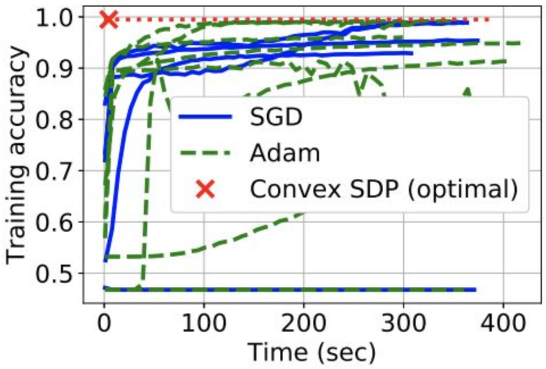
A Convex Approach to Two-Layer Convolutional Neural Network

Poojit Hegde, Shreya Shubhangi, & Ananya Karthik

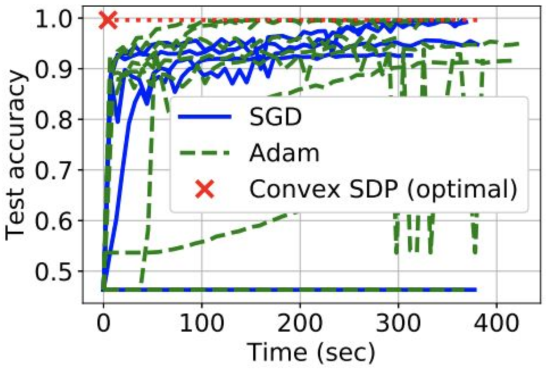
Project Mentor: Burak Bartan

Why convex optimization?

- Optimizer parameters have no influence on model performance
- Hyperparameter tuning becomes less important
- Locally optimal solutions are globally optimal



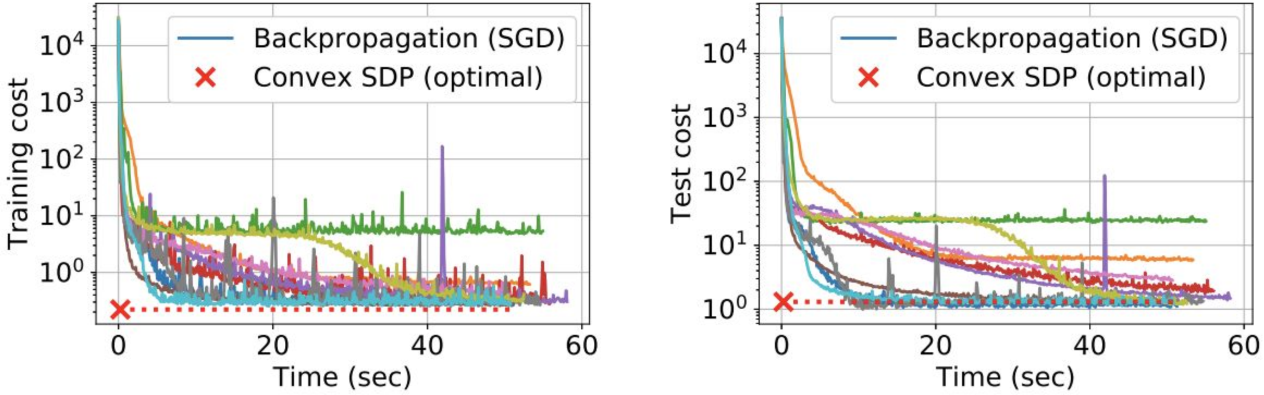
(a) CNN, MNIST, training accuracy



(b) CNN, MNIST, test accuracy

Convex Binary Convolutional Neural Network

- Binary classification accuracy performance on the CNN architecture with global average pooling on MNIST, Fashion MNIST, and Cifar-10 datasets
- Accuracy of stochastic gradient descent is slightly worse than convex Semidefinite Programming (SDP)
- Time that it takes for SGD to converge is consistently larger than the run time for convex SDP



Our Experiments

Datasets: CIFAR-2
CIFAR-10

Test 1 → Convex Binary Classification CNN

Baseline 1 → Non-Convex Binary Classification CNN

Baseline 2 → Non-Convex Multi-Class Classification CNN

Test 2 → Convex Multi-Class Classification CNN

Convex SDP (Scalar Output)

$$\min_{\{Z_k=Z_k^T, Z_k'={Z_k'}^T\}_{k=1}^{K/P}} \ell(\hat{y}, y) + \beta \sum_{k=1}^{K/P} (Z_{k,4} + Z'_{k,4})$$

