equality of the dimensions of S_{m+1} and S_m means the oneness of the subspaces. Hence, we have the following lemma.

Lemma 8: $S_{m+1} = S_m$ if and only if $\alpha = 0$. Also, $S_{m+1} \supset S_m$ if and only if $\alpha > 0$.

Lemma 8 and Lemma 5 imply that for the case when $\alpha = 0$, we need not carry out learning for such data since the learned function does not vary even after training with the new data, i.e., $f_{m+1} = f_m$. Therefore, we will now deal with the case $\alpha > 0$ only.

Theorem 1: Let s_m, σ, t_m and α be defined as in eqs.(24), (25), (26) and (27),respectively. If $\alpha > 0$, then the new learning operator X_{m+1} can be computed incrementally from the previous learning operator X_m as

$$X_{m+1} = (X_m + Y_m)T^* + (v_m \otimes \overline{e_{m+1}^{(m+1)}}), \tag{35}$$

where

$$v_m = \frac{1}{\alpha}(R\psi_{m+1} - RA_m^* t_m), \tag{36}$$

$$Y_m = -(v_m \otimes \overline{t_m}), \tag{37}$$

Proof: It follows from eqs.(20) and (23) that

$$X_{m+1} = R[TA_m + (e_{m+1}^{(m+1)} \otimes \overline{\psi_{m+1}})]^* U_{m+1}^\dagger. \tag{38}$$

Since U_{m+1} in eq.(38) is non-negative definite and has the structure as shown in Fig.4, we can apply Lemma 1 to eq.(38). Hence, we have

$$X_{m+1} = R[A_m^* T^* + (\psi_{m+1} \otimes \overline{e_{m+1}^{(m+1)}})] \begin{bmatrix} U_m^\dagger + \Gamma & \rho \\ \rho^T & \tau \end{bmatrix}, \tag{39}$$

where Γ, ρ, and τ are defined as below.

$$\Gamma = \frac{1}{\alpha}(t_m \otimes \overline{t_m}), \tag{40}$$

$$\rho = -\frac{1}{\alpha} t_m, \tag{41}$$

$$\tau = \frac{1}{\alpha}. \tag{42}$$

Applying Lemma 2 to eq.(39), we have

$$\begin{aligned} X_{m+1} &= R[A_m^* T^* + (\psi_{m+1} \otimes \overline{e_{m+1}^{(m+1)}})][T(U_m^\dagger + \Gamma)T^* \\ &\quad + (T\rho \otimes \overline{e_{m+1}^{(m+1)}}) + (e_{m+1}^{(m+1)} \otimes \overline{T\rho}) \\ &\quad + \tau(e_{m+1}^{(m+1)} \otimes \overline{e_{m+1}^{(m+1)}})]. \end{aligned} \tag{43}$$

Since $T^*T = I$ and $T^* e_{m+1}^{(m+1)} = 0$, it follows from eqs.(43)and (20) that

$$\begin{aligned} X_{m+1} &= RA_m^*(U_m^\dagger + \Gamma)T^* + (RA_m^*\rho \otimes \overline{e_{m+1}^{(m+1)}}) \\ &\quad + (R\psi_{m+1} \otimes \overline{T\rho}) + \tau(R\psi_{m+1} \otimes \overline{e_{m+1}^{(m+1)}}) \\ &= (X_m + RA_m^*\Gamma + (R\psi_{m+1} \otimes \overline{\rho}))T^* \\ &\quad + (RA_m^*\rho + \tau R\psi_{m+1}) \otimes \overline{e_{m+1}^{(m+1)}}. \end{aligned} \tag{44}$$

Substituting eqs.(40),(41), and (42) into eq.(44), we have

$$\begin{aligned} X_{m+1} &= X_m T^* \\ &\quad + [\frac{1}{\alpha}(RA_m^*(t_m \otimes \overline{t_m}) - R(\psi_{m+1} \otimes \overline{t_m}))]T^* \\ &\quad + [\frac{-1}{\alpha}(RA_m^* t_m - R\psi_{m+1}) \otimes \overline{e_{m+1}^{(m+1)}}] \qquad (45) \\ &= [X_m - (v_m \otimes \overline{t_m})]T^* + (v_m \otimes \overline{e_{m+1}^{(m+1)}}) \qquad (46) \\ &= (X_m + Y_m)T^* + (v_m \otimes \overline{e_{m+1}^{(m+1)}}). \qquad (47) \end{aligned}$$

