Validating Efficiency of the Forward-Forward Algorithm

Nowell Kevin Reyes (20658133) and Gabriel Schussler (20470601)

Project proposal for COMP3035 Discovery Project

Supervisor: Dr. Rosalind Wang

School of Computer, Data and Mathematical Sciences, Western Sydney University

Spring, 2020

1 Background

This section is written by Kevin

Artificial Neural Networks (NN) have made significant progress since the introduction of the Perceptron in the late 1950s. The development of the backpropagation (BP) algorithm, introduced in 1986, became the backbone of NN training and has since become the fundamental approach to train most large-language models (LLMs) like ChatGPT, known for their impressive ability to generate human-like text [14].

The strength of the BP algorithm lies in its ability to fine-tune the network's parameters through an iterative process of forward and backward passes. In a forward pass, the input data moves through the network, generating predictions. The backward pass then calculates the error between these predictions and the actual results, sending this error back through the network to adjust the connection weights[14, 10]. This repeated process, also known as epoch, enables the network to learn complex patterns from data.

However, as these NN models grow in size and complexity, the limitations of BP are becoming more evident. While undeniably effective, the algorithm's high computational power and memory requirements lead to scalability issues, especially in environments with limited resources [16]. In 2020, OpenAI released the technical overview of its famous model GPT-3, with its 175 billion parameters, would take 355 years and cost $4.6 million to train, even on the lowest price cloud infrastructure at the time [17]. Beside the high costs, there are also concerns about the environmental impact of training such models [13]. These challenges have led researchers to look for alternative training methods. Techniques like contrastive divergence [4] and target propagation [8] have shown potential in overcoming some of BP's drawbacks, but none have yet to replace it as the main method for training NNs.

In 2022, one of the authors of the original back propagation algorithm, Geoffrey Hinton, proposed a new approach to NN training, called the Forward-Forward algorithm (FF) [3]. The FF's design was motivated by the search for more biologically realistic learning processes, and the desire for more efficient, localised learning rules, to better model the human brain [3, 1]. This method replaces the traditional forward-backward cycle with two forward passes, each with its own objective. Unlike BP, which relies on error gradients moving backwards through the network, FF uses two forward passes: a positive pass for real data, and a negative pass for generated or corrupted data. Each layer in the network has its own objective function, which aims to maximize the positive pass and minimize the negative pass. This localised learning approach might allow for more parallel computation and help address issues like BP's vanishing gradient problem [6].

Although the FF is still in its early stages, it offers an exciting new direction for research in NN training. Upon initial literature review, early studies have looked into different architectures [11, 15], hyperparameters [2], and potential applications [9, 12] of the algorithm, but a comparison of the FF and BP in terms of training speed and memory efficiency has yet to be done. This gap in the research field offers a valuable opportunity that we'll explore with this project - to extend our knowledge of the literature, and upon reproduction of the `Python` implementation of the algorithm, we hope to contribute to the understanding of the Forward-Forward algorithm's practical advantages and limitations, pushing forward its potential applications in neural network training.

2 Objective

In this project, we aim to investigate the Forward-Forward algorithm, focusing on its performance in a supervised learning task, as described in Part 3.3 of the Hinton's paper.

This section is written by Kevin & Gabriel

Our first objective is to develop a sound understanding of the FF method by studying, reproducing and validating the `Python` implementation of the algorithm, to ensure its accurate representation of Hinton's version (originally written in MATLAB).

We then aim to quantify and compare the computational efficiency of the FF and BP by measuring training time (per epoch and as a whole) and memory usage allocation for both algorithms using benchmark datasets and simple network architectures while maintaining similar accuracy results from the paper. This comparison will help determine which algorithm is more efficient in terms of computational resources and evaluate potential performance trade-offs against efficiency gains.

Finally, we will analyse our results in the context of practical applications, identifying scenarios where the FF might be preferable to BP (and vice versa), and outline potential directions for future research in this area.

3 Hypothesis/Research question

Does the Forward-Forward algorithm offer significant improvements in training time and/or memory efficiency compared to backpropagation for supervised classification tasks?

This section is written by Kevin

4 Methodology

4.1 Data

This section is written by Gabriel

In this project, two widely used datasets will be used to benchmark the efficiency of the Forward-Forward algorithm compared with the traditional backpropagation algorithm in training image processing neural networks.

