Vol.2019-HPC-168 No.24
2019/3/7

Darknet53を用いたImageNetによる一般物体認識とYOLOv3を用いた物体検出の分散深層学習

西川 由理[1,2,a)]　佐藤 仁[1]　小澤 順[1]

概要：本稿では産業技術総合研究所の大規模 AI クラウド計算システム ABCI を用い，一般物体認識 Darknet53 および一般物体検出 YOLOv3 の分散深層学習による台数効果と学習効果について述べる．前者には大規模物体認識データセットである ImageNet を適用し，また後者には Darknet53 をプレトレーニングモデルとした Pascal VOC データセットによるファインチューニングを行い，NVIDIA Tesla V100 GPU を最大で 512 台用いて評価した．その結果，プレトレーニング，ファインチューニングのいずれにおいても，Goyal らの Linear Scaling 法に基づく分散深層学習が有効であることが確認された．

Yuri Nishikawa[1,2,a)]　Hitoshi Sato[1]　Jun Ozawa[1]

1. はじめに

近年，撮像された画像から，複数の物体の位置と大きさを自動的に抽出するため，深層学習ベースの一般物体検出手法が多く提案されている．特に YOLO は，単一の CNN で分類タスクと領域推定を行う single-shot な物体検出手法であり，高精度か高スループットであることから広く応用されている．さらに，2018 年には，学習モデルを 19 層から 53 層の CNN に変更して精度向上を図った YOLO version 3 (以後 YOLOv3)[1] が発表され，注目されている．

一方，深層学習における高速化のために，大規模な計算機資源を用いた分散深層学習に関する研究が広く行われている．分散深層学習では，一般に，並列数の増加に伴うバッチサイズ増大による汎化性能低下が課題と言われてきたが [2]，Goyal らの linear scaling 法が，汎化性能低下の解決手法として注目され [3]，画像認識タスクでその有効性が報告されている．特に，学習データに ImageNet を用い，学習モデル ResNet50 で物体認識を行うタスクでは，NVIDIA Tesla V100 GPU を 2,176 台用いた大規模並列分散環境により 3.7 分で学習が完了する [4] など，大幅な性能向上の事例が報告されている．さらに，物体検出手法の分散深層学習においては，YOLO version 2 (以後 YOLOv2)[5] に対して，Goyal らの手法の適用が良好なスケーラビリティを示すとの知見も発表されている [6]．

本研究では，計算機資源を活用した一般物体認識の学習の高速化を目指し，Goyal らの linear scaling 法による学習の性能評価について報告する．具体的には，まず ImageNet を用いて YOLOv3 用の物体認識モデルである Darknet53 をプレトレーニングし，その後，Pascal VOC (学習データ数約 16,000，分類数 20) を用いて YOLOv3 の物体検出モデルのファインチューニングを行う．計算機環境としては，産業技術総合研究所の AI 橋渡しクラウド ABCI[7][8] における NVIDIA Tesla V100 GPU を最大 512 台用い，GPU の台数を変化させたときの精度の推移とスケーラビリティを示す．

2. 一般物体検出 YOLOv3

2.1 Darknet53 と YOLOv3 の概要

YOLO は深層学習ベースの一般物体検出アルゴリズムの一つである．単一の CNN で分類と領域推定を同時に行う single-shot な手法であるため，コンシューマ向け GPU でリアルタイムに検出が行えることが利点である．さらに，2017 年には 19 層の畳み込み層を持つ Darknet19 に基づく YOLOv2[5]，2018 年には本稿で対象とする，53 層の Darknet53 で特徴抽出を行う YOLOv3[1] が公開された．YOLOv3 は v2 に比べ精度が高く，特に小さい物体の検出率が向上したとされる．

YOLOv3 の特徴抽出器である Darknet-53 の構造を**図 1** に示す．本ネットワークは，ResNet 構造 [9] を参考に，従

1　国立研究開発法人 産業技術総合研究所
2　パナソニック株式会社
a)　nishikawa.yuri@aist.go.jp

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

図 1: Darknet53 のアーキテクチャ

来のアーキテクチャに shortcut path を加えた Residual Block を有する．これにより，直前の層への入力を参照した残差関数の学習が可能となり，層数が増加しても学習が可能になるという特色がある．

