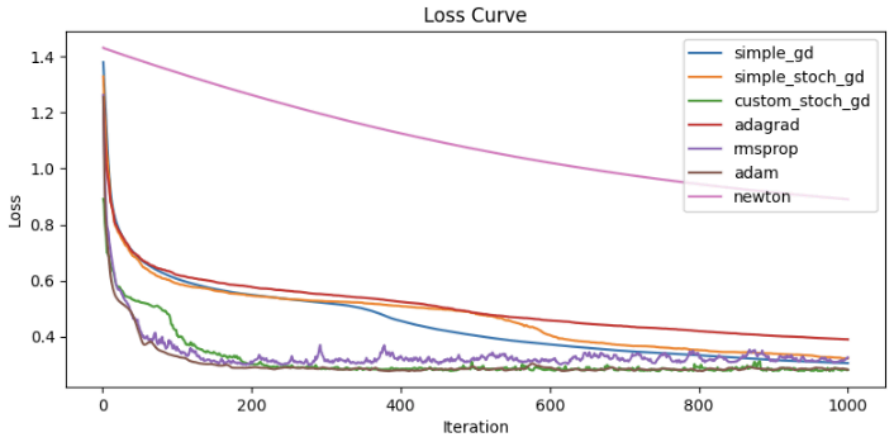
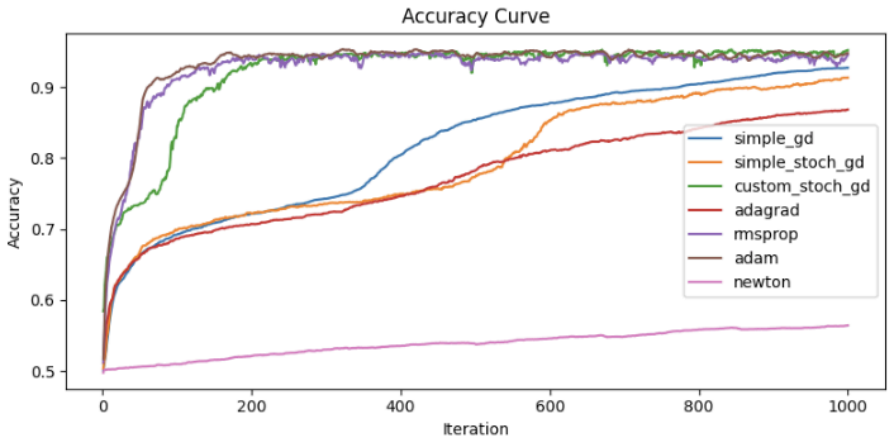
Ahson Saiyed
5-7-2024

Exploring Optimizers

A Logistic Regression model was trained to classify “Good” and “Bad” Bananas from the Banana Dataset. Below is a breakdown of different optimizers performance for fixed max iteration and learning rate. Also included in this report is analysis of varying and fixed floating point impact on execution time, as well as how batched sizes across varying and fixed floating pointing influence numerical stability across different optimizers. Optimizers that were built and tested include Adam, Newton’s Method, Stochastic Gradient Descent, AdaGrad, RMSProp, Custom Stochastic Gradient Descent.



