TENSORFLOW FOR ABACUS PROCESSING UNITS

Robert McCraith
Abacus Computing Society

ABSTRACT

Machine learning has swept the world of computing solving all kinds of problems from image recognition to natural language processing. Current Machine learning frameworks require years of artisan Python coding experience which sets a high barrier to entry for the ambitious young researcher. In recent months supply constraints have resulted in a shortage of Graphical Processing Units for the average consumer resulting in renewed interest in alternative computing techniques being desirable. With these difficulties in mind we introduce $Tensorflow$ for Abacus Processing Units. $Tensorflow$ for Abacus Processing Units utilises bespoke hardware for highly parallel low power compute, requiring 0 watts of electricity to run some of the most complex neural networks. We also demonstrate that the performance of many popular operations are performed in sub linear time and model parameters are fully interpretable unlike many other systems.

I have an abacus at home

Conan O'Brien,
NeurIPS, 2021

1 INTRODUCTION

After the dark AI winter of the late 90's, early 2000's a proliferation of Machine Learning Frameworks has inundated the discerning pythonista. These frameworks all require a large number of CUDA cores and a expensive NVIDIA GPU which is often better used mining Bitcoin.
The Tensorflow library has been optimised to run on low power devices from Mobile Phones, Raspberry Pi's and even on web through JavaScript. These platforms however can also be used to watch Tik Tok videos.
With this in mind we propose a new computing platform for training Machine Learning Models without having to pay for electricity so you can save funding to travel to exotic parts of the house to ~~play pokemon~~ attend conferences on Gather.town. Our contributions are as follows:

- We show how an Abacus Processing Unit can be used to learn complex functions from high dimensional inputs
- Demonstrate how this system saves the user electricity, improves parameter interpretability and ensures understanding of fundamental concepts of machine learning making the initial learning curve the lowest of all libraries
- Lazy evaluation first: our method only evaluates the outputs of layers when you want them, which also has an effect similar to dropout but with even grater stochasticity
- Sparse computation: $Tensorflow$ for Abacus is sparse first computation, no need to waste resources calculating unnecessary values

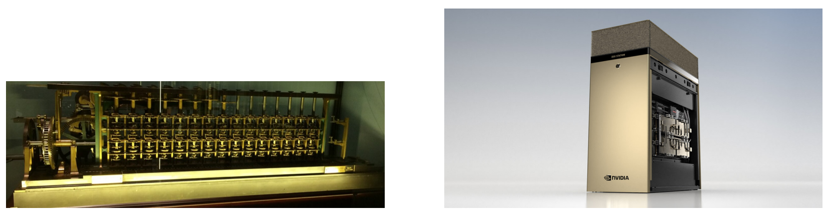
2 RELATED WORK

Computing devices throughout history have varied widely in their implementations. Many societies made markings on stone to count resources as in Fig. 1(a) where the Mesopotamian's provide the first example of a non-overwritable set of weights for scene comprehension. The next major

(a) Mesopotamian mathematical tablet (8) predating the iPad by about 3800 years

(b) Roman Abacus (7) allowing the first portable computing device with write/write io

Figure 1: Early computation platforms. Note that both ancient societies understood the importance of $\mathbb{N}$ as the one true number system



