DIGITS RECOGNITION WITH BEAGLEBOARD PLATFORM

Richard Borbély[1], Michal Blaho[1,2], Leo Mrafko[1], Tatiana Mudráková[1,2]

[1]Institute of Robotics and Cybernetics

Faculty of Electrical Engineering and Information Technology

Slovak University of Technology

Ilkovičova 3, 812 19 Bratislava, Slovak Republic

[2]HUMUSOFT, s.r.o.

Cabanova 13/D, 841 02 Bratislava, Slovak Republic

Abstract

A single-board computer (SBC) is a platform that represents the computer on a single circuit board. Thanks to the affordable price, compact size and adequate power they started to be used in education and practice, as well. The MathWorks Company supports these devices using Support Packages in MATLAB and Simulink environment. This article deals with the design of neural networks for the recognition digits needs.

1 Introduction

Nowadays the open-source platforms have a very large potential usage. For their success it is the reasonable price, compact size and sufficient power. The typical SBC representatives are platforms as BeagleBone or Raspberry Pi [1,2]. They can be used in the science and research, but also in the practice, for example in robotics, because they work very well together with various external devices for the multimedia use, and also they can be easily attached to a variety of robotic devices. BeagleBone platform has been used in the several interesting projects. Veter - the robotic device that is intended for researchers and developers, as well. Veterobot project integrates the BeagleBoard xM device, different cameras, GPS receivers and the light sensors. This project allows cloud robotics and the adaptive video playback for remote control via the internet [3]. Another interesting project is the model space camera that made the Dutchman Tim Zaman in 2011. As a major part of his project he took BeagleBoard platform, attaching all relevant components such as the webcam image capture, GPS module required for the position tracking and GPRS module necessary for the data transfer [4]. The BeagleBone platform was used in this article for the numbers recognition using the neural network.

2 BeagleBoard

BeagleBoard is low-voltage, open-source minicomputer developed by Texas Instruments in the collaboration with Digi-Key and Newark element14. These boards have the capabilities of the most of today's desktop computers, of course, without a big weight, high costs and excessive noise. This board was developed by a team of engineers particularly as an educational product that could be used for learning about the possibilities of open-source hardware and software at universities around the world [5]. BeagleBoard xM used in this article can be seen in the Figure 1.

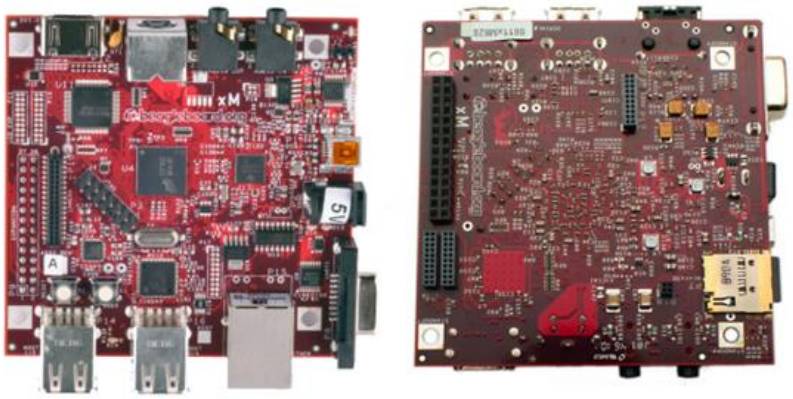

Figure 1: BeagleBoard xM rev. C

BeagleBoard contains AM37x ARM Cortex-A8 processor compatible with a maximum frequency of 1 GHz and with 512 MB operational memory. BeagleBoard supports multiple interfaces such as RJ45 Ethernet connector for the network connection, the slot for MMC / Micro-SD card and 4 USB 2.0 ports for the peripherals connection. To work with the video there are graphical DVI-D port, S-Video port, and to work with the audio these is the IN / OUT 3.5 mm Jack. The further parts of the board are the indicatory LEDs and buttons. BeagleBoard is software-compatible with the Ångström Linux, Android, Ubuntu and XBMC. The installation cost of this platform is about $ 149.

