Beyond Backprop:

Online Alternating Minimization with Auxiliary Variables

NYU

Anna Choromanska

Benjamin Cowen

IBM Research

Sadhana Kumaravel

Ronny Luss

Mattia Rigotti

Irina Rish

Brian Kingsbury

Paolo Di Achillele

Viatcheslav Gurev

Djallel Bouneffouf

MIT

Ravi Tejwani

WHAT'S WRONG WITH BACKPROP?

Computational Issues:

- Vanishing gradients (due to chain of derivatives)
- Difficulty handling non-differentiable nonlinearities (e.g., binary spikes)
- Lack of cross-layer weight update parallelism

Biologically implausibility:

- Error feedback does not influence neural activity, unlike biological feedback mechanisms
- Non-local weight updates, and more [Bartunov et al, 2018]

ALTERNATIVES: PRIOR WORK

- **Offline Auxiliary-variable methods**
  - MAC (Carreira-Perpiñán & Wang, 2014) and other BCD methods (Zhang & Brand, 2017; Zhang & Kleijn, 2017; Askari et al., 2018; Zeng et al., 2018; Lau et al., 2018; Gotmare et al., 2018)
  - ADMM (Taylor et al., 2016; Zhang et al., 2016)
  - offline (batch) is not scalable to large data and continual learning



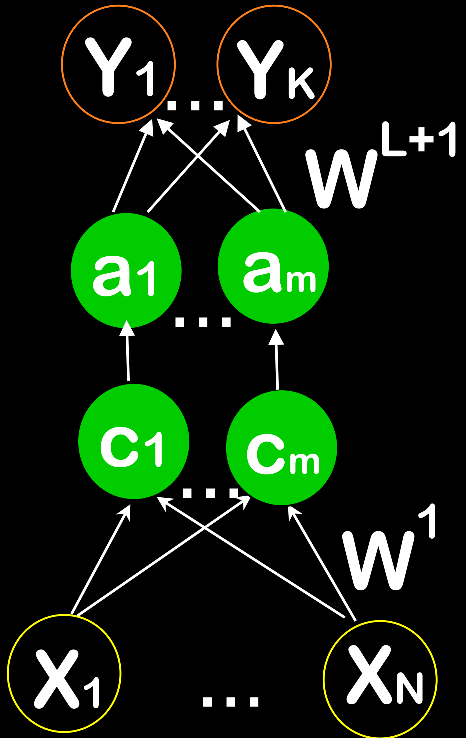
- **Target propagation methods**
  - [LeCun 1986] [Lee, Fisher, Bengio 2015] [Bartunov et al, 2018]
  - Below backprop-SGD performance levels on standard benchmarks

- **Proposed method:**
  - Online (mini-batch, stochastic) auxiliary-variable alternating-minimization

OUR APPROACH

Breaking gradient chains with auxiliary activation variables:

- Relaxing nonlinear activations to noisy (Gaussian) linear activations followed by nonlinearity (e.g., ReLU)
- Alternating minimization over activations and weights: explicit activation propagation
- Weight updates are layer-local, and thus can be parallel (distributed, asynchronous)



NEURAL NETWORK FORMULATIONS

Standard neural network objective function:

Nested

$$\min_{W} \quad \mathcal{L}(y, f(\boldsymbol{W}, \boldsymbol{x}_L))$$
$$\text{where} \quad f(\boldsymbol{W}, \boldsymbol{x}_L) = f_{L+1}(\boldsymbol{W}_{L+1}, f_L(\boldsymbol{W}_L, f_{L-1}(\boldsymbol{W}_{L-1}, ...f_1(\boldsymbol{W}_1, \boldsymbol{x})...)$$

Add auxiliary activation variables (hard constrained problem)

Constrained

$$\min_{\boldsymbol{W}, \boldsymbol{C}} \quad \sum_{t=1}^{n} \mathcal{L}(\boldsymbol{y}_t, \boldsymbol{a}_t^L, \boldsymbol{W}^{L+1}), \text{ where } \boldsymbol{a}_t^l = \sigma_l(\boldsymbol{c}_t^l),$$
$$\text{s.t.} \quad \boldsymbol{c}_t^l = \boldsymbol{W}^l \boldsymbol{a}_t^{l-1}, \ l = 1, ..., L, \ \text{and} \ \ \boldsymbol{a}_t{}^0 = \boldsymbol{x}_t$$

