Parallelization of Machine Learning Algorithms

Ayobami Ogunmolasuyi, Joseph Anderson
University of Maryland Eastern Shore, Salisbury University

ABSTRACT

The idea of machine learning is giving a computer the ability to make connections between different objects or events that may not be explicitly captured by the data A problem that arises in this quest to train computers is the time it takes to train them.

GPUs have been used to parallelize the training phase. However, these GPU-based machines have reached the limit of their speed, hence more speedup for machine learning training does not seem feasible.

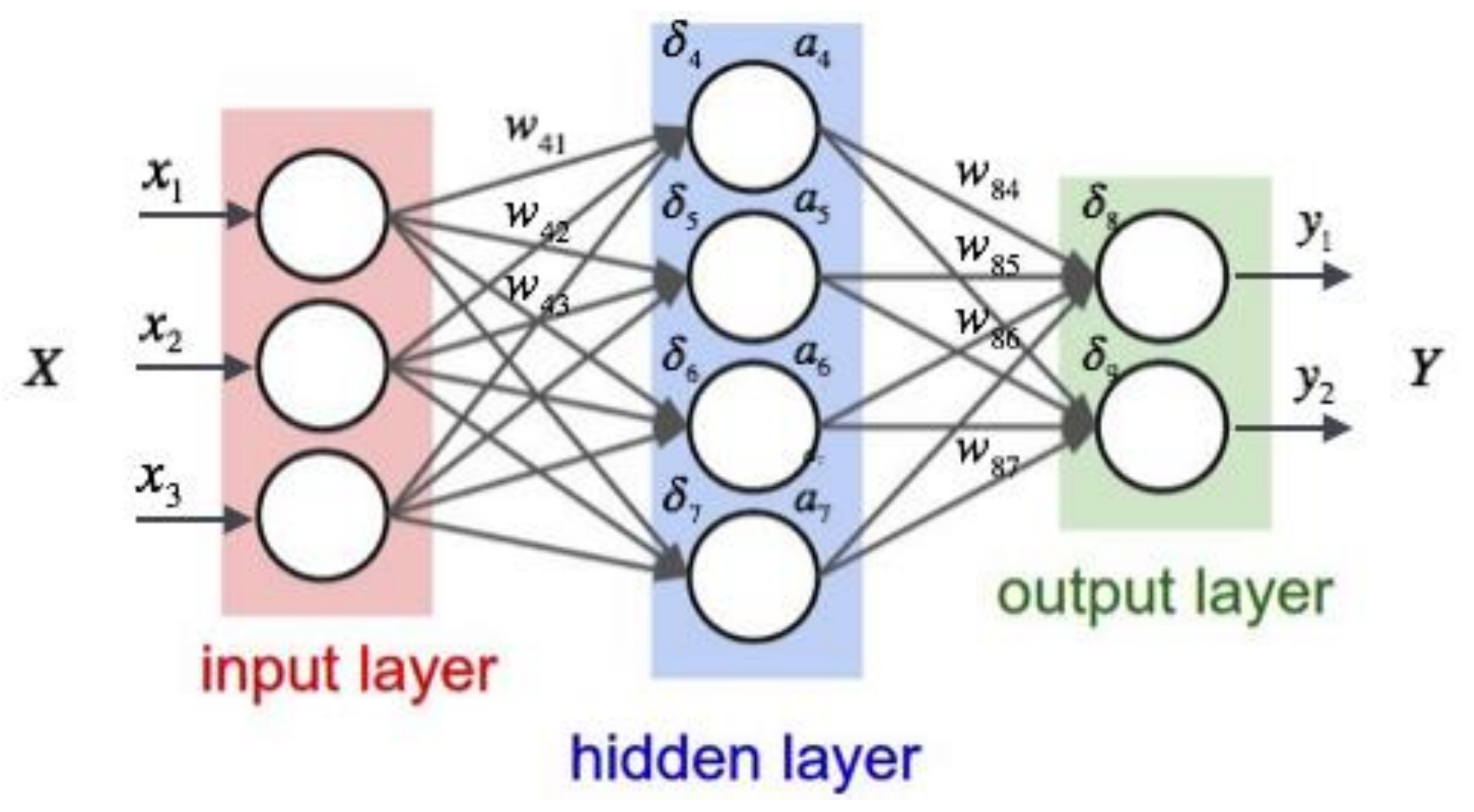
This work examines the Multigrid Reduction in Time algorithm(MGRIT) that parallelizes over the time domain.

