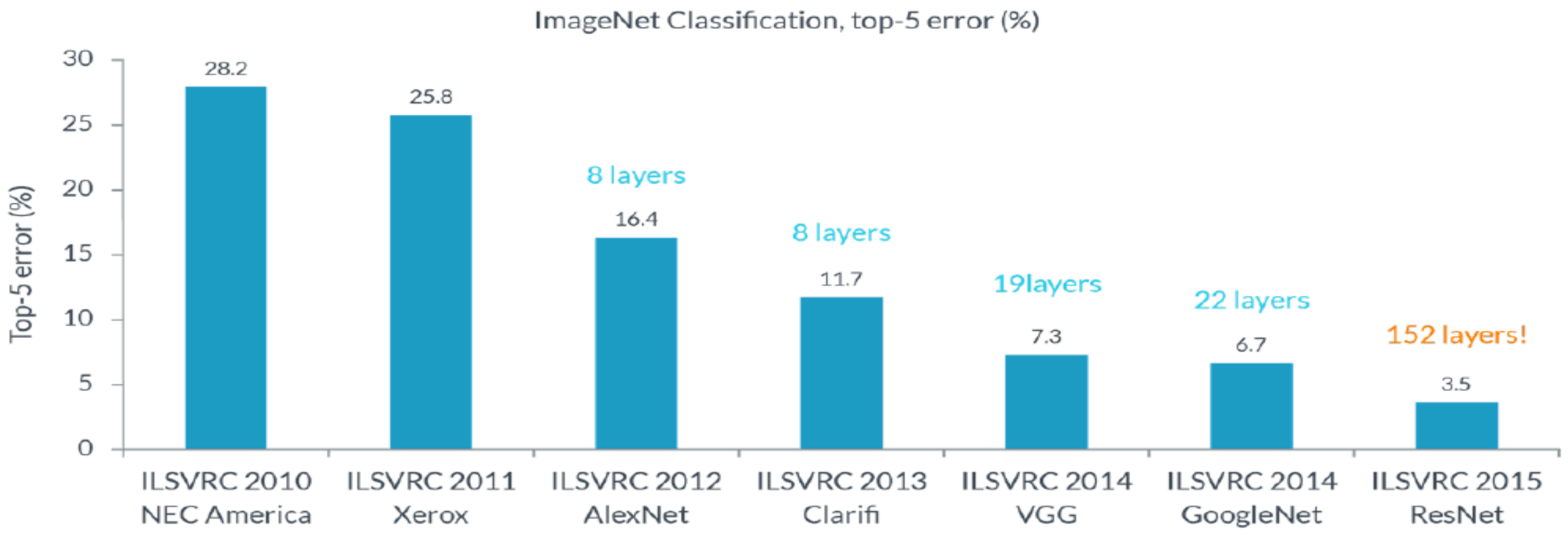
Implicit Convex Regularizers of CNN Architectures: Convex Optimization of Two- and Three-Layer Networks in Polynomial Time

Tolga Ergen

joint work with Mert Pilanci

ICLR2021

Deep Learning Revolution



Deep learning models:

- often provide the best performance due to their large capacity
  - **challenging to train**
- are complex black-box systems based on non-convex optimization
  - **hard to interpret what the model is actually learning**

Prior Work on Convex Neural Networks

Prior Work

Model FC : $\sum_{j=1}^{m}(\boldsymbol{X}\boldsymbol{u}_j)_+ w_j$

Complexity: $O\left(d^6\left(\frac{n}{d}\right)^{3d}\right)$

More than 2 layers: 

Our work

Model CNN : $\sum_{j=1}^{m}\sum_{k=1}^{K}(\boldsymbol{X}_k\boldsymbol{u}_j)_+ w_j$

Complexity: $O\left(h^6\left(\frac{nK}{h}\right)^{3h}\right)$

More than 2 layers: 

