Neural Networks, Vol. 2, pp. 387–393, 1989
Printed in the USA. All rights reserved.

0893-6080/89 $3.00 + .00

ORIGINAL CONTRIBUTION

A Neural Network Approach to Character Recognition

A. RAJAVELU, M. T. MUSAVI, AND M. V. SHIRVAIKAR

University of Maine

(*Received* 18 *July* 1988; *revised and accepted* 12 *April* 1989)

Abstract—*An application of neural networks in optical character recognition (OCR) is presented. The concept of learning in neural networks is utilized to a large extent in developing an OCR system to recognize characters of various fonts and sizes, and hand written characters. Parallel computational capability helps reduce recognition time which is crucial in a commercial context. The sensitivity of the network is such that small variations in the input do not affect the output and this results in an improvement in the recognition rate of characters with slight variations in structure, linearity, and orientation.*

Keywords—Neural networks, Optical character recognition, Pattern recognition, Walsh function.

1. INTRODUCTION

Character recognition has been defined as the conversion of text characters into machine readable code. The range of applications is wide and includes postal code reading (Lam and Suen, 1988), automatic data entry into large administrative systems, recognition of print and script, automated cartography, banking, and reading devices for the blind. It is an integral part of office automation in the form of text processing. The features which characterize a good optical character recognition (OCR) system are accuracy, flexibility, and speed. In the past several algorithms for character recognition have been developed (Kahan, Pavlidis, & Baird, 1987; Stentiford, 1985). Some of them have been found commercially viable and have gone into production. Despite their high cost, the performances of these systems have been constrained by their dependence on font, size, and orientation. Lately, the emphasis has been to develop a versatile system which should be capable of multifont character recognition and be independent of size and orientation of text characters. The necessity for a system whose performance is independent of text orientation arises from the fact that a slight tilt in the text causes the recognition rate of the existing machines to reduce drastically. This places undue constraints in the positioning of the document to be processed.

One of the critical issues in character recognition is feature selection as the recognition rate is dependent on the choice of features. Every character has some features which distinguish it from the other characters. Some of the prominent features used for character recognition are loops, holes, strokes, vertical lines, cusps, etc. A majority of previous works uses these features as they appeal to the human intuitive logic, since the human eye uses geometrical and topological features for recognition of objects. Grimsdale, Sumner, Tunis, and Kilburn (1958) made one of the earliest feature-based attempts at character recognition where the input is scanned by a flying spot scanner to extract basic features. Further research during the 1960s and 1970s resulted in a variety of improved techniques.

Most of the existing algorithms involve extensive processing on the image before the features are extracted. When the approach is based on a global feature of the character, processing is required at every location in the image. Some techniques include thinning, smoothing, contour analysis, etc. during preprocessing and this results in increased computation time. The delay may be significant when such algorithms are implemented in commercial OCR systems. The solution to this lies in the selection of an algorithm which effectively reduces image processing time. Moreover, the comparison of features so obtained, with those in the database to determine the character, accounts for a considerable amount of the recognition time.

The focal point of this paper is to develop an algorithm which effectively reduces image processing time, while maintaining efficiency and versatility. A methodology for developing an OCR system based on neural networks is described here. Although efforts have already been made to implement neural networks in character recognition (Burr, 1986; Mehr & Richfield, 1987), a complete system which encompasses all features of a practical OCR system is yet to be realized. The key factors involved in the implementation are: an optimal selection of features which categorically defines the details of the character, the number of features, and low image processing time. The parallel computational capability of neural networks ensures a high speed of recognition which is crucial to a commercial environment.

A systematic description of the system and implementation is presented in the following sections. In section 2 the methodology used to extract character features is described. This involves scanning the character and transforming the obtained data into a form suitable for processing by the neural network. This is implemented with orthonormal representations of the character patterns. For this purpose the well known Walsh functions (Beauchamp, 1975; Bennett, 1976; Carl, 1974) are used. The advantages of using Walsh functions are that they reduce computation time. As this is a standard set of functions the implementation is less complex compared to the techniques discussed earlier. In section 3 the construction of a two-layer neural network is described which is followed by a discussion on its functional features and compatibility criteria. The implementation of neural networks in an OCR system is described. The concept of learning in neural networks becomes a powerful tool in its application to OCR. This highlights the possibility of developing a system whose functionality is independent of font, style, and orientation. Characters of different fonts are presented and the neural network is trained to recognize the characters. In section 4 experimental results are presented for various character patterns followed by a discussion on system performance.