PROJECT GOALS

- Gain adequate knowledge of machine learning algorithms and what is involved in parallelizing these algorithms.
- Gain in-depth understanding of the concept of the multigrid reduction In Time algorithm.
- Apply this algorithm to a non-machine learning problem as the algorithm is initially employed in physics for time-independent problems.
- Go on to apply this algorithm to a basic neural network.
- Implement the possible improvements to the papers upon which this work is based.

BASE NEURAL NETWORK

A neural network is a system of interconnected nodes that perform computations to learn by analyzing data and making connections between the data . Neural networks are a means of doing machine learning.



The goal is to update the weights of the network automatically from data such that the predicted output is close to the target output for all inputs x.
It does this by a method called forward propagation where the network applies an activation function to the inputs, x.
Hence, y = f(x) gives y.
It seeks to correct the error by a back propagation mechanism where the derivative of the error function is used to update the initial

$$w_{ij} = w_{ij} - \alpha \frac{dE}{dw_{ij}}$$

The neural network used in this project is one that does a basic labelling operation.
The aim is to scale the performance of MGRIT with a simple neural network and then move on to more complex neural networks.
In the MGRIT framework, $\Phi(w_i)$ represents one forward propagation of the weights and a back propagation of the y outputs to update the weights.

THE MGRIT ALGORITHM

In order to efficiently understand the concept of MGRIT, it would be better to think of the training phase of a neural network as a time-dependent process.

Let w_i be a vector representing the solution(weights) at time $t = t_i$.

$w_{i+1} = \Phi(w_i)$, for some initial w_0, and i = 0, 1, . . . , N

Next, we define a uniform–temporal grid with time -step δt and nodes t_i , i = 0, . . , N_t .

Furthermore, define a coarse temporal grid with time-step $\Delta T = m\delta t$ and nodes $T_i = i\Delta T$, i = 0, 1, . . . , N_t/m, for some coarsening factor, m.

Let w_i be a vector representing the solution at time $t = t_i$, discretized on some spatial grid.

When vector w_i is is discretized on the time grid, the equation becomes:

$$w_{i+1} = \Phi(w_i , \delta t) + g_i ,$$

We would consider the two-level version of the multigrid algorithm.