Hence, the theorem holds. ■

5.3 Incremental computation of f_{m+1}

Theorem 2: When $\alpha > 0$, the learned function f_{m+1} due to $m+1$ sample points can be computed incrementally from the previous learned function f_m and the new sample value $f(x_{m+1})$ as

$$f_{m+1} = f_m + Y_m y^{(m)} + f(x_{m+1})v_m, \tag{48}$$

where Y_m and v_m are given by eqs.(37) and (36), respectively.

Proof: From the general form of eq.(9), we can write the newly learned function f_{m+1} as

† $P_{B,C}$ refers to the projection onto a subspace B along a subspace C.

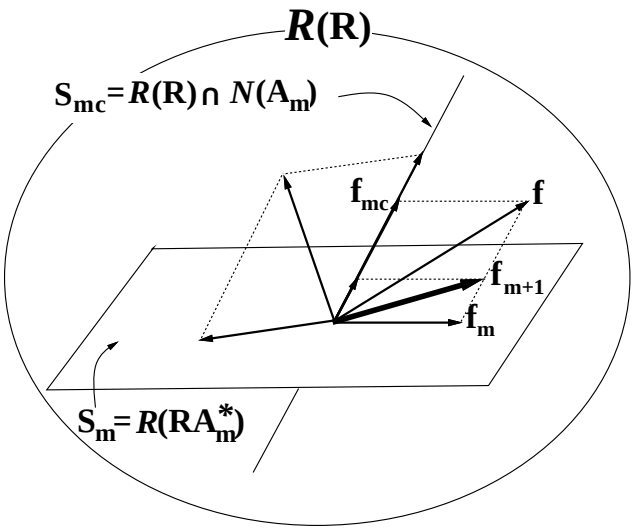

Fig. 5 Relationship betweem f_m and f_{m+1}.

$$f_{m+1} = X_{m+1} y^{(m+1)}. \tag{49}$$

Using eq.(35) of Theorem 1 to compute X_{m+1} in eq.(49), we have

$$\begin{aligned} f_{m+1} &= (X_m + Y_m) T^* y^{(m+1)} + (v_m \otimes \overline{e_{m+1}^{(m+1)}}) y^{(m+1)} \\ &= (X_m + Y_m) y^{(m)} + (y^{(m+1)}, e_{m+1}^{(m+1)}) v_m, \end{aligned}$$

which yields eq.(48). ■

On reducing eq.(48), we can further clarify the relationship between f_m and f_{m+1}, which we will proceed to do in the next theorem. As shown in Lemma 4, f_m is a projection of f onto S_m along S_{mc}. Let f_{mc} be a projection of f onto S_{mc} along S_m.

Now, consider a function ϕ defined as

$$\phi \equiv R\psi_{m+1}. \tag{50}$$

Let ϕ_m amd ϕ_{mc} be the projection of ϕ onto S_m and S_{mc} , respectively. Then, it can be proved, as shown in Appendix B, that α of eq.(18) is equivalent to the value of the function ϕ_{mc} at the new sampled location x_{m+1}, that is,

$$\alpha = \phi_{mc}(x_{m+1}). \tag{51}$$

Using these relations and further reducing eq.(48), we have the following theorem.

Theorem 3: When $\alpha > 0$, it holds that

$$f_{m+1} = f_m + \frac{f_{mc}(x_{m+1})}{\phi_{mc}(x_{m+1})} \phi_{mc}. \tag{52}$$

A proof of the theorem is provided in Appendix C.

