Stanford University

Fast Convex Optimization for Two-Layer ReLU Networks:

Equivalent Model Classes and Cone Decompositions

Aaron Mishkin
amishkin@cs.stanford.edu

Motivation: Neural Networks

Neural networks are building blocks for modern learning systems.

- **Vision**: object recognition, localization, and bounding.
- **Language**: audio transcription and machine translation.
- **Control**: robotics, autonomous cars, game-playing, . . .

A bowl of soup that is a portal to another dimension as digital art.

Generated by DALL·E 2

But optimizing neural networks is hard!

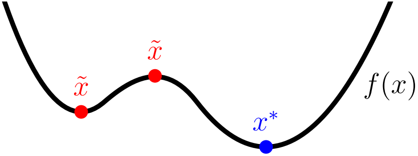
https://openai.com/dall-e-2/

Motivation: Non-Convex Optimization

DALL·E 2 has 5.5 billion parameters and took billions of iterations to fit [2].

Neural network optimization is non-convex!

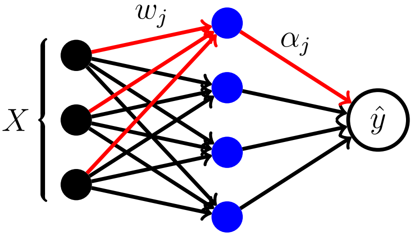
- **NP-Hard**: many sub-optimal local minima, saddles.
- **Speed/Stability**: tuning is critical for performance.
- **Model Churn**: hyper-parameters affect the final model [1].



Convex Reformulations

Non-Convex Problem

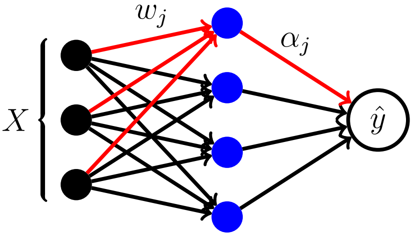
$$\min_{w,\alpha} \| \sum_{j=1}^{m} (Xw_j)_+ \alpha_j - y\|_2^2 + \lambda \sum_{j=1}^{m} \|w_j\|_2^2 + |\alpha_j|^2$$



Convex Reformulations

Non-Convex Problem

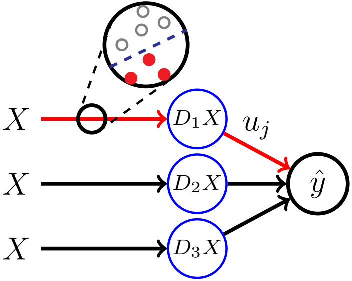
$$\min_{w,\alpha} \| \sum_{j=1}^{m} (Xw_j)_+ \alpha_j - y\|_2^2 + \lambda \sum_{j=1}^{m} \|w_j\|_2^2 + |\alpha_j|^2$$



Convex Reformulation

$$\min_{u} \| \sum_{j=1}^{p} D_j X u_j - y\|_2^2 + \lambda \sum_{j=1}^{p} \|u_j\|_2,$$

where $D_j = \mathrm{diag}[\mathbb{1}(Xg_i \geq 0)]$



Convex Reformulations: A Huge-Scale Linear Model

Convex Form : $$\min_{u} \|\sum_{j=1}^{p} D_j X u_j - y\|_2^2 + \lambda \sum_{j=1}^{p} \|u_j\|_2,$$

$$\text{where } D_j = \text{diag}[\mathbb{1}(X g_i \geq 0)]$$

- Exponential in general: $p \in O(r \cdot (\frac{n}{r})^r)$, where $r = \text{rank}(X)$.
- But, $D_j X$ is row-sparse, u is sparse when $\lambda \gg 0$, and the objective is quadratic $+$ simple non-smooth.

