Learning-Rate-Free Learning by D-Adaptation

Authors: Aaron Defazio (Meta AI), Konstantin Mishchenko (Samsung AI)

Presented by Arseniy Kan arseniykan@unist.ac.kr, Seong Ju Meang

AI51101 Optimization for AI

CONTENTS

Motivation

Motivation – The Problem

- We usually train ML models using a variant of **gradient descent**

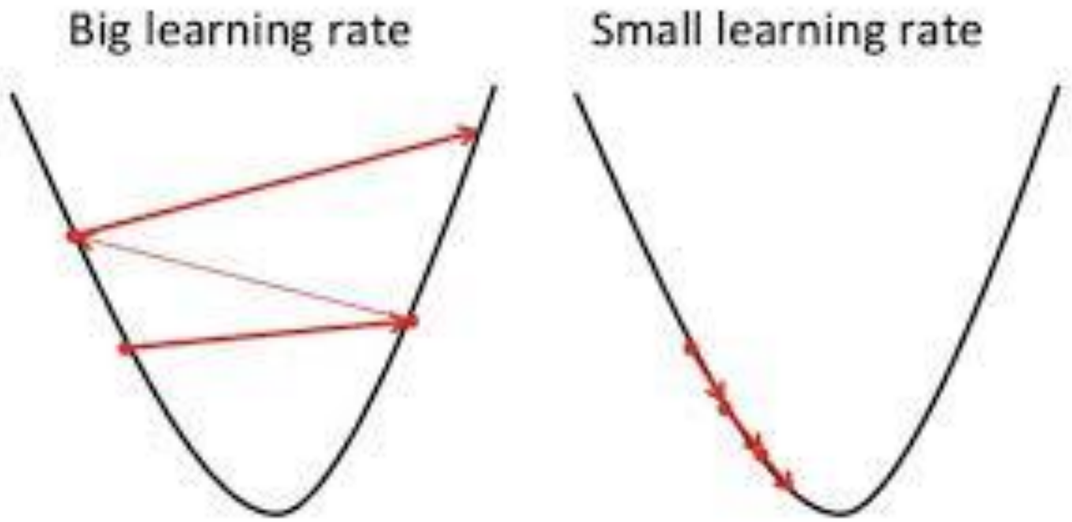
- The standard update rule is

$$x_{k+1} = x_k - \gamma_k g_k,$$

where γ_k is the learning rate

Motivation – The Problem

- The learning rate (γ_k) is the most **critical hyper-parameter** in deep learning.
- **Too large:** The optimizer oscillates and diverges.
- **Too small:** Training is extremely slow.



D-Adaptation Motivation

- Automatically sets the learning rate for the minimization of convex Lipschitz functions.

Motivation – The Problem

- The optimal learning rate (γ_k) requires knowing D.

$$\gamma_k = \frac{D}{G\sqrt{n}}$$

- D - the **distance from our starting point to the true solution**

$$D = \|x_0 - x_*\|$$

- Core Problem: We never know D in advance!

What if, instead of *guessing* D, we could **estimate** it while we train?

Approach

Instead of *guessing* D, the algorithm dynamically computes and maintains **a provable lower bound on D** throughout training.

Approach

1. We use our current d_k to determine the learning rate and take a step.
2. Using the new gradient information from that step, we calculate a new **lower bound. (Measure)**
3. We compare it to our old and update our best (maintaing best bound part)
4. This is then used to determine the learning rate for the next step.

Estimating the lower bound

- **What does it observe? (Input)**: The Directional Efficiency of the optimization path.

- **What does it infer? (Inference)**: The magnitude of the Effective Remaining Distance to travel.

- **What does it determine? (Output)**: The absolute Scale of the learning rate.

Maintaining the best bound

$$d_{k+1} = \max\left(d_k, \hat{d}_{k+1}\right)$$

Type equation here.

Why D adaptation is effective

$$f(\hat{x}_n) - f(x_*) = \mathcal{O}\left(\frac{DG}{\sqrt{n+1}}\right)$$

Theorem 1

Time Constraint

Theorem 2

Generality

Theorem 3

Scalability

Theorem 4

Main Results

Main Results - Overview

- **Broad Success: D-Adaptation** (SGD/Adam) matches or exceeds the performance of carefully hand-tuned baselines across a huge range of tasks and architectures.
- **Where it Works Best:**
  - Convex Problems: (Logistic Regression)
  - CNNs: (ResNet, DenseNet, Wide ResNet)
  - Transformers: (RoBERTa, GPT)
  - LSTMs: (Machine Translation)
  - Other: (Object Detection, MRI, Recommendation)
- **The Main Failure Case:**
  - **Vision Transformer (ViT):** This was the one task where D-Adapt Adam visibly under-performed the hand-tuned baseline.
- **Comparison to Other "Parameter-Free" Methods:**
  - D-Adaptation was compared to **COCOB**, another parameter-free method.
  - D-Adaptation was comparable to the hand-tuned baseline on almost every task.
  - COCOB failed to match the baseline on any task and diverged on several.

Main Results – Convex Problems

- **Task:** Logistic Regression on **12 LIBSVM benchmark** problems.
- **Baseline:** Standard Adam with a fully grid-searched, hand-tuned learning rate.
- **Result:** D-Adapt Adam (green) **matched or exceeded** the performance of the hand-tuned Adam (gray) on all 12 datasets—with zero manual tuning.