A schematic diagram of the relationship is shown in Fig.5. It is clear from eq.(52) and Fig.5 that the newly learned function f_{m+1} can be expressed as the sum of f_m and a scalar multiple of the function ϕ_{mc}. Here, ϕ_{mc} is the projection of ϕ onto S_{mc}, which again depends on the location of the new sample due to eqs.(50) and (4). Another point to be noted is that the term $f_{mc}(x_{m+1})$ which appears in the scalar part of eq.(52) is, in fact, the error between the actual function and the learned function at the new sample point, i.e.,

$$f_{mc}(x_{m+1}) = (f - f_m)(x_{m+1}). \tag{53}$$

6. Conclusion

A framework for incorporating new training data is presented and the concept of incremental learning is formalized to this effect. As an example, the concept of incremental learning is applied to compute the new Wiener learning operator X_{m+1} from the intermediate results of the previous stage like X_m, A_m, and the next sample point x_{m+1}. The incremental approach not only allows computationally efficient incorporation of newly available training data but also outputs learned results as in f_{m+1} , hence, opening possiblities of modifying sampling based on the newly gained data.

References

[1] Albert,A. "Regression and the Moore-Penrose Pseudoinverse", Academic Press, New York and London, (1972) .

[2] Jutten,C. and Chentouf,R. "A new scheme for incremental learning", Neural Processing Letters, vol.2, no.1, pp.1-4, (Jan. 1995) .

[3] Luttrell,S.P. "The use of transinformation in the design of data sampling schemes for inverse problems", Inverse Problems, vol.1, no.1, pp.199-218, (Feb. 1985) .

[4] MacKay,D.J.C. "Information based objective functions for active data selection", Neural Computation, vol.4, pp.590-604, (1992) .

[5] Ogawa,H. and Oja.E. "Projection filter, Wiener filter and Karhunen-Loeve subspaces in digital image restoration", J.Math. Anal. and Appl., vol.114, no.1, pp.37-51, (Feb. 1986) .

[6] Ogawa,H. and Nakamura,N. "Projection filter restoration of degraded images", Proc. IEEE Intl. Conf. on Patttern Recognition, Montreal, pp.601-603, (Jul.30 - Aug.2, 1984) .

[7] Ogawa,H. "A generalized sampling theorem", J. IEICE, Japan, vol.J71-A, no.2, pp.163-170, (Feb. 1988) (In Japanese) ; it's Eng. translation appeared in Elec. and Commn. in Japan, Part 3, Scripta Technnica Inc., vol.72, no.3,pp.97-105,(March 1989) .

[8] Ogawa,H. "Tutorial reviews: Image and signal restoration [1-4]", J. IEICE, Japan, vol.71, no.5-8, pp.491-497, 593-601, 739-748, 828-835, (1988) (In Japanese) .

[9] Ogawa,H. "Inverse problem and neural networks", Proceedings of IEICE 2nd Karuizawa Workshop on Circuits and Systems, pp.262-268, (1989) (In Japanese) .

[10] Ogawa,H. "Neural network theory as an inverse problem", J. IEICE, Japan, vol.73, no.7, pp.690-695, (July 1990) (In Japanese) .

[11] Ogawa,H. "Neural network learning, generalization and over-learning", Proc. ICIIPS'92 (Beijing), vol.2, pp.1-6, (Oct.30 - Nov.1,1992) .

[12] Poggio,T. and Girosi,F. "Networks for approximation and learning", Proceedings of the IEEE, vol.78, no.9, pp.1481-1497, (Sep. 1990) .

[13] Sethu,V. "An incremental approach to training data selection in neural networks", Master thesis, Tokyo Institute of Technology, (Feb. 1995) .
[14] Yamashita,Y. and Ogawa,H. "Properties of averaged projection filter for image restoration", J.IEICE,Japan, vol.J74-D-II, no.2, pp.142-149, (Feb. 1991) (In Japanese) .
[15] Yamashita,Y. and Ogawa,H. "Mutual relations among optimum image restoration filters", J.IEICE,Japan, vol.J75-D-II, no.5, pp.890-898, (May 1992) (In Japanese) .
[16] Zhang,B. T. "An incremental learning algorithm that optimizes network size and sample size in one trial", Proc. of ICNN'94(Orlando, Florida), pp.215-220, (June 28 - July 2, 1994) .