その他の YOLOv2 と v3 の違いとしては，(1) YOLOv3 ネットワークにおける FPN(Feature Pyramid Network)[10] の採用，(2) 最終の Softmax 層の禁止などが挙げられる．(1) は，YOLOv2 で課題であった小さい物体の検出率向上の施策である．一般に，特徴マップの大きさは層の深さと共に小さくなるため，小さい物体の検出を異なる層で行う feature map の考え方を取り入れている．Feature map 自体は，SSD[11] などの検出器にも採用されるが，SSD では大きい物体を認識する際に高次の層の学習結果が反映できないという課題があった．Feature Pyramid Network は，その課題を解決するものであり，低次の層でも高次の層の特徴をアップサンプリングすることで，様々なスケールの物体検出率が向上した．YOLOv3 では，3 つのスケールに対応した FPN 構造を有する．

(2) は，多重分類に対応するための施策で，YOLOv3 では Softmax の代わりに 1 層の畳み込み層とロジスティック関数を用いる．Softmax は，入力画像を 1 クラスに分類することを前提としているが，クラス数が増え，例えば woman と person のように，分類に包含関係がある場合には適していない．ロジスティック関数により，マルチクラスの分類が可能となっている．

2.2 損失関数

YOLOv3 では，入力画像を $S \times S$ のグリッドに区切り，グリッド毎に，あるアスペクト比を持つ B 個の矩形領域 (anchor box) の中心座標 (x, y) および幅と高さのスケール (w, h)，そして矩形内に物体が存在する確率 (confidence) を予測する．さらに，各矩形に何らかの物体が存在するとき，その物体が属するクラスを示す事後確率も予測する．クラス予測に用いられるのが Darknet53 である．また YOLOv3 では，YOLOv2 同様に，矩形領域，物体の存在確率，クラスの事後確率の予測を一つの損失関数 (loss function) に統合する．

2.3 学習手順

YOLOv3 の学習手順は，[1] によれば YOLOv2[5] に準拠している．プレトレーニングでは，ImageNet の学習データを 224×224 の画像サイズに変換し，Darknet53 により 160 epoch 学習する．学習率の初期値は 0.1, polynomial rate decay は 10^{-4}, weight decay は 0.0005，momentum は 0.9 に設定する．この際に，random crop, rotation, color shift (hue, saturation, exposure) 等のデータオーグメンテーションも行う．次に，画像サイズを 4 倍の 448×448 に設定し，学習率の初期値を 10^{-3}，その他のハイパーパラメータは同一のまま，10 epoch だけファインチューニングする．

ImageNet による Darknet53 の学習後，最初の 52 層分の重みを抽出し，YOLOv3 モデルに適用する．これにより，物体検出用の学習データで，ファインチューニングが可能になる．この際には，Darknet53 と同様にデータオーグメンテーションすると共に，入力画像サイズを変更して学習する multi-scale training も行う．学習率の初期値は 10^{-3} とし，60 epoch と 90 epoch で段階的に 1/10 に設定する．

3. 画像分類の分散深層学習における関連研究

一般に深層学習では，学習データを「ミニバッチ」と呼ばれる単位に分割して学習モデルの更新を行う．分散深層学習では，プロセスごとにミニバッチ単位の学習データを割り当てるため，一度の並列計算における総バッチサイズ (以後，バッチサイズと表記) は，並列プロセス数とミニバッチサイズに比例する．一方で並列数が増え，バッチサイズが大きくなるとすると，汎化性能が低下することが報告され [2]，分散深層学習の課題とされてきた．

解決策として Goyal らが提示した linear scaling 法は [3]，バッチサイズと学習率を比例させることで，汎化性能を保てるという経験則であり，ImageNet/ResNet50 に対し，256 台の GPU(バッチサイズ 8,192) で学習を 1 時間で完了できることを示した．Linear scaling 法の妥当性は，Smith

らにより数理的にも示され [12][13]，以後 Goyal らの手法を基本とした改良により，従来は約 1 週間を要した ImageNet の学習時間の記録が更新されている [14][15][16].

