The 33rd Annual Conference of the Japanese Society for Artificial Intelligence, 2019

データ並列深層学習における
短期事前学習を用いた適応的学習係数調節手法

Adaptive Learning Rate Adjustment with Short-Term Pre-Training in Data-Parallel Deep Learning

山田 和樹, 森 陽紀, 陽川 哲也, 宮内 勇貴, 和泉 慎太郎, 吉本 雅彦, 川口 博
Kazuki Yamada, Haruki Mori, Tetsuya Youkawa,Yuki Miyauchi, Shintaro Izumi, Masahiko Yoshimoto, Hiroshi Kawaguchi

神戸大学
Kobe University

Abstract: This paper describes short-term pre-training (STPT) algorism to adaptively select an optimum learning rate (LR). The proposed STPT algorism is beneficial for quick model prototyping in data-parallel deep learning. It adaptively finds an appropriate LR from multiple LR sets by STPT, which means the multiple LRs are evaluated within the beginning few iterations in an epoch. The STPT short cuts the tuning process of LRs that is requested in conventional training procedure as hyper-parameter tuning, even if the unknown models are considered. Therefore, the proposed STPT reduces computational time and increases throughput to find the best LR for network training. This algorism reduces the computational time by 87.5% than the conventional method when the eight-LR sets are evaluated using eight-parallel workers. We verified the accuracy improvement by 4.8 % compared with the conventional one with a reference LR of 0.1; there are no accuracy deterioration is observed. In this algorism, better training convergence is shown and expresses the advantage in terms of training time especially for the unknown models than other cases such as fixed LR.

1. はじめに

畳み込みニューラルネットワーク(CNN)をはじめとした深層学習は，画像認識分野において圧倒的な認識精度を示している [1]. より高精度な認識性能獲得のためには，より深くより大規模なネットワークが期待される. このようなネットワーク学習を考慮した場合，学習に係る計算機の占有は長時間に及び，現在の深層学習実用化にとって大きな障壁となっている. これまで，深層学習を高速化する手法として，データ並列及びモデル並列という 2 つの概念をベースとした多数の並列手法が研究されてきた [2]. さらなる学習時間の短縮のためには，ニューラルネットワークによって自動的に決定されない，ハイパーパラメータの設計がその学習速度に大きな影響を及ぼす.

これまで，LR がその最終精度に与える影響が大きいことは報告されているが，多くの場合で実験的または経験的に LR が設計されていた [3]. また，未知のネットワークに対して，単一プロセスのみの学習で適切な LR パラメータを設計し十分な精度を達成することは困難であった. つまり，ハイパーパラメータの決定のために，複数回の学習を繰り返し試行する必要があった. これに対し，適応的な LR を決定するためのアルゴリズムが多数提案されている. 特に，Adam [4], AdaGrad [5], AdaDelta [3]等は，蓄積した過去の勾配の変化量を基準として LR を適応的に変化させることで勾配降下法の収束速度を改善している. 一方で，これらの適応的手法ではむしろハイパーパラメータが増える傾向にある. また，解くべき問題やネットワークに対してそれぞれ，向き不向きがあり，問題やネットワークに依存しない最適な方法は見つかっていない. 従って，未知のネットワークを考慮した際には，プロセス毎に様々な試行を施し，実験結果に基づいた最適点を選択する必要がある.

本稿では，通常の誤差逆伝搬で計算された勾配を用いた学習手法において，その更新量を調節する学習係数（LR）を対象とし，短時間の事前学習だけで未知のネットワーク構造に適した LR を選択する，短期事前学習（Short-Term Pre-Training: STPT）を用いたアルゴリズムを提案する. また，本手法を用いることで，未知のネットワークの性能を素早く評価できる.

連絡先：山田和樹，神戸大学大学院 システム情報学研究科 情報科学専攻 アーキテクチャ研究室，yamada.kazuki@cs28.cs.kobe-u.ac.jp

2. 短期事前学習（STPT）による適応的 LR 調節

2.1 アルゴリズム概要

提案アルゴリズムの概要図を図 1 に示す. 提案アルゴリズムの主な特徴は，複数の LR 候補を用いて学習を行う点と，1エポック以内の僅かなミニバッチの反復回数（イタレーション）のみで各 LR 候補の評価を行う，短期事前学習（STPT）を用いる点である. 提案アルゴリズムでは，1 つのエポックを Pre-train と Main-train の2つのフェーズに分けて学習を行う.

