Journal of Electronics and Information Science(2017) 2: 76-79
Clausius Scientific Press, Canada

Implementation and Research of LSTM Neural Network Based on the FPGA

Xintao Huang[a], Jun Yang[b *]

School of Information Science and Engineering, Yunnan university, Kunming, China

[a]173471638@qq.com, [b*]junyang@ynu.edu.cn

Keywords: RNN; LSTM; FPGA

Abstract: Over the past decade, artificial intelligence has reached a stage of rapid development, and deep learning has played a main role in this development. Despite of its strong ability to simulate and predict, deep learning is faced with the problem of large computational complexity. At the hardware level, GPU, ASIC, FPGA are ways to solve the huge amount of computing. This paper will explain the deep learning, FPGA structure and the reason why the use of FPGA to accelerate the deep learning is effective. Also, it will introduce a recursive neural network (RNN) implementation on the FPGA platform.

1. Introduction

The rapid growth of data volume and accessibility in recent years, Making the artificial intelligence algorithm design concept has changed[1].The practice of manually creating algorithms is replaced by the ability of computers to automatically acquire combined system from a large number of data, resulting in significant breakthroughs in critical areas such as computer vision, speech recognition, and natural language processing[2]. Deep learning is the most commonly used in these areas of the technology, the industry has also been of great concern.However, the depth learning model requires a very large amount of data and computing power, and only better hardware acceleration conditions can meet the demand of the existing data and model size that continues to expand.

Long short-termmemory (LSTM)is a recurrent neuralnetwork (RNN)architecture (an artificialneural network, deep learning)published[1] in 1997 by Sepp Hochreiter and Jürgen Schmidhuber. Like most RNNs, an LSTM network is universal in the sense that given enough network units it can compute anything a conventional computer can compute, provided it has the proper weight matrix, which may be viewed as its program. Unlike traditional RNNs, an LSTM network is well-suited to learn from experience to classify, process and predict time series when there are very long time lags of unknownsize between important events.

The existing solution uses a graphics processing unit (GPU) cluster as a general purpose graphics processing unit (GPGPU)[3], but the field programmable gate array (FPGA) provides another solution worth exploring.The growing popularity of FPGA design tools makes it easier for top-level software to be used in deep learning areas, making FPGAs easier for model builders and deployers[4]. The FPGA architecture is flexible, allowing researchers to explore model optimization. outside a fixed architecture such as a GPU[5]. At the same time, FPGAs are more powerful in terms of unit energy consumption, which is critical to the study of large-scale server deployments or resource-constrained embedded applications. This article will introduce a recursive neural network (RNN) implementation on the FPGA platform.

2. Introduction of LSTM

LSTM is a kind of RNN (recurrent neural network), which is the most widely used in processing time series data. It is controlled by three gates: input gate, forget door ,output gate, and hidden layer and Memory cell .The following Figure is its topology:

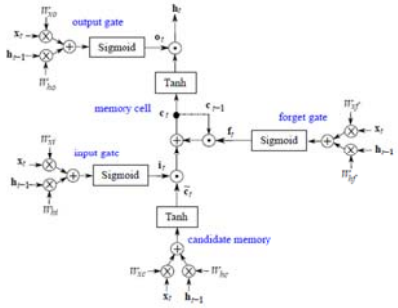


Fig 1. Topology of LSTM

The input gate controls the input of a certain moment; By acting on the memory cell of the last moment, the forget gate control how much of the data stream is going to flow into the next moment; The memory cell is determined by the input of the previous moment and the candidate input of this moment; The output gate acts on the memory cell, determines the information of hidden layer at this moment, and sends it to the next layer of neural network.

All the equations are as follows:

$$i_t = \sigma\left(w_{xi}x_t + w_{hi}h_t + b_i\right) \tag{1}$$

$$f_t = \sigma\left(w_{xf}x_t + w_{hf}h_t + b_f\right) \tag{2}$$

$$o_t = \sigma\left(w_{xo}x_t + w_{ho}h_t + b_o\right) \tag{3}$$

$$\hat{c} = \tanh\left(w_{xc}x_t + w_{hc}h_t + b_c\right) \tag{4}$$

$$c_t = f_t c_{t-1} + i_t \hat{c} \tag{5}$$

$$h_t = o_t \tanh\left(c_t\right) \tag{6}$$

Where w represents the respective weight, b represents the respective offset, σ is the logistic sigmoid function:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{7}$$

3. Design of FPGA Module

The FPGA used in the study was Cyclone II EP2C35F672C6N, , the following figure is its overview:

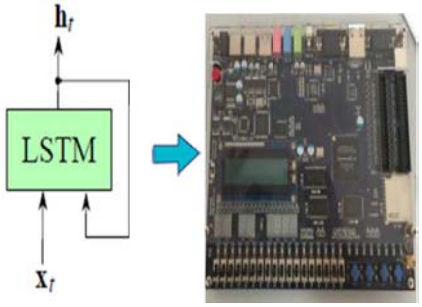
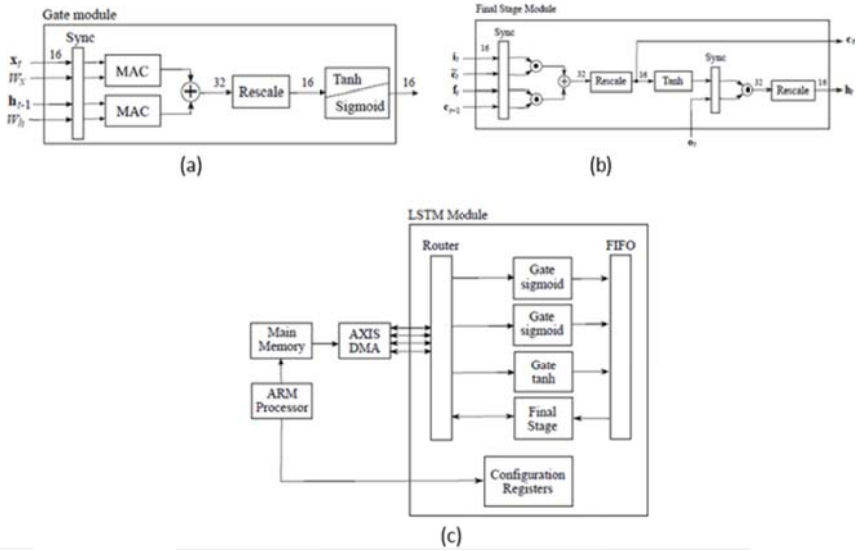


