Diagnosing diabetes using neural networks on small mobile devices

Oğuz Karan a,*, Canan Bayraktar a, Haluk Gümüşkaya b, Bekir Karlık c

a Haliç University, Department of Computer Engineering, Şişli, İstanbul, Turkey
b Gediz University, Department of Computer Engineering, Menemen, İzmir, Turkey
c Mevlana University, Department of Computer Engineering, Selçuklu, Konya, Turkey

ARTICLE INFO

Keywords:
Pervasive healthcare
Artificial neural networks
Diabetes

ABSTRACT

Pervasive computing is often mentioned in the context of improving healthcare. This paper presents a novel approach for diagnosing diabetes using neural networks and pervasive healthcare computing technologies. The recent developments in small mobile devices and wireless communications provide a strong motivation to develop new software techniques and mobile services for pervasive healthcare computing. A distributed end-to-end pervasive healthcare system utilizing neural network computations for diagnosing illnesses was developed. This work presents the initial results for a simple client (patient’s PDA) and server (powerful desktop PC) two-tier pervasive healthcare architecture. The computations of neural network operations on both client and server sides and wireless network communications between them are optimized for real time use of pervasive healthcare services.

1. Introduction

The vision of pervasive computing systems is to create environments saturated with computing and communication capabilities and gracefully integrated with human users. Technology was not available when this vision was first stated about 30 years ago (Weiser, 1991).

The use and capabilities of small mobile communication devices that include many all-in-one features, such as Pocket PCs, Tablet PCs, smart mobile phones, and smart wireless sensors that have become embedded in everyday life are growing rapidly especially in healthcare. Today new mobile communication devices that combine several heterogeneous wireless technologies, e.g. cellular like UMTS/GSM and GPRS/EGDE, GPS as well as wireless data communication LAN technologies, like IEEE 802.11 and Bluetooth are becoming widely available. These recent advances in mobile devices and wireless communications provide a strong motivation to develop new software techniques and mobile services for pervasive healthcare computing.

A distributed end-to-end 3-tier pervasive healthcare system architecture has been developed utilizing artificial neural network (ANN) computations. At the first tier, there are sensors and wearable devices for monitoring vital signs on the human body. At the second tier, user end devices such as PDAs and computers play a mediator and communicator role between the first tier and the last tier. The last third tier end has powerful desktop servers for providing healthcare services and database operations to the users. ANN models are used at both the second tier and third tier for diagnosing illnesses.

This work presents the initial results for a simple client (patient’s PDA) and server (powerful desktop PC) two-tier pervasive healthcare architecture. The model and the client and server applications are based on ANN computations. The computations of ANN operations on both client and server sides and wireless network communications between them are optimized for real time use of pervasive healthcare services. The model can be applied to diagnose different illnesses. In this paper a simple ANN model is presented to diagnose diabetes as a case study.

2. Related work

The current healthcare models proposed by many previous studies (Bardram, Mihailidis, & Wan, 2007; Stanford, 2002; Varshney, 2009) are mainly based on remote access to healthcare services provided by centralized servers (Mikkonen, Vayrynen, Ikonen, & Heikkila, 2002) or remote patient monitoring (Boric-Lubecke & Lubecke, 2002; Ogawa & Togawa, 2003). Many of these applications have appeared in the popular press and are actually starting to be deployed.

Today many consumer mobile devices are easily connected to a network with home PCs, and let users gather data from sensors located on different parts of body in the home. As a result of these advances in technology, physicians at remote locations can access the patient data in real time and telesurgery is becoming a practical reality. Remote physicians are able to consult on a patient’s
condition as well as take part in a surgical operation. Moreover, medical-informatics research has recognized the need for making medical applications that are aware of clinicians' tasks for example, as workflow support systems or clinical guideline systems. However, these systems are clinical applications supporting the flow of medical work and, as such, are not basic middleware support for pervasive computing (Ciccarese, Caffi, Quaglini, & Stefanelli, 2005; Malamateniou & Vassilacopoulos, 2003).