Convex Reformulations: A Huge-Scale Linear Model

Convex Form : $$\min_u \|\sum_{j=1}^{p} D_j X u_j - y\|_2^2 + \lambda \sum_{j=1}^{p} \|u_j\|_2,$$

where $D_j = \text{diag}[\mathbb{1}(X g_i \geq 0)]$

- Exponential in general: $p \in O(r \cdot (\frac{n}{r})^r)$, where $r = \text{rank}(X)$.
- But, $D_j X$ is row-sparse, u is sparse when $\lambda \gg 0$, and the objective is quadratic + simple non-smooth.

Solve with (accelerated) **proximal-gradient** methods:

$$u_i^+ = u_i^k - \eta_k X^\top D_i^\top \left(\sum_{j=1}^{p} D_j X u_j - y \right)$$

$$u_i^{k+1} = u_i^+ \left(1 - \frac{\eta_k \cdot \lambda}{\|u_i^+\|_2} \right)_+$$

Convex Reformulations: Performance

Fast solvers use **numerical tricks** and **hardware acceleration**:

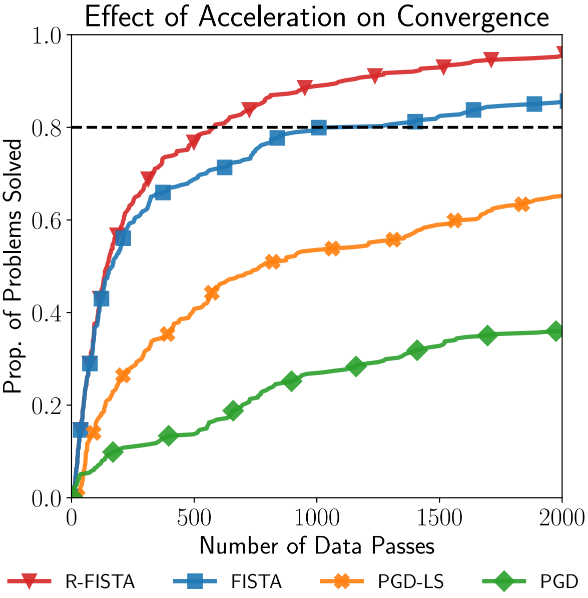
- Classic Tricks: faster convergence via line-search, restarts, ...
- CUDA GPUs: $70\times$ faster Mat-Vec operations using `float32`.
- Code Optimization: tensor operations remove intermediate computations, data normalization improves conditioning, ...

Scaling is a still a problem!

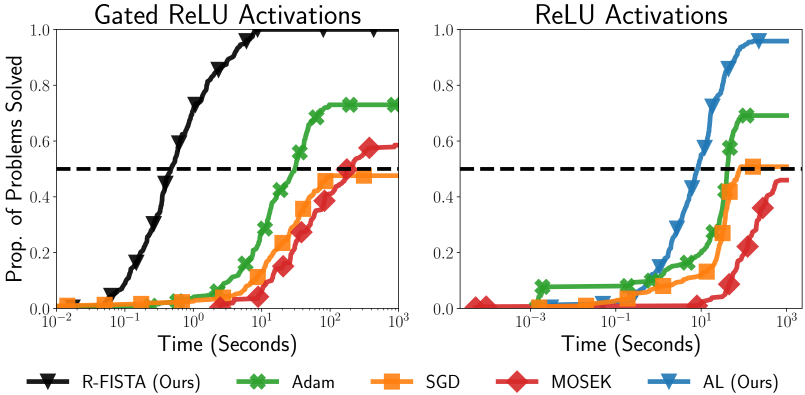
- Dense operations on GPUs are faster than sparse computations with OpenMP — does this reverse at scale?
- GPU memory is typically 32GB, but ImageNet is 150GB — can we use multi-GPU programming models?
- How can we leverage the (dynamic) sparsity pattern of u^k?

Thanks for Listening!

Bonus: Numerical Results (1)



Bonus: Numerical Results (2)



References I

Peter Henderson et al. "Deep Reinforcement Learning That Matters". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018.* 2018, pp. 3207–3214.

Aditya Ramesh et al. "Hierarchical Text-Conditional Image Generation with CLIP Latents". In: *CoRR* abs/2204.06125 (2022). arXiv: 2204.06125.