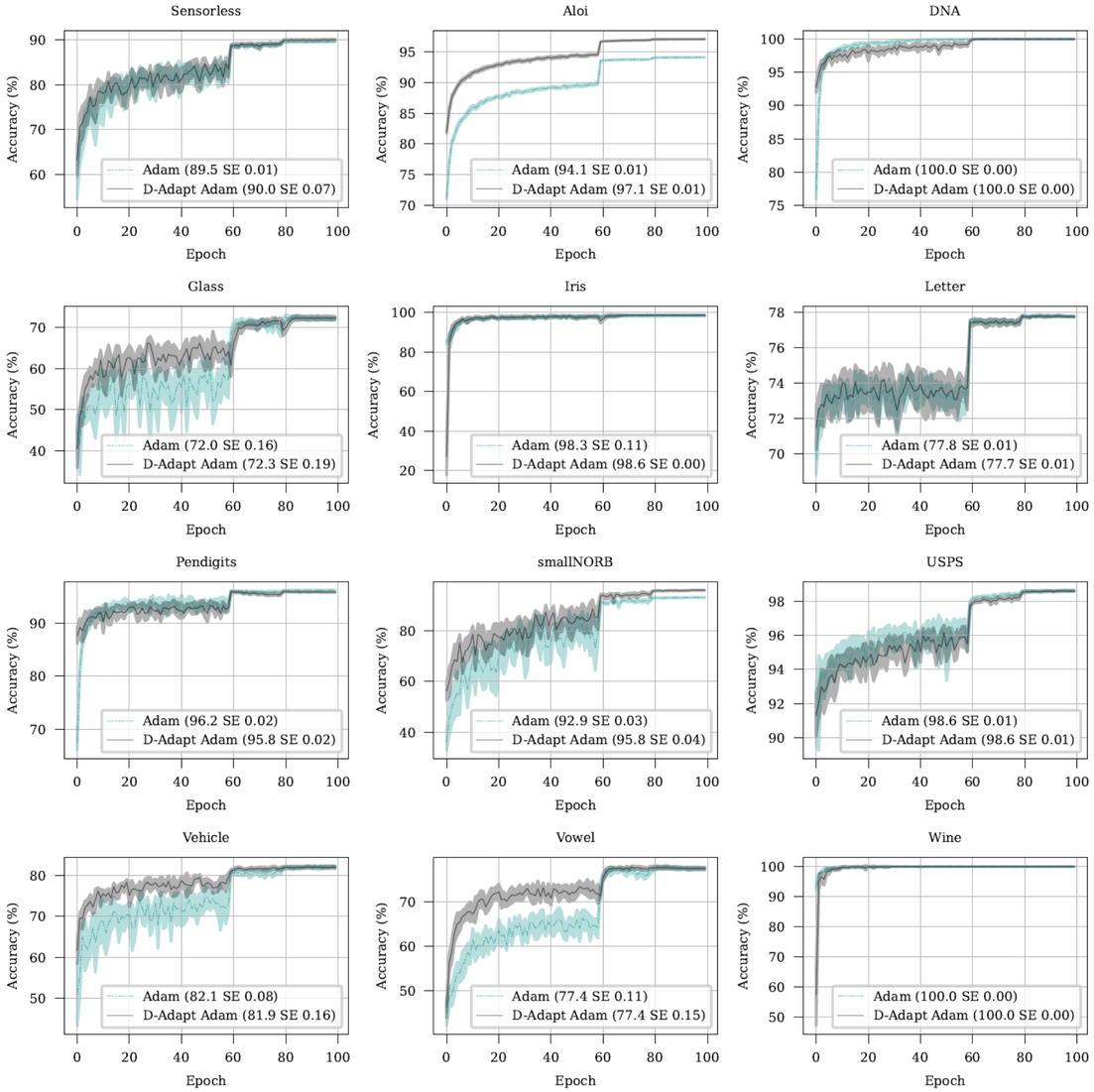


Figure 2: Logistic Regression experiments.

Main Results – Convolutional Image Classification

- **Task:** Standard CNN benchmarks on CIFAR-10 (Wide ResNet), CIFAR-100 (DenseNet), and ILSVRC 2012 ImageNet (ResNet-50).
- **Result:** D-Adapt SGD (black) closely matches the performance of the highly-tuned baseline SGD (red) on all three major datasets.

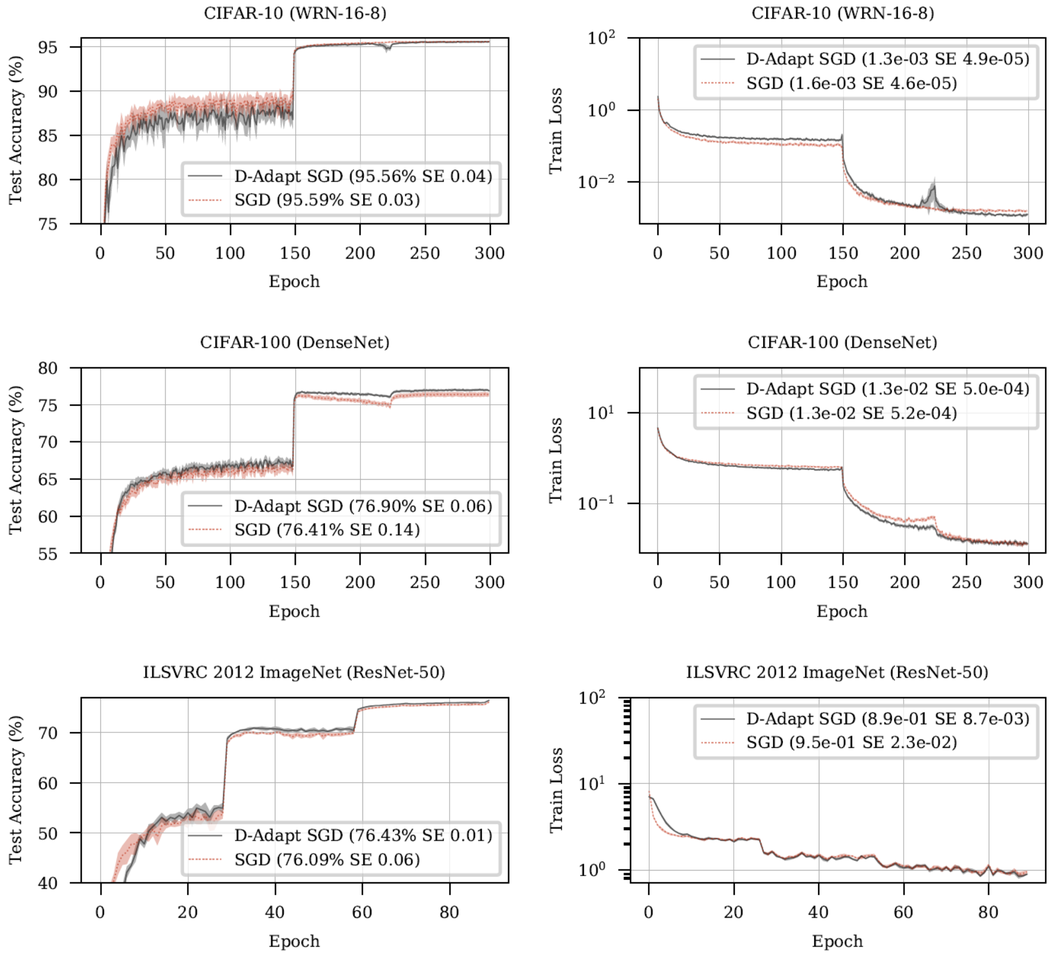
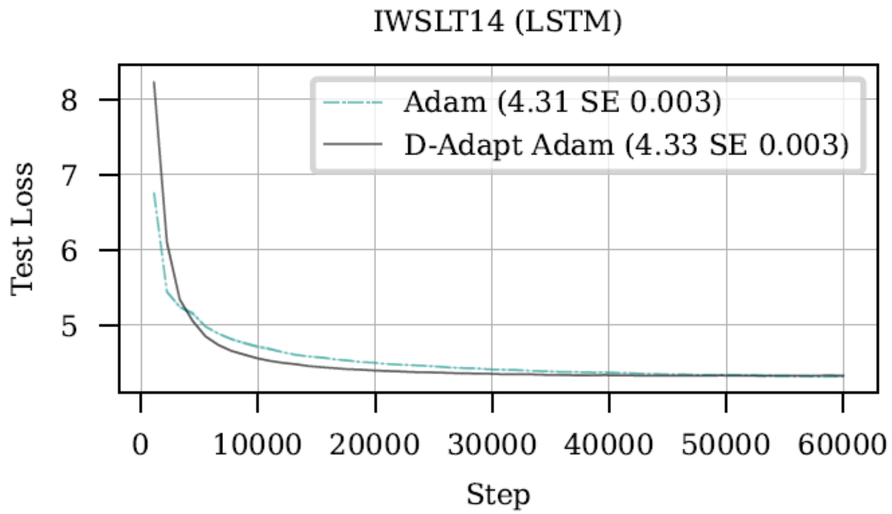
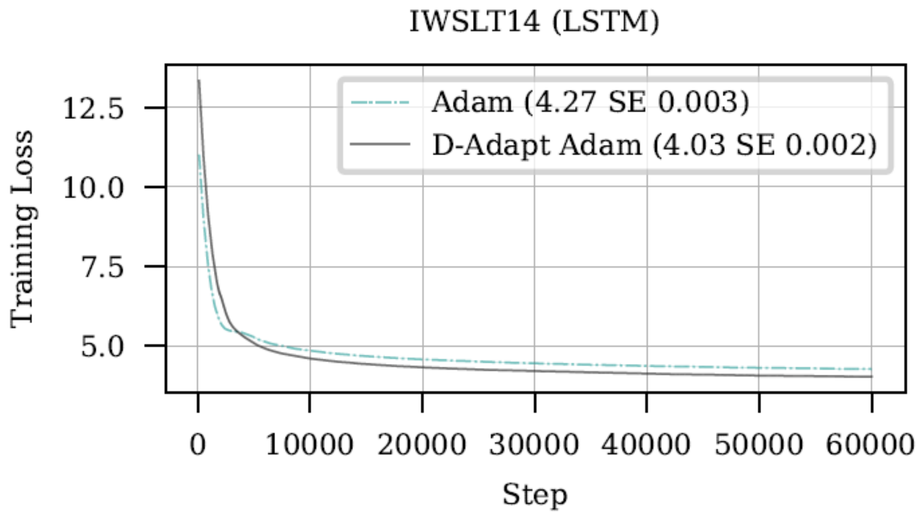


