Significance of Opportunity

An optical character recognition (OCR) system is needed that can reliably operate in a wide variety of conditions to enable automated road signage recognition. Current systems have limited capability and depend on fixed angle, range and lighting. A system based on neural network pattern recognition of images at a variety of conditions could be trained to recognize text for use in real world applications.

Software has been successfully developed that can detect typed and even handwritten characters. However, the environments that these systems are developed and tested under often include controlled angle, range and lighting. Thus, this technology is not yet usable in applications where these variables cannot be controlled.

Artificial neural networks, modeled after biological nervous systems, are versatile computer structures composed of simple elements operating in parallel. As in nature, the network function is determined largely by the connections between elements. CRIA Corp. can train a neural network to perform a particular function by adjusting the values of the connections (weights) between elements. A sufficiently large neural network can represent any continuous or discontinuous function[1] (this includes time/frequency domain transformations). An outline of how neural networks will be applied to OCR is shown in Figure 1.

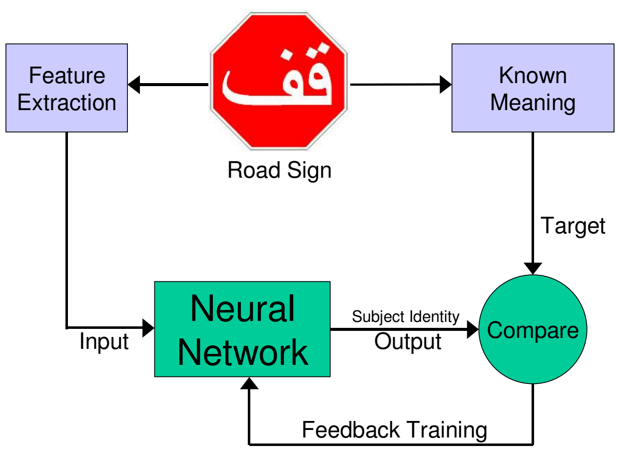


Figure 1: Flow chart showing how a neural network, given handwritten text can be trained to recognize patterns that identify characters in such text.

Commonly neural networks are adjusted, or “trained,” so that a particular input leads to a specific target output. The network is adjusted based on a comparison of the output and the target until the network output matches the target. Typically, many such input/target pairs are used, in this supervised learning, to train a network. Neural networks have been trained to perform complex functions in various areas of pattern recognition including speech, vision, identification, classification and control systems.[2]

Technical Objectives

To prove the concept of a neural network based OCR system, CRIA Corp. will design, build and test a prototype system in Phase I. Sign features will be input into neural networks for training and testing. Through repetitious training and testing, various neural network configurations and feature extraction techniques can be quantitatively and objectively compared.

Outline

In Phase I, CRIA Corp. will design, train and test a neural network that can recognize road signage. Once trained with the trial data, the neural network will be able to take handwritten characters and output which subject is being shown. Testing will be conducted on data not used for training to prove the performance of the system.

In Phase II, the system will be further developed through extensive data collection by CRIA Corp.. This will provide both a larger database of subjects and a comprehensive variety of conditions. A portable system will then be developed to demonstrate the effectiveness of the technology for live translation.

Figure 2: Signage OCR in military scenarios will require the use of various languages. Ability to recognize letters, phrases, symbols, shapes and numbers will need to be flexible to maximize the effectiveness of such a system.

Feature Extraction

Before input into a neural network, feature extraction techniques will be applied to the image data. While a neural network has the computational capability to extract information itself from the raw image data, the optimum approach will involve processing the data before it reaches the neural network to provide features that are highly correlated with the differences between the image patterns to be identified.

Feature extraction techniques to be explored in Phase I include:

- Fourier Transform (FFT)
- Wavelets
- AR Models
- Complexity measures
    - Optimal AR model order
    - Spectral Entropy
    - Approximate Entropy
    - Embedded Space Decomposition
    - Fractal Dimension
    - Global Field Strength
    - Global Frequency of Field Changes
    - Spatial Complexity

Many of these techniques are traditionally considered time/frequency methods for single-channel time-varying signals, however they are just as applicable to images of Arabic text. One example is a windowed Fourier transform where repetitive frequency domain calculations are made on data in overlapping windows, thereby producing spatial frequency information at various image spatial samples. Wavelets are another viable option. The Discrete Wavelet Transform (DWT) can be computed as a series of filters as shown in Figure 3.