2018 年 11 月には，2,176 台の NVIDIA Tesla V100 GPU を，論理的に 2 次元トーラス状に接続し，縦横方向が等遅延で通信できるクラスタ構成を採用することで AllReduce 処理を高速化し，学習時間 3.7 分，精度 75.03% の記録が報告された [4]．さらに同時期には，Google TPUv3 Pod チップを 1,024 個用い，複数プロセス単位での batch normalization と入力パイプライン最適化により，学習時間 2.2 分，精度 76.3%での訓練の成功事例も報告されている [17].

4. YOLOv3 の分散深層学習

4.1 ImageNet/Darknet53 のプレトレーニング

まず，ImageNet データセットを用い，Darknet53 の分散深層学習を行う．前述のとおり，分散深層学習では学習率が重要なハイパーパラメータである．今回は Goyal らの linear scaling 法に基づき，非分散時の学習率が $\eta(=0.1)$，ミニバッチサイズが $n(=32)$，GPU 数が k のとき，$kn/256\times\eta$ となるように設定する．ただし，ネットワークの重みが大きく変化する学習開始直後は，徐々に学習率を増加させる gradual warmup を適用する．具体的には，最初の 1 epoch の学習率を η とし，5 epoch 程度をかけて $kn/256\times\eta$ に到達するよう，線形に学習率を増加させる．

また，学習が進むに従って学習率を単調減少させる，polynomial learning rate decay を適用することとし，本稿では 2 乗の polynomial decay ([5] では 4 乗) を採用する．また epoch 数は 200 とする．その他のハイパーパラメータ (weight decay 等) は [5] と同様とする．

4.2 Pascal VOC/YOLOv3 の学習

ImageNet/Darknet53 の学習モデルを用い，YOLOv3 ネットワークのファインチューニングを行う．非分散時の学習率を $\eta=0.001$，ミニバッチサイズを $n=16$ とし，GPU 数が k のときの学習率が $kn\times\eta$ となるように設定する．またここでも，最初の 5 epoch の学習率を徐々に増加させる gradual warmup を適用する．なお，Darknet53 では polynomial decay により学習率を減衰させたが，YOLOv3 では [5] と同様に，学習率を 60, 90epoch で 1/10 に設定した．Epoch 数は 100 に設定し，その他のハイパーパラメータ (weight decay 等) も，[5] と同様とする．

4.3 画像検出用データセット

本研究では，画像検出用データセットとして，Pascal Visual Object Classes (VOC)[18][19] を用いる．コンピュータビジョン分野における画像検出用ベンチマークデータの一つであり，1 枚の写真画像に対し，人物，犬，車など 20 クラスの対象物が映っている．正解データには，対象物の位置と大きさを表す矩形領域と，そのクラス ID がアノテーションされている．1 枚の画像中に含まれる矩形数は画像により異なる．なお Pascal VOC コンテストのデータセットは，2005 年から 2012 年開催分が公開され [20]，本稿では，2007 年データセットの画像から 5,011 枚，2012 年から 11,540 枚の合計 16,551 枚を学習用データとする．また YOLOv3 では，Pascal VOC の学習用画像を 608 x 608 ピクセルにリサイズしてから学習する．

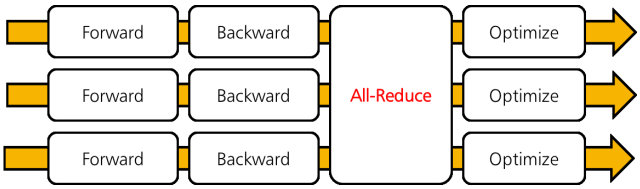


図 2: ChainerMN による分散深層学習

表 1: ABCI の計算ノードのスペック

CPU	Intel Xeon Gold 6148 (27.5M Cache, 2.40 GHz, 20 core) × 2
GPU	NVIDIA Tesla V100 SXM2 × 4
Mem	384 GiB
SSD	1.6TB NVMe SSD × 1

4.4 分散深層学習フレームワーク

分散深層学習フレームワークには ChainerMN[14] を用いる．[21] の先行研究と同様，データセットを分散して学習モデルを計算する「データ並列」の手法を採用する．概要を**図 2** に示す．学習過程では，1) 予測を行ってその誤差を計算 (Forward 処理) し，2) 誤差を減らす方向の勾配を計算 (Backward 処理) し，勾配を用いて学習モデルを更新する．ChainerMN は図 2 に示すように，1) 2) を行った後，複数の計算資源で分散して計算した勾配から平均を求めて配り直す AllReduce 処理を行う．