2.1 MATLAB Support

BeagleBoard is compatible with MATLAB and Simulink environment, that is why there is the possibility to create the own programs and to start Simulink models on the device. The BeagleBoard helps students understand the workflow for designing an embedded system without using manual programming. Students can use Simulink to create algorithms for the audio processing and computer vision applications. Industry-proven techniques for the Model-Based Design can be applied to verify that their algorithms work during the simulation. These algorithms can be further implemented as the standalone applications on the BeagleBoard [6].

To connect the BeagleBoard with MATLAB Simulink environment it is necessary to install the support package - a package of support for the BeagleBoard device. After passing all the steps of installation until the termination, all needed third-party development tools will be installed in the computer, as the Simulink block library: Simulink Support Package for BeagleBoard Hardware and Examples. In the BeagleBoard platform it is necessary to install the firmware for the correct cooperation with MATLAB Simulink. The installer writes the firmware image to the microSD card with the login details to an environment that will run on the device. After inserting the microSD card into the board, the support package installer applies the settings that were selected. Therefore in the Simulink pre-built block shown in the following figure can be used.

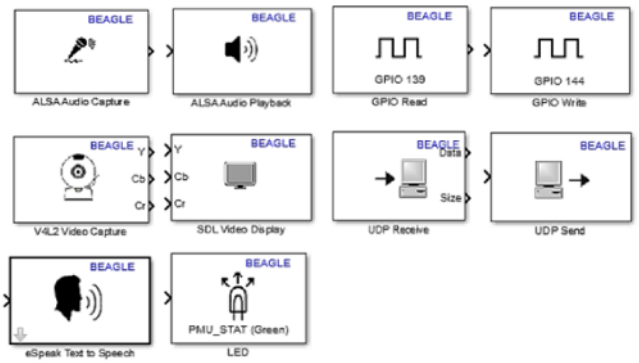

Figure 2: BeagleBoard Library

3 Character Recognition Using the Neural Networks

Artificial neural networks represent models inspired by the human nervous system, which is able to learn. One reason why it is possible to have the computers with the features similar to human is the use of neural networks. Neural networks are particularly useful to solve problems that cannot be interpreted as a sequence of steps, such as pattern recognition, classifying them into the groups, forecasting data, or their acquisition. One example of pattern recognition is the recognition of the handwritten characters. This part of the recognition is in fact particularly difficult task in the areas such as image processing or machine vision, since every human has a very different handwriting. This work will be focused on the exclusively recognition of handwritten Arabic numerals. This application could later be extended to include all alphanumeric characters, or even other than Slovak characters.

3.1 Data Acquisition

The first step is to gain the adequate amount of input data that can be used to train the neural network. The data were obtained from the sufficient number of persons to be our data sufficiently variable, and to increase the character recognition rate among the test persons. Samples of eight persons were used for training, whereby the data intended for the verification of recognition is different from the training samples. The application is able to recognize numbers from 0-9, and from each digit ten samples from each person was collected, so it is all together the 800 samples digits. The digits were written using the freely downloadable application for Android devices and each digit represents the image in PNG format with dimensions of 20x20 pixels, as can be seen in the Fig. 3.

Figure 3: Sample data for the network training

3.2 Data Adjustment

Before the data sent to the neural network, the matrix X for patterns and the target matrix T to identify them need to be created. The matrix X is a matrix of 800 columns, where the each column represents one particular digit (pattern). Digits are progressively-loaded, so every 100 columns of the resulting matrix X represents the same digit. The matrix T represents the final numbers, with a single line containing the number 1 at the corresponding position.

3.3 Neural Network Creating and Training

When recognizing 10 digits from 0 to 9 it needs to be distinguished a total of 10 groups. The neural network has in this case 10 neurons in the output layer. If the digit 0-9 is detected, only the one output neuron corresponding to detected digit sends a signal. In the Fig. 4 can be seen the network model for the digits recognition. On the input the signs that represent the unknown numbers are detected. In the first case the digit “1” was detected and the second one the digit “3”. This allows to detecting any number of classes. The output data need not be the discrete, as shows in the example, but it is sufficient to determine the maximum of all values, and this corresponds to the detected class.