最初に，第 1 フェーズである Pre-train を行う. Pre-train では n 台のワーカにそれぞれ異なる LR 候補を分配し，各ワーカの LR と LR 初期値をそれぞれ掛け合わせた値を，実行的 LR として，学習を行う. これにより LR 初期値に対してある一定の幅を持たせた m 種類の実行的 LR を評価することができる. Pre-train 開始時の各ワーカの初期値は，LR 候補以外すべて同じである. また，m ($\leq$ n)個の LR の評価を想定し，各ワーカでそれぞれ独立した学習を行う. ここで最も重要な点は，Pre-train では α イタレーションのみ学習を行う点である. 尚，学習全体のイタレーション数を $\alpha + \beta = 1$ エポックとなるように設定する. この時，α は Pre-train のイタレーション数，β は Main-train のイタレーション数を意味する. 各ワーカはそれぞれ異なる LR を用いて，α イタレーションの学習を行い，その時点の精度を出力する. 次に，各実行的 LR の精度を比較し，最も精度が高かった実行的 LR を bestLR として選択する.

第2フェーズでは，Main-train を行う. Main-train では Pre-train において bestLR で学習したモデルを，n 台のワーカに分配し，実行的 LR に bestLR を用いた n 並列のデータ並列学習を β イタレーション行う.

以上が提案アルゴリズムにおける1エポック(α + βイタレーション)の学習フローである. Main-train 終了後のモデルを次のエポックの Pre-train の初期値として用いる. また実行的 LR は, 前のエポックの bestLR に各 LR 候補を掛け合わせた物を用いて, それぞれ学習を行う. 以上を繰り返すことで, 学習を進めていく.

2.2 例

用いる LR 候補群(LR set)を 3.0, 1.0, 0.5 として, 学習を行った場合の学習時の精度推移のグラフを図 2 に示す. 各エポックの初期段階にある3本の短い線は, Pre-train における各 LR 候補を用いた場合の精度を示し, 黒い線は bestLR を用いた, Main-train の精度推移を示している. 精度推移に注目すると, 各エポックの初期段階では精度が大きく上下するが, すぐに安定していることが分かる. これより, エポックの初期段階のみで実行的 LR の評価が可能と考え, STPT 終了後に実行的 LR の評価を行っている. これを用いることで, 複数の LR 候補の評価に必要なイタレーション数を削減することが可能である. 例として, 8つの LR 候補と8台のワーカを用いた実験において, 各 LR 候補を1エポック学習した場合と, 提案アルゴリズムの αを 200 とした場合を比較した図を図 3 に示す. この場合, 提案アルゴリズムでは 87.5%のイタレーション数を削減することができる.

3. 実験と結果

本実験では, 提案アルゴリズムの性能を評価するために画像認識を用いた実験を行った.

3.1 実装

本実験ではデータセットに ImageNet の ILSVRC2012 [6]で使用されたデータセットを使用した. このデータセットはサイズの違う 1000 カテゴリの一般物体の RGB 画像を, 学習データ 128 万枚, 検証データ 5 万枚, テストデータ 10 万枚を用意したものである. 本稿では画像のサイズを 256×256 に統一し, 実験を行う. またネットワークには ResNet50 [7]を用いた. ResNet は残差と共に学習を行い, 勾配消失問題を回避できる. そのため, ネットワークの層を増やすことが可能である. また, 最適化関数には Momentum SGD [8]を用いており, LR 候補はエポックごとに掛け合わせている. さらに, 提案アルゴリズムにおいては, 学習の初期段階に比較的小さい値の LR で学習を行う, Warmup [9]と呼ばれる手法を用いる. リファレンスとして ResNet の論文 [7]に従い, LR の初期値を 0.1 に設定し, 精度が飽和すると LR を 10 分の 1 にする手法を用いる. また, この場合に Warmup を用いると精度の劣化が見られたので, リファレンスに対して Warmup は用いない.

表 1 に実験で使用したパラメータを示す. 本実験では8並列, 1ノードの同期型データ並列を用いてソフトウエシミュレーションを行う. 提案アルゴリズムの疑似コードをアルゴリズム 1 に示す. また, アルゴリズム 1 に示している通り, Pre-train については, データ並列を使用してシミュレーションを行った. また, 実装のフレームワークとしては Chainer [10]を用いた.

3.2 Pre-train における αイタレーションの決定方法

本節では, Pre-train の実行イタレーション数である, αの決定方法について述べる. この実験ではバッチサイズを 512(=64 バッチ/ワーカ*8 ワーカ)に設定しているため, 128 万枚の学習データを全て学習するのに, 2,500 イタレーション必要である.