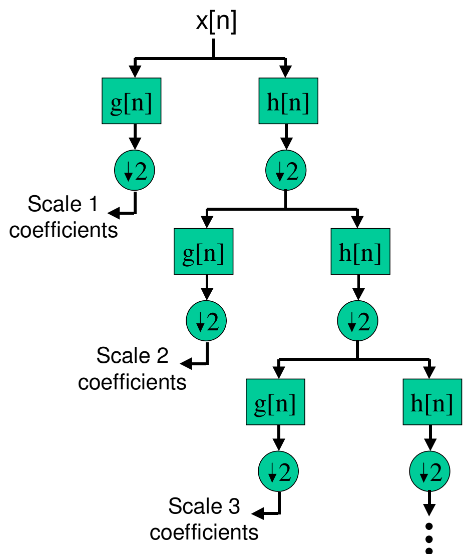


Figure 3: Implementation of the Discrete Wavelet Transform.

The filtering and downsampling cycle shown can be repeated until there is only one sample remaining. Thus, the maximum number of resulting DWT coefficients is equal to the number of samples in the original signal. The computationally expensive Continuous Wavelet Transform (CWT) can produce many more coefficients, however the resulting closely spaced scales are highly correlated and it can be shown that no information is lost (i.e., using an inverse DWT, the original signal can be reconstructed).

A further discrete method, called Wavelet Packets, allows the DWT frequency bands (coefficient scales) to be further subdivided. This may be useful, for example, in OCR if the DWT has a scale that covers a wide band of spatial frequencies, then using Wavelet Packets, one could further break this down into smaller spatial frequency ranges. However, the cost of such an operation is reduced resolution in the spatial-domain (generally in all spatial/spatial-frequency algorithms an increase in the resolution of one results in decrease in resolution of the other).

Comment [DR1]:

Another possible feature for OCR is the use of Autoregressive (AR) techniques. In AR techniques, a model is created where a current amplitude can be predicted from N past amplitudes where the model order is N. Thus the model can be represented as:

$$x_{i,e}(t) = -\sum_{i=1}^{p} a_{i,e} x_{i,e}(t-i)$$

where $a_{i,e}$ is the i^{th} order AR coefficient for channel/position e. These AR coefficients can be used as features. To obtain these coefficients, image data is generally windowed into blocks of data with more than N samples. Then, as the value of t is shifted through the window of data, we obtain numerous model equations which allow us to compute optimum AR coefficients. Thus, these AR coefficients can be used to represent features specific to a certain character.

A final category of feature extraction methods that shows promise for OCR is complexity. Examples include optimal AR model order, Spectral Entropy, Approximate Entropy, Embedded Space Decomposition, Fractal Dimension, Global Field Strength, Global Frequency of Field Changes, and Spatial Complexity.

One method of quantitatively representing signal/image complexity is to find the optimal order of an autoregressive (AR) model for that signal/image. Several different techniques exist for estimating this optimal order, *p* (as seen in the equation shown above for computing AR coefficients). These include Final Prediction Error (FPE), Akaike's Information Criterion (AIC), Schwarz Information Criterion (SIC) [also known as Bayesian Information Criterion (BIC) or the Schwarz Bayesian Criterion (SBC)], and Residual Error.

The AIC algorithm[3] is:

$$AIC(p) = N \ln \det(\hat{\Sigma}_p) + 2d^2 p$$

where d is the number of inputs, p is the model order and $\hat{\Sigma}_p$ is the estimated error covariance matrix.

The SIC algorithm[4] is:

$$SIC(p) = \ln\left|\hat{\Sigma}\right| + \frac{\ln T}{T} p$$

where T is the number of observations, p is the model order and $\hat{\Sigma}_p$ is the estimated error covariance matrix.

