Computer Systems Department
Faculty of Computer & Information Sciences
Ain Shams University

SPEEDING UP LARGE SCALE MACHINE LEARNING ALGORITMS USING GPGPU

This thesis is submitted as a partial fulfillment of the requirements for the degree of Master of Science in Computer and Information Sciences.

BY

Islam Ahmed Hamed Elgarhy
B.Sc. in Computer and information Sciences,
Demonstrator at Computer Systems Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

Under Supervision of

Prof. Dr. Hossam El-Deen Mostafa Fahim
Professor in Computer Systems Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

Prof. Dr. Rania Abd El-Rahman El Gohary
Professor in Information Systems Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

Dr. Heba Ahmed Khaled
Assistant Professor in Computer Systems Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

Cairo 2019

قسم نظم الحاسبات
كلية الحاسبات و المعلومات
جامعة عين شمس

تسريع أداء خوارزميات تعلم الآلة باستخدام كروت الشاشة عالية الأداء متعددة الاستخدام

رسالة مقدمة لإستيفاء الحصول على درجة الماجستير فى الحاسبات و المعلومات

إعداد
إسلام احمد حامد الجارحي
بكالوريوس الحاسبات و المعلومات،
المعيد بقسم نظم الحاسبات،
كلية الحاسبات و المعلومات،
جامعة عين شمس.

إشراف
ا.د/حسام الدين مصطفى فهيم
أستاذ بقسم نظم الحاسبات,
كلية الحاسبات و المعلومات,
جامعة عين شمس.

ا.د/رانية عبد الرحمن الجوهري
أستاذ بقسم نظم المعلومات,
كلية الحاسبات و المعلومات,
جامعة عين شمس.

د/هبه خالد احمد
مدرس بقسم نظم الحاسبات,
كلية الحاسبات و المعلومات,
جامعة عين شمس.

القاهرة 2019

This thesis is dedicated to my parents and my brothers for their love, continuous support and encouragement. Many Thanks to my wife. Without you, I couldn't complete my work. To my daughters, I'm fond of you.

ACKNOWLEDGMENT

All gratitude is to ALLAH for the strengths and his blessing in completing this thesis. Profound appreciation goes to *prof. Dr. Hossam Faheem* for his supervision and continuous support and invaluable help by constructive comments and suggestions during the study and fulfillment of this thesis.

I would like to acknowledge and express my appreciation to *prof. Dr. Rania El Gohary* for her support, suggestions and kind instructions. No words to express my sincere thanks and gratitude to *Dr. Heba Khaled* for her guidance, supports at all phases of this research work. My appreciation goes to my professors, staff members of FCIS Ain Shams University.

ABSTRACT

As many of the machine learning algorithms, SVM requires a high computational cost (memory and time) to solve a complex quadratic programming (QP) optimization problem, so SVM necessitate a high computing hardware capabilities.

Due to the physical limitation in miniaturization process, the central processing unit (CPU) clock frequency can’t be increased, therefore the huge improvements done in packaging multiple CPU cores onto the same silicon chip and also using graphical processing unit (GPU) for general purpose numerical computing.

With the advantages of parallel multi-architecture in both multi-core CPU and a high-scalable GPU, there is a promising candidate to enhance the performance of the algorithms that fits well to run in parallel multi-architecture, so there is a chance to enhance the SVM high computational time for solving the optimization problem.

Moreover, Tensorflow is an open-source machine learning framework library that allows to implement machine learning algorithms using Application program interfaces (APIs). Tensorflow has the ability to migrate to alternative hardware components, and it will reduce time for developing alternative algorithms, so there is a chance to use Tensorflow library for implementing a cross-platform SVM implementation with short development time.

This thesis, presents the design and implementation of a hybrid parallel implementation for SVM algorithm, also it show a comparative study between this hybrid parallel implementation and tensorflow implementation.

The benchmark shows a significant improvements in speed up for hybrid parallel implementation over sequential implementation, other parallel implementation and tensorflow implementation.

The proposed hybrid parallel implementation achieves a speed up of 40X over the sequential open-source library (LIBSVM), a speed up of 7.5X over the CUDA-OPENMP for training process with (44442 records, 102 features size, and 9 classes), a speed up of 13.7X over LIBSVM in classification process for 60300 records, and a speed up of 14.9X over the SVM Tensorflow implementation on pavia centre dataset.

CUDA-GPU achieves a speed up of (154.3X, 60.5X, and 119.7X) over Tensorflow-GPU for three different training datasets (pavia centre hyperspectral, breast cancer, and iris folower) respectively. Also, Experimental results show that the explicit control in CUDA API have a speed up over the implicit control in Tensorflow. However, Tensorflow is a cross-platform implementation where it can be migrated to alternative hardware components, which will reduces the development time.

TABLE OF CONTENTS

Chapter 3: Proposed Parallel Implementations for Solving SVM Optimization Problem

Chapter 4: Experimental Results and Discussion

LIST OF FIGURES

LIST OF TABLES

List of Abbreviations

ALU	**A**rithmetic **L**ogic **U**nit
API	**A**pplication **P**rogram **I**nterface
CPU	**C**entral **P**rocessing **U**nit
CU	**C**ontrol **U**nit
CUDA	**C**ompute **U**nited **D**evice **A**rchitecture
GDDR	**G**raphic **D**ouble **D**ata **R**ate
GPGPU	**G**eneral **P**urpose **G**raphics **P**rocessing **U**nit
GPU	**G**raphics **P**rocessing **U**nit
IC	**I**ntegrated **C**ircuit
KKT	**K**arush **K**uhn **T**ucker
MIMD	**M**ultiple **I**nstruction **M**ultiple **D**ata
MISD	**M**ultiple **I**nstruction **S**ingle **D**ata
ML	**M**achine **L**earning
MPI	**M**essage **P**assing **I**nterface
OpenCL	**Open C**omputing **L**anguage
OpenMP	**Open M**ulti **P**rocessing
QP	**Q**uadratic **P**rogramming
RAM	**R**andom **A**ccess **M**emory
SIMD	**S**ingle **I**nstruction **M**ultiple **D**ata
SISD	**S**ingle **I**nstruction **S**ingle **D**ata
SMO	**S**equential **M**inimal **O**ptimization
SMs	**S**treaming **M**ultiprocessor**s**
SPs	**S**treaming **P**rocessor**s**
SVM	**S**upport **V**ector **M**achine
UPC	**U**nified **P**arallel **C**