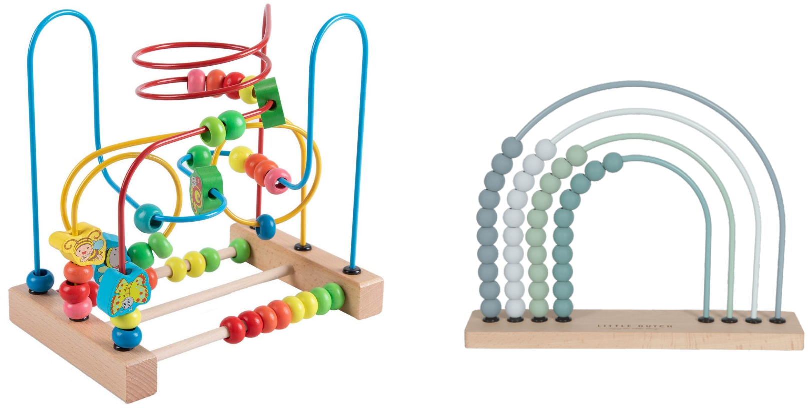
(a) Difference Engine

(b) NVIDIA DGX Station

Figure 2: Two contemporary compute devices

development is the romans abacus seen in Fig. 1(b) which reduced weight while also allowing frequent read/write cycles. As the first Turing complete computation device the abacuses simplicity allows for multiple variables to be manipulated simultaneously. After the creation of the Abacus many other computing form factors have been created but mostly very derivative form factors and typically at greater cost and higher likelihood of coding errors.

More modern computation platforms such as the Babbage Difference Engine as seen in Fig. 2(a) demonstrate the backwards thinking of modern device design. The difference engine weighs orders of magnitude more than the Roman Abacus Portable Compute Platform not to mention the additional manufacturing difficulties. The NVIDIA DGX Station also consumes high levels of electricity and makes reading parameter values difficult in hardware as the transistors are very small. The Abacus Computing Society's latest standardised compute platform on the other hand is very approachable to not only all budget levels but also to a wide range of ages, while retaining the excellent portability that made the earlier iterations so compelling. We believe that devices such as the one depicted in Fig. 3(a) makes machine learning approachable to practitioners of all ages with some syntactic sugar (represented here by a butterfly), syntax highlighting (multiple colours depicting weight magnitude), and complex geometry.



(a) An highly interactive REPL environment for quick prototyping of neural networks (5)

(b) Non-Linear Operator layer in $Tensorflow$ for Abacus (2)

Figure 3

3 Forward Pass Computations

Forward Pass computations are simple on $Tensorflow$ for Abacus, in Fig 4 the top left abacus is used to represent the input variables. Each layer of the network is then represented by an additional abacus. This allows for computation graphs custom built for deployment in real time systems. The Abacus naturally implements many useful activation functions, namely $ReLUX$ where X is chosen at training time and is performed in place with a computation time of $\mathcal{O}(0)$ making it more efficient than other computation devices. We also have natural quantisation of weights and input/output parameters. We can also implement other learning algorithms such as Logistic Regression, Support Vector Machines, and K-means as in Fig. 3(b) which allows us separate inputs with a high dimensional plane. In fact with more complex 3 dimensional Abacus processing units having weights in a superposition allows us to compute an ensemble of models simultaneously. In theory using gravity and other physical phenomena the possibility of an Abacus Processing unit with weights and parameters constantly allowing us to explore an immeasurable number of possible weights and network architectures which gives us the possibility of excellent performance even without training as in (3).

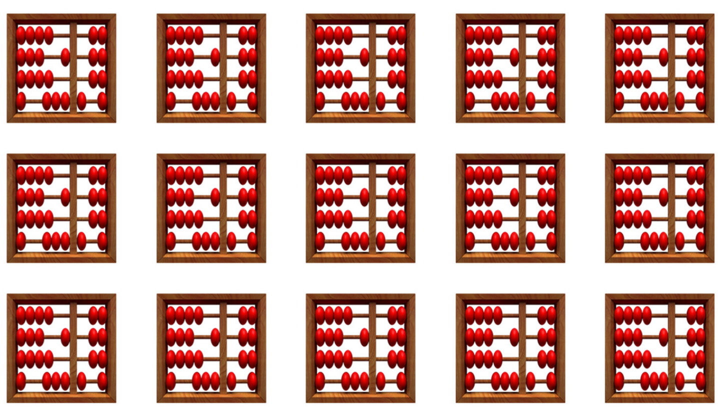


Figure 4: Example Computation graph for a simple Neural network (4)

Figure 5: Example of a model performing backprop (9)

4 Backwards Pass

As in the forward pass, backward passes through the network can be quickly performed by the user. To as with other libraries $Tensorflow$ for Abacus Processing Units provides a simple programatic hook to direct computations backwards through the graph by simply picking up the layer (Abacus) used to derive the current layers input values and using the chain rule to derive the updates to parameters required to improve performance. This has two benefits:

- Allows the user to constantly remember the basics of machine learning, differential calculus and linear algebra
- The values parameters take can be explicitly understood as the changing of weights is exposed to the user

Fig. 5 demonstrates a computation graph in performing backpropagation. Note that the abacus does not provide numbers below 0 as you don't need that kind of negativity in your life, the supremacy of $\mathbb{N}$ numbers should be obvious to the reader. While many other frameworks view quantisation and interpretability as advanced features these are considered fundamental to our system which we view as a fundamental building block on the path to applying machine learning to the real world where understanding the outputs of a network relative to all possible inputs is highl desirable in safety critical situations.