2. FEATURE EXTRACTION USING WALSH FUNCTIONS

In this technique the character data is expanded in a set of orthogonal functions. The expansion coefficients form the features of the character. Walsh functions have been used in a number of applications to represent data as a series of orthogonal functions. This process is a form of bandwidth reduction. We have used this method for feature extraction as it effectively reduces complexity without loss of relevant data. In general, the character image undergoes a preprocessing stage before meaningful features are extracted. This usually involves thinning, noise removal, thresholding, etc. In this technique though, the only form of preprocessing required is binarization based on a fixed threshold. The features extracted depend on the intensity distribution of the binarized image. This results in reduced computation time as any of the complex feature extraction procedures discussed earlier is not required.

The general equation is of the form

$$V(x) = \Sigma C_n f_n(x) \qquad \text{for } 0 \leq x \leq 1 \qquad (1)$$

where $V(x)$ is the unknown function which is expanded in a series of n known orthogonal functions $f_n(x)$. The set of coefficients C_n obtained from the above expansion characterize the data represented by the function $V(x)$. In this application $V(x)$ is the intensity distribution function of the character image. Consider two functions $f_i(x)$ and $f_j(x)$. They are said to be orthogonal in the interval $0 \leq x \leq 1$ if

$$\int f_i(x) f_j(x)\, dx = 0 \qquad \text{for } i \neq j \qquad (2)$$

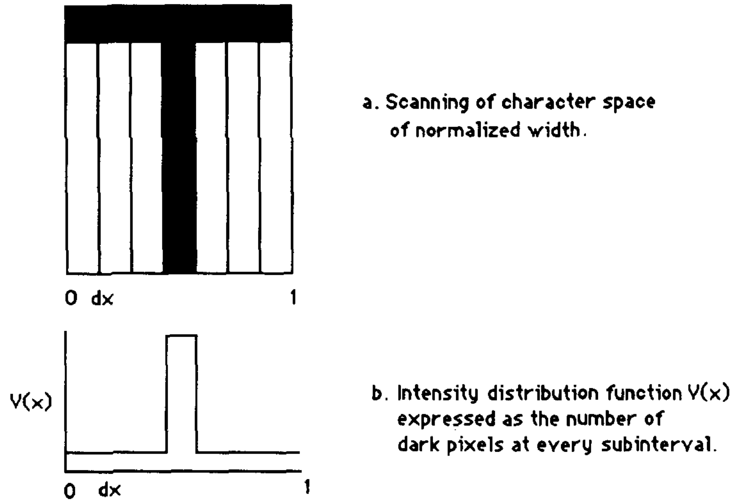
and are normalized over this interval if

$$\int [f_i(x)]^2\, dx = 1 \qquad \text{for all } i. \qquad (3)$$

A set of functions which satisfies both these properties is termed orthonormal. Consider the intensity distribution function $V(x)$ defined earlier. The space in terms of pixel length occupied by the character horizontally is normalized to the interval $0 \leq x \leq 1$. This normalized space is then divided into n subintervals of width dx, where n is the number of features in the horizontal direction. Figure 1(a) shows the intensity distribution for the letter T. The number of dark pixels in the vertical direction represented by $V(x)$ at intervals of dx is plotted in Figure 1(b). This is done by digitizing the image based on a threshold and expressing $V(x)$ as the number of dark pixels in the vertical direction. For the sake of simplicity of analysis let's assume that $V(x)$ represents the normalized function. This can now be expanded in a set of orthogonal functions of the form defined by eqn (1), where $f_1, f_2 \ldots f_n$ are a standard set of orthogonal functions, and $c_1, c_2 \ldots c_n$ are the expansion coefficients which can be individually determined by property of orthogonal functions.