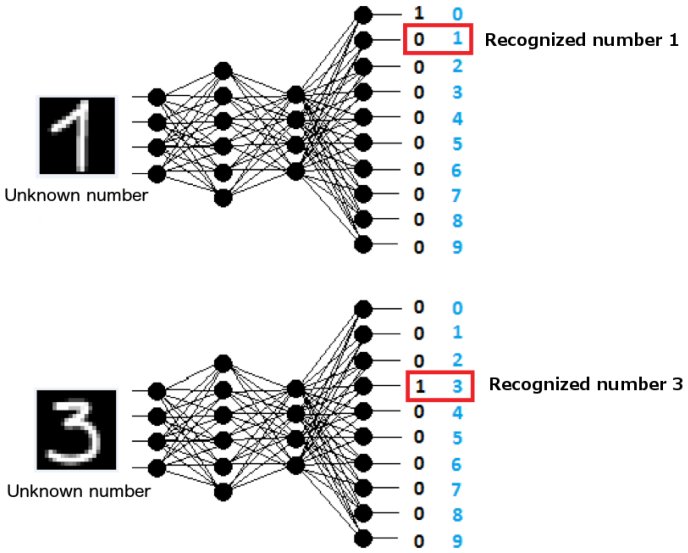

Figure 4: Scheme for the number detection [7]

The algorithm used for the learning is called Backpropagation algorithm – algorithm with the back propagation error. It works like that the evaluated solution is compared with the expected one and thus determines how much is neural network mistaken. Retrospectively, based on this data, it is calculated how much the weights of neurons need to be changed, to reduce as much as possible this deviation from the correct solution. Initially, random parameter settings are selected, and afterwards the aim of the whole learning is to find the global minimum, meaning the correct setting of network parameters to minimize the error.

4 BeagleBoard Implementation

The neural network trained as described in the previous section can be implemented for various target components. Neural network generated for the Simulink environment can be extended to the supported blocks. Afterwards it can be translated and uploaded to the device. This section describes the procedure of trained neural network implementation to the BeagleBoard xM device.

4.1 Simulink Model

When the neural network was already trained, it was able to generate a new model in Simulink using the command *gensim*, which will include a neural network as shown the Fig. 5. The data has been sent into the neural network using a network interface with UDP protocol. The output of the neural network is a vector which determines the weight of each number. To find the greatest weight in the vector, the custom function was written. The function output is a number represented with the ASCII character. After the conversion to data type uint8, the number was afterwards sent to the block eSpeak Text to Speech. Using this block the BeagleBone can sent the recognized number to the audio output using a speech synthesizer. The block UDP Receive for receiving data holds the input data several loops. The variable Timer was added to the function to eliminate the number to be said more times. Mentioned variable counts the number of program cycles and it blocks the output of the same nonzero number a selected number of times in a row. If on the output of the block UDP Receive was no input data, the function ensures that the output value is zero. In this case the speech synthesizer did not say anything. The scheme created as was described above has been translated and recorded to the BeagleBoard device.

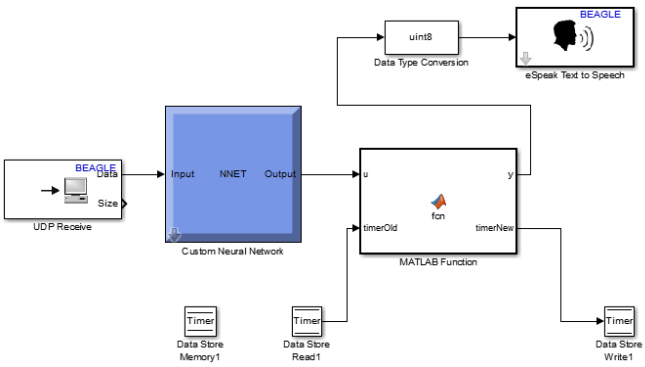

Figure 5: Simulink model

4.2 User Interface

The graphical user interface was designed to communicate with a program that runs on the target machine. In this interface IP address and port number of the device can be set. BeagleBoard receives on the IP address and port the data assigned for processing with a neural network. The graphical interface allows the user to select an image from disk, to load it into the variables and to depict it on the desktop. With the button the selected image can be sent to the device. The graphical user interface for the communication with the device is in the following picture.