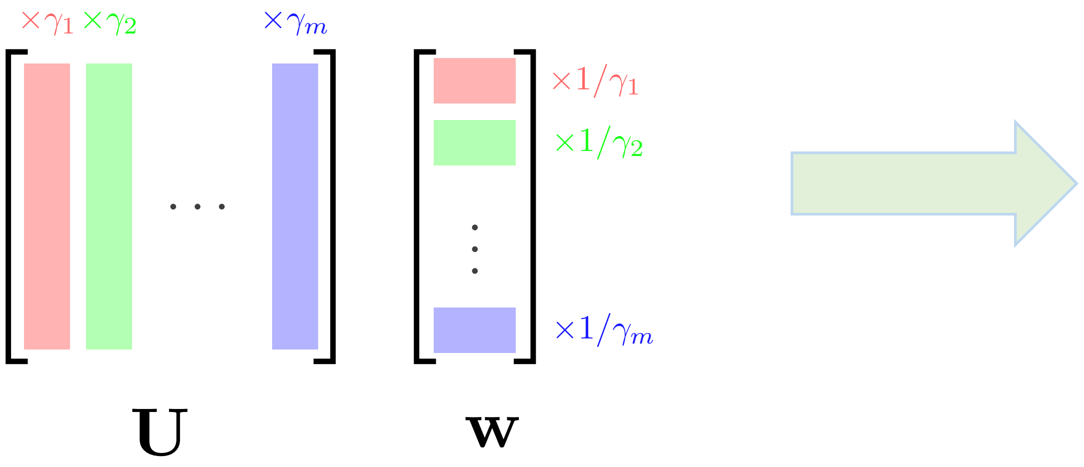
n: #of samples
d: #of features
h: filter size
K: #of patches

M. Pilanci and T. Ergen, "**Neural Networks are Convex Regularizers: Exact Polynomial-time Convex Optimization Formulations for Two-layer Networks**", **ICML2020**

Standard 2-layer CNN Training Problem

$$p^* := \min_{\{\boldsymbol{u}_j, w_j\}} \frac{1}{2} \left\| \sum_{j=1}^{m} \sum_{k=1}^{K} (\boldsymbol{X}_k \boldsymbol{u}_j)_+ w_j - \boldsymbol{y} \right\|_2^2 + \frac{\beta}{2} \sum_{j=1}^{m} \left(\|\boldsymbol{u}_j\|_2^2 + w_j^2 \right)$$

$\boldsymbol{u}_j \in \mathbb{R}^d$: Filter weights
$w_j \in \mathbb{R}$: Output layer weights
$\boldsymbol{X}_k \in \mathbb{R}^{n \times d}$: Patch matrix
$\beta > 0$: Regularization parameter
$(\cdot)_+$: ReLU activation



$$p^* = \min_{\substack{\{\boldsymbol{u}_j, w_j\} \\ \|\boldsymbol{u}_j\|_2 \leq 1}} \frac{1}{2} \left\| \sum_{j=1}^{m} \sum_{k=1}^{K} (\boldsymbol{X}_k \boldsymbol{u}_j)_+ w_j - \boldsymbol{y} \right\|_2^2 + \beta \|\boldsymbol{w}\|_1$$

Convex Duality

$$p^* \geq d^* := \max_{\boldsymbol{v}} -\frac{1}{2}\|\boldsymbol{v}-\boldsymbol{y}\|_2^2 + \frac{1}{2}\|\boldsymbol{y}\|_2^2 \text{ s.t. } \max_{\|\boldsymbol{u}\|_2 \leq 1} \left|\sum_{k=1}^{K} \boldsymbol{v}^T (\boldsymbol{X}_k \boldsymbol{u})_+\right| \leq \beta$$

Let m be a number such that $m \geq m^*$ for some $m^* \in \mathbb{N}, m^* \leq n+1$,
then strong duality holds, i.e., $p^* = d^*$, and the equivalent convex program is

$$p^* = \min_{\boldsymbol{c}_j, \boldsymbol{c}'_j \in \mathcal{H}_j} \frac{1}{2}\left\|\sum_{j=1}^{P_{conv}} \sum_{k=1}^{K} \boldsymbol{D}_{j,k}\boldsymbol{X}_k\left(\boldsymbol{c}'_j - \boldsymbol{c}_j\right) - \boldsymbol{y}\right\|_2^2 + \beta \sum_{j=1}^{P_{conv}} \left(\|\boldsymbol{c}'_j\|_2 + \|\boldsymbol{c}_j\|_2\right)$$