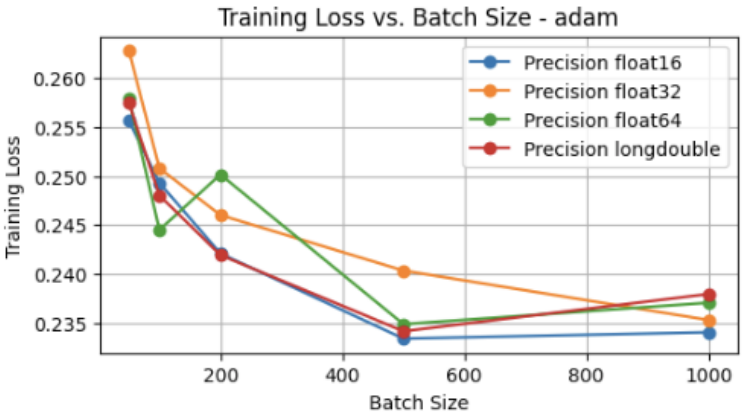


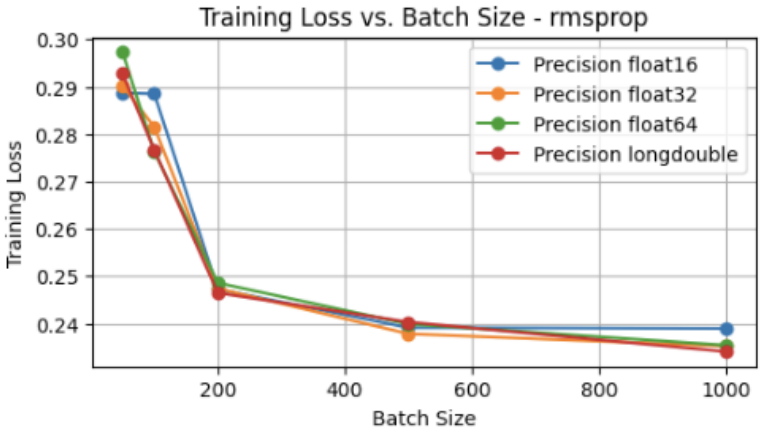
It was found, of the optimizers tested, ADAM had the highest training, and validation accuracy, and the lowest training and validation loss. ADAM also had the fastest execution time across varying and floating point precision. ADAM was most performant when using “float16” precision and given a batch_size of 1000, which are the parameters selected for the model submitted for evaluation. Precision did seem to influence ADAM’s performance across batch sizes, as compared to RMSPROP which was less sensitive to intermediate batch sizes and precision, as shown in the heatmaps below. ADAM and RMSPROP converted to a stable loss in fewer iterations than other optimizers.

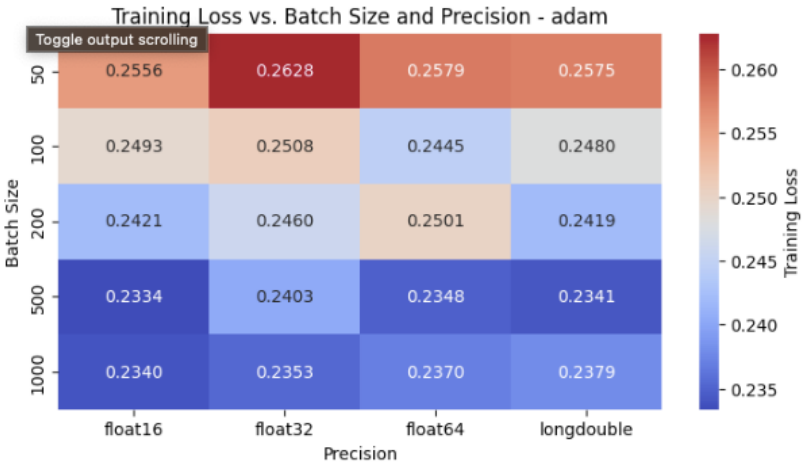
Throughout each implementation, there were steps taken to preserve numerical stability. Examples of this include using numpy functions like np.clip to control magnitude of gradient updates, and adding episolons in np.log calculations to prevent undefined states, and for newton's method, calculating the condition number and setting a condition threshold to enforce stronger regularization, if the condition threshold was exceeded.