Figure 3: Image Classification experiments.

Main Results – LSTM RNNs

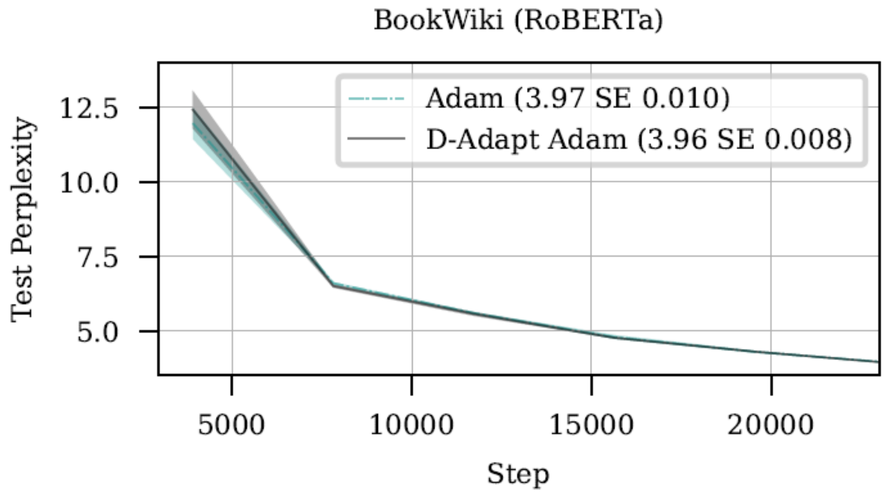
- **Task:** Machine Translation on IWSLT14 (German-to-English).
- **Result:** D-Adapt Adam (black) achieves comparable performance to the hand-tuned baseline Adam (green). The test loss curves are nearly identical.

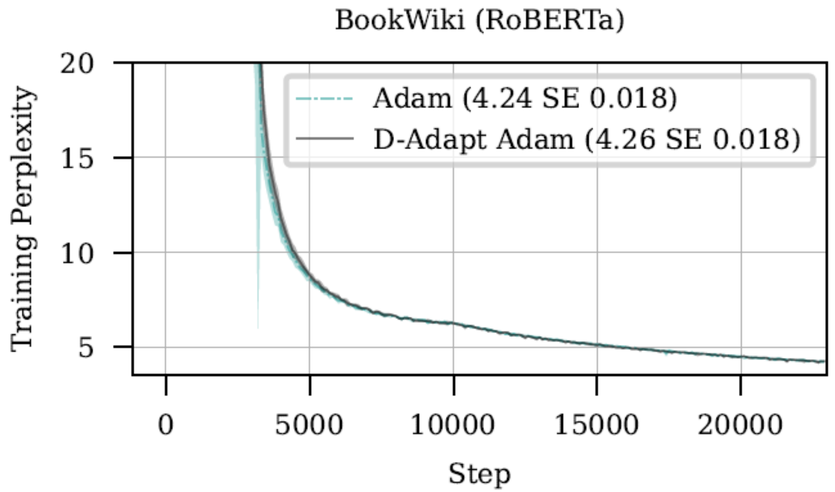




Main Results – Masked Language Modeling

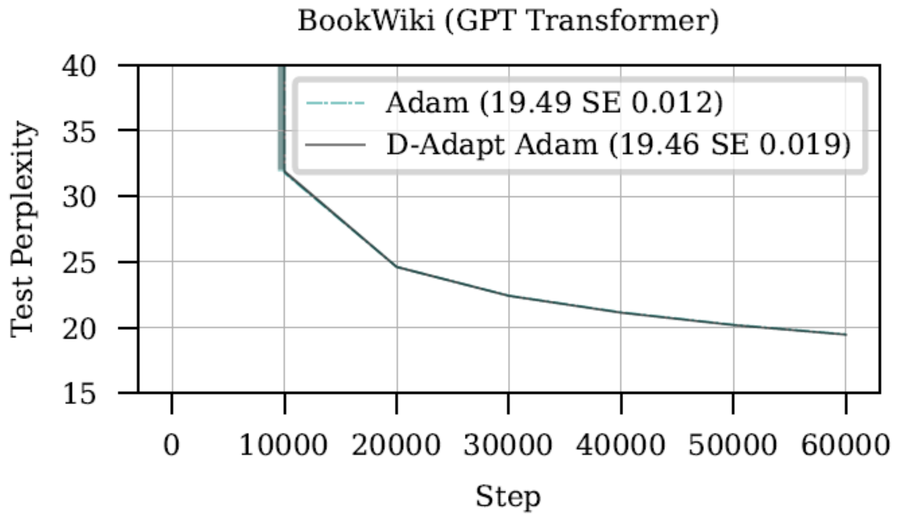
- **Task:** Pre-training on the BookWiki corpus.
- **Model:** RoBERTa (110M parameter Transformer), a large-scale, realistic problem.
- **Result:** D-Adapt Adam (black) perfectly matches the baseline (green) in final test set perplexity.