4.1.1 MNIST

The first data set to be used in this project is the `MNIST` [7] data set, published by the National Institute of Standards and Technology. The data set consists of 60,000 training images and 10,000 testing images of Hand written digits converted to 28 * 28 pixel images. The `MNIST` data set is a commonly used dataset to benchmark the performance of image processing Neural Networks, hence it has been chosen for this project.

4.1.2 CIFAR-10

The second data set to be used in this project is the `CIFAR-10` [5] data set, published by the Canadian Institute of Advanced Research. The data set consists of 10 classes of image, varying widely in type, with 60,000 images in total. Similar to the `MNIST`

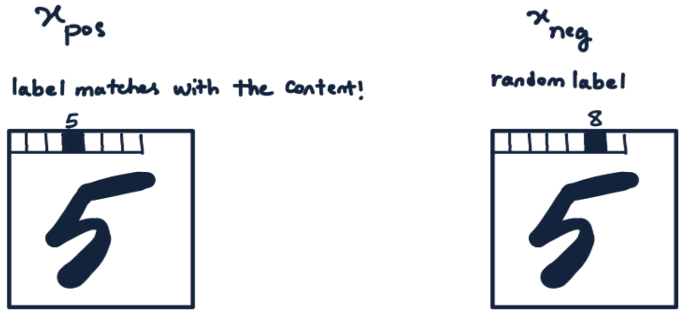


Figure 1: Visual depiction of positive and negative labels will be appended to each image. Sourced from `https://github.com/mpezeshki/pytorch_forward_forward`

data set, the `CIFAR-10` data set is commonly used to benchmark image processing neural networks.

In the case of the Forward-Forward model, the image class will be appended to the beginning of each image, this will allow the model to train on positive and negative images required in the learning process of the FF algorithm, as shown in Figure 1. In the case of the backpropagation, the image and class will be fed into the model as separate variables.

4.1.3 Tools/Code

The tools and code in the project will consist of open-source software and data in the effort to maintain repeatability. The neural network models will be created using the `Python` library `PyTorch`, a widely used machine learning package.

Another `Python` library, `psutil`, will be used to measure the usage of computing resources of each model. This library will give us the tools to measure several compute resources important to main objectives of this project, namely, RAM, GPU utilization and GPU memory.

The code for the neural networks have will be sourced from other open-source projects online. This is done to ensure that implementation of the Forward-Forward algorithm is done correctly.

Forward-Forward Algorithm Implementations The forward-forward algorithm code will be sourced from external code repositories as an in-house implementation of the forward-forward algorithm falls outside the scope of this report. In the preliminary research for this project multiple code repositories have been found to have working implementations of the forward-forward algorithm.

The first implementation is a `Pytorch python` script with the forward-forward algorithm written by Sindy Löwex. While this implementation is sophisticated, and appears to be an accurate representation of the forward-forward algorithm, it may be difficult to implement alongside a traditional BP neural network such that the network architecture is comparable.

The second implementation is also a `Pytorch Python` script authored by Mohammed Pezeshki and Junho Yun. This version of the forward-forward algorithm is a simpler implementation, and will be easier to work with.

The third implementation is one `Pytorch` has implemented already. Further investigation is need to determine if this version is suitable for the needs of this project.

4.2 Task Overview

Below is break down of the overall structure of the steps used to produce the results of this project:

1. Build/Source two similar (same number of parameters, and overall network architecture) neural network models, one using backpropagation, the other using the forward-forward algorithm.

2. Train/test both model on the same dataset(s), recording both GPU and Memory usage at set intervals throughout the training/testing process.

3. Compare the resources usage of across each model to see if there are significant difference in resource usage while maintaining similar test error rates.