J. L. Walsh (1923) developed a complete set of normal orthogonal functions which are defined over the interval $0 \leq x \leq 1$ and have values $+1$ or -1. These functions have already been used in pattern recognition applications (Beauchamp, 1975; Carl, 1974). These functions may be arranged in pairs of functions having even- and odd-symmetry about the point $x = 1/2$ and increasing numbers of sign changes. The functions are denoted by $f_n(x)$ where n is the number of sign changes in the defined interval. The first few Walsh functions are shown in Figure 2. These functions are generated using the algorithm described in Bennett (1976). The unknowns $c_1, c_2, \ldots c_n$ in (1) are computed as follows. Both sides of (1) are multiplied by $f_m(x)$ and then integrated over the interval $0 \leq x \leq 1$.

$$\int V(x) f_m(x)\, dx = \int \Sigma c_n f_n(x) f_m(x)\, dx. \qquad (4)$$



a. Scanning of character space of normalized width.

b. Intensity distribution function V(x) expressed as the number of dark pixels at every subinterval.

FIGURE 1. Intensity distribution function.

By property of orthogonal functions all terms in the sum except $n = m$ vanish leaving behind

$$\int V(x)f_m(x)\,dx = \int c_m[f_m(x)]^2\,dx. \qquad (5)$$

Since the set of orthogonal functions described earlier is also orthonormal

$$\int [f_m(x)]^2\,dx = 1. \qquad (6)$$

Therefore

$$c_m = \int V(x)f_m(x)\,dx \qquad m = 1, 2 \ldots, n. \qquad (7)$$

The coefficients can be calculated by a number of numerical integration methods. They have values between -1 and $+1$. Each character in the character set results in a unique set of expansion coefficients. This is due to the disparity in the shapes of different characters, which result in varying intensity distributions. Thus it is possible to categorize the characters based on these coefficients. A similar set of expansion coefficients are obtained by expressing the intensity in the horizontal direction as a function of the character height and performing the same operations explained above. The number of Walsh functions required to encode the image data in the orthogonal series expansion depends on the accuracy needed for the application. The computational time increases with increasing number of Walsh functions. In applications of pattern recognition like map and aerial photograph processing every detail in the image has to be preserved. This involves more detailed image analysis. There is a need for more accuracy and hence the number of Walsh functions required is comparatively high. In character recognition the number is constrained as both accuracy and computation time are important. Since the accuracy in this application is not dependent on every detail in the image, fewer Walsh functions are necessary to encode the character data. Various combinations were tried and best results in terms of accuracy and computation time were obtained when we implemented ten Walsh functions. The number of expansion coefficients is equal to the number of Walsh functions chosen. Hence the horizontal and vertical scans result in ten expansion coefficients each. Therefore, a total of twenty expansion coefficients will be used as the input to the neural network.

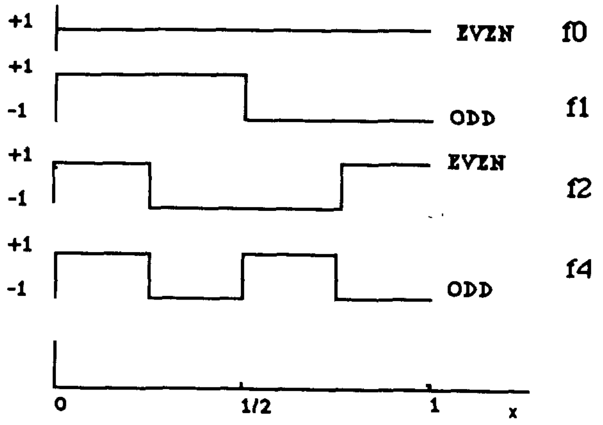


FIGURE 2. The first four Walsh functions showing even- and odd-symmetry about the point $x = 1/2$.

The character image is acquired through a charged coupled device (CCD) camera and stored in the frame buffer. The character is enclosed (justified at the upper left corner) in a square box of specified height and width. The dimensions of the box are chosen so as to accommodate the largest character in the character set taking into consideration all the available fonts. The box is segmented into 10 intervals each in the vertical and horizontal directions. The intervals are dx and dy, respectively. The image is scanned at these intervals and the number of pixels below the selected threshold value is determined. A plot of the number of pixels against distance is shown in Figure 1b. Each character in the character set results in a unique set of coefficients which form the feature vectors for that character.