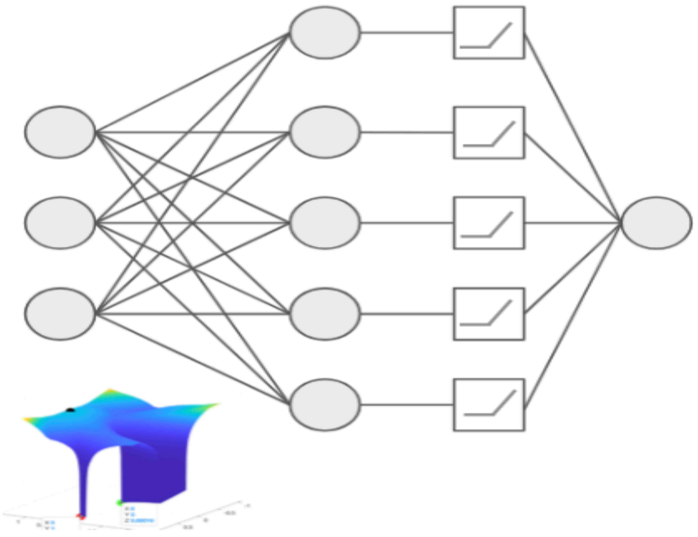
Filter sparsity via group ℓ_1 regularization

Our Convex Model

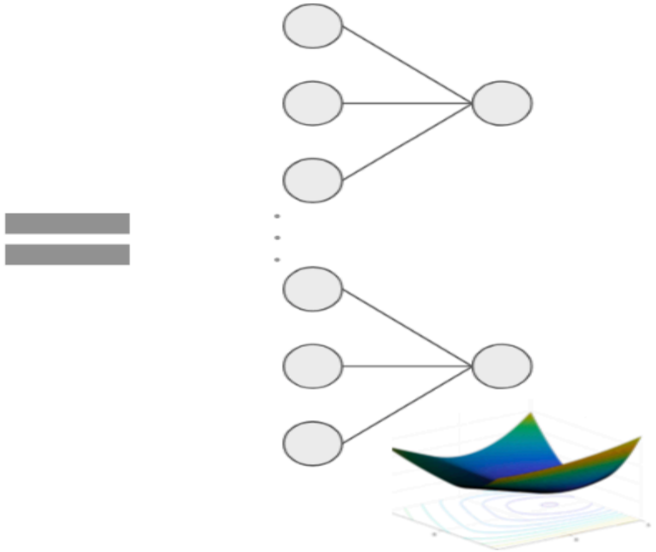
$$p^* = \min_{\{\boldsymbol{u}_j, w_j\}} \frac{1}{2} \left\| \sum_{j=1}^{m} \sum_{k=1}^{K} (\boldsymbol{X}_k \boldsymbol{u}_j)_+ w_j - \boldsymbol{y} \right\|_2^2 + \frac{\beta}{2} \sum_{j=1}^{m} (\|\boldsymbol{u}_j\|_2^2 + w_j^2)$$

$$p^* = \min_{\boldsymbol{c}_j, \boldsymbol{c}'_j \in \mathcal{H}_j} \frac{1}{2} \left\| \sum_{j=1}^{P_{conv}} \sum_{k=1}^{K} \boldsymbol{D}_{j,k} \boldsymbol{X}_k (\boldsymbol{c}'_j - \boldsymbol{c}_j) - \boldsymbol{y} \right\|_2^2 + \beta \sum_{j=1}^{P_{conv}} (\|\boldsymbol{c}'_j\|_2 + \|\boldsymbol{c}_j\|_2)$$

$$(\boldsymbol{u}_j, w_j) = \begin{cases} \left(\frac{\boldsymbol{c}'_j}{\sqrt{\|\boldsymbol{c}'_j\|_2}}, \sqrt{\|\boldsymbol{c}'_j\|_2} \right), & \|\boldsymbol{c}'_j\|_2 > 0 \\ \left(\frac{\boldsymbol{c}_j}{\sqrt{\|\boldsymbol{c}_j\|_2}}, -\sqrt{\|\boldsymbol{c}_j\|_2} \right), & \|\boldsymbol{c}_j\|_2 > 0 \end{cases}$$