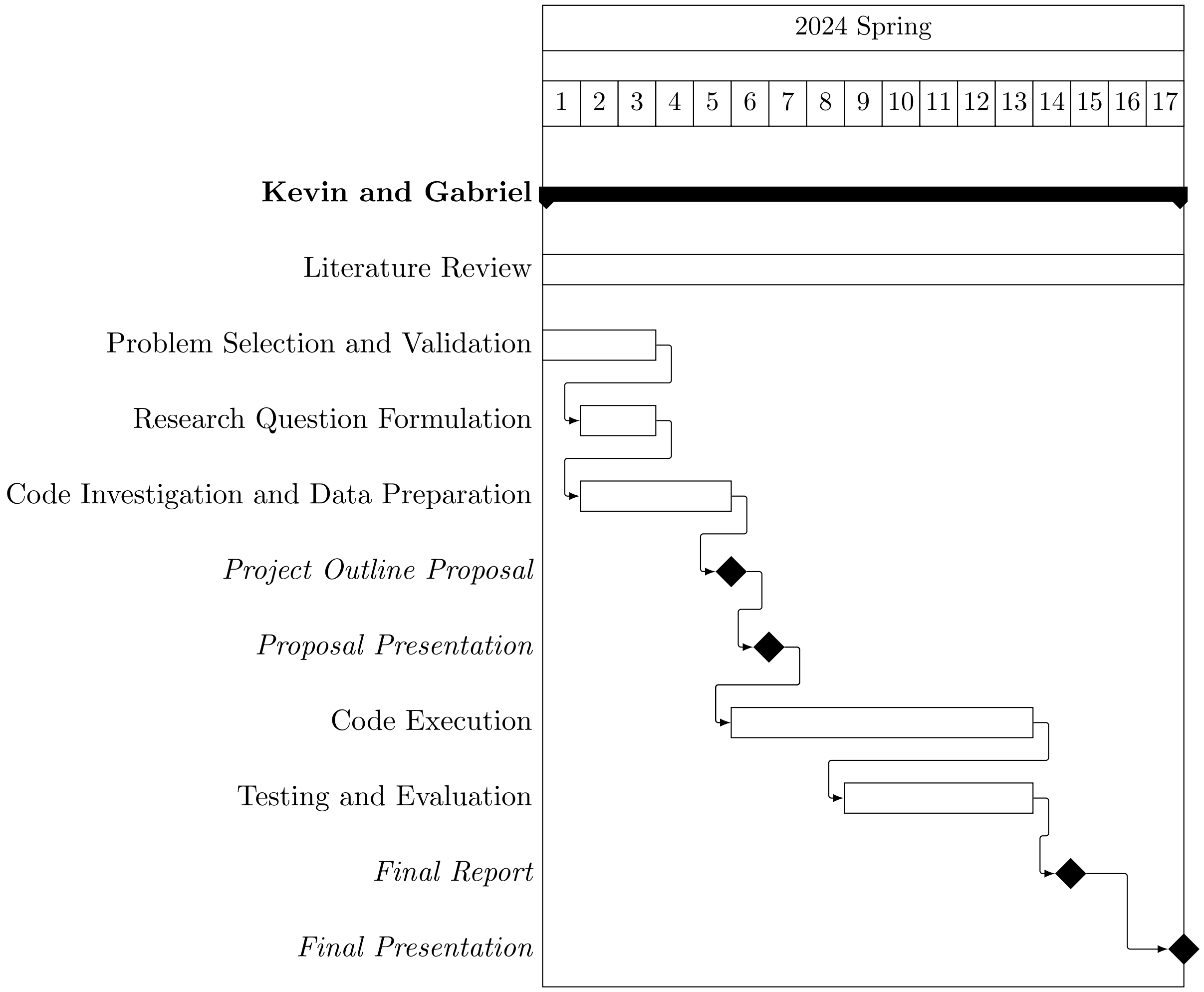
5 Expected outcomes

This section is written by Kevin & Gabriel

There are two key outcomes that we expect our research to cover. First, we expect to gain a better understanding of the Forward-Forward algorithm by successfully reproducing and validating its implementation, which will give a solid foundation for comparing the FF algorithm against traditional methods and expand our knowledge of the literature. Second, by comparing the computational efficiency of the FF and BP algorithms we expect to identify which method is more efficient with computing resources. This outcome will offer insights into potential performance trade-offs and help us determine which scenarios the FF algorithm could be preferable over BP.

Ultimately, this project aims to make a meaningful contribution to the field of machine learning, particularly in the development of novel learning techniques for training neural networks.

6 Program of work



References

[1] Avishan Bewtra. “A Further Investigation of the Forward-Forward Algorithm”. In: ().