The FPE algorithm[5,6] is:

$$FPE(p) = \varepsilon_p \left(\frac{N + p + 1}{N - p - 1}\right)$$

where N is the number of data samples used, p is the model order and ε_p is the prediction error power.

Spectral Entropy[5] is given by:

$$H = \sum_f p_f \log(1/p_f)$$

where *f* is frequency and p_f is a normalized Power Spectral Density (PSD) function. One method of obtaining a PSD is using the Fourier Transform.

Approximate Entropy is given by[5]:

$$ApEn(m,r,N) = \Phi^{m+1}(r) - \Phi^{m}(r)$$

where:

$$\Phi^{m} = \log[P\|u_{jm} - u_{im}\| \le r]$$

and where *r* is a fixed tolerance parameter.

One method[5] of representing embedded space decomposition is using a fractional spectral radius (FSR):

$$FSR(j) = \frac{\sum_{i=0}^{j} \sigma_i^2}{\sum_{l=0}^{m} \sigma_l^2}$$

where σ_i^2 are the eigenvalues of $\boldsymbol{U}^T\boldsymbol{U}$ and $\boldsymbol{U}$ is an embedding matrix made up of vectors $u_1^T,...,u_{N-(i-1)}^T$ such that $u_i = (x_i, x_{i+j},...,x_{i+(m-1)j})^T$. Thus u_i is the embedding vector whose elements are *m* samples taken at intervals of *J* samples along the observed time series.

Fractal Dimension is a chaotic method of estimating signal/image complexity. Several fractal dimension algorithms have been tested, including algorithms by Petrosian[7] and Katz[8], however an algorithm by Higuchi[9,10,11,12,13,14] appears to be the most widely used. Under Higuchi's algorithm, one finds the Fractal Dimension, D_F, by a linear least-squares best fit as follows:

$$D_F = \frac{n * \sum\left(x_k * y_k\right) - \sum x_k \sum y_k}{n * \sum\left(x_k^{\,2}\right) - \left(\sum x_k\right)^2}$$

where: $y_k = \ln L(k)$, $x_k = \ln(1/k)$, $k = k_{\min},...,k_{\max}, Q$, Q is n minus the total number of different values of k in the interval [k_{min} k_{max}] and:

$$L_m(k) = \frac{\sum_{i=1}^{\left\lfloor \frac{N-m}{k} \right\rfloor} \left| x(m+ik) - x(m+(i-1)k) \right| (n-1)}{\left\lfloor \frac{N-m}{k} \right\rfloor k}$$

A measure of global field strength, (Σ), is defined as:

$$\sum = \frac{1}{L}\sqrt{\frac{\sum_n \|u_n\|_2^2}{16L}}$$

where u_n is the row vector ofa 2-D video image, L is the length of the signal (# of rows) and $\| \,.\, \|_2$ is the 2-norm.

A measure of global frequency of field changes, (Φ), is defined as:

$$\Phi = \frac{1}{2\pi}\sqrt{\frac{\sum_n \|u_n\|_2^2}{\sum_n \left\|(u_n - u_{n-1})/\Delta t\right\|_2^2}}$$

where Δt is the sampling period, u_n is the row vector of an image and $\| \,.\, \|_2$ is the 2-norm.

A measure of spatial complexity, (Ω), is defined as:

$$\log\Omega = -\sum_{i=1}^{16} \xi_i \log \xi_i$$

where ξ represents the normalized ($\xi_i = \lambda_i / \Sigma_i \lambda_i$, where λ_i for i=1:16 represents the eigenvalues of C) eigenvalues of C:

$$C = \frac{1}{L}\sum_n u_n u_n^T$$

where u_n is the row vector of an image and L is the length of the signal (# of rows).

Overall, many feature extraction methods will be explored to find which method or combination of methods contains the best data for a neural network to be able to identify characters. It is important to note that the most likely input to an optimum neural network will be data from several feature extraction methods applied to the same image.