Relax constraints and now amenable to alternating minimization

Relaxed

$$\min_{\boldsymbol{W}, \boldsymbol{C}} \quad \sum_{t=1}^{n} \mathcal{L}(y_t, \sigma_L(c_t^L), W^{L+1}) \quad + \quad \mu \sum_{t=1}^{n} \sum_{l=1}^{L} ||c_t^l - W^l \sigma_{l-1}(c_t^{l-1})||_2^2$$

ONLINE ALTERNATING MINIMIZATION

Offline algorithms of prior works are not scalable to extremely large datasets and not suitable for incremental, continual/lifelong learning, hence …

while more samples **do**
  Input (x_t, y_t)
  $C \leftarrow \text{encodeInput}(\, x_t, W_{t-1})$ **Forward: compute linear activations at layers 1,…,L**
  $C \leftarrow \text{updateCodes}(\, C, y_t, \; W_{t-1}, \mu)$ **Backward: error propagation by code changes**
  $W_t \leftarrow \text{updateWeights}(\, W_{t-1}, \; x_t, \; y_t, \; C, \; \mu, \; \eta, \; Mem)$ **Parallelizable**
end while
return W_t

Note: **updateWeights** has two options: Apply SGD to the current mini-batch or apply BCD to version that includes memory of previous samples using the following (via Mairal et al., 2009):

$$\sum_{i=1}^{t} ||\boldsymbol{c}_i^l - \boldsymbol{W}\boldsymbol{a}_i^l||_2^2 = Tr(\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{A}_t^l) - 2Tr(\boldsymbol{W}^T\boldsymbol{B}_t^l)$$

FULLY-CONNECTED NETS

AM greatly **outperforms all off-line** methods (ADMM of Taylor et al, and offline AM), and often matches Adam and SGD (50 epochs)

MNIST

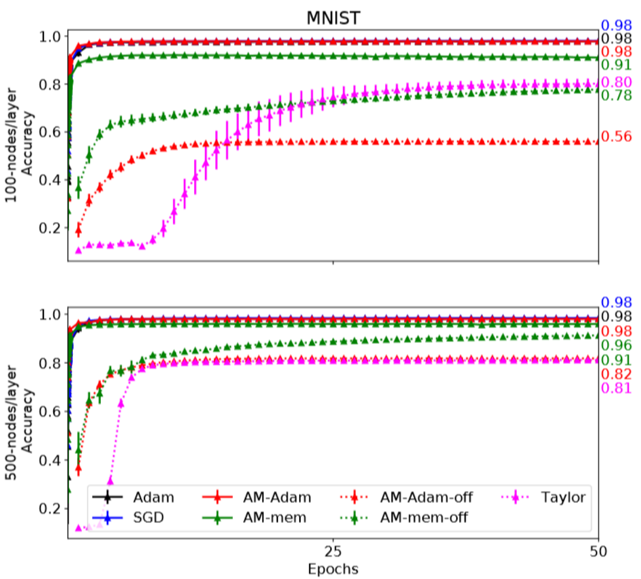


Figure 2. MNIST (fully-connected nets, 2 layers): online vs. offline methods vs. Taylor's ADMM, 50 epochs.