Appendix A: Proof of Lemma 6

Since U_{m+1} is a non-negative matrix and has the structure represented by eq.(16), it can be decomposed as

$$U_{m+1} = \begin{pmatrix} G \\ g^T \end{pmatrix} \begin{pmatrix} G \\ g^T \end{pmatrix}^T, \tag{54}$$

where G is an $m \times m$ matrix and g is an m-dimensional column vector. It follows from the structure of eq.(16) and eq.(54) that

$$Gg = s_m. \tag{55}$$

Solving for g, we have for some u

$$g = G^\dagger s_m + Cu, \tag{56}$$

where C is defined as

$$C = I - G^\dagger G. \tag{57}$$

Since $Cu \perp G^\dagger s_m$, from eqs.(16), (54) and (56), we have

$$\sigma = \|g\|^2 = \|G^\dagger s_m\|^2 + \|Cu\|^2. \tag{58}$$

We also have

$$Cg = C(G^\dagger s_m + Cu) = C^2 u = Cu. \tag{59}$$

Hence, from eqs.(58) and (59)

$$\begin{aligned} \|Cg\|^2 &= \sigma - \|G^\dagger s_m\|^2 \\ &= \sigma - s_m^T G^{\dagger T} G^\dagger s_m \\ &= \sigma - (U_m^\dagger s_m, s_m), \end{aligned} \tag{60}$$

and hence, eqs.(26) and (27) yield

$$\|Cg\|^2 = \alpha. \tag{61}$$

From eqs.(61) and (57), we have

$$\alpha = \|(I - GG^T)g\|^2. \tag{62}$$

Hence, $\alpha = 0$ if and only if $(I - GG^T)g = 0$. In other words, $\alpha = 0$ if and only if $g \in \mathcal{R}(G^T)$. Therefore, $\alpha = 0$ if and only if

$$rank(\begin{pmatrix} G^T & g \end{pmatrix}) = rank(G^T). \tag{63}$$

On the other hand, from eq.(54) and Fig.4, we have

$$rank(U_m) = rank(G^T) \tag{64}$$

and from eq.(54), we have

$$rank(U_{m+1}) = rank(\begin{pmatrix} G^T & g \end{pmatrix}). \tag{65}$$

Looking at eqs.(63), (64) and (65), we conclude that $\alpha = 0$ if and only if

$$rank(U_{m+1}) = rank(U_m). \tag{66}$$

Since $\alpha \geqq 0$ because of Lemma 1, eq.(66) implies Lemma 6. ■

Appendix B: Proof of eq.(51)

From eqs.(27) and (25), we have

$$\alpha = (R\psi_{m+1})(x_{m+1}) - (t_m, s_m). \tag{67}$$

Looking at the second term of the expansion in eq.(67) and using eqs.(24) and (26), we have

$$\begin{aligned} (t_m, s_m) &= (U_m^\dagger A_m R\psi_{m+1}, A_m R\psi_{m+1}) \\ &= (RA_m^* U_m^\dagger A_m R\psi_{m+1}, \psi_{m+1}). \end{aligned} \tag{68}$$

Using eq.(20) and Lemma 4, we can write eq.(68) as

$$\begin{aligned} (t_m, s_m) &= (X_m A_m R\psi_{m+1}, \psi_{m+1}) \\ &= (P_{S_m, S_{mc}} R\psi_{m+1}, \psi_{m+1}) \\ &= (P_{S_m, S_{mc}} R\psi_{m+1})(x_{m+1}). \end{aligned} \tag{69}$$

Substituting eq.(69) in eq.(67) and using the definition of ϕ,ϕ_m and ϕ_{mc}, we have

$$\begin{aligned} \alpha &= (R\psi_{m+1})(x_{m+1}) - (P_{S_m, S_{mc}} R\psi_{m+1})(x_{m+1}) \\ &= \phi(x_{m+1}) - \phi_m(x_{m+1}) \\ &= \phi_{mc}(x_{m+1}). \end{aligned} \tag{70}$$