Neural Network: Design and Training

Neural networks have proven to be a powerful tool in complex pattern recognition. A neural network is composed of a number of nodes connected by links. Each link has a numeric weight associated with it. Weights are the primary means of long-term storage in neural networks, and learning generally takes place by updating weights. Some of the nodes are connected to the external environment, and can be designated as input or output nodes. The remaining nodes (called hidden nodes) are connected to inputs, outputs and each other. The weights are modified by the training algorithm in an attempt to bring the network's input/output behavior more into line with that of the environment providing the actual input. In this system, weights will be modified so that inputs recorded from a specific subject cause the network to output recognition of that subject.

Each node has a set of input links from other nodes, a set of output links to other nodes, a current activation level and a means of computing the activation level at the next step in time, given its inputs and weights. The idea is that each node does a local computation based on inputs from its neighbors, but without the need for any global control over the set of nodes as a whole.

As applications for neural networks are greatly expanding, commercial tools are becoming available to improve their use. In this system, CRIA Corp. will use MATLAB with the neural network toolbox, available from MathWorks Inc., for the design, training and testing of the network. This software will allow CRIA Corp. to implement and test a variety of neural network parameters in an efficient manner.

An example of a neuron is shown in Figure 4. The scalar input p_j is transmitted through a connection that multiplies its strength by the scalar weight $w_{i,j}$, to form the product $w_{i,j}p_j$ *(vector* $\mathrm{W}_i p_j$*)*. Here the weighted input $\mathrm{W}_i p_j$ is the only argument of the transfer function *f*, which produces the scalar output *a*. The neuron on the right has a scalar bias, *b*. The bias may be viewed as simply being added to the product $\mathrm{W}_i p_j$ as shown by the summing junction or as shifting the function *f* to the left by an amount *b*. The bias is much like a weight, except that it has a constant input of 1.

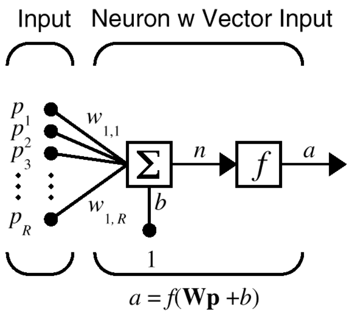


Figure 4: Diagram of a Single Node

The transfer function's input *n*, again a scalar, is the sum of the weighted input vector $W_i p_j$ and the bias *b*. This sum is the argument of the transfer function *f*. Several examples of transfer functions are shown in Figure 5. Here *f* is a transfer function, typically a step function or a sigmoid function, that takes the argument *n* and produces the output *a*. In this project, the input layer and hidden layers will use sigmoid transfer functions, but the output layer will use a linear transfer function so that outputs can be of any value. Note that $w_{i,j}$ and *b* are both adjustable scalar parameters of the neuron. In this system, these parameters will be adjusted by the training algorithm so that the network will produce the proper output (subject identification) for the given input (processed image data).

