UTM Computing Proceedings
Innovation in Computing Technology and Applications
Volume: 2| Year: 2017 |ISBN: 978-967-0194-95-0

Web Engine Integration for Self-Organizing Map Learning

Mohamad Nizul Zaim Bin Anuar[2], Shafaatunnur Binti Hasan[1,2]

[1]UTM Big Data Centre, Universiti Teknologi Malaysia, 81310 Johor Bahru, Johor, Malaysia
[2]Faculty of Computing, Universiti Teknologi Malaysia, 81310 Johor Bahru, Johor, Malaysia

mnizulzaim@gmail.com, shafaatunnur@utm.my

Abstract.As of the big data application start to grow and business to seek of different kind of solution to analyse their data in data warehouse. However, the current technology of data processing is using CPU processing which is very slow in performance because of CPU are not optimize for parallel computing. Many researchers discover that GPU that can run parallel computing can solve this problem. However, the desktop application that being created by researchers can only run in computer that have GPU and otherwise the data still need to run in CPU unit. The rising of web application already help many type of application to be run without need to have certain specifications to be able to do the job. It is also able to run many applications in many platforms without to worry software constraint Using web application with GPU can improve the performance of processing and the result can display to web browser more quickly. In this paper, it is focus on implementation and integration of web engine for web application and GPUMLib testing it using Self-Organizing Map algorithm that are already available in GPUMLib. This study analyses MNIST handwriting datasets and do pre-processing of the data to make sure it can be run to solve clustering problem to make sure it will suit with SOM algorithm. Then, the web engine has been created to ensure that it can bridge the web application with current GPUMLIB-SOM application and the time taken to execute the GPUMLIB-SOM using client web browser has been recorded with the network performance data. Both CPU and GPU device performance has been recorded and result show that GPU are running 44 times faster than using CPU device to do the data analysis using SOM algorithm. To ensure the result also can be visualize, the web application also generate the subsequent heatmap where the clustering can be view in web browser.

Keywords: Self-Organizing Map (SOM), Big Data, Web Engine, Application, GPU, Data Analysis

1. Introduction

Machine learning (ML) are very hot topic in this past years. However, ML is not new things for researchers where most of researchers start to study about this topic half a decade ago. Arthur Samuel has defined ML is machine that can learn without explicitly programmed. ML is subfield of Artificial Intelligence (AI) and very related to Probability and Statistic. This is because one contribution of ML is for prediction where it is also the important of statistic.

The problem of ML arises when to get most accurate output or result from a given data. The more data is given for training can give better and accurate result but it can give impact of performance of processing time. The main concern of every researcher is to find the best way to overcome this issue because increase in processing time will also effect the increase of cost indirectly. For big company, this problem is not their main concern as they can use a super-computer to overcome this problem.

The ability of web application to use in every device without worry about the operating system is something that change most of human activity. From social to business application, the web application can save the data to the server and be access the same data without need to worry about what is operating system and software needing to make sure the data can be access. This advantage of web application has widely use and Machine Learning web application has been created before. However, the lack of processing speed in current application is the main reason this application is not something reliable to use. This is because, most of the server use middle end-performance CPU to analyse the big data. So, most of the ML application still stick on window-based application as the processing speed is the most important thing for big data analysis.

GPU processing is something that can overcome this problem. However, the problem of GPU processing is this application do not have enough support for web application. Although The World Wide Web Consortium (W3C), main international standards organization for the World Wide Web has started working to include support for standard GPU API for web application, this is far from accomplish. The GitHub repository to implement this standard only created on 10 February 2017 which is usually take about 5 years before it will available to all modern browsers (W3C Community website, 7 June 2017). GPUMLib for example is written in CUDA C++ so that it is difficult to make it work with web application unless the server-side web engine is written in C++. The lack of C++ support to create web engine is the main concern of any web developer to create the web engine in this language. GPU programming is widely use in window-based application but the computer without the GPU programming will not get fully potential from this device.

The research aim is to web engine that can bridging the web application and SOM algorithm application that have been created in GPUMLib to test performance of GPU and compare it with CPU performance. The research also will also need to find the network latency time and then substitute it from the finding to make sure that this variable will not affect the result.