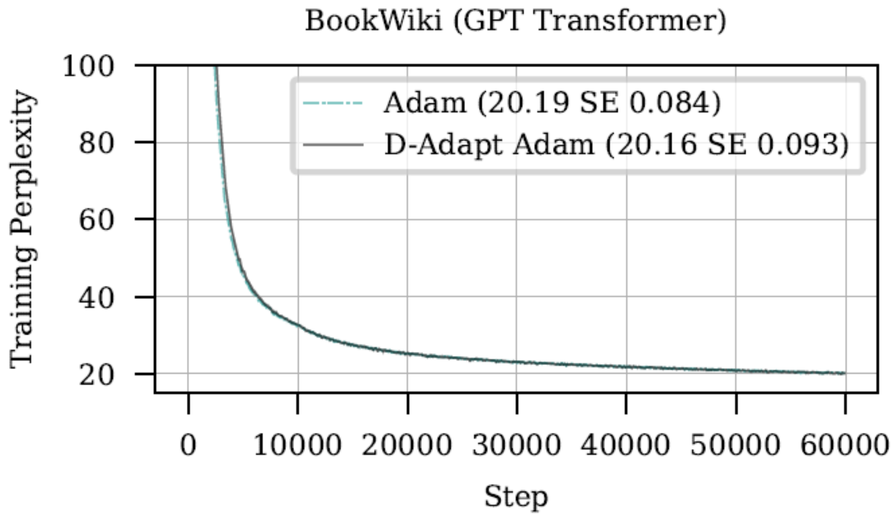




Main Results – Auto-regressive Language Modeling

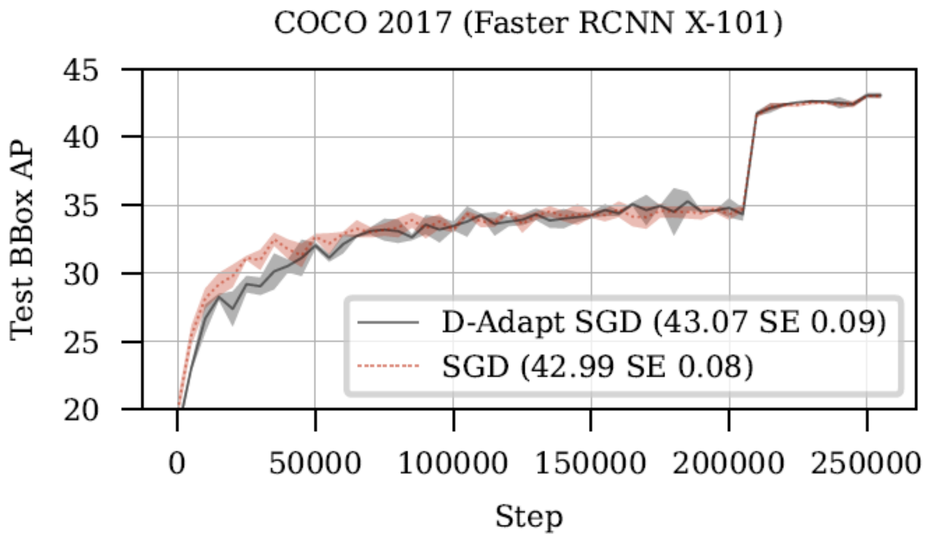
- **Task:** Auto-regressive language modeling on the BookWiki corpus.
- **Model:** GPT-style decoder-only Transformer.
- **Result:** D-Adapt Adam (black) performance is **virtually identical** to the tuned baseline (green), with only a negligible difference in perplexity.