表 1 各パラメータ

Network	ResNet50
Number of workers	8
Batch size (per worker)	512 (64)
Training max epoch	40
Pre-train (α) iterations	200, 600, 1,250, 2,500
Weight initializations	He's initialization [11]
Momentum cofficient	0.9
Initial LR	0.1
LR set	表 2, 表 3
Warmup epoch	4
Warmup LR	0.005

従って αの最大値は2,500となる. また, 実装上の理由から αの最小値を200とする. よって αは 200～2,500 の範囲で決定することとする. ここで, αが小さいほど, 演算量は削減されるが, 学習が十分にできていない段階での LR の評価は, 学習の最終精度を悪化させる可能性がある. 逆に, αが大きいほど, 最終精度への悪影響は小さくなるが, 演算量が増えてしまう. そこで, 本実験では αの値が最終精度にどのように影響するかを調査する. 実験では αを 200, 625, 1,250, 2,500 の 4 つの値でシミュレーションを行い, 評価する. また, 本実験で用いた LR set を表 2 に示す.

図 4 は各 αを用いたときの最大精度を示している. αが 200 の時, 最も高い精度である, 0.639 を示した. 625, 1,250, 2,500 はそれぞれ 0.628, 0.622, 0.628 と, 比較的低い精度となった. また, αによる精度の変化幅は, 最大で 1.6%である. これは Pre-train におけるイタレーション数は最終精度に大きな影響を与えないことを示している. 従って本稿では αの値を 200 とした.

表 2 αを決定の際に用いる LR set

LR set	#1	#2	#3	#4	#5	#6	#7	#8
LRs	10.0	5.0	3.0	1.5	1.0	0.9	0.5	0.4

3.3 LR set の決定方法

提案アルゴリズムにおいて, 使用する LR 群である LR set の選択は重要である. この節では, LR set の決定方法について述べる. 表 3 に示す, LR 候補間の間隔の違う3種類の LR set, (a) Narrow set, (b) Middle set, そして (c) Wide set を用意し, 精度の収束性を比較する. 各 LR set の最大値と最小値をそれぞれ定め, その他の値は 1.0 を中心に等間隔になるように設定した. 他のパラメータは表 1 と同様である. 図 5 及び図 6 は(a),(b),および(c)を用いて, 40 エポック学習を行った際の精度及び LR の推移を示したグラフである. これによると,

(a) Narrow set: 最終精度は 65%. 最終精度は最も高いが, 25 エポック以降精度の減衰が見られる. 図 5 によると, 25 エポック以降は推移にばらつきがあることが分かる. これは LR 候補間の差が小さくなると, 精度の差が GPGPU の演算時に生じるノイズに負けてしまうためであると考えられる. 結果的にこのばらつきが精度に悪影響を与えてしまっている.

(b) Middle set: 最終精度は 63.4%. 最終精度は 2 番目だが, 学習が途中で止まっている.

(c) Wide set: 最終精度は 61.4%. 最終精度は最も悪く, 学習速度も遅いが, 40 エポック時点でも学習が進んでいる. 理由として, LR が小さくなりすぎているためである. しかし, LR 候補間の差が大きく, LR の評価を正しく行えるため, 学習後期でも学習が進んでいると考えられる.

以上の結果を基に，表 4 に示すスケジュールで LR set を組み合わせた．これを Mix と呼ぶ． Mix のシミュレーション結果の精度及び LR の推移を図 7，図 8 に示す．結果として，Mix は，最も高い精度であった(a)と比べて，1.2%向上した．(a)では 25 エポック以降に精度の減衰が見られたが，Mix ではこれを回避することができた．本スケジュールの様に LR set を組み合わせることで，各 LR set の欠点を補うことができる．また，Mix はリファレンスと比較して精度が 4.8%向上している．

以上より，STPT を用いた適応的学習係数調整手法では，従来手法と比較して，87.5%の演算量を削減し，4.8%の精度向上を達成した．

表 3 LR sets

Values	(a) Narrow set	(b) Middle set	(c) Wide set
#1	1.25	2.50	5.00
#2	1.17	2.00	3.67
#3	1.08	1.50	2.33
#4	1.00	1.00	1.00
#5	0.95	0.85	0.80
#6	0.90	0.70	0.60
#7	0.85	0.55	0.40
#8	0.80	0.40	0.20