5. 評価

5.1 実験環境

分散深層学習の評価は，産総研の AI 橋渡しクラウド ABCI 上で行った．**表 1** に計算ノード 1 台のスペックを示す．ABCI は 1 ノードあたり 4 台の GPU を搭載する．また計算ノード間は EDR Infiniband により，Full-bisection Fat Tree 構成で接続される．ただしラックを跨ぐ計算ノード間は Full-bisection の帯域が 1/3 となるように接続されている．

計算ノードの OS は CentOS 7.4, Linux のカーネルは v3.10.0 である．またソフトウェアについて，GPU に対しては，CUDA Toolkit v9.2.148.1，CuDNN v7.3.1 を使用し，GPU 間の集団通信を行うため NCCL v2.3.5-2 と OpenMPI v2.1.5 を使用した．Python およびライブラリ

は、Python v.3.6.5, Chainer v5.0.0，CuPy v5.0.0，ChainerCV v0.11.0, mpi4py v3.0.0 を使用した．ChainerMN は，Chainer v5.0.0 に含まれるものを用いた．

その他，次の ChainerMN の設定を適用した．まず複数 GPU 間通信を行う communicator には `pure_nccl` を指定した．また，cuDNN による batch normalization の高速実装を適用するため，`cudnn_fast_batch_normalization` を使用した．なお，本研究の計算には FP32 を使用したが，AllReduce には FP16 を利用して通信時間を短縮する ChainerMN の機能を用いた．

5.2 ImageNet/Darknet53 の評価

5.2.1 汎化性能

まず，4.1 節で述べた Goyal らの手法を適用し，異なるバッチサイズで ImageNet を Darknet53 モデルで学習させた時の，検証 (validation) データを用いた epoch 数に対する精度の変化を**図 3** に示す．ミニバッチサイズ $n = 32$, GPU 台数 $k = 64$，128，256, 512 である．グラフの青線は学習データを用いた時，オレンジ色の線は検証データを用いた時の画像認識の正解率 (validation accuracy) を表す．

図 3 が示すように，GPU 256 台の時の正解率は 77.2% と [1] と同程度の汎化性能を示した．この時のバッチサイズは 8192 だったが，Goyal らもバッチサイズ 8192 までの汎化性能の維持について報告しており，同様の結果が得られることが分かった．一方，GPU 512 台の学習曲線を見ると，検証データを用いた精度が，75 epoch までに数回低下したのち後半で学習が進行する現象が見られ，256 台までと比べやや低下したが，75.4%の正解率を確認した．

5.2.2 Darknet19 との汎化性能比較

次に YOLOv3 の特徴抽出器である Darknet53 における ImageNet の汎化性能を，また YOLOv2 の Darknet19 と比較した結果を**図 4** に示す．共に，16 ノード 64GPU での分散深層学習の結果である．ハイパーパラメータは前節と同様に設定した．

[5] によれば，Darknet19 の正解率は 74.1%であり，**図 4a** でも同等であることを確認した．また，Darknet53 が Darknet19 と比べ高精度であること，分散深層学習によって先行研究と同様の汎化性能が得られることも示された．

5.2.3 スケーラビリティ

図 3 を取得したときの実行時間を計測し，スケーラビリティを評価した．結果を**図 5** に示す．横軸は GPU の台数，縦軸は 1 台の GPU を用いた際の実行時間を基準としたときのスケーラビリティを表す．概ねスケーラブルな処理性能を示すことを確認し，GPU256 台で約 109.5 倍, 512 台で約 256 倍の高速化を達成した．

表 2: Pascal VOC/YOLOv2 と YOLOv3 の精度

GPU 台数	8	16	32	64	96	128
バッチサイズ	128	256	512	1024	1536	2048
mAP(YOLOv2)	78.1	78.1	78.0	75.5	0.3	-
mAP(YOLOv3)	78.7	78.7	78.2	77.3	76.3	65.8

