A load-aware scheduler for large-scale neural network autotuning

PROJECT THESIS II / T2000

for the study program
Computer Science

at the
Baden-Wuerttemberg Cooperative State University Stuttgart

by
Dominik Stiller

Date	September 2019
Project Period	18 Weeks
Matriculation Number, Course	4369179, TINF17A
Company	Hewlett Packard Enterprise
Corporate Supervisor	Junguk Cho
University Supervisor	Prof. Dr. Bernd Schwinn

Declaration of Authorship

I hereby declare that the thesis submitted with the title *A load-aware scheduler for large-scale neural network autotuning* is my own unaided work. All direct or indirect sources used are acknowledged as references.

Neither this nor a similar work has been presented to an examination committee or published.

Sindelfingen	September 3rd, 2019	
Place	Date	Dominik Stiller

Abstract

Real-time computer vision applications with deep learning-based inference require hardware-specific optimization to meet stringent performance requirements. However, this approach requires vendor-specific libraries developed by experts for some particular hardware, limiting the set of supported devices and hindering innovation. The deep learning compiler stack TVM is developed to address these problems. TVM generates the optimal low-level implementation for a certain target device based on a high-level input model using machine learning in a process called autotuning.

In this paper, we first explore the capabilities and limitations of TVM's autotuning implementation. Then, we develop a scheduler to orchestrate multiple, parallel autotuning jobs on shared computation resources such as CPUs and GPUs, allowing us to minimize resource idle time and job interference. Finally, we reflect our design choices and compare the efficiency of our approach with the default, scheduler-less design.

Contents

List of Figures

List of Tables

List of Source Codes

1 Introduction

1.1 Motivation

AI is increasing in popularity include studies no automated performacne optimization for real time CV applications not easy for non-expert users required: automated performance optimization as a service

1.2 State of the Art

Test.2019

1.3 Task Description

research project

2 Deep Learning

2.1 Machine Learning

2.2 Neural Networks

2.3 Convolutional Neural Networks

describe convolutions

3 Autotuning

3.1 Deep Learning Inference Optimization

many possible implementations only few optimal ones for target device

3.2 Manual Optimiziation

requires deep knowledge of target device limitations

3.3 Automated Optimization

using machine learning vendor-agnostic describe autotuning process

3.4 Issues with current autotuning design

lots of idle time

4 SimpleTVM

create wrapper for simpler usage of TVM expose easy, chainable interface

4.1 Automated Benchmarking Framework

Explore TVM behavior

5 Scheduler Design and Implementation

5.1 Design Principles

prototype touch TVM code as little as possible build on SimpleTVM should work with a variable number of clients

5.2 Scheduler Design

interleaving while keeping dependencies agnostic of stages, necessity for call next/is done methdods allows for multiple strategies

5.3 Approaches

In process vs RPC

5.4 RPC

schedulers and clients describe endpoints

6 Evaluation

Comparison of interleaved design vs synchronous and sequential

7 Conclusion

Describe results

7.1 Future Work

make into mature product

Predictive scheduler using times for task to make scheduling more intelligent, as idea for future work Keep trained model and update it every n new entries to skip loading history time for every task Check currently known best configurations and see if SLA is met before actually starting autotuning