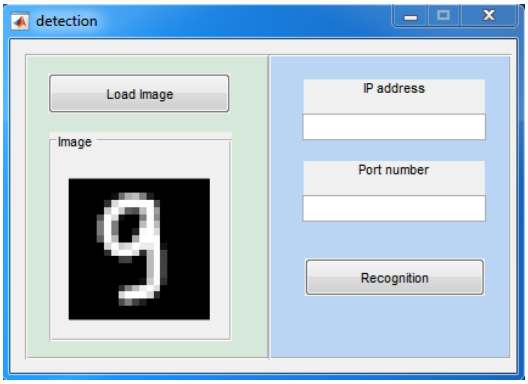

Figure 6: Graphical user interface

The button to perform the character recognition creates in the background several operations. In the first step it transfers the representation of the image of the matrix form to the vector. After the conversion, Simulink opens a simple block scheme and the vector is entered in the block constant. Before the data sending, the data need to be transposed for the correct input into the neural network. The data are sent via UDP protocol for the specified IP address and port. The block scheme of the data sending to the device is shown in the Fig.7.

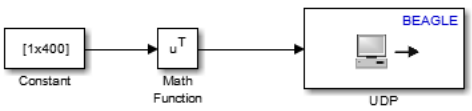

Figure 7: Data sending to the device

4.3 Evaluation

The effectiveness of the proposed solution depends on the several factors. The first factor is data used during the neural network training. 800 samples were used during the network training and were divided into 10 groups. Each group therefore contained 80 samples. Another factor is the separate neural network, suitable selecting and setting its parameters. To verify the effectiveness of neural network, the algorithm was created. Mentioned algorithm verified all numbers using the neural network and the result was compared with its group. The success rate of the result varies with the

testing samples. From the testing data the best result has the digit 0 with 100% success rate. On the contrary, the worst one was the digit 6 with a success rate of 85%. These data provide an idea how effectively is the neural network trained, which may lead to the parameters adjustment. It needs to be noted that mentioned digits were received from the 8 users. For the better detection it would be necessary to use a larger number of standalone samples.

5 Conclusion

BeagleBoard platform has proven to be suitable for use in the small and also in the bigger projects, offering adequate power, the huge amount of input-output ports, as well as the ability to connect more external devices. The interesting points are its compatibility with the MATLAB environment, leading to the practical use and further expand its use. BeagleBoard platform is afterwards not limited only to the classical programming under the Linux operating system in C, C ++, Python, or other. For the BeagleBoard platform the neural network for digits recognition was designed. The procedure consisted of several steps from the initial data collection, through the algorithm design for inputs adjustment, afterwards the neural network design and up to the resulting link of all elements into the one unit. In our test the best solution has the digits 0,1,2,7 and 9, where the probability of defective sample ranged maximally on the level of 0.25%. The worst digits were 8 and 6, where the probability was approaching to the 15%. For the better accuracy, it is necessary to obtain even more input data for training. With hand written characters, it is important to have the huge amount of the input data and they have to be blurred as much as possible. And in that case the probability of wrong trained network and the wrong recognized output will be much smaller.

Acknowledgement

This paper was supported by the Scientific Grant Agency of the Ministry of Education, Science, Research and Sport of the Slovak Republic under the contract No. VEGA-1/1256/12.

References

[1] BeagleBoard.org. *BeagleBoard-xM*, [online], 2015. Available at: http://beagleboard.org/getting-started
[2] RASPBERRY PI. F. *What is a Raspberry pi?*, [online], 2015, Available at: http://www.raspberrypi.org
[3] Veter. *Robotic vehicle*. [online]. 2015. Available at: http://veterobot.com/
[4] T. Zaman. Space Camera 3. [online]. 2011. URL: http://www.timzaman.com/?page_id=1106
[5] M. Richardson. *Getting Started With BeagleBone*. Published by Marker Media, Inc., Sebastopol. 2013, ISBN 978-1-449-34537-2.
[6] Mathworks, *Matlab support for BeagleBone*. [online]. 2015. Available at: http://www.mathworks.com/hardware-support/beagleboard.html
[7] V. Chalupník. *Biologické algoritmy – neuronové sítě.* [online]. 2012. Available at: http://www.root.cz/clanky/biologicke-algoritmy-5-neuronove-site/
[8] I. SEKAJ. *37123_3B Aplikovaná výpočtová inteligencia. Prednáška č. 6*, Available at: Document server AIS STU

Michal Blaho
blaho@humusoft.sk

Leo Mrafko
leo.mrafko@stuba.sk

Tatiana Mudráková
tatianam@humusoft.sk