This paper is organized as follows: In Section 1, we present the introduction of this paper. Then, in Section 2, we discuss more details of some related works. While in Section 3, we describe the overview of the research methodology that used in this research. Next, we provide the experimental results in Section 4. In section 5, we analyse the results from the research conducted. Finally, Chapter 6 states the conclusion and future works in enhancing this research.

2 Related Work

Web Application refers to a software that are store in remote server and can be access by multiple nodes or client computers through the web browser. The ability of web application that can be access remotely without direct accessing the server are the vital part of acceptance of web application to the end users become key to the birth of mobile application era. Although mobile application is right now suddenly increasing in term of user and number of application, the web application still is not going to extinct soon. This is because there are many business solutions still need to use desktop application for high complexity system.

Graphical Processing Unit (GPU) programming are very hot topic right now. The ability of this hardware can enhance the performance of Central Processing Unit (CPU). The GPU nowadays are more powerful, highly parallel and programmable device that can be used for general purpose computing application (Khajeh, 2015). Although GPU are build high performance where repeated are much more common, they are also effective for parallel processing and this can be benefit for data processing where the data might happen being a very large dataset. Hence, it might be not uncommon when the very big data that need to be process using CPU where it need to take many days to accomplish the desire result can be reduce to only a few hours.

Self-Organizing Map (SOM) algorithm is one type of classification algorithm where this algorithm can self-organize the data and this type of algorithm are being called as unsupervised competitive learning or unsupervised learning by many researchers. They can also be called Feature Maps, as in Self-Organizing Feature Maps. Retaining principle features of the input data is a fundamental principle of SOMs, and one of the things that makes this algorithm so valuable (Wei Gong et. al, 2015)

3 Methodology

There are 7 main phase to complete this research. Figure 1 illustrates the proposed research methodology for this research.

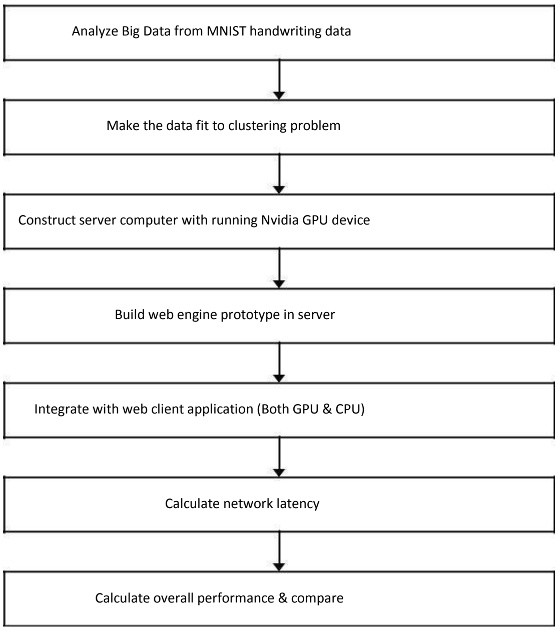
Analyze Big Data from MNIST handwriting data
Make the data fit to clustering problem
Construct server computer with running Nvidia GPU device
Build web engine prototype in server
Integrate with web client application (Both GPU & CPU)
Calculate network latency
Calculate overall performance & compare

Figure 1 Operational Framework

There are two main layer that is client and server and three sublayers in server. It starts with analyze the big data from real running company data to make sure this application is fit to real world problem. Because of this research are using SOM algorithm, this research need to consider the data that can fit to clustering problem which can be solve by using SOM algorithm. Next, this research should start with constructing the server that have NVidia GPU on it and install GPUMLib on it. Then, the prototype of web engine should start being build follow by integrating the web client application to make sure it can be run both on CPU & GPU processing. Then, it should be next measurement and testing phase where it should be start to run both on this CPU and GPU and calculate the network latency between server and client computer. The last step is needed to calculate overall performance in term of speed of time of processing and compare it between CPU and GPU result.

4 Experimental Results

In this research, we conduct several experiments using constant of configuration which can be seen on Table 1. The only things are different in the configuration is either running it in CPU or GPU.

Table 1: Constant Configuration Settings

Constant Name	Value
Number of Samples	1000
Number of Iterations	100
Number of Features	784
Map X	10
Map Y	10

Table 2 show the average performance between CPU and GPU which is both has been running about 10 times.