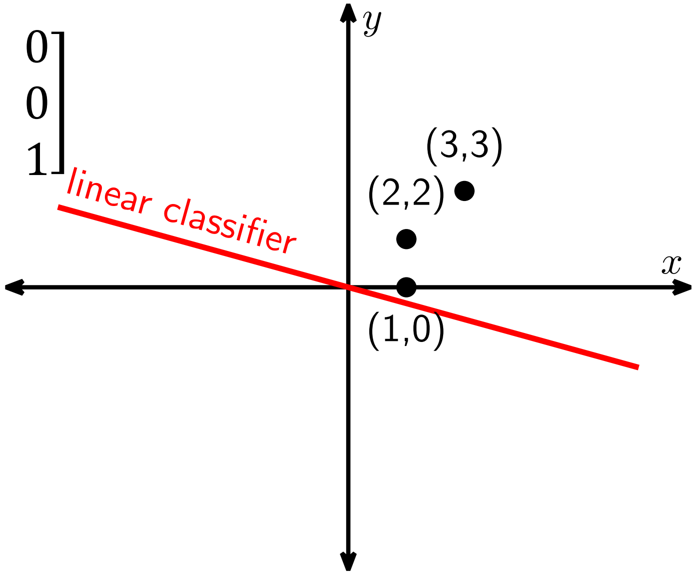
Non-convex Neural Network



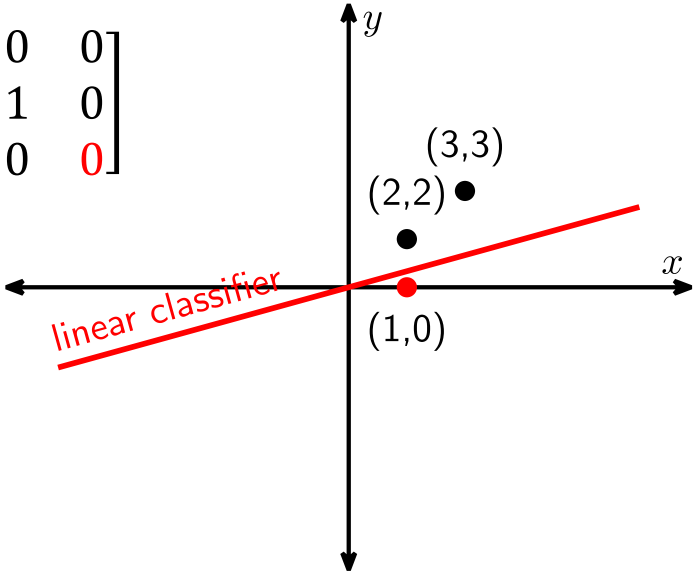
Sparse Mixture of Convex Models

Hyperplane Arrangements

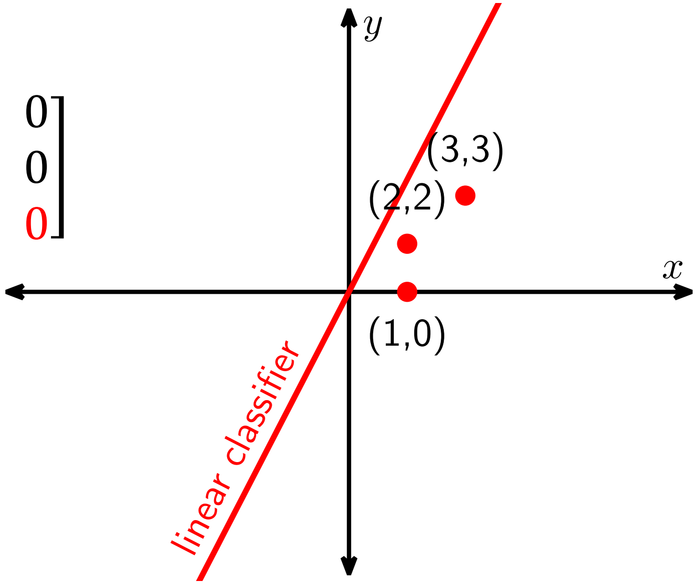
$$\mathbf{D}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