In this paper a novel end-to-end distributed pervasive health-care system architecture utilizing ANN computations was proposed. ANN models have been widely used to examine the complex relationships between input and output variables (Nelson & Illingworth, 1994) in many scientific and technological areas including diagnosing illnesses (Karlık, Tokhi, & Alci, 2003). Since many ANN algorithms such as Back-Propagation (BP), Radial Basis Function (RBF), and Learning Vector Quantization (LVQ) require CPU, memory and I/O intensive operations, it was difficult and sometimes impossible to use these algorithms on mobile devices which are the main user devices of pervasive health-care computing. Because small mobile devices generally do not have powerful CPUs, large amount of memory and high-speed input/output (I/O) and networking capabilities compared to desktop PCs. However, their CPU speeds, memory and I/O capacities have increased recently, and now new applications and software communication models, which require processing power, large memory, and high speed communications, can be used on these devices. In a previous study (Gümüşkaya, Gürel, & Nural, 2008) researchers analyzed basic TCP socket connections, Java RMI distributed object technology and service oriented approaches, which are the representatives of three important generations in distributed systems. Client/server architectures were studied and the time analyses for different wireless network connections were presented. One of the results of this study is that these small mobile devices are ready for the development of more advanced applications and distributed software communication technologies including Java RMI and service oriented computing.

Diabetes is a chronic disease with varying clinical presentations, depending on the type of diabetes, ethnicity, age and relative insulin level, blood pressure and cholesterol. There are many research studies using different ANN approaches for diagnosing different characteristics of diabetes. Tresp, Briegel, and Moody (1999) have presented a recurrent neural networks (RNN) model for the blood glucose metabolism of diabetic. Venkatesan and Anitha (2006) have presented a Radial Basis Function (RBF) neural networks model for the diagnosis of diabetes mellitus. Polat and Gunes (2007) have presented an adaptive neuro fuzzy inference and support vector machine models (Polat, Gunes, & Arslan, 2008) to diagnose the diabetes disease. Ucman et al. have used both Multilayer Perceptron (MLP) and RBF for classification of MCA stenosis in diabetes (Ucman et al., 2004). Temurtas, Yumusak, and Temurtas (2009) have proposed a comparative pima-diabetes disease diagnosis. They use a MLP neural network model which is trained by Levenberg–Marquardt (LM) algorithm and a probabilistic neural network. Karlık and Al-Bastaki (2004) have presented a MLP model to diagnose the level of sugar in diabetics from bad breath using electronic nose.

3. The ANN model for diagnosing diabetes

Medical information systems in modern hospitals and medical institutions have become larger and this trend causes big difficulties for extracting useful information for decision support systems. Medical decision-support systems are computer systems designed to assist physicians or other healthcare professionals in making clinical decisions (Karlık & Öztöprük, 2007; Miller, 1994; Öztekin, Kong, & Delen, 2011). These systems deal with medical data and knowledge domain in diagnosing patients' conditions as well as recommending suitable treatments for the particular patients. Traditional manual data analysis has become inefficient and new methods such as the use of ANNs for efficient computer-based analysis are essential for diagnosing illnesses (Karlık & Öztöprük, 2007).

In this study, three-layered Multilayer Perceptron (MLP) feed-forward neural network architecture was used and trained with the error back propagation algorithm. The back propagation training with generalized delta learning rule is an iterative gradient algorithm designed to minimize the root mean square error between the actual output of a multilayered feed-forward neural network and a desired output. Each layer is fully connected to the previous layer, and has no other connection (Nelson & Illingworth, 1994). The back propagation algorithm is described step by step as follows:

1. Initialization: Set all the weights and biases to small random values.
2. Presentation of input and desired outputs: Present the input vector \( x(1), x(2), \ldots, x(N) \) and corresponding desired response \( d(1), d(2), \ldots, d(N) \), one pair at a time, where \( N \) is the number of training patterns.
3. Calculation of actual outputs: Use Eq. (1) to calculate the output signals.
\[
y_j = \varphi \left( \sum_{i=1}^{N_w} w_{ij}^{(l-1)} x_i + b_j^{(l-1)} \right), \quad i = 1, \ldots, N_M
\]  
4. Adaptation of weights \( (w) \) and biases \( (b) \):
\[
\Delta w_{ij}^{(l-1)}(n) = \mu \cdot x_i(n) \cdot \delta_i^{(l-1)}(n)
\]
\[
\Delta b_i^{(l-1)}(n) = \mu \cdot \delta_i^{(l-1)}(n)
\]