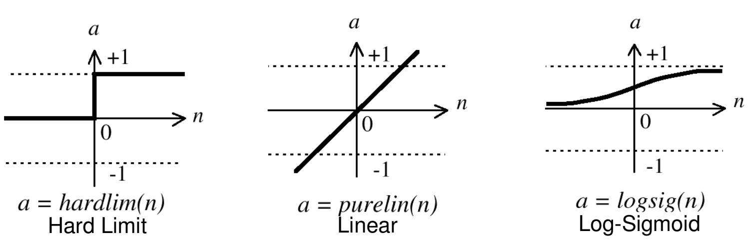


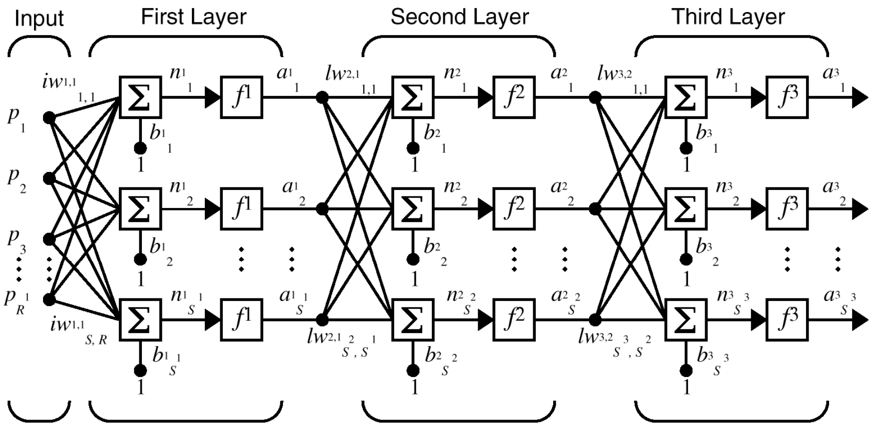
Figure 5: Examples of three common transfer functions

To build a neural network, to perform some task, one must first decide how many nodes are to be used, what kind of nodes are appropriate, and how the nodes are to be connected to form a network. One then initializes the weights of the network, and trains the weights using a learning algorithm applied to a set of training examples for the task. The use of examples also implies that one must decide how to encode the examples in terms of inputs and outputs of the network.

The input/output format of the neural network must take into consideration the desired number of inputs, outputs, nodes, layers and the computation time needed to train the network. Without actually constructing the network, the ideal number of nodes and layers in unknown, however we do know that there is a limit to performance improvement with increase in network size. For example, if a network is too small, it will be incapable of representing a given function and if it is too large, it will be able to memorize all trained examples (by forming a large lookup table), but will not generalize to inputs it has not see before. While there is a limit to the desired number of nodes and layers, the optimal amount of input information is infinite. In other words, the more inputs we use, the more information the neural network will have to formulate an accurate output. However, because of the exponential growth characteristic of neural networks, a computer cannot be expected to train a network with millions of inputs. Therefore, it is obvious that a method must be outlined to properly balance the number of inputs and nodes to create optimal results under reasonable computation times.

The network will have an output for each subject. For training, the output will be set to 0 for all outputs that do not correspond to the subject being used and to 1 for the output that does correspond to the current subject. During testing the highest output will be the most likely subject and a confidence level can be computed by looking at how many outputs were well above 0.

Now that the input and output of data with the neural network has been laid out, the internal structure can be outlined. The internal structure of a feed-forward neural network, shown in Figure 6, consists of a given number of layers each containing a certain number of nodes which are selectively interconnected with nodes of neighboring layers. First, it is important to know that with one sufficiently large hidden layer it is possible to represent any continuous function and with two layers discontinuous functions can be represented. The number of nodes needed in each layer is an unknown. It can be shown that, for n inputs, $2^n / n$ hidden nodes (2^n weights) are needed to represent all boolean functions of the inputs, however most problems can be solved with many fewer nodes. The third variable for network size is the interconnectivity, or the pattern of connections between in outputs of one layer and the inputs of the next layer. Ideally, a network would be fully interconnected, where every output from one layer is connected to every input of the next layer. However, when computational power is a limitation as in this case, the interconnectivity can be selectively set up in ways that will greatly reduce network size. For example, between the input layer and the first hidden layer, the network could be grouped into frequencies at a specific time, where the nodes would only be interconnected within that group; then between the first hidden layer and the second hidden layer the nodes could be fully interconnected. The optimal combination of number of layers, number of nodes and interconnectivity for this application can only be determined through the actual training and testing of the neural network that will be conducted in Phase I.



$$\mathbf{a}^1 = \mathbf{f}^1(\mathbf{IW}^{1,1}\mathbf{p}+\mathbf{b}^1) \qquad \mathbf{a}^2 = \mathbf{f}^2(\mathbf{LW}^{2,1}\mathbf{a}^1+\mathbf{b}^2) \qquad \mathbf{a}^3 = \mathbf{f}^3(\mathbf{LW}^{3,2}\mathbf{a}^2+\mathbf{b}^3)$$

$$\mathbf{a}^3 = \mathbf{f}^3(\mathbf{LW}^{3,2}\,\mathbf{f}^2(\mathbf{LW}^{2,1}\mathbf{f}^1(\mathbf{IW}^{1,1}\mathbf{p}+\mathbf{b}^1)+\mathbf{b}^2)+\mathbf{b}^3)$$

Figure 6: Structure of a multi-layer feed-forward neural network.