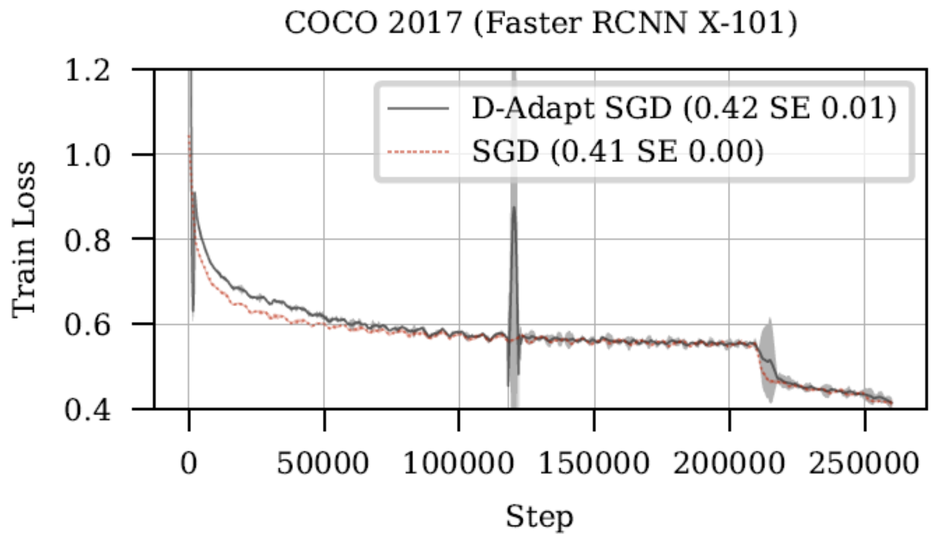




Main Results – Object Detection

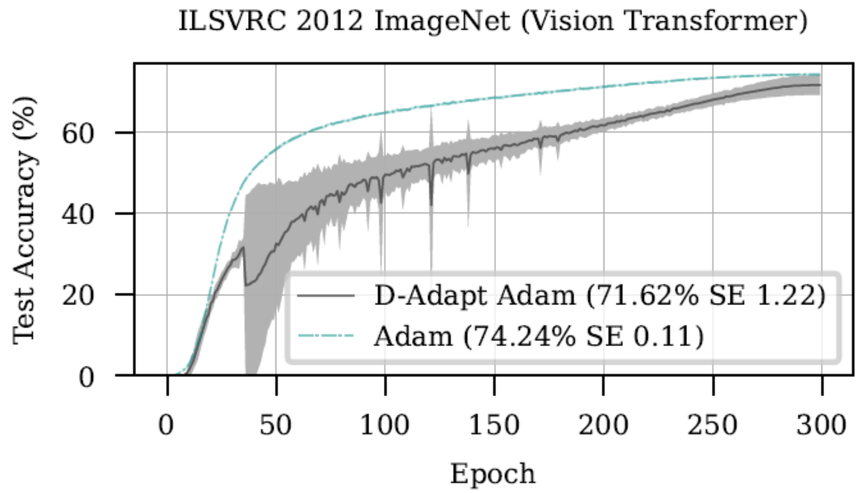
- **Task:** COCO 2017 Object Detection, a popular computer vision benchmark.
- **Model:** Faster R-CNN with a ResNeXt-101 backbone.
- **Result:** D-Adapt SGD (gray) **matches the final test AP** (Average Precision) of the tuned baseline (red).
- **Note:** This experiment highlighted that other hyper-parameters (like weight decay) still matter and need to be set appropriately (changed default decay 0.0001 -> 0.00015).

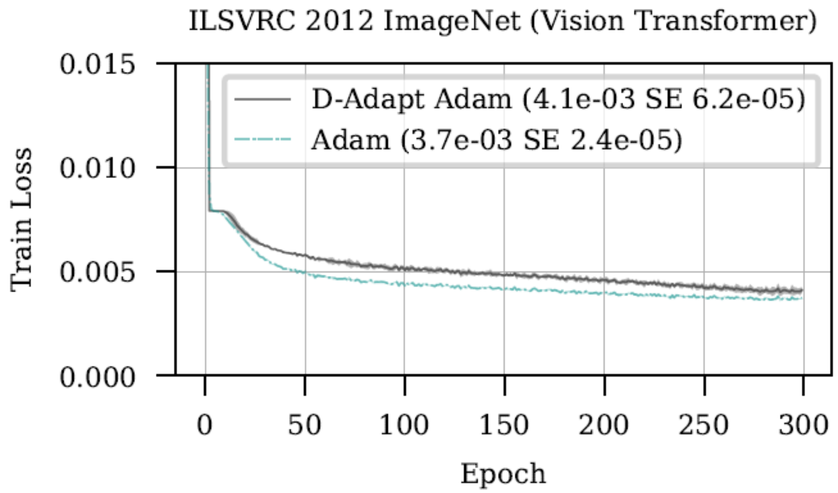




Main Results – Vision Transformers

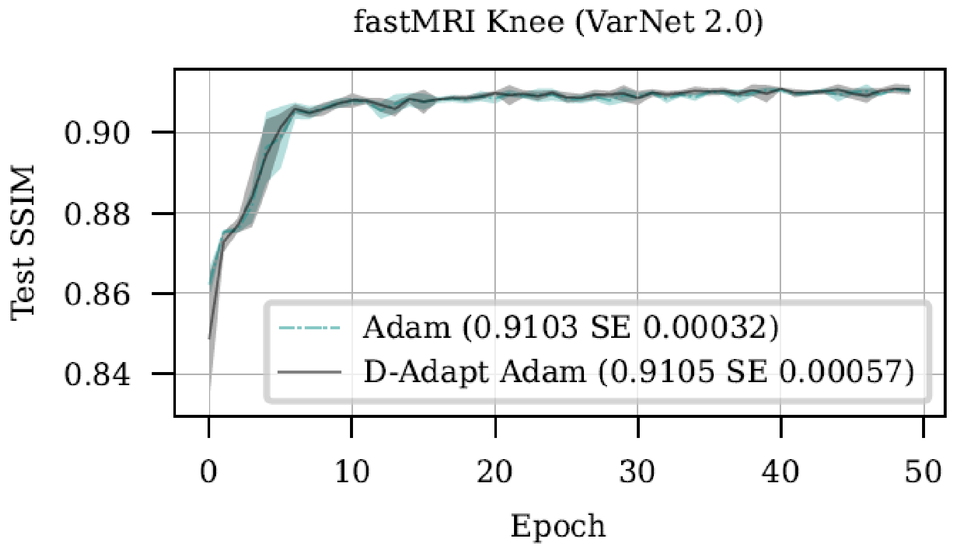
- **Task:** ILSVRC 2012 ImageNet classification.
- **Model:** Vision Transformer (ViT).
- **Result (Failed Testcase):** D-Adapt Adam (gray) **visibly under-performs** the hand-tuned baseline (green) by ~3% accuracy.
- **Hypothesis:** The authors note this problem is "highly sensitive to the initial learning rate," which may be a limitation of the D-Adaptation method.