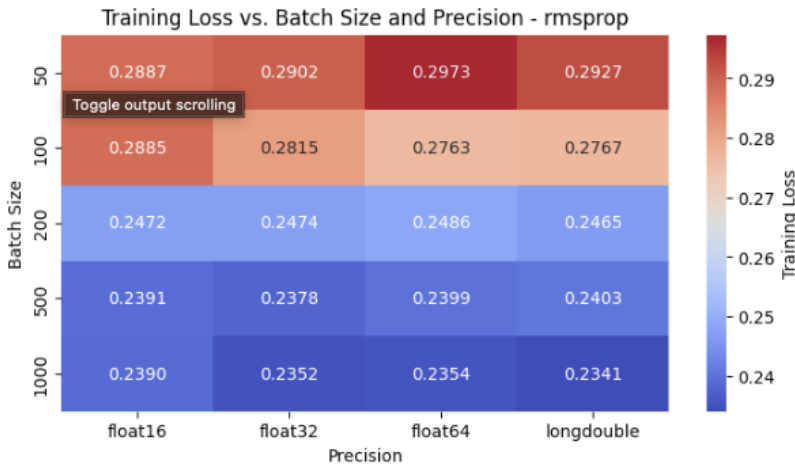
Optimizer	Max Training Accuracy	Final Training Accuracy
simple_gd	0.927551	0.927347
simple_stoch_gd	0.913469	0.913469
custom_stoch_gd	0.953061	0.952245
adagrad	0.868571	0.868571
rmsprop	0.950204	0.945918
adam	0.953469	0.948163
newton	0.564286	0.564286

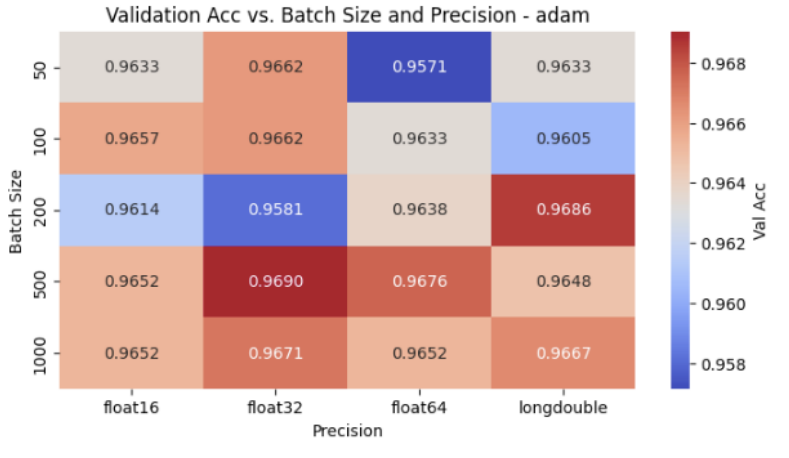
Optimizer	Min Training Loss	Final Training Loss
simple_gd	0.305967	0.305967
simple_stoch_gd	0.322478	0.322511
custom_stoch_gd	0.277563	0.281564
adagrad	0.390600	0.390600
rmsprop	0.298321	0.325978
adam	0.277567	0.282579
newton	0.890258	0.890258

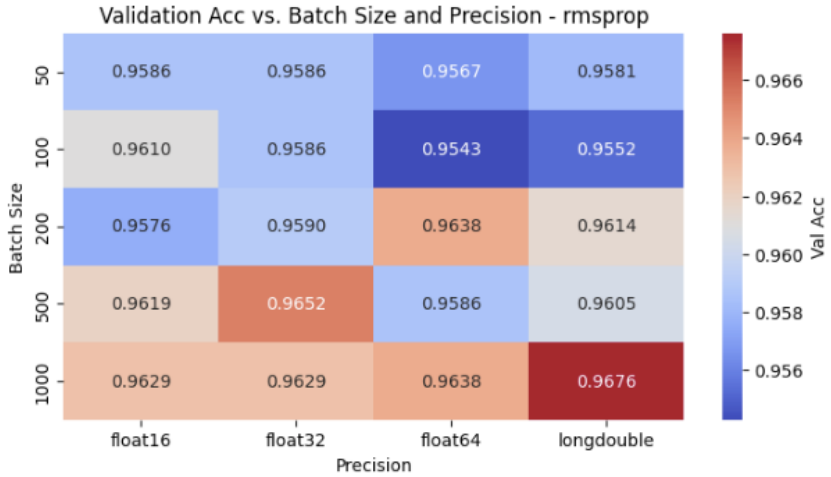

Training Loss vs. Batch Size - adam
Precision float16
Precision float32
Precision float64
Precision longdouble
Training Loss
Batch Size