$$\text{s.t.} \quad \hat{y}_i = a \frac{1}{P} \sum_{k=1}^{K/P} \sum_{l=1}^{P} x_{i,(k-1)P+l}^T (Z_{k,1} - Z'_{k,1}) x_{i,(k-1)P+l} + b \frac{1}{P} \sum_{k=1}^{K/P} \sum_{l=1}^{P} x_{i,(k-1)P+l}^T (Z_{k,2} - Z'_{k,2}) +$$

$$+ c \sum_{k=1}^{K/P} (Z_{k,4} - Z'_{k,4}), \quad i \in [n]$$

$$\operatorname{tr}(Z_{k,1}) = Z_{k,4}, \ \operatorname{tr}(Z'_{k,1}) = Z'_{k,4}, \quad k = 1, \ldots, K/P$$

$$Z_k \succeq 0, \ Z'_k \succeq 0, \quad k = 1, \ldots, K/P \,. \tag{95}$$

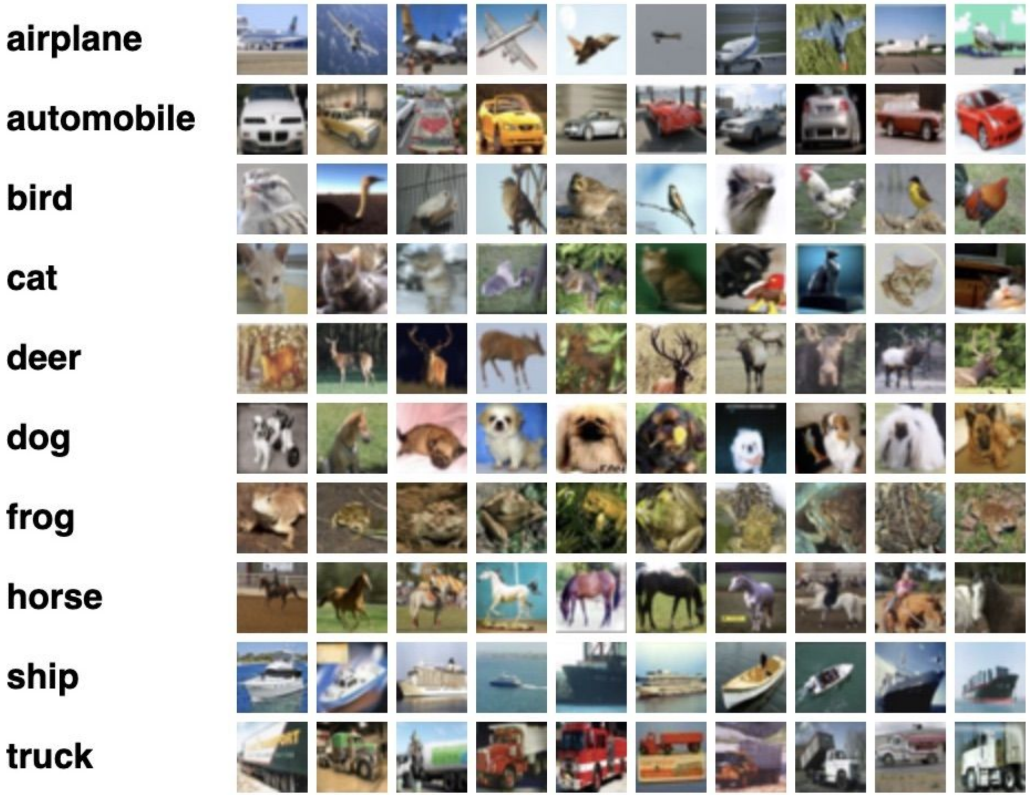
Convex SDP (Vector Output)

$$
\begin{aligned}
\min_{Z_k^{(t)}, Z_k'^{(t)}} \ & \ell(\hat{Y}, Y) + \beta \sum_{t=1}^{C} \sum_{k=1}^{K/P} (Z_{k,4}^{(t)} + Z_{k,4}'^{(t)}) \\
\text{s.t.} \quad & \hat{Y}_{it} = a \frac{1}{P} \sum_{k=1}^{K/P} \sum_{l=1}^{P} x_{i,(k-1)P+l}^T (Z_{k,1}^{(t)} - Z_{k,1}'^{(t)}) x_{i,(k-1)P+l} + b \frac{1}{P} \sum_{k=1}^{K/P} \sum_{l=1}^{P} x_{i,(k-1)P+l}^T (Z_{k,2}^{(t)} - Z_{k,2}'^{(t)}) + \\
& + c \sum_{k=1}^{K/P} (Z_{k,4}^{(t)} - Z_{k,4}'^{(t)}), \quad i \in [n], t \in [C] \\
& \mathrm{tr}(Z_{k,1}^{(t)}) = Z_{k,4}^{(t)}, \ \mathrm{tr}(Z_{k,1}'^{(t)}) = Z_{k,4}'^{(t)}, \quad k \in [K/P], t \in [C] \\
& Z_k^{(t)} \succeq 0, \ Z_k'^{(t)} \succeq 0, \quad k = [K/P], t \in [C], \qquad (1)
\end{aligned}
$$