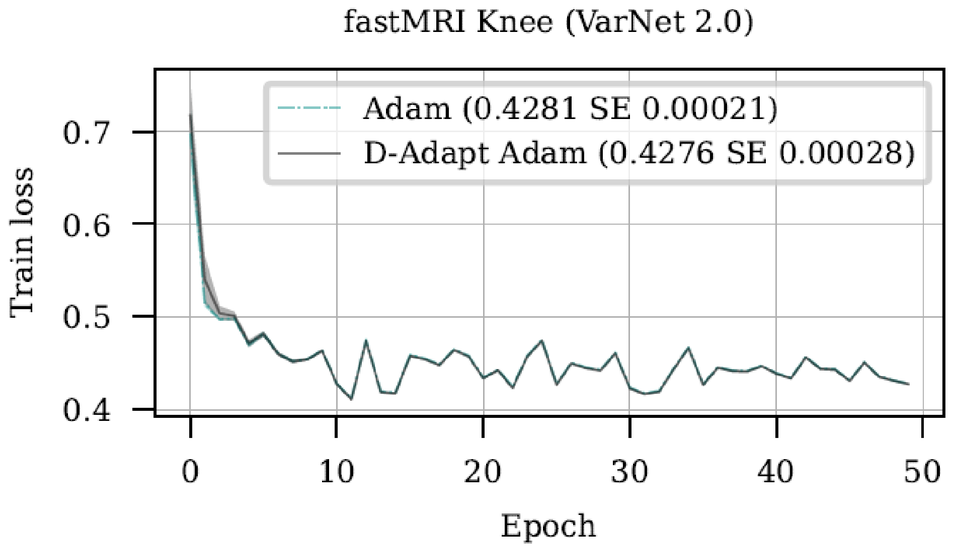




Main Results – fastMRI

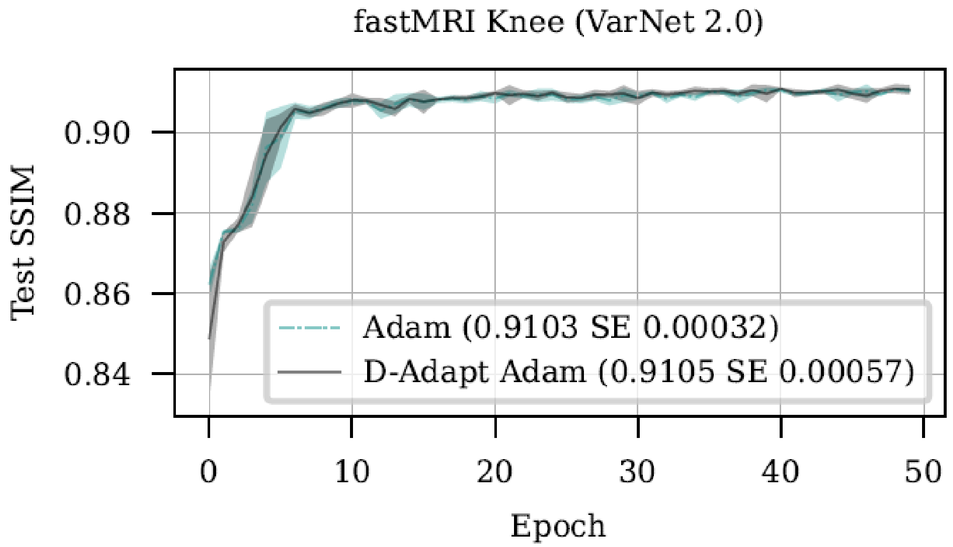
- **Task:** fastMRI Knee Dataset, an image reconstruction problem.
- **Model:** VarNet 2.0.
- **Result:** D-Adapt Adam (gray) **matches the performance** of the highly tuned baseline (green) on the Test SSIM (Structural Similarity) metric.

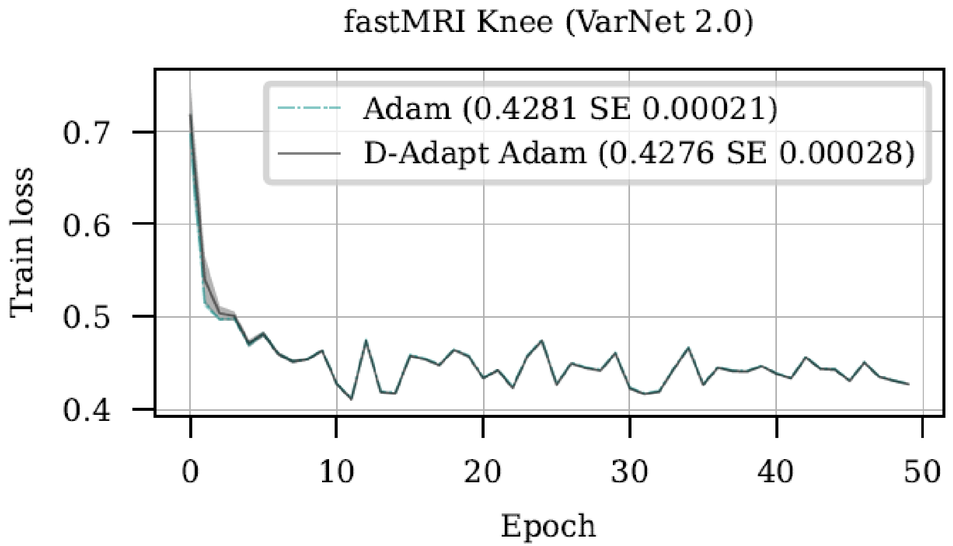




Main Results – fastMRI

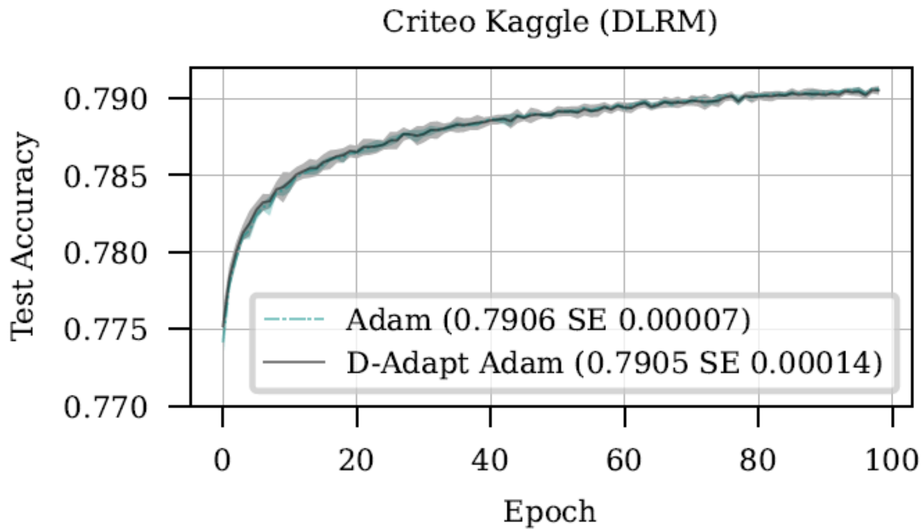
- **Task:** fastMRI Knee Dataset, an image reconstruction problem.
- **Model:** VarNet 2.0.
- **Result:** D-Adapt Adam (gray) **matches the performance** of the highly tuned baseline (green) on the Test SSIM (Structural Similarity) metric.

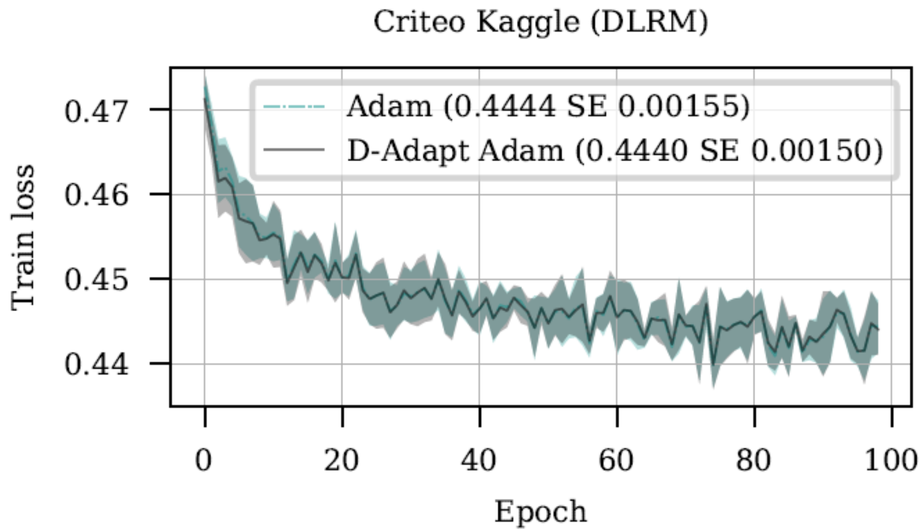




Main Results – Recommendation Systems

- **Task:** Criteo Kaggle Display Advertising challenge, a large, sparse dataset for predicting user click-through events.
- **Model:** DLRM (Deep Learning Recommendation Model), a benchmark for industrial-scale personalization and recommendation.
- **Result:** D-Adapt Adam (gray) closely matches the performance of the hand-tuned baseline Adam (green). The final test accuracy curves are nearly identical.