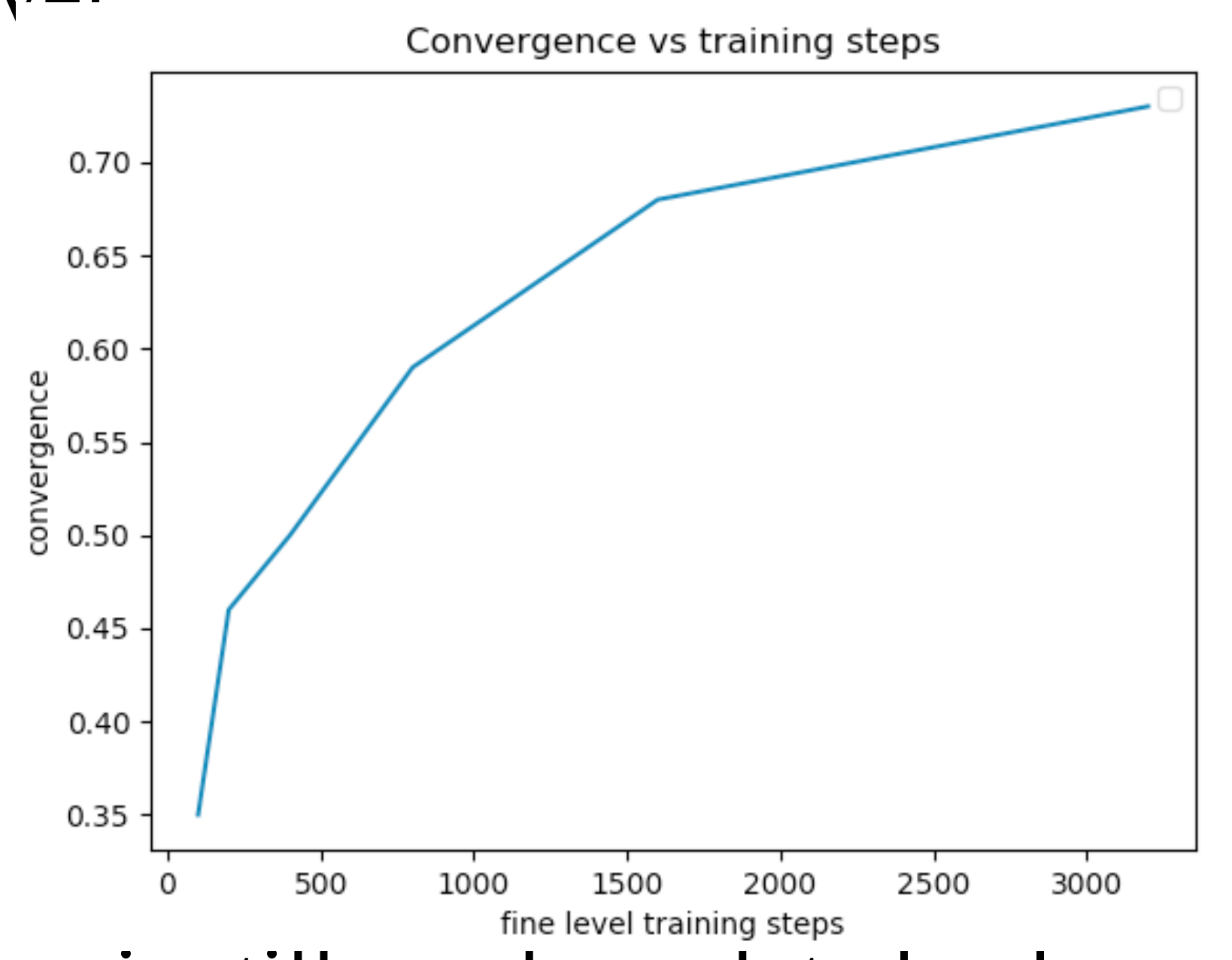
After this, MGRIT employs four mechanisms called Restriction, Interpolation, F-Relaxation and C-relaxation that aim to increase the convergence rate and ensure the neural network converges to the same result as sequential time stepping.

```
Algorithm 1 MGRIT($A^{(0)}$, $w^{(0)}$, $g^{(0)}$, $tol$)
1: Apply F- or FCF-relaxation to $A^{(0)}(w^{(0)}) = g^{(0)}$.
2: Inject the fine level approximation and its residual to the coarse level:
   $w_i^{(1)} \leftarrow w_{mi}^{(0)}, \quad r_i^{(1)} \leftarrow g_{mi}^{(0)} - \left(A^{(0)}(w^{(0)})\right)_{mi}$.
3: Solve $B^{(1)}(v^{(1)}) = B^{(1)}(w^{(1)}) + r^{(1)}$.
4: Compute the coarse level error approximation: $e^{(1)} \simeq v^{(1)} - w^{(1)}$.
5: Correct $w^{(0)}$ at $C$-points: $w_{mi}^{(0)} = w_{mi}^{(0)} + e_i^{(1)}$.
6: if $\|g^{(0)} - (A^{(0)}(w^{(0)}))\| \le tol$ then
7:     Update $F$-points with apply F-relaxation to $A^{(0)}(w^{(0)}) = g^{(0)}$, and halt.
8: else
9:     Go to step 1.
10: end if
```

The paper upon which we hope to build on with our research has shown that it is possible to parallelize across training steps with MGRIT.

RESULTS, CONCLUSIONS AND FUTURE WORK

Much of the work that has been done on the Multigrid Reduction in Time algorithm is still very basic. The graph below quantifies convergence of the algorithm with respect to the number of training steps on the finest level



There is still much work to be done with this algorithm. We look forward to applying this algorithm to larger neural networks and neural networks with larger examples and also what could be done to provide possible speedup. One of these possibilities would involve making the learning rate, alpha dynamic across the different levels considering how they have different training steps.

REFERENCES

[1] Jacob B. Schroder. Parallelizing Over Artificial Neural Network Training Runs with Multigrid. (n.d.). Retrieved

from https://www.semanticscholar.org/paper/Parallelizing-Over-Artificial-Neural-Network-Runs-Schroder/beeada3fa99961ac951282535b0632ce5b832a60?tab=abstract

[2] N. P. V. Dobrev, Tz. Kolev and J. Schroder, Two-level convergence theory for multigrid reduction in time (mgrit), SIAM J. Sci. Comput. (to appear), (2016). LLNL-JRNL-692418.

ACKNOWLEDGEMENTS

This project was funded by the National Science Foundation.