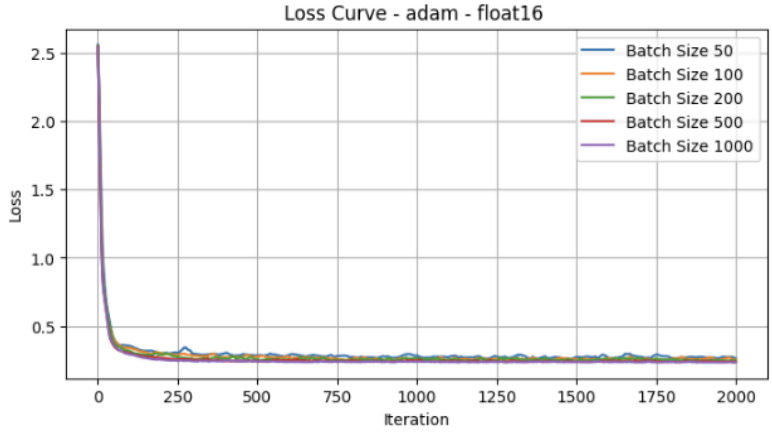

Training Loss vs. Batch Size - rmsprop
Precision float16
Precision float32
Precision float64
Precision longdouble
Training Loss
Batch Size

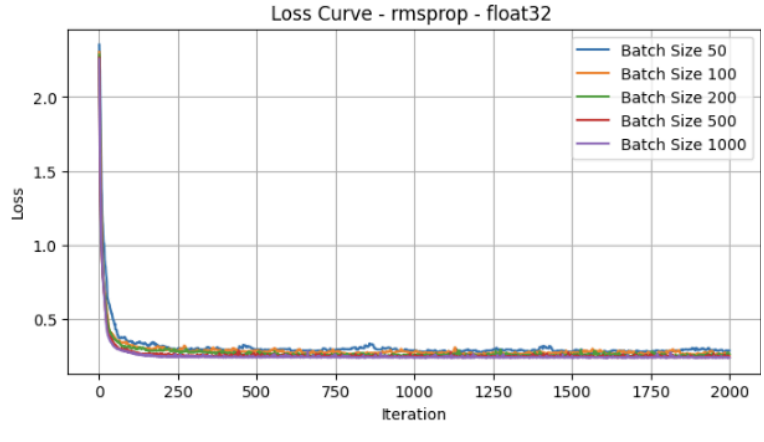

Training Loss vs. Batch Size and Precision - adam
Toggle output scrolling
Batch Size
Precision
Training Loss


Training Loss vs. Batch Size and Precision - rmsprop
Toggle output scrolling
Batch Size
Precision
Training Loss


Validation Acc vs. Batch Size and Precision - adam
Batch Size
Precision
Val Acc


Validation Acc vs. Batch Size and Precision - rmsprop
Batch Size
Precision
Val Acc


Loss Curve - adam - float16
Batch Size 50
Batch Size 100
Batch Size 200
Batch Size 500
Batch Size 1000
Loss
Iteration


Loss Curve - rmsprop - float32
Batch Size 50
Batch Size 100
Batch Size 200
Batch Size 500
Batch Size 1000
Loss
Iteration

References

Sean Harrington, “Solving Logistic Regression with Newton’s Method.” *The Laziest Programmer*, 6 July 2017, thelaziestprogrammer.com/sharrington/math-of-machine-learning/solving-logreg-newtons-method.

Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen, and Kailash Gopalakrishnan. 2018. Training deep neural networks with 8-bit floating point numbers. In Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS'18). Curran Associates Inc., Red Hook, NY, USA, 7686–7695.

Stephen Baek. Gradient-based Optimization Slides

Oppermann, Artem. “Optimization in Deep Learning: Adagrad, RMSPROP, Adam.” *KI Tutorials*, 21 Dec. 2021, artemoppermann.com/optimization-in-deep-learning-adagrad-rmsprop-adam/.

Yu, Hao, and Bogdan Wilamowski. “Levenberg–Marquardt training.” *Electrical Engineering Handbook*, 28 Feb. 2011, pp. 1–16, https://doi.org/10.1201/b10604-15.