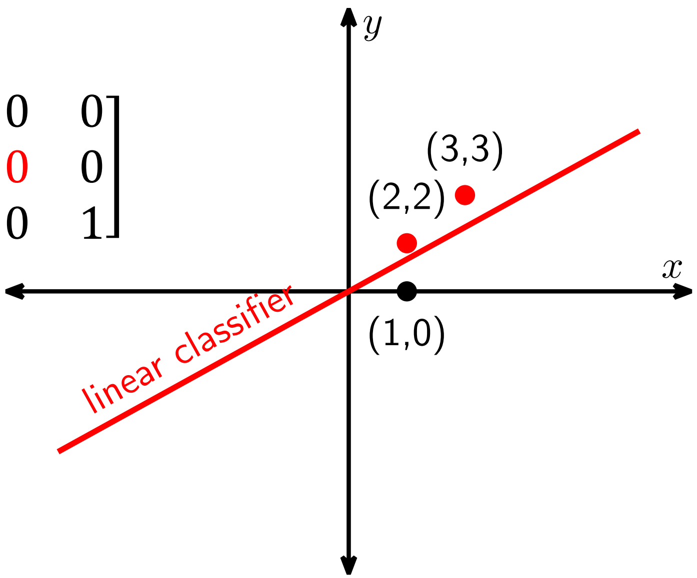
$$\mathbf{D}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$



$$\mathbf{D}_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$



$$\mathbf{D}_4 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



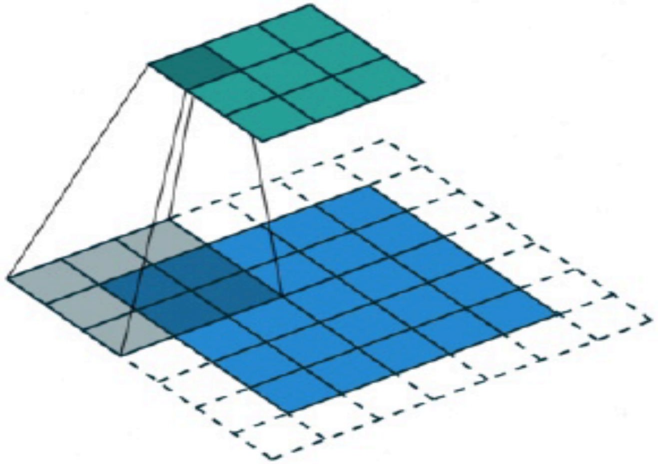
Convolutional Hyperplane Arrangements

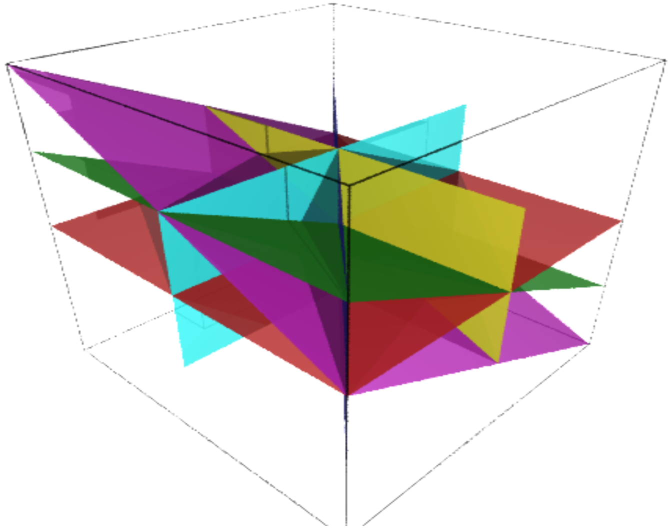
Given a data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ partitioned into the patches as $\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_K \in \mathbb{R}^{n \times h}$ we define **convolutional hyperplane arrangements** as

$\{\mathbb{I}(\boldsymbol{X}_k \boldsymbol{u}) : \boldsymbol{u} \in \mathbb{R}^h\}_{k=1}^{K}$, where $\mathbb{I}(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$

$$P_{conv} \leq O\left(\left(\frac{nK}{h}\right)^h\right),$$

h: filter size
K: #of patches
n: #of samples





Convex optimization complexity: $O\left(h^6 \left(\frac{nK}{h}\right)^{3h}\right)$ **polynomial in all the problem parameters n, m, and d**

3-layer CNNs

$$p^* := \min_{\substack{\{u_j, w_j\} \\ ||u_j||_2 \le 1}} \frac{1}{2} \left\| \sum_{j=1}^{m} \left(\sum_{k=1}^{K} (\boldsymbol{X}_k \boldsymbol{u}_j)_+ w_{1jk} \right)_+ w_{2j} - \boldsymbol{y} \right\|_2^2 + \frac{\beta}{2} \sum_{j=1}^{m} \left(\|\boldsymbol{w}_{1j}\|_2^2 + w_{2j}^2 \right)$$