CIFAR-10

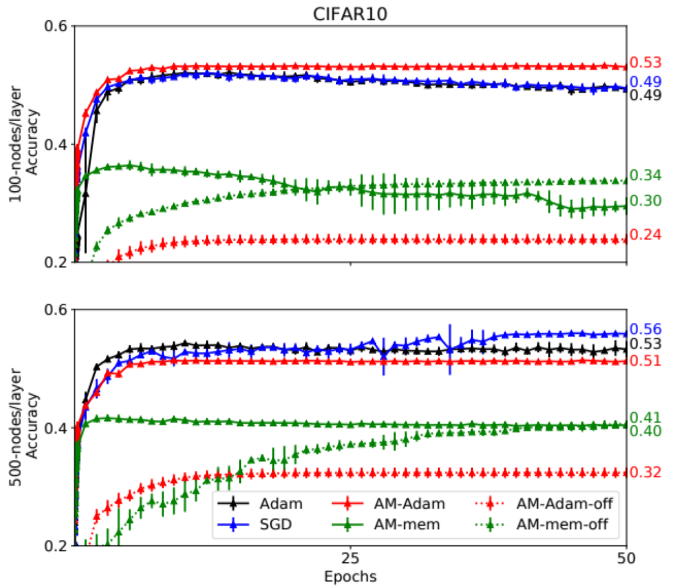


Figure 4. CIFAR10 (fully-connected networks): online vs. offline, 50 epochs. Similar experiments to Figure 2.

FASTER INITIAL LEARNING: POTENTIAL USE AS A GOOD INIT?

- AM often learns faster than SGD & Adam (backprop-based) in the 1st epoch, then matches their performance

MNIST

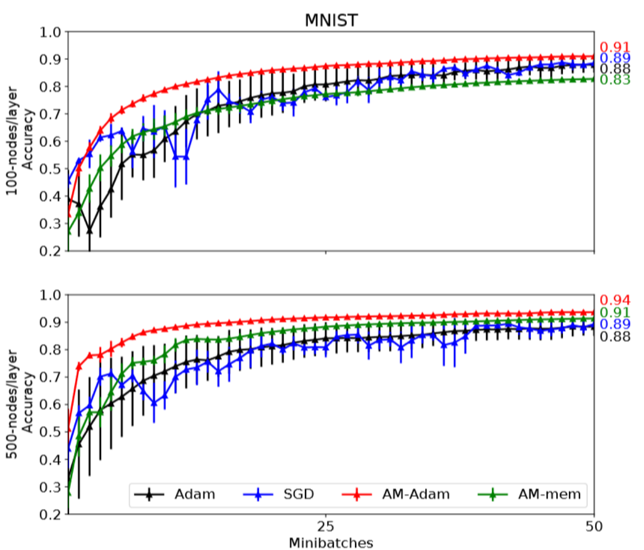


Figure 1. MNIST (fully-connected nets, 2 layers): online methods, first epoch; 50 mini-batches, 200 samples each.

CIFAR-10

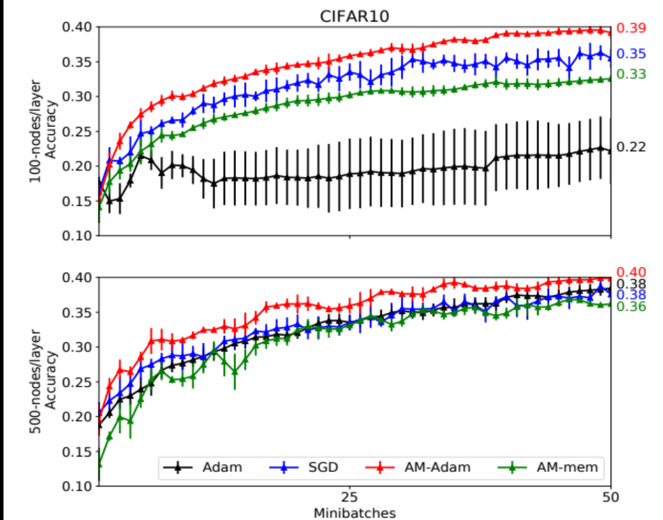


Figure 3. CIFAR10 (fully-connected nets): online methods, 1st epoch. 2 hidden layers with 100 (top) and 500 (bottom) units each; 250 mini-batches, 200 samples each.