3. NEURAL NETWORK

3.1. Configuration

The implementation of neural networks for this application eliminates the need for a large database. The recognizing capacity of the neural network lies in the weights interconnecting the nodes in the different layers. We have implemented a layered feed forward network with one hidden layer. As described earlier, the feature extraction process results in a total of twenty features (expansion coefficients), ten each, in the horizontal and vertical directions. Hence the neural network has 20 input units in the input layer. The number of output units is based on the number of characters in the standard character set. If only the alphanumeric characters are considered then a six bit binary word is required for binary representation of every character. If the character set includes special characters, also, then seven bits are necessary. In the neural network, every bit in the binary word is represented by an output unit. Therefore the network has seven output nodes. For selection of hidden nodes the factors to be considered are input nodes and convergence rate. It has been heuristically determined that in order to ensure convergence in our system the minimum number of hidden units must be the same as the number of input units, that is, 20 hidden nodes. Increasing the hidden units causes a corresponding increase in the convergence rate and slight improvement in the recognition rate. However, the number of hidden units is constrained in hardware realization of the neural network. This is explained as follows. Each hidden unit has $n + m$ connections where n and m denote the input and output units, respectively. Hence increasing the hidden units by p results in $p(n + m)$ additional connections, which adds to the complexity of hardware realization. Though not very important in this implementation, it is critical for applications which require a large number of input and output units. The system was tested with 20, 30, 40, and 50 hidden units, and convergence rates for a known number of test patterns were studied. As expected, the convergence rate improved with an increase in the hidden units. Although the convergence rate was better for the 50 hidden node case, the recognition time increased due to increased computations. Therefore the neural network has been implemented with 40 hidden units in our experiments.

3.2. Learning

Learning is implemented using the back-propagation technique. During initialization, the weights interconnecting the various units are assigned small arbitrary values. Every input node is set to the respective expansion coefficient of the first pattern. The seven bit binary code of the character forms the target output vector.

The total input, x_j, to unit j is a linear function of the outputs, y_i, of the units that are connected to j and of the weights, w_{ji} on these connections

$$\begin{aligned} x_j &= \Sigma_i\, y_i w_{ji} \\ j &= 1, 2 \ldots 40 \quad \text{and} \\ i &= 1, 2 \ldots 20 \text{ for hidden layer.} \\ j &= 1, 2 \ldots 7 \quad \text{and} \\ i &= 1, 2 \ldots 40 \text{ for output layer.} \end{aligned} \tag{8}$$

The output of every unit is a nonlinear function of its total input and is defined by

$$y_j = [1 + e^{-x_j}]^{-1} \tag{9}$$

A more detailed description of the back propagation technique can be found in Jones and Hoskins, (1987), Rumelhart, McClelland and the PDP Research Group, (1988), and Rumelhart, Hinton, and Williams (1986). In the first iteration the values of the output nodes are determined using (8) and (9). These are then compared with the target output values and the resulting errors are propagated back to the penultimate layer to compute changes in the weights interconnecting the nodes in the hidden and output layers. These modified weights are then used to change the weights between the input and hidden layer. The second iteration commences with the input vector being the second set of expansion coefficients. The target output vector is set accordingly and the procedure is repeated. Iterations continue until convergence is attained which signifies the completion of learning. The system has been tested with a number of training sets and the results are discussed in the following section.

4. EXPERIMENTAL RESULTS AND SYSTEM PERFORMANCE

In the first session the system was trained with the upper-case alphabets of Times font shown in Figure 3. Each alphabet was scanned once and features of 20 expansion coefficients were extracted and stored. These were presented to the network and iterations were performed. The number of learning trials needed by the network to converge was 1215. The system was tested five times with the characters being presented in a random fashion and recognition was achieved for all alphabets. One exception was the character *B* which the system failed to recognize