where
\[
\delta_i^{(l-1)}(n) = \begin{cases} \varphi'(\text{net}_i^{(l-1)})[d_i - y_i(n)], & l = M \\ \varphi'(\text{net}_i^{(l-1)}) \sum_k w_{ik} \cdot \delta_k^{(l)}(n), & 1 \leq l < M \end{cases}
\]
in which \( x_i(n) \) is output of node \( j \) at iteration \( n \), \( l \) is layer, \( k \) is the number of output nodes of neural network, \( M \) is output layer, \( \varphi \) is activation function. One of the most important units in a neural network structure is their net inputs by using a scalar-to-scalar function called the "activation function". Tanh and sigmoid activation functions used in the vast majority of MLP applications are a good choice to obtain high accuracy (Karlık & Olgaç, 2011). The learning rate is represented by \( \mu \). It may be noted here that a large value of the learning rate may lead to faster convergence but may also result in oscillation. In order to achieve faster convergence with minimum oscillation, a momentum term may be added to the basic weight updating equation. After completing the training procedure of the neural network, the weights of MLP are frozen and ready for use in the testing mode.

3.1. Training data for diabetes

In the MLP model for diagnosing diabetes, one input, one hidden layer and one output layer are used as seen in Fig. 1. There are 11 inputs and 2 outputs for diagnosing diabetes. Input data names and their units are shown in Table 1. The outputs in the ANN model are NORMAL and ABNORMAL. The dataset has 456 observations used as training data. The first 228 data are from normal people and others belong to abnormal people. These samples were taken from real patients of a hospital.
3.2. Training phase

At the training phase, the system was trained using different learning rates, momentums and number of iterations starting from 1000, continuing with 2000, 3000, 4000, 5000, 6000 and ending with 7000.

First, a 1000 iterations test was conducted for different momentums and learning rates. The first test results show that the minimum error occurs with the learning rate and the momentum as 0.75 and 0.25, respectively are shown in Table 2. The relations of the results for 1000 iterations are given in Fig. 2.

In the second test, 2000 iterations were performed for different momentums and learning rates. Learning rates, momentums and total square errors for these values and their relations are seen in Table 3 and Fig. 3, respectively. The second test results show that the minimum error occurs with the learning rate and the momentum as 0.95 and 0.05, respectively.

In the third test, 3000 iterations were used with the same values of learning rates and momentums as seen in Table 4. The relations of the results for 3000 iterations are given in Fig. 4.
The third results show that 2.119 total square errors occur in
0.85 learning rate and 0.25 momentum values. This error value is
the minimum for 3000 iterations.

In the fourth test, 4000 iterations were used with the same
values of learning rates and momentums as seen in Table 5. The
relations of the results for 4000 iterations are given in Fig. 5.

The fourth results show that 2.086 total square errors occur in
0.95 learning rate and 0.15 momentum values. This error value is
the minimum for 4000 iterations.

In the last test, 5000 iterations were used for the same values.
The last test results are seen in Table 6 and the relations of the re-
sults are given in Fig. 6.

Table 6 shows that 1.142 total errors occur with 0.95 learning
rate and 0.05 momentum values. After the 5000 iterations test,
6000 and 7000 iteration tests were conducted with 0.95 learning
rate and 0.05 momentum value, and 1.045 and 1.047 error values
were reported, respectively. It was seen that there was no signifi-
cant changes in total square errors in the 6000 and 7000 iterations
tests. Consequently, it was decided to use 5000 iterations with 0.95
learning rate and 0.05 momentum values for the best solution for
fast training.

4. The implementation of the ANN model

In this study, client and server applications were also developed
for the implementation of the ANN model described in Section 3
for diagnosing diabetes. The user interface of the server application
as shown in Fig. 7 has basically two modules. The first module is
used to train the data. As seen from Fig. 7, the user can change many parameters of the ANN model. This server module gets the training data which are inputs and related outputs from text files. After reading the training data, this module produces the trained data. This trained data is served to PDA clients as needed over wireless network connections by the second server module. The server application was developed as a concurrent server hence it can handle multiple client requests at a time.