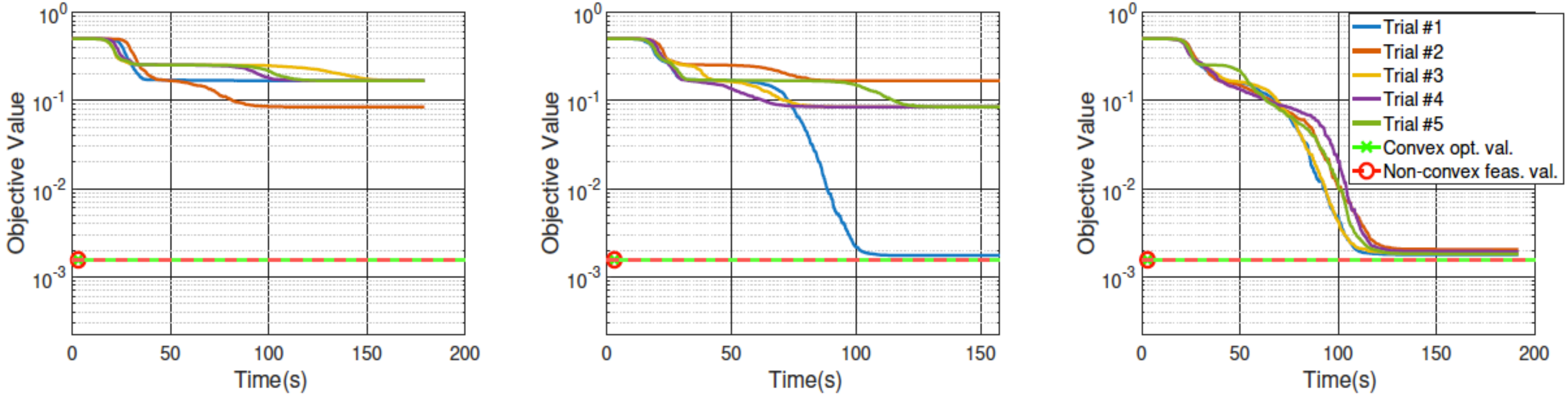
Let m be a number such that $m \geq m^*$ *for some* $m^* \in \mathbb{N}, m^* \leq n+1$,
then strong duality holds, and the equivalent convex program is

$$p^* = \min_{\boldsymbol{c}_{jlk}, \boldsymbol{c}'_{jlk} \in \mathcal{H}_{jlk}} \frac{1}{2} \left\| \sum_{l=1}^{P_2} \sum_{j=1}^{P_1} \sum_{k=1}^{K} \boldsymbol{D}_{l,k}^{(2)} \boldsymbol{D}_{j,k}^{(1)} \boldsymbol{X}_k (\boldsymbol{c}'_{jlk} - \boldsymbol{c}_{jlk}) - \boldsymbol{y} \right\|_2^2 + \beta \sum_{l=1}^{P_2} \sum_{j=1}^{P_1} \left(\|\boldsymbol{C}'_{jl}\|_F + \|\boldsymbol{C}_{jl}\|_F \right)$$

where $\boldsymbol{C}_{jl} = [\boldsymbol{c}_{jl1} \ \dots \ \boldsymbol{c}_{jlK}]$.

