Branch Prediction Through Neural Networks

Minchan & Kwon
1012871803, minchan.kwon@mail.utoronto.ca
https://colab.research.google.com/drive/1TJG8gOfIr_fHjWi9OP46okk2Iw-5RrvZ?usp=sharing

Introduction

Branch predictors are digital circuits inside a computer chip that tries to guess in which direction a branch should take place. The purpose of a branch predictor is to improve performance in modern computer chips that take advantage of deeply pipelined architecture. A branch happens in if-then-else statements, where a branch is taken when a certain condition is not met. If it weren't for branch predictors, modern computer chips that process multiple instructions at a time should halt the execution of instructions when it runs into an if-then-else statement. Therefore, it is crucial that an accurate branch predictor exists to enhance computational power.

Modern branch predictors are based on hash tables that keep track of the history of branches that have taken place inside a running program. This history can be interpreted as a sequential data that a neural network can learn from. A deep learning algorithm can learn dependencies between branches that are far away along the time domain through this data. Since RNNs are designed to process sequential data and time series, it would be a reasonable idea to employ RNNs for predicting branches. This adoption of DL to branch prediction may have the potential to reduce computational bottleneck.

Illustration

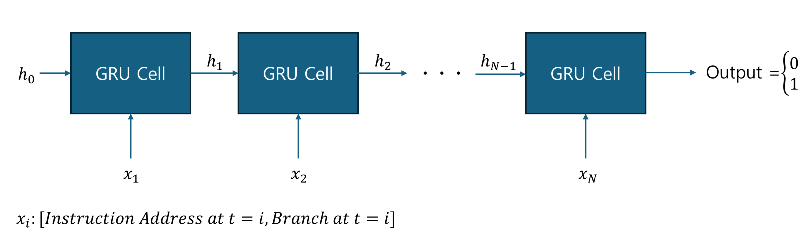

Figure 1: Simple illustration of a branch predicting RNN model.

Background & Related Work

(Lin, 2019) argues that modern branch predictors that achieve near-perfect accuracy still struggle with hard-to-predict(H2P) branches. By correcting the mispredictions of modern branch predictors, we can achieve improvements in performance at the same extent as furthering a generation of computer chips. (Tarsa, 2019) demonstrated that implementing a CNN model to predict the H2P branches yielded a 36.6

(Zangeneh, 2020a) built up on previous work, introducing BranchNet, a CNN model that can accurately predict H2P branches. He points out that modern branch predictors have no way of knowing which branch in the global history is actually useful for prediction. Modern predictors that store branch history in run-time don't have the capacity to store branch history when two correlated branches are very far way due to memory constraints. On top of that, he argues that despite the strong performance of CNNs as branch predictors, researchers should prioritize designing custom logic and specialized structures to predict branches efficiently((Zangeneh, 2020b)).

An older paper published in 2004((Smith, 2004)) studies the potential use of RNNs and concludes it is not as effective due to its longer delay in predicting a branch. He argues that the perceptron predictor is the best overall predictor that shows acceptable performance in both computation time and accuracy.

Data Processing

Our data source will be a pre-collected public dataset from Kaggle, containing benchmark traces from the 3rd Championship Branch Prediction (CBP). Each branch instance in the dataset is represented by a 480-bit feature vector composed of three main components: the binary address of the instruction, the global history of previous branch outcomes, and the global address history of previous instructions. The corresponding label is a single bit indicating if the branch was taken or not-taken. Our main processing step will be to load and combine these components into a final input matrix, where each row represents a single branch and has 480 columns.

Architecture

We will primarily implement Recurrent Neural Networks (RNNs), using LSTM or GRU cells to sequentially process the 200-step branch history and capture temporal patterns. The output will be processed using a sigmoid activation to classify the branch as taken or not-taken. As a point of comparison, we will also develop a 1D Convolutional Neural Network (CNN) to act as a parallel pattern detector which will also use a sigmoid activation to output a binary prediction for the branch.

Baseline Model

Our baseline model is a non-machine learning heuristic: the 2-bit saturating counter. For each unique PC encountered in the trace, a 2-bit counter tracks one of four states: 'Strongly Not-Taken' (00), 'Weakly Not-Taken(01)', 'Weakly Taken(10)', and 'Strongly Taken' (11). Predictions are based on the most significant bit (0 for Not-Taken, 1 for Taken). After the branch outcome is known, the counter is updated. The counter is incremented on a taken branch and decremented on a not-taken branch, saturating at both ends. We will compare our model with the baseline model on the set of H2P branches.

Ethical Considerations

The open source dataset from Kaggle consists of instruction addresses and branch decisions at the specific address. Therefore, it doesn't contain any personally identifiable information. In terms of data collection, this project is free of ethical concerns. The goal of our algorithm is to enhance prediction accuracy during program execution. It aims to improve CPU performance and energy efficiency which doesn't carry any ethical challenges

References

Lin, C. K., T. S. J. (2019). Branch prediction is not a solved problem: Measurements, opportuni- ties, and future directions. *arXiv preprint*.

Smith, A. (2004). Branch prediction with neural networks: Hidden layers and recurrent connections. **Department of Computer Science University of California, San Diego La Jolla, CA*, *92307*.*

Tarsa, S. J., L. C. K. K. G. C. G. W. H. (2019). Improving branch prediction by modeling global history with convolutional neural network. *arXiv preprint*.

Zangeneh, S., P. S. L. S. P. Y. N. (2020a). ranchnet: A convolutional neural network to predict hard-to-predict branches. **2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)* (pp. 118-130).*

Zangeneh, Siavash, P. S. P. Y. (2020b). Branch prediction with multilayer neural networks: The value of specialization. **Machine Learning for Computer Architecture and Systems*.*