Main Results – Observed Learning Rates

Problem	Baseline LR	D-Adapted LR	Std. Dev.
CIFAR10	1.0	2.085	0.078
CIFAR100	0.5	0.4544	0.029
ImageNet	1.0	0.9227	0.084
IWSLT	0.01	0.003945	0.000086
GPT	0.001	0.0009218	0.000014
RoBERTa	0.001	0.0009331	0.000011
COCO	0.2	0.2004	0.0026
ViT	0.001	0.0073	0.00085
fastMRI	0.0003	0.0007596	0.00022
DLRM	0.0001	0.0001282	0.000056

Table 1: Comparison of baseline learning rates against final D-Adapted learning rates for the deep learning experiments, with average and standard deviation shown over multiple seeds.

Critical Review

Critical Review - Strengths

✓ Solves a Real, Expensive Problem

Eliminates the need for costly, manual **learning rate (LR) grid search.**

✓ Theoretically Optimal

Achieves the best possible $O\left(\frac{1}{\sqrt{n}}\right)$ convergence rate for convex problems.

✓ "Log-Free" Rate

The first hyper-parameter-free method to achieve this optimal rate without slower log factors.

✓ Broad Empirical Validation

Matches hand-tuned LRs on diverse tasks (Vision, NLP, RecSys) and architectures (CNN, LSTM, Transformers).

✓ Robust

Final accuracy is highly **insensitive** to the initial d_0 parameter.

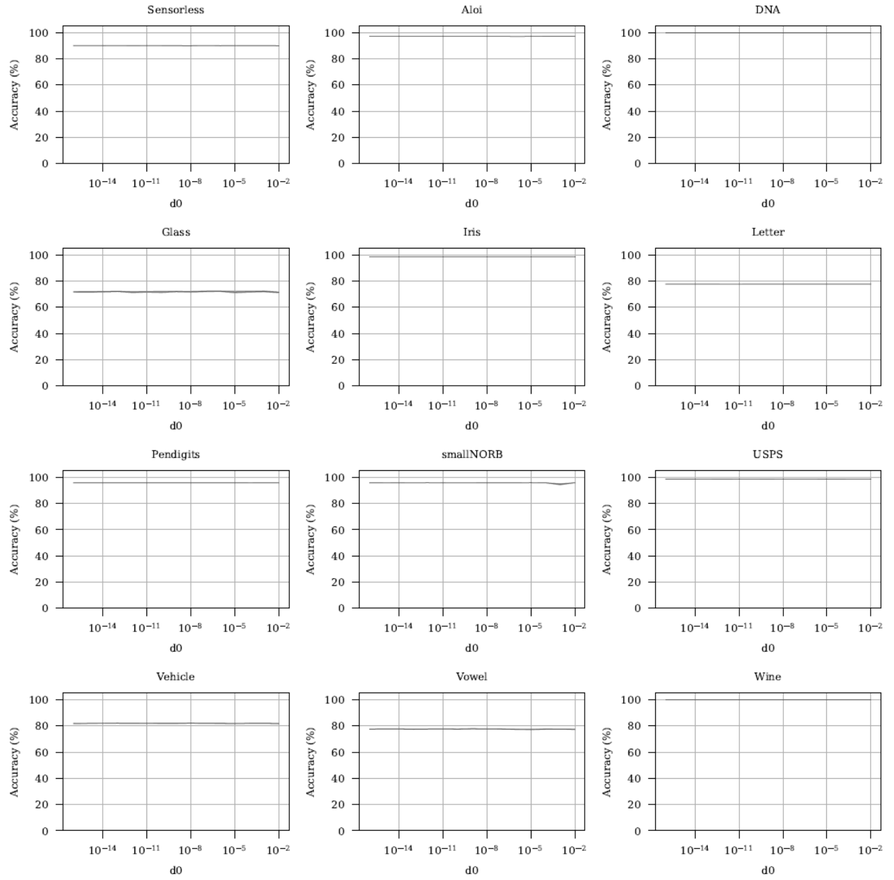


Figure 7: Final accuracy as a function of d_0. Setup described in Section 7.1. Error bars show a range of 2 standard errors above and below the mean of the 10 seeds. For most problems error bars are too narrow to be visible.

Critical Review - Weaknesses

⊗ The Theory-Practice Gap

- D adaptation convergence rate is only valid for convex Lipschitz function.
- Deep learning is famously **non-convex and stochastic** (SGD with mini batches).
- Although empirical results are good in deep learning, the **theoretical success is not guaranteed**.

⊗ ViT Failure Case

- The method under-performed the hand-tuned baseline on the Vision Transformer experiment.
- The problem appears to be highly **sensitive to the initial learning rate**, which may explain the observed differences.

⊗ Average vs. Last Iterate Guarantee

- The theory provides guarantees for the average of all iterates $(\hat{x}_n)$.
- In deep learning practice, we use **the last iterate** (x_n). The paper doesn't explore this gap.

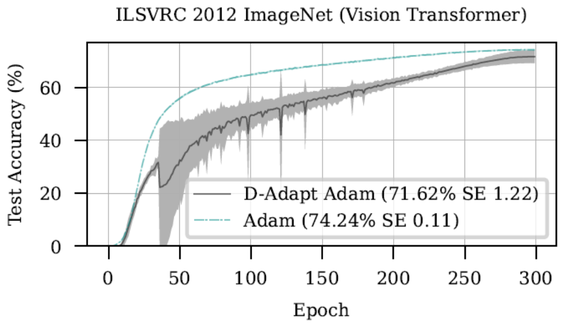
6. Machine Learning Applications

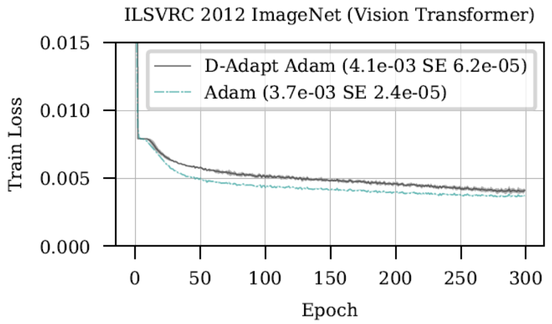
It is straightforward to adapt the D-Adaptation technique to stochastic optimization, although the theory no longer directly supports this case. Algorithm 4 and 5 are versions of D-Adaptation for SGD and Adam respectively. Both of the two methods solve the stochastic optimization problem,

$$\min_{x \in \mathbb{R}^p} \mathbb{E}[f(x, \xi)]$$

using stochastic subgradients $g_k \in \partial f(x_k, \xi_k)$.