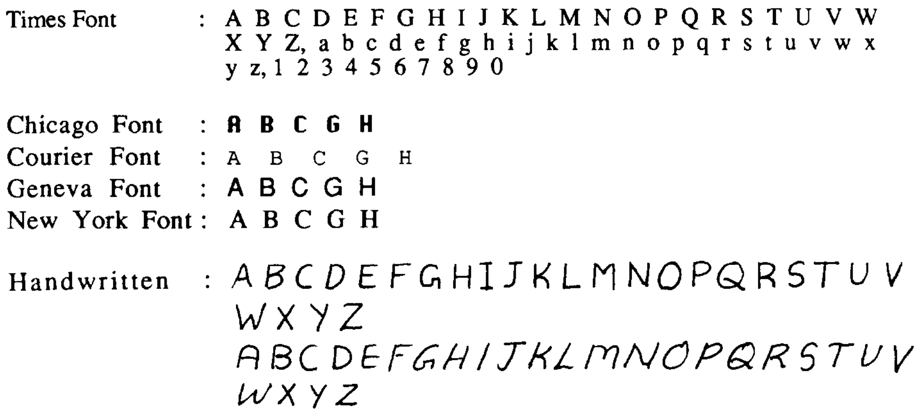

Times Font : A B C D E F G H I J K L M N O P Q R S T U V W X Y Z, a b c d e f g h i j k l m n o p q r s t u v w x y z, 1 2 3 4 5 6 7 8 9 0
Chicago Font : A B C G H
Courier Font : A B C G H
Geneva Font : A B C G H
New York Font : A B C G H
Handwritten : A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

FIGURE 3. Training sets.

twice. This was believed to be due to inconsistency in the expansion coefficients of the original and the test patterns. This was eliminated by training the network with three patterns of each character. There is no specific rule for selecting three patterns, but it was observed that accuracy improved if the system was trained with more number of patterns for each character.

Next, both upper and lower case letters of Times font were used to train the system. As described earlier three patterns of each character were presented. The results of training are summarized in Table 1, which shows the convergence and recognition rates for the various training sets. Initially the system exhibited oscillatory behavior and finally converged in the third attempt. The system failed to distinguish between lower case letter "l" and numeral "1." Both the characters are similar and generate identical expansion coefficients. This was the cause for the oscillatory behaviour during the initial learning stages. By training the system with the character set exclusive of the lower case letter "l," convergence was attained in the first attempt.

In the next experiment five alphabets from four different fonts, Chicago, Courier, Geneva, and New York formed the learning set. These are shown in Figure 3. This resulted in a recognition rate of over 99%. This experiment was restricted to a few characters as it was deemed sufficient, to emphasize the multi-font handling capacity of the system. This result can be extended to all characters in the font. The success of the system in this experiment led us to test for recognition of handwritten characters. The training set for handwritten characters consists of characters of two distinct styles, Figure 3. The system was trained with three patterns of each character. This experiment proved to be successful resulting in a recognition rate of over 98%. However it is to be noted that the system does not generalize for characters with styles different from the ones in the training set.

The highlight of the experimental session was training the system with characters oriented at angles of 60 and 120 to the horizontal, and recognition was achieved for characters with orientation between 60 degrees and 120 degrees. As the expansion coefficients are typical of each character and have very small values (between −1 and +1), a slight variation in the orientation of the character will result in only a small change in the coefficients. As the neural network is immune to small changes in the inputs, the recognition capability of the system is preserved.

It has been established that the network exhibits high recognition rates for all the characters in the training set. For characters not included in the training set, the system performs well for machine printed characters of the same fonts as those in the training set. However, the system is size dependent. Prior to decomposition into features, the character is enclosed in a box whose dimensions are independent of those of the character. Hence a character of different size will result in a change in the features which affects recognition. One way of tackling this drawback would be to make the bounding box size dependent by making it the same size as the character. This resulted in a reduction in recognition rate as the characters "S," "O" (upper-case) and "s," "o" (lower case) generate identical expansion coeffi-

TABLE 1
Experimental Results

Training Sets	Number of Trials for Convergence	Recognition Rate
Times Font Character Set	5,127	97%
Times Font Character Set Without lower-case 'l'	2,950	99%
Multi-Font Characters	2,563	99%
Handwritten Characters	1,153	98%
Misoriented Characters	1,360	98%

cients. The size dependence does not pose a serious problem in a commercial environment as the system can be trained beforehand with characters of expected sizes. Also, the problem can be remedied by adding a length (size) feature to differentiate between upper and lower case characters of the same shape.

Another factor of interest is the configuration of output nodes of the network. It is possible to dedicate every output node to a character. If we consider alphanumeric characters, then the network will have 62 output nodes instead of 7 as selected previously. The advantages of this configuration are improved convergence and response to characters not in the training set. Apart from the constraints of hardware realization this is a much better configuration of the neural network for this application. It was observed that increasing the number of inputs of the neural network, that is, the number of expansion coefficients, improved the accuracy of the system especially for recognition of handwritten characters, the tradeoff being larger convergence rate.

As compared to the existing techniques this system exhibits the following advantages: (a) high speed of recognition, (b) recognition for characters with improper alignment, and (c) font independence. To further highlight the performance aspects of this technique, we shall compare the preprocessing, feature extraction and classification stages of this technique with those of the conventional systems.