The neural network described so far has been outlined as a multilayer feed-forward network that will be trained with the back-propagation algorithm. Feed-forward describes that connections between nodes are always unidirectional with no cycles. In back-propagation learning, example inputs are presented to the network and if the network computes an output that matches the target, nothing is done. If the network computes an output that is different from the target (i.e., if the network computes an output corresponding to the wrong character), then the weights are adjusted to reduce this error. The algorithm must determine all of the weights that contributed the incorrect output so that it can divide blame among them. There are many variations of the back-propagation algorithm. The simplest implementation of back-propagation learning updates the network weights and biases in the direction in which a performance function decreases most rapidly (the negative of the gradient). This performance function evaluates similarity between network output and desired output (i.e., mean squared error). One iteration of this algorithm can be written:

$$X_{k+1} = X_k - \alpha_k g_k$$

where X_k is a vector of current weights and biases, g_k is the current gradient, and α_k is the learning rate. There are two different ways in which this gradient descent algorithm can be implemented: incremental mode and batch mode. In the incremental mode, the gradient is computed and the weights are updated after each input is applied to the network. In the batch mode all of the inputs are applied to the network before the weights are updated. Another training variable is the use of momentum. Momentum allows a network to respond not only to the local gradient, but also to recent trends in the error surface. Acting like a low pass filter, momentum allows the network to ignore small features in the error surface, thus preventing is from becoming fixed at a shallow local minimum.

Manipulation of the following variables followed by neural network training and testing will allow an understanding of the optimal parameters:

- Input data type
- Number of Layers
- Number of Nodes in Each Layer
- Interconnectivity
- Training Algorithm and Parameters
- Output Threshold

While predictions can be made about how each of these parameters will affect the performance of the network, the optimal balance of them can only be determined by training the neural network under many variations. In Phase I, CRIA Corp. will, by trying several variations of the parameters, show conclusive results that a neural networks can recognize road signs. In Phase II, a large variety neural network parameters and a large selection of different characters written by different subjects under various conditions will be used so that performances can be evaluated and compared to create a system with maximum performance.

Testing and Evaluation of Results

Once the network has been trained using a database of sign images, it can be used to recognize subjects in additional samples not used for the training. Performance can be tabulated by comparing the output of the neural network to the known subject in the trial and percentage of error will be calculated. This will show proof of principle for the system.

Summary

In Phase I, CRIA Corp. will train and test a neural network to recognize road signage. In Phase II, CRIA Corp. can then develop an optimized neural network road signage recognition system that is capable of performing in any application.

Work Plan

A work plan has been developed which will prove the feasibility of the system. CRIA Corp. will research and test a variety of promising feature extraction techniques and neural network configurations on road sign images. A summary of will be provided in a final report summarizing usefulness for this application.

The Phase I and Phase I Option work plan has been divided into tasks:

1) Review Project Requirements & Conduct Extensive Literature Review
2) Acquire Road Signage Data
3) Compare Feature Extraction Methods
4) Train & Test Neural Networks (Option)
5) Compile Results & Publish Final Report

Task 1: Review Project Requirements & Conduct Extensive Literature Review

The initial task in Phase I will require the review of the project requirements through discussions with the US Army and further review of the solicitation, proposal and related work. This will ensure research to be conducted will meet the needs of the US Army to the fullest extent.

In addition, an extensive literature review will be conducted. In addition to any recent developments specific to OCR, the general topics of image processing, feature extraction methods and neural networks will be reviewed. Further information may be discovered recommending additional techniques to test as well as data on optimization of the proposed techniques for this application.

Task 2: Acquire Road Signage Data

Another task to be initiated at the onset of the project will be to acquire data to be used in Phase I. In addition, any required pre-processing will be conducted. For example, data will be imported in to MATLAB and image samples will be matched with known character identifications. In addition, data sorting, windowing or resampling may be conducted.