For the SGD variant (Algorithm 1), we multiply the D bound by a factor of two compared to Algorithm 4. This improves the practical performance of the method. Our theoretical rate is still valid up to constant factors, for any constant multiplier applied to the step size, so this change is still covered by our theory. For the denominator of the step size, we use $G = \|g_0\|$, which is a crude approximation to the true G but appears to work very well in practice.





Critical Review - Future Directions

❖ Bridge the Theory-Practice Gap

Extend proofs from convex to **non-convex** and **stoch astic** settings (deep learning).

❖ Investigate Failure Modes

Why did it fail on **Vision Transformers (ViT)**? Improve robustness for sensitive architectures.

❖ Fully Automate Schedules

Move beyond just the base LR; can D-Adaptation also automate **warmup** and **decay schedules**?

❖ Explore Last-Iterate Convergence

Provide theoretical guarantees for the **last iterate** $(\boldsymbol{x_n})$, not just the average iterate $(\hat{x}_n)$

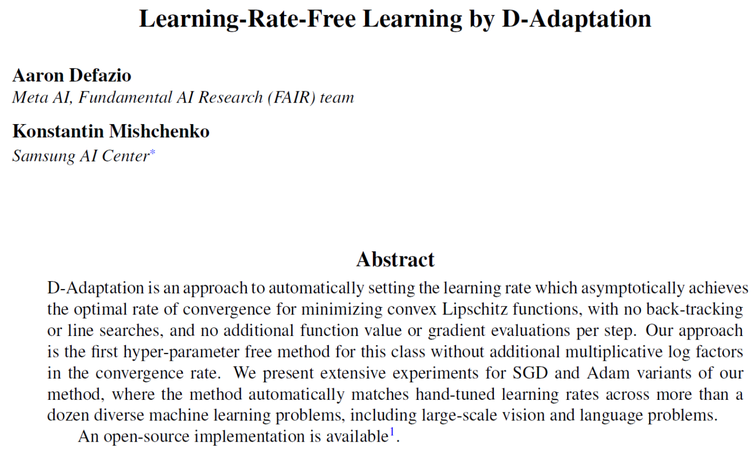
Conclusion

Summary

- **What is it? D-Adaptation** is a simple, robust method for "learning-rate-free" optimization.
- **How?** It cleverly **estimates the unknown distance to the solution D** on the fly and adapts the learning rate accordingly.
- **Why is it good?**

1. It has optimal theoretical guarantees for convex problems.
2. It shows outstanding *empirical* performance on a wide range of deep learning tasks, matching hand-tuned LRs.

- **The Takeaway:** While a theory-practice gap remains, D-Adaptation is a significant and practical step toward fully automated, "plug-and-play" machine learning.



Learning-Rate-Free Learning by D-Adaptation

Aaron Defazio
Meta AI, Fundamental AI Research (FAIR) team

Konstantin Mishchenko
*Samsung AI Center**

Abstract

D-Adaptation is an approach to automatically setting the learning rate which asymptotically achieves the optimal rate of convergence for minimizing convex Lipschitz functions, with no back-tracking or line searches, and no additional function value or gradient evaluations per step. Our approach is the first hyper-parameter free method for this class without additional multiplicative log factors in the convergence rate. We present extensive experiments for SGD and Adam variants of our method, where the method automatically matches hand-tuned learning rates across more than a dozen diverse machine learning problems, including large-scale vision and language problems.

An open-source implementation is available[1].

Q&A Session

ULSAN NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY

2 0 0 7

THANK YOU

Approach

$$\hat{d}_{k+1} = \frac{\gamma_{k+1} \|s_{k+1}\|^2 - \sum_{i=0}^{k} \gamma_i d_i^2 \|g_i\|^2}{2 \|s_{k+1}\|}$$

$$d_{k+1} = \max\left(d_k, \hat{d}_{k+1}\right)$$

Theorem 1,2

Theorem 1 *For a convex G-Lipschitz function f, Algorithm 1 returns a point $\hat{x}_n$ such that:*

$$f(\hat{x}_n) - f(x_*) = \mathcal{O}\left(\frac{DG}{\sqrt{n+1}}\right),$$

as $n \to \infty$, where $D = \|x_0 - x_*\|$ for any x_* in the set of minimizers of f, as long as $d_0 \leq D$.

Theorem 2 *Consider Algorithm 1 run for $n \geq 2\log_2(D/d_0)$ iterations with the step size modified to be*

$$\gamma_{k+1} = \frac{1}{\sqrt{G^2 + \sum_{i=0}^{k} \|g_i\|^2}}. \tag{1}$$

If we return the point $\hat{x}_t = \frac{1}{\sum_{k=0}^{t} d_k} \sum_{k=0}^{t} d_k x_k$ where t is chosen to be

$$t = \arg\min_{k \leq n} \frac{d_{k+1}}{\sum_{i=0}^{k} d_i},$$

then using the notation $\log_{2+}(x) = \max(1, \log_2(x))$, we have:

$$f(\hat{x}_t) - f_* \leq 16\frac{\log_{2+}(d_{n+1}/d_0)}{n+1} D\sqrt{\sum_{k=0}^{t} \|g_k\|^2} \leq 16\frac{DG\log_{2+}(D/d_0)}{\sqrt{n+1}}.$$

Theorem 3,4

Theorem 3 *Gradient Descent with D-Adaptation (Algorithm 2), under the assumptions of Theorem 1, returns a point* $\hat{x}_n$ *such that:*

$$f(\hat{x}_n) - f = \mathcal{O}\left(\frac{DG}{\sqrt{n+2}} \log(n+2)\right).$$

This log factor arises whenever any-time step sizes are used on top of gradient descent when applied to unbounded domains, and is not specific to our method (Beck, 2014).

Theorem 4 *For a convex p-dimensional function with* $G_\infty = \max_x \|\nabla f(x)\|_\infty$, *D-Adapted AdaGrad (Algorithm 3) returns a point* $\hat{x}_n$ *such that*

$$f(\hat{x}_n) - f_* = \mathcal{O}\left(\frac{\|a_{n+1}\|_1 D_\infty}{n+1}\right) = \mathcal{O}\left(\frac{pG_\infty D_\infty}{\sqrt{n+1}}\right),$$

as $n \to \infty$, *where* $D_\infty = \|x_0 - x_*\|_\infty$ *for any* x_* *in the set of minimizers of* f, *as long as* $d_0 \leq D_\infty$.