Appendix C: Proof of Theorem 3

From eqs.(36) and (26),

$$\begin{aligned} v_m &= \frac{1}{\alpha}(R\psi_{m+1} - RA_m^* t_m) \\ &= \frac{1}{\alpha}(R\psi_{m+1} - RA_m^* U_m^\dagger s_m). \end{aligned} \tag{71}$$

Applying eqs.(20) and (24) to eq.(71), we have

$$\begin{aligned} v_m &= \frac{1}{\alpha}(R\psi_{m+1} - X_m s_m) \\ &= \frac{1}{\alpha}(R\psi_{m+1} - X_m A_m R\psi_{m+1}) \\ &= \frac{1}{\alpha}(I - X_m A_m) R\psi_{m+1}. \end{aligned} \tag{72}$$

Using the result of Lemma 2, we write eq.(72) as

$$\begin{aligned} v_m &= \frac{1}{\alpha}(I - P_{S_m,S_{mc}})R\psi_{m+1} \\ &= \frac{1}{\alpha}P_{S_{mc},S_m}R\psi_{m+1} = \frac{1}{\alpha}P_{S_{mc},S_m}\phi. \end{aligned} \tag{73}$$

By definition of ϕ_{mc} and α(eq.(51)), we can reduce eq.(73) to

$$v_m = \frac{\phi_{mc}}{\phi_{mc}(x_{m+1})}. \tag{74}$$

On the other hand, on substituting the value of Y_m from eq.(37) into eq.(48), we have

$$\begin{aligned} f_{m+1} &= f_m - (v_m \otimes \overline{t_m})y^{(m)} + f(x_{m+1})v_m \\ &= f_m - (y^{(m)}, t_m)v_m + f(x_{m+1})v_m \\ &= f_m + [f(x_{m+1}) - (y^{(m)}, t_m)]v_m. \end{aligned} \tag{75}$$

Now, let us look at the term $(y^{(m)}, t_m)$ in the above equation. It can be written as

$$(y^{(m)}, t_m) = (A_m f, t_m) = (f, A_m^* t_m). \tag{76}$$

Assume $f \in \mathcal{R}(R)$, which holds true for Wiener learning. Then, $f = Ru$ for some u. Hence, eq.(76) can be written as

$$(y^{(m)}, t_m) = (Ru, A_m^* t_m) = (u, RA_m^* t_m). \tag{77}$$

Expanding for t_m using eqs.(26) and (24) in eq.(77),

$$\begin{aligned} (y^{(m)}, t_m) &= (u, RA_m^* U_m^\dagger A_m R\psi_{m+1}) \\ &= (u, X_m A_m R\psi_{m+1}). \end{aligned} \tag{78}$$

Using Lemma 2, eq.(78) can be written as

$$\begin{aligned} (y^{(m)}, t_m) &= (u, PR\psi_{m+1}) \\ &= (Pu, R\psi_{m+1}) \\ &= (RPu, \psi_{m+1}), \end{aligned} \tag{79}$$

where P is the projection on S_m along S_{mc}. It can be easily shown that $PR = RP$. Using this result in eq.(79), we have

$$\begin{aligned} (y^{(m)}, t_m) &= (PRu, \psi_{m+1}) \\ &= (Pf, \psi_{m+1}) \\ &= (f_m, \psi_{m+1}). \end{aligned} \tag{80}$$

Using the property of the function ψ_{m+1}, eq.(80) becomes

$$(y^{(m)}, t_m) = f_m(x_{m+1}). \tag{81}$$

Now, using eqs.(74) and (81) in eq.(75), we have

$$\begin{aligned} f_{m+1} &= f_m + [f(x_{m+1}) - f_m(x_{m+1})]\frac{\phi_{mc}}{\phi_{mc}(x_{m+1})} \\ &= f_m + \frac{f_{mc}(x_{m+1})}{\phi_{mc}(x_{m+1})}\phi_{mc}. \end{aligned} \tag{82}$$

■