A High Performance Parallel Ranking SVM with OpenCL on Multi-core and Many-core Platforms

Huming Zhu, Xidian University, Xi’an, China
Pei Li, Xidian University, Xi’an, China
Peng Zhang, Xidian University, Xi’an, China
Zheng Luo, Xidian University, Xi’an, China

ABSTRACT

A ranking support vector machine (RSVM) is a typical pairwise method of learning to rank, which is effective in ranking problems. However, the training speed of RSVMs are not satisfactory, especially when solving large-scale data ranking problems. Recent years, many-core processing units (graphics processing unit (GPU), Many Integrated Core (MIC)) and multi-core processing units have exhibited huge superiority in the parallel computing domain. With the support of hardware, parallel programming develops rapidly. Open Computing Language (OpenCL) and Open Multi-Processing (OpenMP) are two of popular parallel programming interfaces. The authors present two high-performance parallel implementations of RSVM, an OpenCL version implemented on multi-core and many-core platforms, and an OpenMP version implemented on multi-core platform. The experimental results show that the OpenCL version parallel RSVM achieved considerable speedup on Intel MIC 7110P, NVIDIA Tesla K20M and Intel Xeon E5-2692v2, and it also shows good portability.

KEYWORDS

GPU, MIC, Multi-core, OpenCL, OpenMP, Parallel Computing, RSVM

INTRODUCTION

Learning to rank (LTR) is a kind of supervised learning method, and it has been proved effective in ranking problem. LTR has been widely applied in the field of information retrieval (IR), data mining, text retrieval, anti-spam, keyword extraction, and search engines (McFee & Lanckriet, 2010; Pan & Luo, 2011; Li & Zhou, 2013). LTR has developed into a series of mature methods, which can be divided into three categories: 1) Point-wise approaches (Li & Burges, 2007), 2) Pair-wise approaches (Freund & Iyer, 2003; Matveeva & Burges, 2006), 3) List-wise approaches (Cao & Qin, 2007; Xu & Huang, 2007). Ranking SVM (RSVM) is a typical algorithm of Pair-wise approaches, its main idea is to transform a target data points ranking problem into a binary classification problem of ordered Pair-wise points, and solve it with support vector machine (SVM) (Joachims, 2002; Cao & Xu, 2006).

RSVM has many unique advantages in dealing with small samples, nonlinear and high dimensional ranking problems, but like most of learning to rank methods, RSVM suffered a computation bottleneck in training, especially when solving large-scale data ranking problem. Thus, accelerating the training process of RSVM is a very valuable work. As known to all, the most time-consuming part of SVM is
to solve the quadratic programming (QP) problem. Sequential minimal optimization (SMO) algorithm is a popular and efficient solution of QP problem that can reduce the training time of SVM (Platt & Scholkopf, 1999). Thus, using parallel programming to accelerate SMO algorithm is a promising method to reduce the training time of RSVM.

Recent years, the rapid development of multi-core and many-core platform set off an upsurge of parallel programming. And that leads to appearance of many parallel programming models (CUDA, OpenCL, OpenMP, etc.). Under this circumstances, parallel programming has become a routine tool for accelerating algorithms in various domains, including image processing, scientific computing and data mining (Andrade & Trabasso, 2017; Schmiescheka & Shamardin, 2017; Yu, & Liu, 2015). Thus, the researchers suggest that accelerating RSVM with parallel programming is a practical method (Ksiaà & Ben Rejab, 2017). There are several researches about accelerating SVM has been done in these years. Previous research focuses on either designing efficient SVM tool for multi-core CPUs and many-core co-processors(MIC), or developing parallel strategies of SVM on GPU (Yan & Ren, 2015; You & Song, 2014; Kim & Kim, 2016). All these experiments achieved considerable speedups when comparing with serial algorithm. However, none of the research has realized a portable implementation of SVM for different platforms and compared the performance of RSVM with different parallel programming models when there are varieties of computing platforms and programming models (Pennycook & Hammond, 2013; Zhang & Sinclair, 2013).

In this paper, the authors proposed parallel RSVM based on OpenCL for multi-core and many-core platforms and compared the performance of RSVM with OpenMP on multi-core CPUs. Our main contributions are to do the comparison of the performance of OpenCL-based parallel RSVM on different computing platforms, and verify the portability based on OpenCL.

The rest of this paper is organized as follows: Section 2 briefly introduces basic concepts, including ranking SVM, SMO algorithm, computing platforms, and parallel programming model. Introduction of design strategy and implementation details of parallel RSVM algorithm with OpenCL and OpenMP are presented in section 3. Experimental results and analysis are shown in Section 4. At last, our work of the paper is concluded in Section 5.

BACKGROUND STUDY

Ranking SVM

RSVM is a kind of learning to rank method based on Pair-wise that transform the ranking problem into two-class classification problem and use SVM to solve the classification problem. Different with the SVM for classification, the output of ranking SVM is the grade of each object, i.e., for any \( x_i \succ x_j \), the output of ranking function is \( F(x_i) > F(x_j) \).

Suppose the training samples \( x_i \in \mathbb{R}^n, i = 1,2,...,l \) and the corresponding labels \( y_i \in \{+1,-1\}, i = 1,2,...,l \) are given, and \( l \) denotes the total number of training samples, \( n \) is the number of features. Training SVM for classification is equivalent to solving this SVM model:

\[
\begin{align*}
\min_{w,b,\varepsilon} & \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \varepsilon_i \\
\text{s.t.} & \quad y_i (w \cdot K(x_i) + b) \geq 1 - \varepsilon_i \\
& \quad \varepsilon_i \geq 0, i = 1,2,...,l
\end{align*}
\]

(1)

where the parameter \( C(C > 0) \) stands for penalty factor, the parameter \( w \) is the weight vector, \( K(x) \) represents kernel function. Then the original ranking problem can be transformed to its dual problem:
Guaranteeing Correctness for Collaboration on Documents Using an Optimal Locking Protocol
www.igi-global.com/article/guaranteeing-correctness-collaboration-documents-using/58631?camid=4v1a

Peer-to-Peer Desktop Grids Based on an Adaptive Decentralized Scheduling Mechanism
www.igi-global.com/article/peer-peer-desktop-grids-based/38975?camid=4v1a