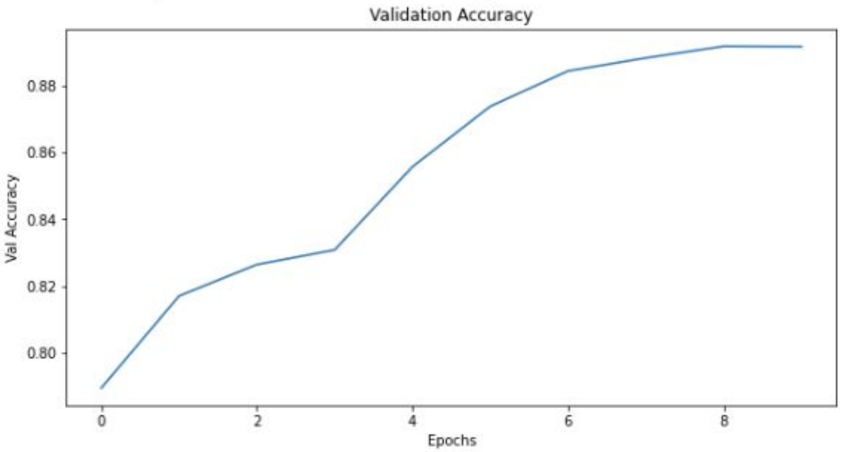
Dataset

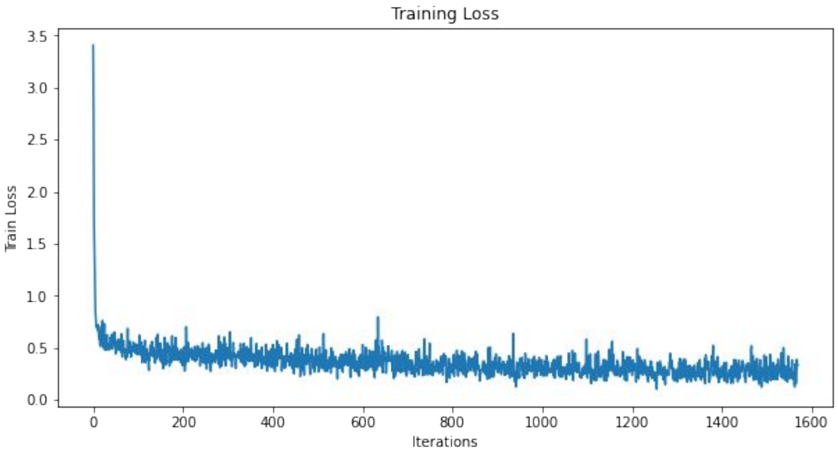
Subset of CIFAR-10

- Binary Classification (First two classes)
- Multi-Class Classification (All Classes)



