Mideval

ADAM: A Method for Stochastic Optimization

Introduction

Team name: Entropy Death

- **Vishal Reddy Mandadi** (2019101119)
- **Pranshul Chawla** (2019101057)
- **Kadari Ruthwik Reddy** (2019101121)
- **Naman Verma** (2019101066)
- TA advisor: **Abhinaba Bala**

GitHub Link: https://github.com/vishal-2000/SMAI-project (Restricted Access)

Mid Eval

Submission contents:

Directory structure

```
.
├── ADAM.py
├── CNNCIFAR10.ipynb
├── graphs
│   ├── MNIST_mideval_plot.png
│   ├── MNIST_Vishal_linear_activation.png
│   ├── MNIST_Vishal_linear_act_resize=14.png
│   └── Pranshul_linear_activation_resize=original=28.png
├── Initial.ipynb
├── Report.pdf
├── MNISTCNN.ipynb
└── Modified_MLP.ipynb
```

Datasets and Models

Implemented a multi-layer neural network with ReLU activation functions (section 6.2 in the research paper) and used multiple optimizers named RMSprop, SGD, Adadelta, and Adam to train the model and compared the loss functions after each epoch to see which one is better among these four.

Adam was implemented from scratch whereas for other optimizers we called the inbuilt optimizers to compare.

The Results were as Follows (T.L means training loss) for a few epochs:

Results (linear output, 28*28 input, 3-layered network)

Aa Title	# Epoch	# T.L RMSprop	# T.L SGD	# T.L Adam	# T.L Adadelta
Untitled	1	9.529574	1.809672	0.305558	1.960298
Untitled	2	0.235609	0.667916	0.175069	0.874818
Untitled	3	0.181927	0.440802	0.146869	0.534714
Untitled	4	0.151211	0.372952	0.11998	0.435051

Adam did perform better compared to other optimizers as it was argued on the paper here is the graph for more epochs to visualize the difference (observe that ADAM performs better with both linear and Sigmoid output activations).

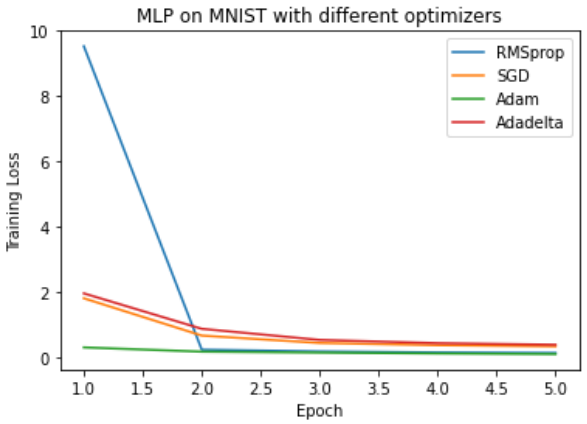

Image with original size (28*28). 3-layered network (1000, 1000, 10) with linear output (hidden layers use ReLu)

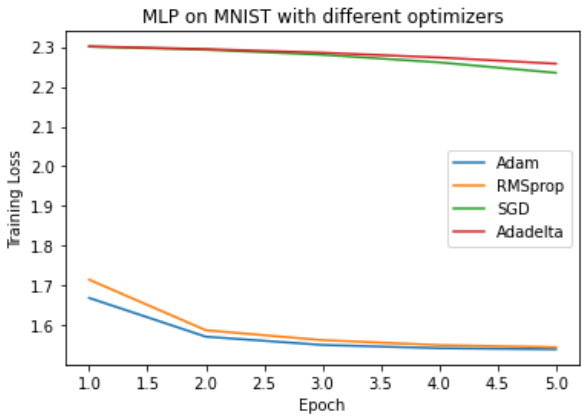

Image resized to 10*10. 3-layered neural network (32, 16, 10) with sigmoid output (hidden layers use ReLU)

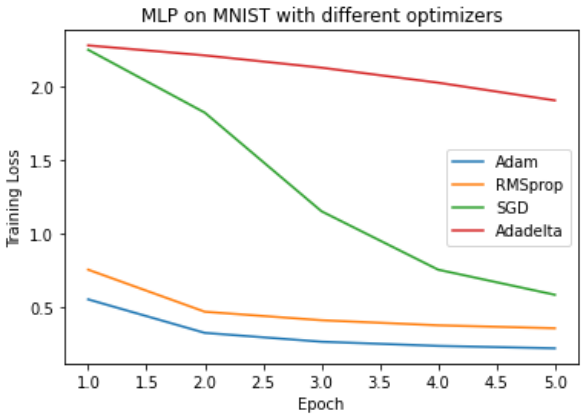

Image resized to 10*10, a 3-layered neural network (32, 16, 10) with linear output (hidden layers use ReLU)

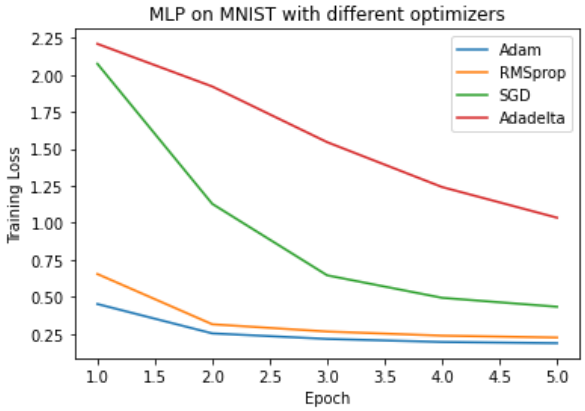

Image resized to 14*14, a 3-layered neural network (64, 32, 10) with linear output (no activation function used in the end) (hidden layers use ReLU)

You can find the code for the above graphs and other related resources here - https://github.com/vishal-2000/SMAI-project

> 💡 Analysis of ADAM's performance compared to others on various different cases will be provided for the final eval as we think it will be more complete by then

Apart from the multi-layer neural network we also built a CNN to feed MNIST data to see if the results follow as shown in the paper however this wasn't completely finished so we don't have data related to it at present we will deliver them at the end of the presentation.

Accuracy of MLP with ADAM on Test data

1. Image with original size (28*28). 3-layered network (1000, 1000, 10) with linear output (hidden layers use ReLu) - 93%
2. Image resized to 10*10. 3-layered neural network (32, 16, 10) with sigmoid output (hidden layers use ReLU) - 78%
3. Image resized to 10*10, a 3-layered neural network (32, 16, 10) with linear output (hidden layers use ReLU) - 88%

4. Image resized to 14*14, a 3-layered neural network (64, 32, 10) with linear output (no activation function used in the end) (hidden layers use ReLU) - 91%

Work Distribution

- Data Loading and preprocessing for MNIST to feed into MLP: Pranshul, Naman, and Vishal
- Building MLP and graphs: Pranshul and Naman
- Implementing Adam: Vishal
- Data Loading and pre-processing for CIFAR 10- Ruthwik
- Report: Ruthwik
- Building CNN to feed MNIST data- Ruthwik (Not finished)