表 4 Mix: LR set 変更スケジュール

Epoch	1–4	5–19	20–29	30–40
LR set	Warmup	(a) Narrow	(b) Middle	(c) Wide

4. 今後の研究

今後の研究では，提案アルゴリズムに Adam や AdaGrad 等他の手法を組み合わせる．また，LARS [9]に適応することで，巨大なバッチサイズの並列学習を行えると考える．

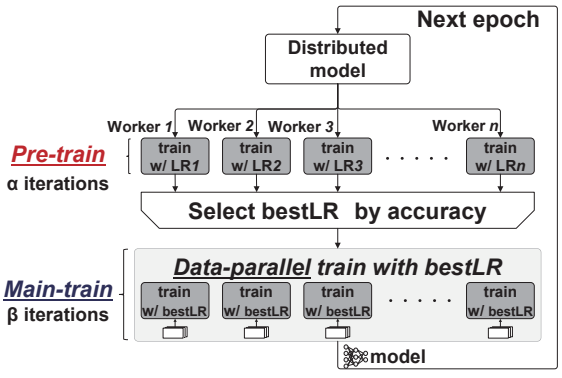

図 1　提案アルゴリズムの概要図

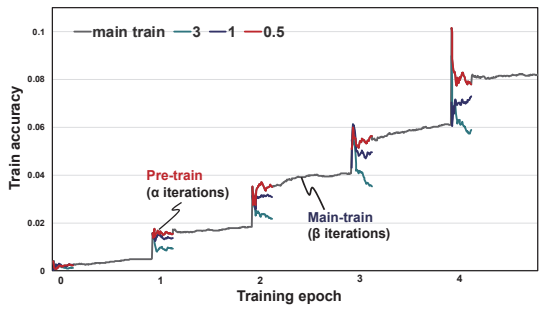

図 2　提案アルゴリズムを実行した際のコンセプト図

謝辞

この成果は、国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）の委託業務の結果得られたものです。

参考文献

[1] A. Krizhevsky, I. Sutskerver and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," Neural Information Processing Systems Conference, 2012.

[2] H. Mori, "A LAYER-BLOCK-WISE PIPELINE FOR MEMORY AND BANDWIDTH REDUCTION IN DISTRIBUTED DEEP LEARNING," MLSP, 2017.

[3] M. D.Zeoler, “AdaDelta: An Adaptive Learning Rate Method,” 2012.

[4] D. Kingma , J. Ba, “Adam A Method for Stochastic Optimization,” 2014.

[5] J. Duchi, E. Hazan , Y. Singer, “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization,” 2011.

[6] "IMAGENET," [Online]. Available: http://www.image-net.org/.

[7] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image recognition," Microsoft Research, 2015.

[8] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural Networks : The Official Journal of the International Neural Network Society,* vol. 12, no. 1, p. 145–151, 1999.

[9] P. Goyal, P. Dollar, R. Girshick , P. Noordhuis, “Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour,” 2017.

[10] P. Networks, "Chainer," [Online]. Available: https://chainer.org/.

[11] K. He, X. Zhang, S. Ren , J. Sun, “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification,” arXiv, 2015.