Table 2: Average performance for both of GPU and CPU

	CPU	GPU
Average Devices Execution Time (s)	416.49	9.5224
Highest Execution Time (s)	438.4	8.46
Lowest Execution Time (s)	404.3	10.43

Result above show that average performance of GPU is 46 times faster than using CPU to analyses the same configuration as set in Table 2. Then, to show that this application already integrated with the GPU devices, the experiments continue with running SOM application using GPU with 60,000 data from MNIST training file. The results still show this integration successful with 424.4 seconds take to finish for 60,000 compare to CPU that only can run 1,000 data in that period. The result with 60,000 training data can be show in Table 3.

Table 3: Performance after running using GPU with 60,000 data

No.	Overall Execution Time (s)	Network Latency (s)	GPU Computation Time (s)
1	449.4	1.22	448.18
2	454.6	5.23	449.37
3	431.5	3.42	428.08
4	473.43	3.61	469.82
5	421.14	2.98	418.16
6	462.34	9.5	452.84
7	491.6	1.75	489.85
8	442.6	2.33	440.27
9	475.6	6.29	469.31
10	451.92	1.22	450.7
Average Computation Time			451.658

The result also can generate the heatmap which is can show the performance of training to cluster the data to their subsequent group as can see in Figure 2 below:

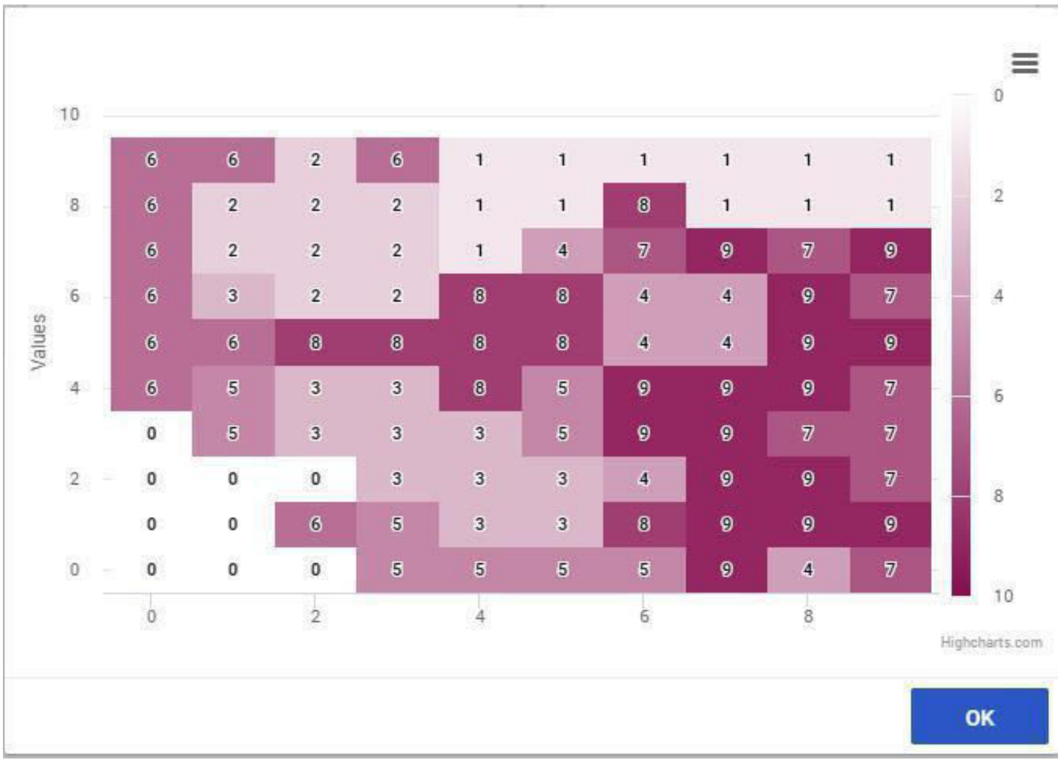

Figure 2 Heatmap of the data when running the GPUMLib-SOM with 60,000 data

5 Discussion

From the result in chapter 4, it shows a performance of this application and can do the calculation without problem in server with Nvidia GPU and process can run more quickly and take less time compare to CPU. This result show the integration of this application successful integrated and ready for production uses although the application itself need some adjustment. Parallel computing in CPU give very high performance of result and it is also costless compare to creating the server to match this result.