Comparison: Non-convex CIFAR-2 vs Convex Binary CIFAR-2

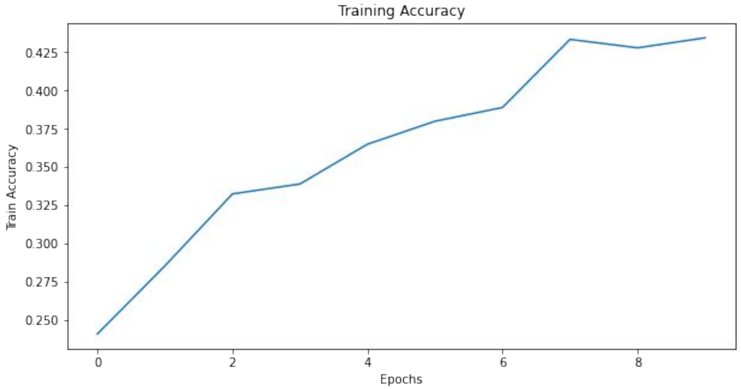


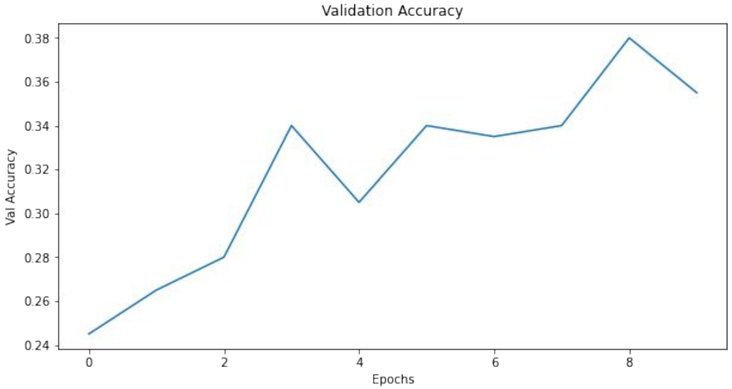


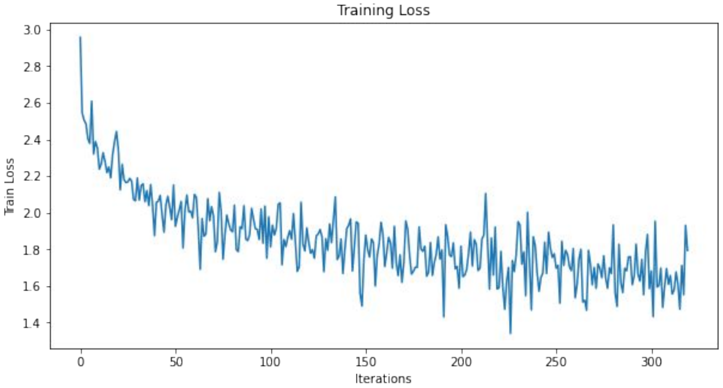
Non-convex CIFAR-2 Binary CNN Test Accuracy: 88.90%

Convex CIFAR-2 Binary CNN Test Accuracy: 84.05%

Baseline 2: Non-convex Multi-Class CNN



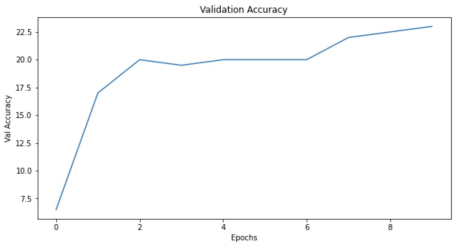


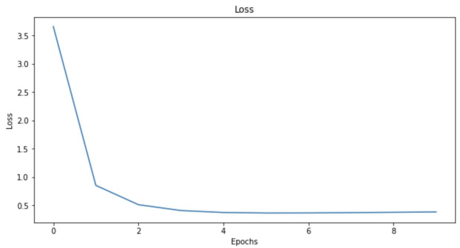


Non-Convex Multi-Class CNN Test Accuracy: 36.50%

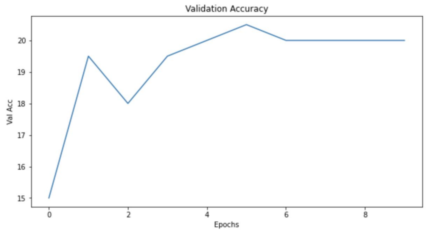
Convex Multi-Class CNN Test Accuracy: 28.55%

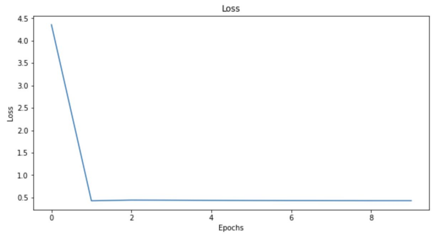
Convex Multi-Class CNN with Different Learning Rates



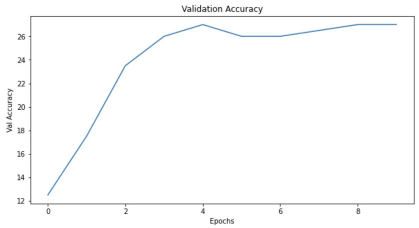


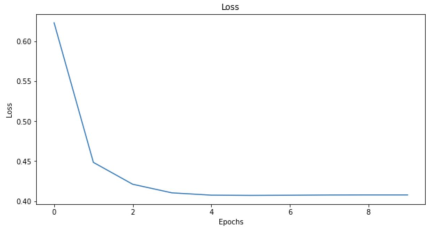
Learning Rate: 1e-4
Validation Accuracy: 23%





Learning Rate: 1e-2
Validation Accuracy: 20%





Learning Rate: 1e-3
Validation Accuracy: 28.5%

What do we see?

- Convex optimization CNNs have slightly lower but comparable accuracies
  - Pytorch version has a forward implementation built from scratch - not equipped for this
  - Non-controls which are positively skewing our baseline accuracies in comparison to our experiments

- For custom code with convex SDPs, the necessary optimizations were likely inadequate, and therefore the training was slow.

Validity of Using Convex Multi-Class CNNs

- More work needed to fully formulate convex optimization
- Convex methods still show great promise
- Next steps:
    - Layerwise learning
    - Deeper architectures
    - Expanding Dataset