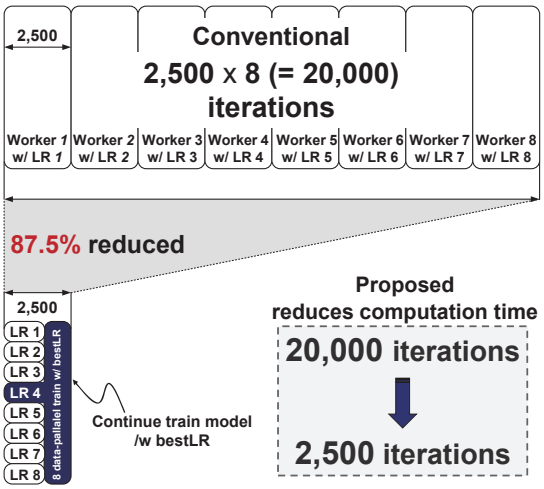

図 3　提案アルゴリズムと従来手法の比較

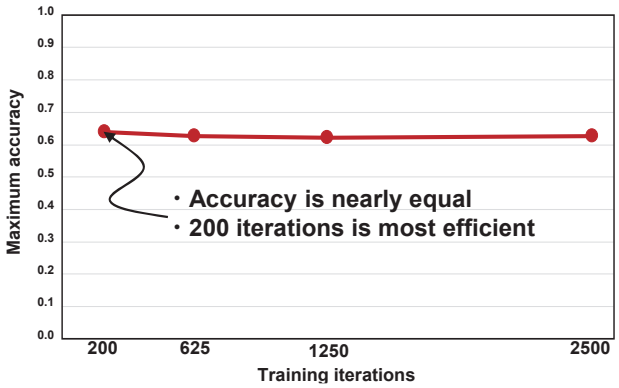

図 4　α が 200，625，1250，2500 の時の最大精度

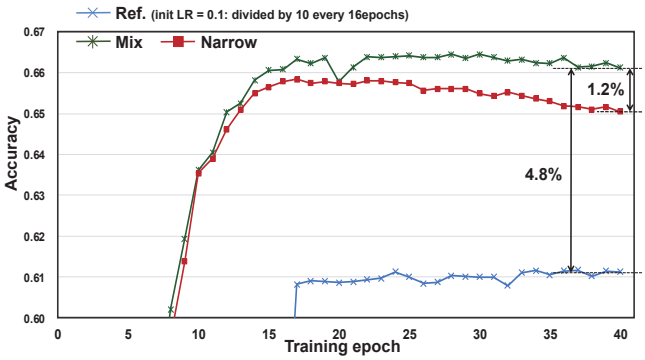

図 7　Mix，(a)Narrow set，Ref.の精度推移

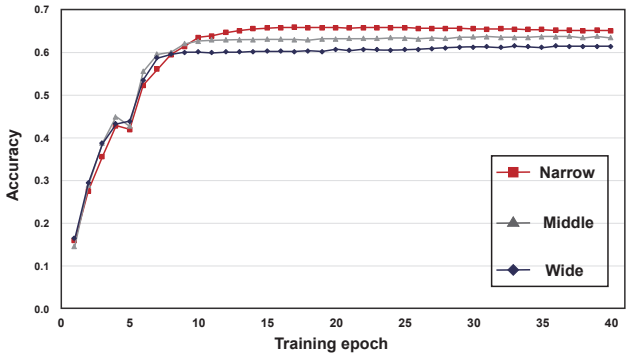

図 5　(a),(b),(c)を用いた際の LR 推移

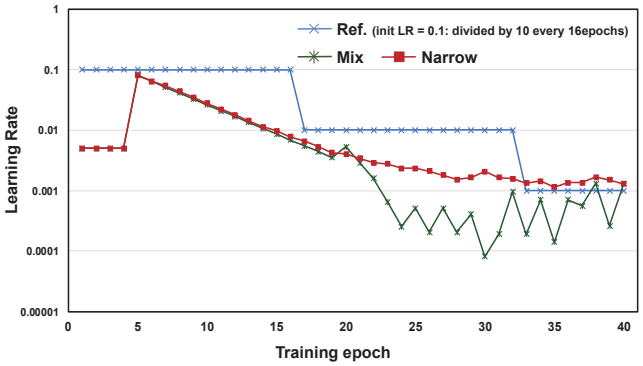

図 8　Mix，(a)Narrow，Ref.の LR 推移

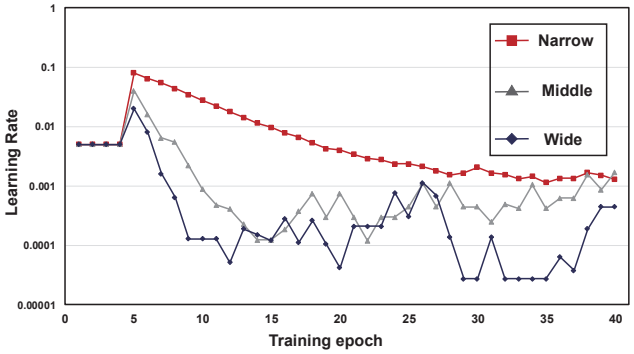

図 6　(a),(b),(c)を用いた際の LR 推移

アルゴリズム 1：提案アルゴリズムの実装

```
for epoch in range(max_epoch + warmup_epoch)
  if epoch < warmup_epoch
      run warmup train with warmup_LR using data parallel
  else
      #initialization
      best_accuracy = 0.0
      bestLR = 0.0
##-------------------------Pre-train-------------------------##
      for LR_index in range( len( LR_set ) )
          new_accuracy = run pre_train with
LR_set[ LR_index ] using data parallel
          if new_accuracy > best_accuracy
              bestLR = LR_set[ LR_index ]
```