Fig.2 Overview of LSTM on FPGA

LSTM mainly carries out the calculation of the multiplication and non-linear functions of the matrix (tanh, sigmoid). Matrix multiplication is performed by the MAC unit (Multiply Accumulate) with two data streams: vector and weight matrix streams, as shown in Figure (a). After the iteration, the MAC is reset to prevent the previous data from being mixed into the next time.The data of the two MAC units are added before the nonlinear function is calculated, While using a rescale module to 32-bit data into 16-bit data.Scalar calculation of the module is to calculate the ct and ht, and finally passed the next moment of calculation, as shown in Figure (b).Figure (c), the Data flow into and out used DMA (Direct Memory Access) serial control.Since the DMA serial port is independent, a clock module is also required for timing control.The clock module is mainly composed of a buffer memory and temporarily stores some data until the data arrives. When the last port data flows into the clock module to start transmitting data, this ensures that the input is associated with the weight matrix at the same time.



Therefore, the LSTM model is divided into three stages:

a.Calculate i_t and $\hat{c}_t$;

b.Calculate f_t and o_t ;

c.Calculate c_t and h_t ;

The first and second stage two gate modules (4 MAC units) are calculated in parallel, and i_t , $\hat{c}_t$, f_t and o_t are obtained and stored in the FIFO (First In First Out).The last stage fetches the vector in the FIFO to calculate c_t and h_t. After that, the LSTM module continues to wait for the calculation of the data of the next layer or the next time. At the last moment LSTM is calculated to the last layer, the model outputs the final target value.

4. Result Analysis

By training the LSTM network on different platforms, we get the comparison of different models. Table 1 is the platform parameters, the results of the operation shown in Figure 3, you can find: Even at 142MHz clock frequency, FPGA platform running time is much smaller than other platforms, parallel to eight LSTM memory cells processing has achieved 16 times faster than Exynos5422 results.

Configuration Type	Tegra K1 development board		Odroid XU4		Zedboard Zynq ZC7020	
CPU/GPU	Cortex-A15	Kepler	Exynos5422(CPU only)		Cortex-A9	FPGA
Cores	4	192	4(High)+4(Low)		2	/
Clock	2320.5 MHz	852 MHz	2000 MHz	1400MHz	667 MHz	142 MHz

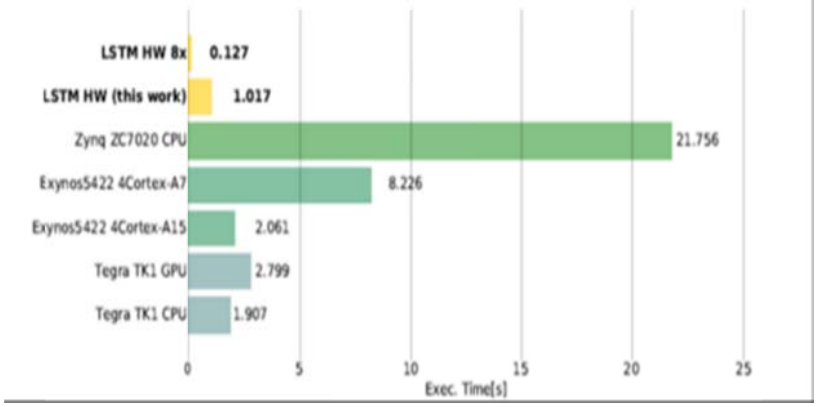


Fig.3 Comparison of different hardware platform parameters

5. Summary

Depth learning uses a deep neural network model with multiple hidden layers. Because of the inherent parallelism of DNN, GPU and FPGA with large-scale parallel architecture become the mainstream hardware platform for accelerating deep learning, and its outstanding advantage is that it can customize the calculation and storage structure according to the characteristics of the application, and achieve the hardware structure and depth learning algorithm Optimal match, get higher performance power ratio; And, FPGA flexible reconstruction function also facilitates the fine tuning and optimization of the algorithm, can greatly shorten the development cycle. There is no doubt that FPGA in the depth of learning the future is very worth looking forward to.

References

[1] Y. Bengio, Learning Deep Architectures for AI, vol. 2, no. 1. 2009.

[2] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[3] Wikipedia.(2015).Field-programmable gate array [Online].

[4] G. Orchard, J. G. Martin, R. J. Vogelstein, and R. Etienne-Cummings, “Fast Neuromimetic Object Recognition using FPGA Outperforms GPU Implementations,” vol. 24, no. 8, pp. 1239–1252, 2015.

[5] S. Chikkerur. (2008). CUDA Implementation of a Biologically Inspired Object Recognition System [Online].