The PDA client user interface has two screens as tabbed windows for diabetes data entry. The first screen as shown in Fig. 8 is for personal data which is entered once by the patient. The second screen as shown in Fig. 9 is used many times during the day when new diabetes tests are performed.

The PDA client application downloads the trained data from the server when it is first started for diabetes measurements. The ANN software for diagnosing diabetes runs locally on the PDA and does not need any open network connection to the server application. The PDA client application only needs the trained data from the server.

When the PDA client application is executed by the user, it first controls the network connection. If the connection is available, it downloads the last updated trained data from the server automatically. As a result the local trained data always stay updated if the PDA is connected to the network.

When a test is performed by the patient by clicking the “Test” button as shown in Fig. 9, the diabetes data are sent to the server if the connection is available. If not, this data become the pending data and are sent to the server automatically when the connection is available. Hence the server periodically updates the trained data using the incoming diabetes data from patients’ PDAs over the network.

In this study, the mobile client was an HTC Diamond 2 Pocket PC running Windows Mobile 6.5 and based on 528 MHz Qualcomm MSM7200A processor with 512 MB ROM, 288 MB RAM, 802.11b+g, GPS, and Bluetooth 2.0. The PDA client application was developed in .NET Compact Framework and the server application was developed in .NET Framework 3.5. Both client and server applications were developed using the C# programming language. TCP sockets

<table>
<thead>
<tr>
<th>Learning rate</th>
<th>Momentum</th>
<th>Total square error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.95</td>
<td>0.05</td>
<td>1.142</td>
</tr>
<tr>
<td>0.85</td>
<td>0.15</td>
<td>1.193</td>
</tr>
<tr>
<td>0.75</td>
<td>0.25</td>
<td>3.828</td>
</tr>
<tr>
<td>0.65</td>
<td>0.35</td>
<td>3.907</td>
</tr>
<tr>
<td>0.55</td>
<td>0.45</td>
<td>3.927</td>
</tr>
<tr>
<td>0.45</td>
<td>0.55</td>
<td>3.940</td>
</tr>
<tr>
<td>0.35</td>
<td>0.65</td>
<td>4.557</td>
</tr>
<tr>
<td>0.25</td>
<td>0.75</td>
<td>3.979</td>
</tr>
<tr>
<td>0.15</td>
<td>0.85</td>
<td>4.168</td>
</tr>
<tr>
<td>0.05</td>
<td>0.95</td>
<td>5.581</td>
</tr>
</tbody>
</table>

Fig. 6. The relations for 5000 iterations.

Fig. 7. The user interface of the server application.
the PDA and server can do computations for diagnosing illnesses in real time.

As mentioned before, there are many research studies using ANN approaches for diagnosing different characteristics of illnesses including diabetes. The study presented in the paper is different from the previous work in many aspects. Current healthcare models proposed by many previous studies are mainly based on remote access to healthcare services provided by centralized servers, or remote patient monitoring. For these models, the client application on the patient side is used only to collect patient data. In the novel model presented in this paper, ANN and other computations are implemented on distributed client server architecture and the client devices are small mobile devices. As a result of this approach, computations and network communications on client and server sides are optimized depending on the nature of illnesses. The client mobile application tries to make its ANN and other complex calculations locally and shows the results to the patient without contacting the server for a range of illnesses including diabetes. If the characteristics of illnesses are not suitable for PDA local processing based on some more complex ANN algorithms which require more CPU and memory power, PDA becomes an input and output device for the patient data and uses the server side for ANN computations.

5. Conclusion

In this paper a neural network algorithm that classifies diabetes illness data on client/server architecture was presented. The obtained results show a competitive accuracy. Moreover, the feasibility of incorporating ANN algorithms into the PDA is also shown. The main questions answered in this paper are how a client PDA can share the ANN computation load with a server and become a data classifier, if the PDA classifier has a good accuracy, and if

References