[2] Saumya Gandhi et al. *Extending the Forward Forward Algorithm*. July 14, 2023. DOI: `10.48550/arXiv.2307.04205`. arXiv: `2307.04205[cs]`. URL: `http://arxiv.org/abs/2307.04205` (visited on 07/24/2024).

[3] Geoffrey Hinton. *The Forward-Forward Algorithm: Some Preliminary Investigations*. Dec. 26, 2022. DOI: `10.48550/arXiv.2212.13345`. arXiv: `2212.13345[cs]`. URL: `http://arxiv.org/abs/2212.13345` (visited on 07/30/2024).

[4] Geoffrey E. Hinton. “Training Products of Experts by Minimizing Contrastive Divergence”. In: *Neural Computation* 14.8 (Aug. 1, 2002), pp. 1771–1800. ISSN: 0899-7667. DOI: `10.1162/089976602760128018`. URL: `https://doi.org/10.1162/089976602760128018` (visited on 08/11/2024).

[5] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. 2009.

[6] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521.7553 (May 2015). Publisher: Nature Publishing Group, pp. 436–444. ISSN: 1476-4687. DOI: 10.1038/nature14539. URL: https://www.nature.com/articles/nature14539 (visited on 08/23/2024).

[7] Yann LeCun, Corinna Cortes, and CJ Burges. "MNIST handwritten digit database". In: *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* 2 (2010).

[8] Dong-Hyun Lee et al. "Difference Target Propagation". In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by Annalisa Appice et al. Cham: Springer International Publishing, 2015, pp. 498–515. ISBN: 978-3-319-23528-8. DOI: 10.1007/978-3-319-23528-8_31.

[9] Ali Momeni et al. "PhyFF: Physical forward forward algorithm for in-hardware training and inference". In: Machine Learning with New Compute Paradigms. Dec. 22, 2023. URL: https://openreview.net/forum?id=aqsFLVg0tY (visited on 07/30/2024).

[10] Deepak Narayanan et al. *Efficient Large-Scale Language Model Training on GPU Clusters Using Megatron-LM*. Aug. 23, 2021. arXiv: 2104.04473[cs]. URL: http://arxiv.org/abs/2104.04473 (visited on 08/10/2024).

[11] Daniele Paliotta et al. *Graph Neural Networks Go Forward-Forward*. Feb. 10, 2023. DOI: 10.48550/arXiv.2302.05282. arXiv: 2302.05282[cs]. URL: http://arxiv.org/abs/2302.05282 (visited on 07/30/2024).

[12] Abel Reyes-Angulo and Sidike Paheding. *The Forward-Forward Algorithm as a feature extractor for skin lesion classification: A preliminary study*. July 2, 2023. DOI: 10.48550/arXiv.2307.00617. arXiv: 2307.00617[cs]. URL: http://arxiv.org/abs/2307.00617 (visited on 07/30/2024).

[13] Matthias C. Rillig et al. "Risks and Benefits of Large Language Models for the Environment". In: *Environmental Science & Technology* 57.9 (Mar. 7, 2023). Publisher: American Chemical Society, pp. 3464–3466. ISSN: 0013-936X. DOI: 10.1021/acs.est.3c01106. URL: https://doi.org/10.1021/acs.est.3c01106 (visited on 08/23/2024).

[14] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors". In: *Nature* 323.6088 (Oct. 1986). Publisher: Nature Publishing Group, pp. 533–536. ISSN: 1476-4687. DOI: 10.1038/323533a0. URL: https://www.nature.com/articles/323533a0 (visited on 08/06/2024).

[15] Riccardo Scodellaro et al. *Training Convolutional Neural Networks with the Forward-Forward algorithm*. Jan. 7, 2024. DOI: 10.48550/arXiv.2312.14924. arXiv: 2312.14924[cs]. URL: http://arxiv.org/abs/2312.14924 (visited on 07/30/2024).

[16] Shivanshu Shekhar et al. *Towards Optimizing the Costs of LLM Usage*. Jan. 29, 2024. arXiv: 2402.01742[cs]. URL: http://arxiv.org/abs/2402.01742 (visited on 08/10/2024).

[17] Craig S. Smith. *What Large Models Cost You – There Is No Free AI Lunch.* Forbes. Section: AI. URL: https://www.forbes.com/sites/craigsmith/2023/09/08/what-large-models-cost-you--there-is-no-free-ai-lunch/ (visited on 08/10/2024).