Task 3: Compare Feature Extraction Methods

Once the first two tasks have been completed, feature extraction techniques will be compared. This will be based on related literature, theoretical analysis and testing on database data. Factors such as computational complexity, statistical significance, number of outputs and correlation with other features will be explored. Variables for individual feature extraction techniques will be examined to narrow the number of neural network tests to be conducted (e.g., order for AR models).

Task 4: Train & Test Neural Networks

Once feature extraction techniques have been explored, neural networks will be built, trained and tested with the processed database character images. Results will be compiled for a variety of feature extraction method combinations and neural network configurations. These results will be used to evaluate the performance of the initial system and the success of the Phase I project. In addition, redundant testing will be conducted where promising feature extraction methods and neural network configurations are explored with slight parameter changes. This task comprises the Phase I Option.

Task 5: Compile Results & Publish Final Report

The final task for Phase I will be to compile all of the testing results and publish a final report documenting system performance and recommendations for Phase II. The final report is the sole deliverable for this project.

Related Work

The principal investigator is an expert in pattern recognition with applications similar to the proposed system for road signage recognition. Related work includes a dissertation as well as an NIH SBIR applying neural networks for pattern recognition using Electroencephalogram (EEG, i.e. scalp voltages) to detect pain in human subjects. Very similar to this application, data was collected from human subjects for training and testing of a neural network. Prior to input to the neural network, feature extraction was conducted with methods such as Fourier, wavelet, autoregressive and various complexity measures.

Additional pattern recognition work by the principal investigator has been conducted applying neural networks to brain-computer interfacing (BCI) and cardiology signals. In BCI, data is collected from a human subject through EEG or implanted electrodes and the pattern recognition system needs to process physiological data produced by the human subject related to a mental state in order to produce a predictable output for applications such as mouse or wheelchair control. Cardiology systems were designed for detecting arrhythmias for such applications as automatic implanted defibrillators.

Relationship with Future Research

In Phase I, CRIA Corp. will design, train and test a neural network that can recognize road signs. Once trained with the trial data, the neural network will be able to take character image data and output what character they correspond to. Testing will be conducted on additional road sign data to prove the effectiveness of the network.

In Phase II, the system will be further developed by extensive training with a large database of Arabic text characters covering an exhaustive list of variables and a significant number of writers. The neural network will then be optimized to provide a system that can accurately identify domestic and foreign road signs in a variety of conditions. A portable system will then be developed to demonstrate the effectiveness of the technology with real-time acquisition/recognition.

Commercialization Strategy

The system is expected to have extensive commercial applications. A system that can recognize characters and symbols in a realistic variety of conditions would have far-reaching effects on the state-of-the-art for OCR by expanding the field to applications beyond written or type text. Applications for an effective road sign detection system include:

- Civilian Vehicle Driver-Assistance Technologies
- Railroad Conductor Assistance and Safety Maximization
- Law Enforcement Automatic License Plate Checking
- Intelligence Gathering
- Commercial Text Digitization Applications
- Unmanned Vehicle Navigation & Traffic Law Obedience Capability

Key Personnel

Dan Rissacher will be the principal investigator on this project. He has 3 years of experience working exclusively on SBIR contracts for North Dancer Labs, Inc. This work has included Phase I and Phase II projects, particularly for the Department of Defense. His role included principal investigator, proposal and report writer and design engineer for electronics, software and optics. Mr. Rissacher has earned a B.S. in Computer Engineering through Clarkson University, a M.S. in Electrical and Computer Engineering through Georgia Institute of Technology and is currently a Ph.D. Candidate at Clarkson University with an expected 2007 graduation. He has experience as an instructor in Advanced Digital Circuit Design at Clarkson University as well. Mr. Rissacher is also a First Lieutenant in the Vermont Air National Guard where he serves as an F-16 Fighter Pilot and as the unit's Electronic Combat Pilot. His security clearance level is TS-SCI.

