Inference at the edge: tuning compression parameters for performance

Deliverable 1: Final year Dissertation

Bsc Computer Science: Artificial Intelligence

Sam Fay-Hunt — `sf52@hw.ac.uk`

Supervisor: Rob Stewart — `R.Stewart@hw.ac.uk`

March 16, 2021

DECLARATION

I, Sam Fay-Hunt confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed: .....Sam Fay-Hunt...........

Date: .....10/12/2020...............

Abstract: *Abstract here*

Contents

1 Introduction **1**

2 Background **2**

3 Methodology **2**

3.1 Overview 2

3.2 Conceptual Process 2

3.3 Filter and channel selection 2

3.4 Engineering/implementation details 3

3.4.1 High level overview of system 3

3.4.2 Benchmarking 4

4 Evaluation **4**

4.1 Evaluation of experimental design 4

4.2 Evaluation of results 5

5 Conclusion **5**

5.1 Further work 5

5.2 Discussion 5

A Back matter **5**

A.1 References 5

1 Introduction

- *Introduce terminology - Inference, neural network model, pruning, layers, channels, filters*
- *Introduce models to be used - high level conceptual representation of the models*
- *Introduce hypothesis*
- *Describe research aims*
- *Define project objectives*
- *Describe how this work contributes to further research*

2 Background

- *Adapt from D1*
- *rewrite with more of a focus on the concrete channel and pruning methodology used*
- *Would be good to include wandb bayse hyperparam optimisation details*

3 Methodology

3.1 Overview

- *Questions to be addressed*
- *Metrics to be measured - why*

3.2 Conceptual Process

- *Sensitivity analysis - filter/channel selection and layer interdependencies*
- *Filter pruning implementation - Theory*
- *Channel pruning implementation - Theory*
- *Retraining pruned model*

3.3 Filter and channel selection

Link back to selected model - concrete examples of process described in previous section

- *Filter selection (visual representation of filters)*
- *Channel selection (visual representation of channels)*
- *Discussion of pruning consequences (and recovery) -¿ top1/top5 before retraining and after*

3.4 Engineering/implementation details

- *High level overview of physical system - justify need for multiple training agents*
- *Pruning & retraining setup - Distiller (Pruning & training)*
- *Benchmarking setup - openvino + benchmark (getting latency/throughput)*
- *Data processing - wandb + data visualisation steps*

3.4.1 High level overview of system

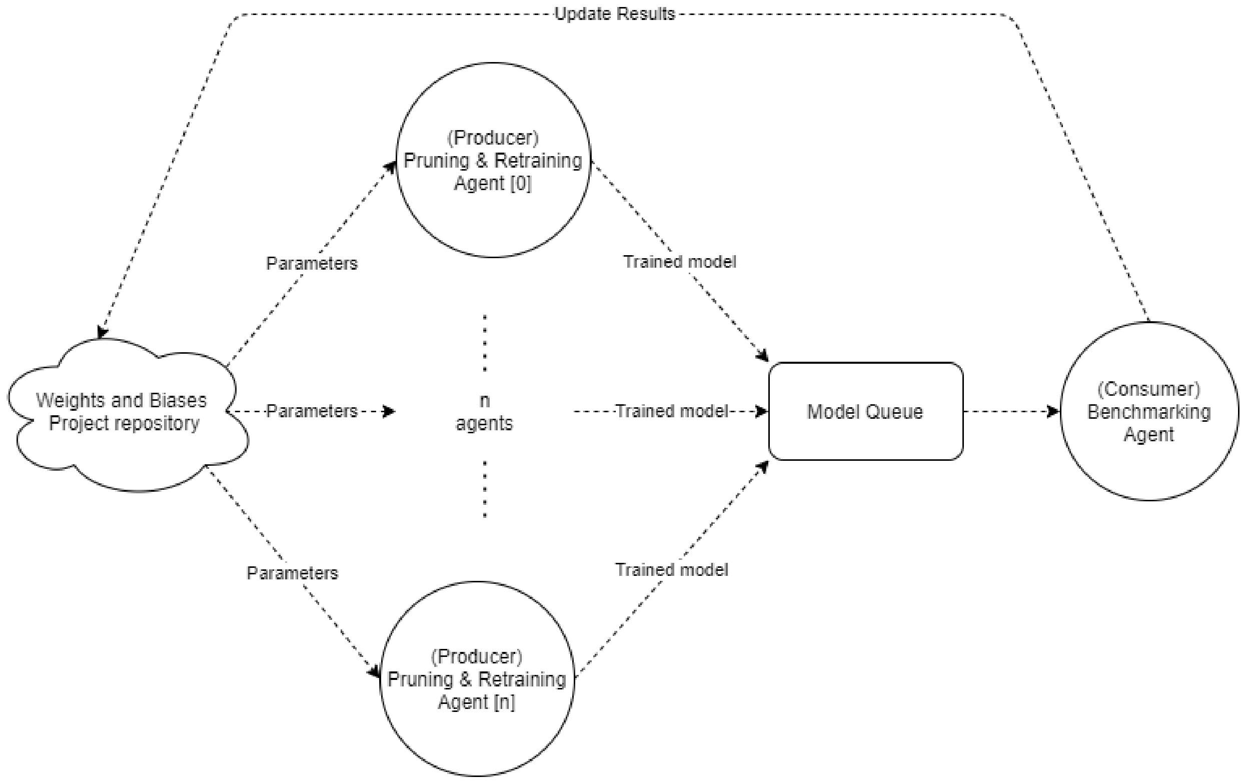
Update Results
(Producer) Pruning & Retraining Agent [0]
Parameters
Trained model
Weights and Biases Project repository
Parameters
n agents
Trained model
Model Queue
(Consumer) Benchmarking Agent
Parameters
Trained model
(Producer) Pruning & Retraining Agent [n]