5 Conclusion

In summary our new framework and hardware platform provides:

- A zero code solution to train neural networks of infinite complexity. Which democratises machine learning beyond those capable of writing code.
- A total of 0 Watts are needed to perform a forward and backwards pass with our low power compute devices
- Ensures the users fundamental understanding of mathematical and computational basics are preserved in user memory making our framework useful for both beginners and advanced users alike

5.1 FUTURE WORK

While Abacus Processing Units are viewed by the authors as complete computation devices we also feel that the Rubik's Cubes are a theoretically interesting physical computation device. With 43 quintillion (6) possible configurations the possibility of encoding complex functions in a physical device which closely resembles a tensor (1). Excel may also be a possible computation platform for training machine learning algorithms, due to the proven Turing completeness of Powerpoint, Excel presents the unique advantage of being a skill that many job applicants possess making the application of machine learning to a wider array of problems a much simpler task except maybe in the public sector (10).

REFERENCES

[1] Introduction to tensors. https://www.tensorflow.org/guide/tensor, 2021. Accessed: 2021-03-26.

[2] Little Dutch. Rainbow abacus blue. https://www.little-dutch.com/en/new/rainbow-abacus-blue, 2021. Accessed: 2021-03-26.

[3] Adam Gaier and David Ha. Weight agnostic neural networks. 2019. https://weightagnostic.github.io.

[4] Chaim Gartenberg. Apple's abacus emoji is wrong. https://www.theverge.com/tldr/2019/5/26/18639006/apple-abacus-emoji-wrong-historically-inaccurate-math, 2019. Accessed: 2021-03-26.

[5] Jacootoys. Jacootoys toddlers bead maze roller coaster animal circle toys educational abacus beads game for boys girls baby. https://www.amazon.co.uk/Jacootoys-Toddlers-Roller-Coaster-Educational/dp/B082W3TKK4/ref=asc_df_B082W3TKK4/?tag=googshopuk-21&linkCode=df0&hvadid=430732564689&hvpos=&hvnetw=g&hvrand=9975284137850090030&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=1006976&hvtargid=pla-909353806781&psc=1&tag=&ref=&adgrpid=102609725880&hvpone=&hvptwo=&hvadid=430732564689&hvpos=&hvnetw=g&hvrand=9975284137850090030&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=1006976&hvtargid=pla-909353806781, 2021. Accessed: 2021-03-26.

[6] Andy Kiersz. Any rubik's cube can be solved in 20 moves, but it took over 30 years for anyone to figure that out. https://www.businessinsider.com/rubiks-cube-gods-number-steps-to-solve-any-cube-2019-1?r=US&IR=T, 2019. Accessed: 2021-03-26.

[7] Evelyn Lamb. Cumbersome calculations in ancient rome. https://thatsmaths.com/2019/06/27/cumbersome-calculations-in-ancient-rome/. Accessed: 2021-03-22.

[8] Peter Lynch. Ancient babylonian number system had no zero. https://blogs.scientificamerican.com/roots-of-unity/ancient-babylonian-number-system-had-no-zero/, 2019. Accessed: 2021-03-22.

[9] Melissa and Doug. Add and subtract abacus. https://www.mulberrybush.co.uk/add-and-subtract-abacus, 2021. Accessed: 2021-03-26.

[10] Simon Thorne. Excel errors: the uk government has an embarrassingly long history of spreadsheet horror stories. https://theconversation.com/excel-errors-the-uk-government-has-an-embarrassingly-long-history-of-spreadsheet 2020. Accessed: 2021-03-26.