CONVNETS: LENET5, MNIST

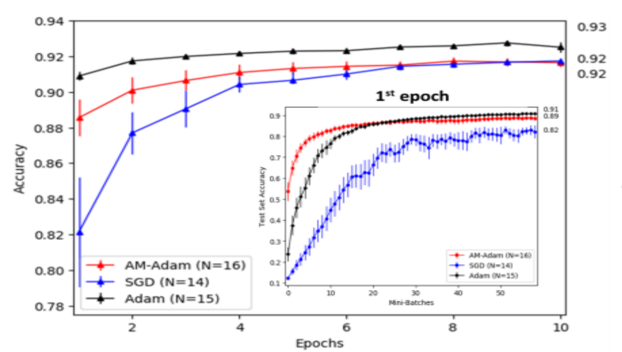


Figure 6. RNN-15, Sequential MNIST.

RNN: SEQUENTIAL MNIST

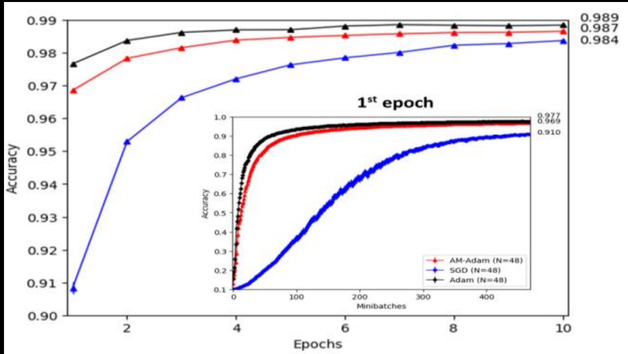


Figure 7. CNN: LeNet5, MNIST.

HIGGS DATASET, FULLY-CONNECTED

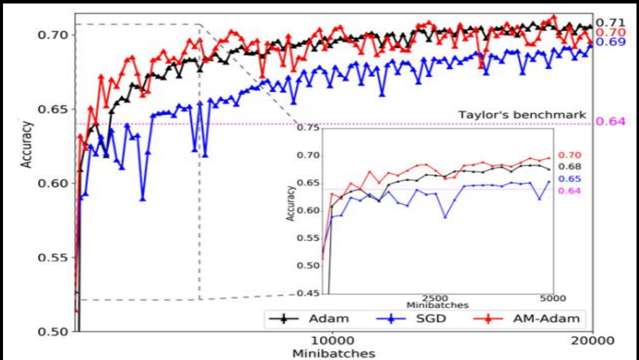


Figure 5. HIGGS dataset.

- AM performs similarly to Adam, outperforms SGD
- All methods greatly outperform offline ADMM (Taylor's 0.64 benchmark) using less than 0.01% of 10.5M-sample HIGGS data

NONDIFFERENTIABLE (BINARY) NETS

- Backprop replaced by Straight-Through Estimator (STE)
- Comparing with Difference Target Propagation (DTP)
- DTP took about 200 epochs to reach 0.2 error, matching the STE performance (Lee et al., 2015)
- AM-Adam with binary activations reaches same error in < than 20 epochs

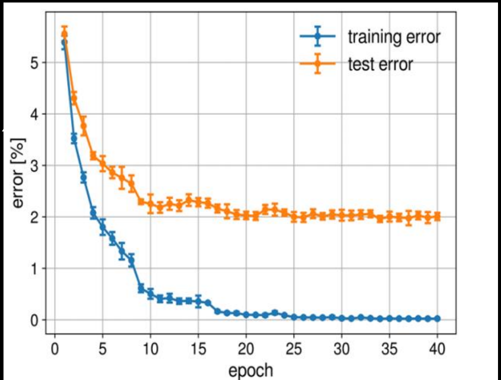


Figure 8. Binary net, MNIST.

SUMMARY: CONTRIBUTIONS

- **Algorithm(s):** novel online (stochastic) auxiliary-variable approach for training neural networks (prior methods are offline/batch); two versions of the approach (memory-based and local-SGD-based)
- **Theory:** first general theoretical convergence guarantees for alternating minimization in the stochastic setting: the error decays at the sub-linear rate $O((1/t)^{3/2} + 1/t)$ in t iterations
- **Extensive Evaluations:** variety of architectures and datasets demonstrating advantages of online vs offline approaches and performance similar to SGD (Adam), with faster initial convergence