Sanjin Bicanic will act as chief software engineer for this project. He has 3 years of experience as a software engineer with Tellabs and is an expert with programming languages useful to this project such as MATLAB and C++. Mr. Bicanic earned his B.S. in Software Engineering from Clarkson University in 2004.

Facilities & Equipment

CRIA Corp. is located in Winooski, Vermont. Facilities include 1,000 square feet of laboratory and office space with contingency plans to expand/relocate in the same area. All basic equipment and facilities are available, however some additional specialized equipment and software will be necessary for this project. The location meets all local and federal requirements for environmental concerns related to this work.

A data acquisition and signal processing workstation along with the proper software will be needed. MATLAB from MathWorks Inc. with the proper Toolboxes would be adequate for all data processing and testing. All equipment would be portable for future demonstrations, presentations and field-testing.

Subcontractors & Consultants

No consultants or subcontractors are needed on this project.

Prior, Current, or Pending Support of Similar Proposals

There is no prior, current or pending support of similar work.

References

[1] S.J. Russell, Peter Norvig, "Artificial Intelligence: A Modern Approach", Prentice Hall (1995).

[2] H. Demuth, Mark Beale, "Neural network toolbox for use with MATLAB, Users guide Version 3", Mathworks Inc. (1998).

[3] Anderson, C. W., Stolz, E., & Shamsunder, S (1998). Multivariate Autoregressive Models for Classification of Spontaneous Electroencephalogram During Mental Tasks. *IEEE Transactions on Biomedical Engineering*, vol. 45, no. 3, pp. 277-286.

[4] Ozcicek, O. & McMillin, W. D. "Lag length selection in vector autoregressive models: Symmetric and asymmetric lags." Retrieved March 20, 2003 from: http://www.bus.lsu.edu/economics/papers/pap97_27.pdf

[5] Rezek, I. A. & Roberts, S. J. (1998). "Stochastic Complexity Measures for Physiological Signal Analysis." IEEE Transactions On Biomedical Engineering, Vol. 45, No. 9, September.

[6] Kay, S. M. & Marple, S. L. (1981). "Spectrum analysis—A modern perspective," *Proc. IEEE*, vol. 69, pp. 1380–1419, Nov.

[7] Petrosian, A. (1995). "Kolmogorov Complexity of Finite Sequences and Recognition of Different Preictal EEG Patterns," *Proc. IEEE Symposium on Computer-Based Medical Systems*, pp. 212-217.

[8] Katz, M. (1998). “Fractals and the analysis of waveforms,” *Comput. Biol Med.*, vol. 18, no. 3,pp. 145-156.

[9] Higuchi, T. (1988). ”Approach to an irregular time series on the basis of the fractal theory”, *Physica D*, vol.31, pp.277-283.

[10] Ciszewski, J., Klonowski, W., Stepien, R., Jernajczyk, W., Karlinski, A. & Niedzielska, K. (1999). "Application of Chaos Theory for EEG-signal Analysis in Patients with Seasonal Affective Disorder", *Med. Biol. Eng. Comput.*, vol. 37, Supplement 1, pp. 359-360.

[11] Klonowski, W., Olejarczyk, E. & Stepien, R. (2000). Nonlinear Quantifiers of EEG-signal Complexity, *Proceedings of 2000 International Symposium on Nonlinear Theory and its Applications (NOLTA'2000)*, Dresden, Germany, 261-264.

[12] Klonowski, W., Olejarczyk, E. & Stepien, R. (2001). Chaocity and Dimensional Complexity of EEG-signal, *Proceedings of 2000 International Symposium on Nonlinear Theory and its Applications (NOLTA'2001)*, Tokyo, Japan.

[13] Esteller, R., Vachtsevanos, G., Echauz, J. & Litt, B. (1999). "A Comparison of Fractal Dimension Algorithms Using Synthetic and Experimental Data," *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS 1999)*, Orlando, FL, May 30 - Jun. 2.

[14] Accardo, A., Affinito, M., Carrozzi, M. & Bouquet, F. (1997). “Use of the fractal dimension for the analysis of electroencephalo-graphic time series”, *Biol.Cybern.* 77, 339-350, 1997.