It is also achieved to measure the real performance of processing without need to have network latency in the result.

6 Conclusion

There is some downside of using GPUMLib SOM application which is obviously the testing or validation of the data cannot being run and the result are not actually being tested with real data. However, the improvement can be done on GPUMLib and this web engine can run the testing if GPUMLib provide the validation of the data in future. There is also some improvement can be made which is creating the same web engine with different algorithm such as Multi-Back Propagation (MBP) algorithm or Non-Negative Matrix Factorization (NMF). Other than that, it is also can be made as library or platform where the new algorithm can be add to this web application easily in future.

With this technology, everyone can use the processing of the data from the web application and will not worry about GPU for data processing. It is also will improve the performance of the data processing which is sometime take hours to finish and can be optimize to only less than an hour.

References

Anthony Ralston (1992), *The influence of computers and informatics on mathematics and its teaching,* Professort Bernard Cornu.

CA, J. Needham (2013), Disruptive Possibilities: *How Big Data Changes Everything*, O'Reilly Media, Sebastopol.

Chris McClanahan (2011), *History and Evolution of GPU Architecture.*

David, Edward E., Jr. (1984), Renewing U. S. Mathematics: Critical Resources for the Future.

Ebrahim Khajeh (August 2015), *Gpu-Based Harmony Search Algorithm for N-Dimensional Problems*, (SCI) 115, 715–762

Glenn Shafer (2005), *Why Should Statisticians Be Interested in Artificial Intelligence*.

Gustafson (2016), David A, Neural network.

GPUMLib Website (May 16, 2016), http://gpumlib.sourceforge.net.

Helmut Heugl (2004), *The influence of technology in several roles of mathematics.*

Hujun Yin (2008), *The Self-Organizing Maps: Background, Theories, Extensions and Applications*

Ivica Mar, Slobodan Ribari (2000), *Comparison of a back propagation and a self-organizing map neural networks in classification of TM Images.*

Jarvis, John F. (2014), Robotics Systems Research Department, Bell Laboratories, Holmdel, New Jersey.

Jeff Offutt (2009), *Modeling Presentation Layers of Web Applications for Testing*, Kohonen Self-Organizing Maps, Shyam M. Guthikonda.

John E. Stone, Ryan McGreevy, Barry Isralewitz, and Klaus Schulten (2014), *GPU-Accelerated Analysis and Visualization of Large Structures Solved by Molecular Dynamics Flexible Fitting.*

McGraw-Hill Education (2003), McGraw-Hill Dictionary of Scientific and Technical Terms, 6th edition.

Noel Lopes and Bernardete Ribeiro (2011), *GPUMLib: An Efficient Open-Source GPU Machine Learning Library*, ISSN 2150-7988 Volume 3, 355-362.

NVidia, July 2013, Cuda, C. Programming guide NVIDIA Corporation.

NVidia (15 May 2016), Nvidia Website, Nvidia Corporation, http://www.nvidia.com/object/cuda_home_new.html.

NVidia (2011) Tesla Case Study: *Plaque*, Nvidia Corporation.

Shafaatunnur Hasan, Siti Mariyam Shamsuddin, Noel Lopes (2014) , *Advance Machine Learning Big Data Framework and Analytics for Big Data Problems*, Soft Compu. Appl, Vol. 6, No. 2

Sugano Shigeki (2016), Department of Modern Mechanical Engineering, Waseda University, Tokyo, Japan.

T. H. Davenport (2014), Big Data at Work: D*ispelling the Myths, Uncovering the Opportunities,* Harvard Business Review Press, Boston

Wei Gong, Kévyn Johannes and Frédéric Kuznik (2015), *A Survey on Parallel Computing and its Applications in Data-Parallel Problems Using GPU Architectures,* Numerical Simulation of Melting with Natural Convection Based on Lattice Boltzmann Method and Performed with CUDA Enabled GPU

W3C Community website (7 June 2017), https://www.w3.org/community/gpu/