Preprocessing in the present technique consists of normalization and binarization only, as compared to thinning, smoothing, contour following, and filtering used in most of the existing techniques. Feature extraction in this approach consists of obtaining the intensity distributions and computing the expansion coefficients (features). The main computation involved here is numerical integration. In terms of complexity and number of operations, this is not as tedious as the other techniques used for feature extraction. For example, in Cash and Hatamian (1987) a method of moments has been described which involves the generation of raw, central, and normalized central moments as features for the character. Another form of feature extraction used in the AT&T Bell Laboratories reading machine (Kahan et al., 1987; Pavlidis, 1986) extracts strokes from a character using a vectorizer which is based on the line adjacency graph. The AT&T method also obtains other features namely, number of holes, their location and size, concavities, crossings of strokes, and endpoints. It is inherent from the above that the techniques used are more complex than the one used in this paper.

Another advantage lies in the use of neural networks in the classification stage. In the conventional techniques the features obtained are compared with the ones in the library set and recognition is achieved based on a variety of classification schemes. This requires a database, whose size and access time depends on the number of characters and fonts. Also, there is a considerable amount of time involved in classification. The neural network approach not only eliminates the need for a large database, but also reduces the classification time.

5. CONCLUSION

In this paper we have successfully shown that Neural Networks can be implemented in Optical Character Recognition. The performance of the system is remarkable as it effectively reduces the size of the database. The system has been designed to accept new patterns. This illustrates the cognitive characteristics of this system, which can expand the scope of recognition. In a user environment the system is not constrained by factors like font dependence and need for perfect alignment of text material which makes this system a practical feasibility.

REFERENCES

Beauchamp, K. G. (1975). *Walsh Functions and their applications* (pp. 171–178). London: Academic Press.

Bennett, W. R., Jr. (1976). *Scientific and engineering problem solving with the computer* (pp. 67–72). Engelwood Cliffs, NJ: Prentice Hall.

Burr, D. J. (1986). A neural network digit recognizer. *Proceedings of IEEE International Conference on Systems, Man and Cybernetics* (Vol. 1, pp. 1621–1625). Piscataway, NJ: IEEE.

Carl, J. W. (1974). On the use of Walsh Functions in man-made and biological pattern recognition systems. In H. G. Schreiber & G. Ferell Sandy (Eds.), *Application of Walsh Functions and sequency theory* (pp. 9–26). New York: IEEE.

Cash, G. L., & Hatamian, M. (1987). Optical character recognition by the method of moments. *Computer Vision, Graphics and Image Processing*, **39**, 291–310.

Grimsdale, R. L., F. H. Sumner, C. J. Tunis, & T. Kilburn (Eds.). (1958). A system for automatic recognition of patterns, *Proceedings of IEE* (Vol. 106B, pp. 210–221). London: IEE.

Jones, W. P., & Hoskins, J. (1987). Back propagation: A generalized learning rule. *Byte*, **12**, 155–158.

Kahan, S. T., Pavlidis, T., & Baird, W. (1987). On recognition of printed characters of any font and size, IEEE *Transactions on Pattern Recognition and Machine Intelligence*, **PAMI-9**, 274–285.

Lam, L., & Suen, C. Y. (1988). Structural classification and relaxation matching of totally unconstrained handwritten zip-code numbers. *Pattern Recognition*, **21**, 19–32.

Mehr, D., & Richfield, S. (1987). Neural net application to optical character recognition. *IEEE First International Conference on Neural Networks*, **4**, 771–776.

Pavlidis, T. (1986). A vectorizer and feature extractor for document recognition, *Computer Vision, Graphics and Image Processing* **35**, 111–127.

Rumelhart, D. E., McClelland, J. L. and the PDP Research Group. (1988). Learning internal representations by error propagation. In J. A. Feldman, P. J. Hayes, & D. E. Rumelhart (Eds.), *Parallel distributed processing* (Vol. 1, pp. 318–363). MIT Press.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back propagating errors. *Nature* **323**, 533–536.

Stentiford, F. W. M. (1985). Automatic feature design for optical character recognition using evolutionary search procedure. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, **PAMI-7**, 349–355.

Walsh, J. L. (1923). A closed set of normal orthogonal functions. *American Journal of Mathematics*, **45**, 5–24.