Figure 1: Diagram showing agent communication

While the process of pruning is relatively fast, retraining the network to regain lost accuracy can very demanding. To handle this problem we separated the benchmarking system (consumer) from the pruning and retraining systems (producer), this made it easy to add new pruning and benchmarking agents to a single experiment or run multiple experiments in parallel.

When pruning begins, the producer agent requests the (initially random) pruning parameters from the Weights and Biases Project server, the producer then applies the pruning algorithm and begins retraining the model. Upon completion of retraining the model is exported into ONNX format and added to a queue for the consumer (the benchmarking agent) to benchmark and record the results, these results are then logged to weights and biases.

3.4.2 Benchmarking

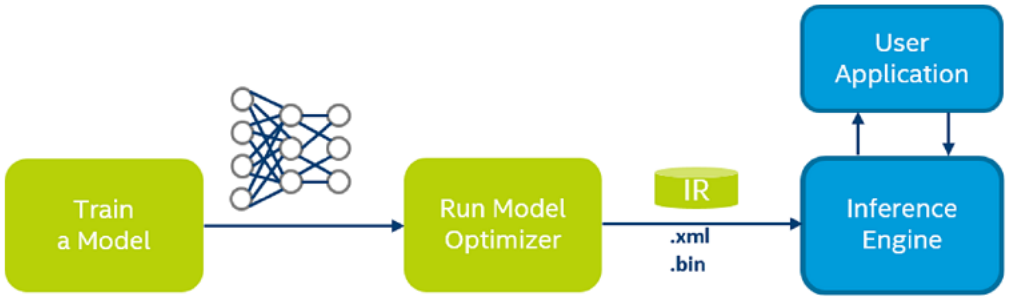

Figure 2: Workflow for deploying trained model onto NCS [1]

To pass the pruned and trained model to the Neural Compute stick we used OpenVino; a toolkit providing a high level **inference engine(Ensure this is defined)** API, this facilitates the process of optimizing the model for specialised hardware (in this case the NCS). OpenVino itself has a Benchmarking tool that we leverage to access to detailed latency and throughput metrics,

4 Evaluation

4.1 Evaluation of experimental design

- *Duration of training*
- *volume of data gathered*
- *(im)practicalities - power consumption?*
- *limitations - single optimisation metric*
- *Criticism of methodology*

4.2 Evaluation of results

- *Summary of results per model/dataset*
- *Deep dive into results, detailed visualisations of accuracy & latency tradeoffs (maybe example with poor quality sensitivity analysis vs higher quality layer selection)*
-

5 Conclusion

5.1 Further work

- *Suggested improvements for methodology*
- *Next steps*

5.2 Discussion

- *Discuss results*

A Back matter

A.1 References

References

[1] (). "Model Optimizer Developer Guide - OpenVINO™ Toolkit," [Online]. Available: `https://docs.openvinotoolkit.org/latest/openvino_docs_MO_DG_Deep_Learning_Model_Optimizer_DevGuide.html` (visited on 03/15/2021).