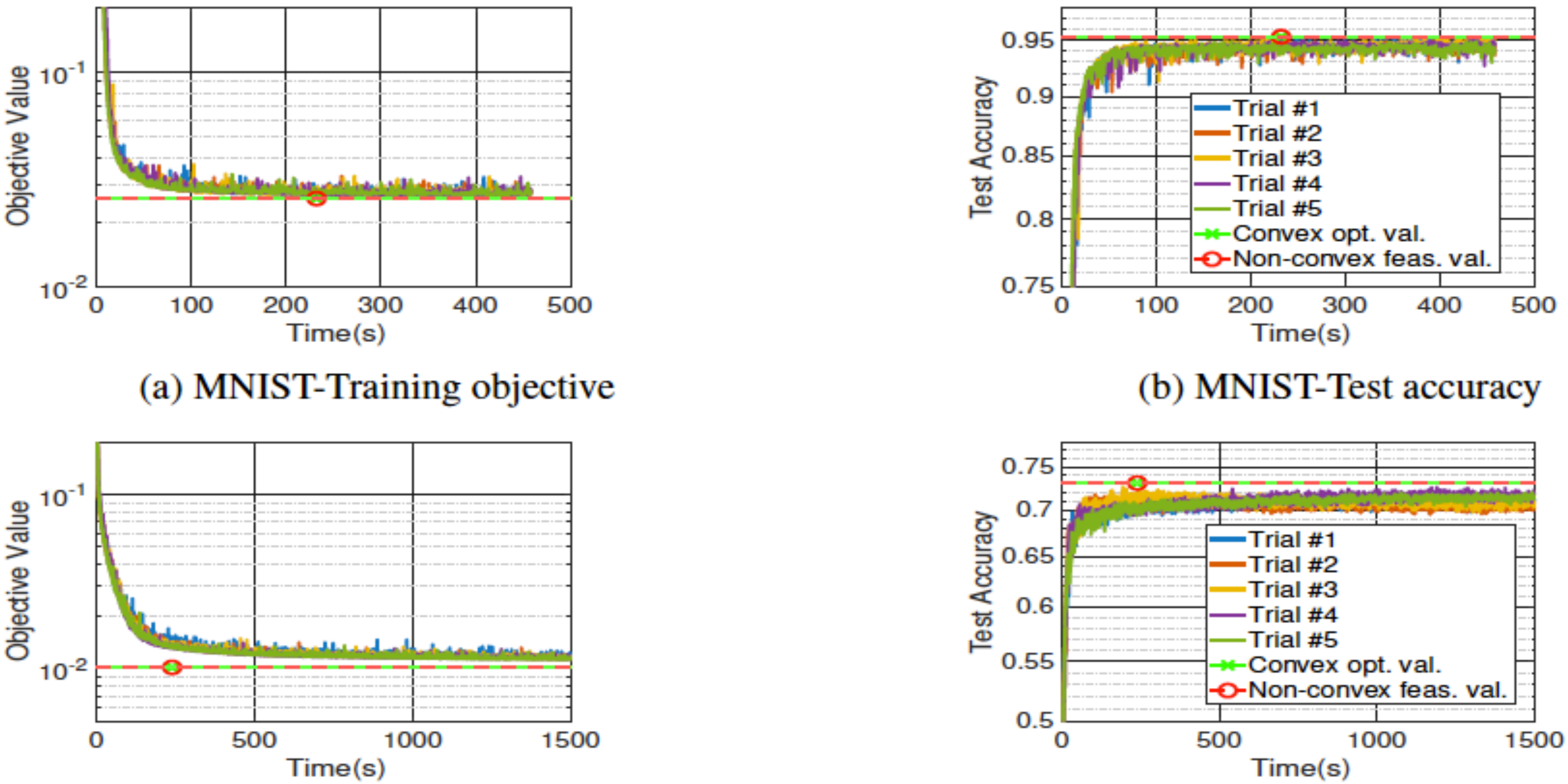
Matrix group sparsity

Numerical Results



(a) $m = 3$ (b) $m = 8$ (c) $m = 15$

Figure: Two-layer CNN trained with SGD (5 initialization trials) on a synthetic dataset ($n = 6$, $d = 15$, $h = 10$, stride $= 5$), where the green and red lines represents the objective value for the convex program and its non-convex equivalent. Here, we use markers to denote the total computation time.



(a) MNIST-Training objective

(b) MNIST-Test accuracy

(c) CIFAR10-Training objective

(d) CIFAR10-Test accuracy

Figure : Evaluation of the three-layer circular CNN trained with SGD (5 initialization trials) on a subset of MNIST ($n = 99$, $d = 50$, $m = 20$, $h = 3$, stride $= 1$) and CIFAR10 ($n = 99$, $d = 50$, $m = 40$, $h = 3$, stride $= 1$).

Conclusion and Open Problems

- We can train ReLU CNNs in polynomial time using convex solvers
  - No need for complex hyperparameter optimization: learning rate & initialization
  - No need for heuristics: batch norm & dropout
  - Interpretable training results

- ReLU CNNs are convex in a higher dimensional space

- Sparsity is promoted via group sparse regularization

- Possible extensions: autoencoders, RNNs, GANs, ResNets, deeper architectures