5.3 Pascal VOC/YOLOv3 の評価

5.3.1 汎化性能

次に，Pascal VOC を用いて，YOLOv3 による画像検出モデルを学習させたときの汎化性能を**図 6** に示す．さらに，Pascal VOC の検証データを用いて計測した画像検出精度を表す mAP(mean Average Precision) の比較結果を**表 2** に示す．なお，Darknet19 と YOLOv2 での学習により得られた mAP[6] も併せて示す．なお YOLOv2 でも，YOLOv3 と同様の学習率とバッチサイズを設定し，学習用画像にも 608 x 608 ピクセルにリサイズしたものを用いた．

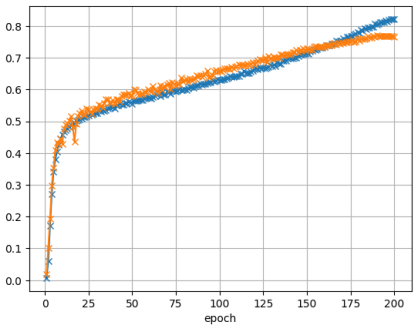
図 6 の学習誤差曲線が示すように，Goyal らの手法によって学習が進んだ．また表 2 を見るといずれの GPU 数でも YOLOv2 より良好な精度を得られた．さらに YOLOv2 では GPU 96 台で精度が大きく低下したのに対し，YOLOv3 では 2.4 の mAP 低下に留まった．YOLOv2 と比べ，YOLOv3 での層数の増加による汎化性能の高さが，分散深層学習にも寄与したと考えられる．なお YOLOv3 では，GPU 8 台での mAP 78.7 に対し，96 台で 76.3, 128 台では 65.8 となり，96 台と 128 台の間で汎化性能が低下した．なお，96 台，128 台の GPU を用いたときのバッチサイズはそれぞれ 1536，2048 であり，それぞれ学習データ数の約 9.3%，12.4%に相当する．

5.3.2 スケーラビリティ

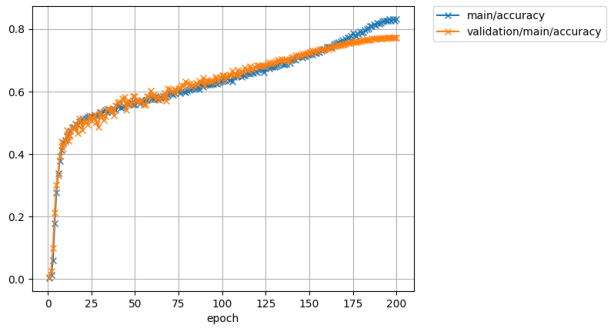
図 6 を取得したときの実行時間から求めた台数効果を評価した．結果を**図 7** に示す．横軸は GPU の台数，縦軸は 1 台の GPU を用いた際の実行時間を基準としたときのスケーラビリティを表す．概ねスケーラブルな処理性能を確認し，GPU 96 台で約 64.4 倍となることを確認した．

6. まとめ

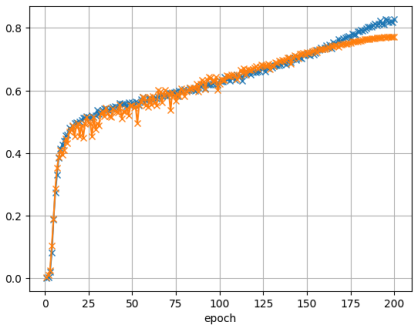
本稿では，産業技術総合研究所の AI 橋渡しクラウド ABCI の GPU を最大 512 台用い，ImageNet/Darknet53 による一般物体認識と，Pascal VOC/YOLOv3 による一般物体検出の分散深層学習を行った．Goyal らの linear scaling 法を適用し，非分散時と比較した学習効果と台数効果を評価した．その結果，Darknet53 では GPU 256 台でも非分散時と同等の正解率である 77.2%を達成し，かつ約 109.5 倍の高速化を確認した．また YOLOv3 によるファインチューニングでは，GPU 96 台において約 2.4% の精度低下で 64.4 倍の高速化を確認し，それ以上の GPU 数では精度が低下することが確認された．この時のミニバッチサイズと並列数の積は，総データサイズの約 9%であった．



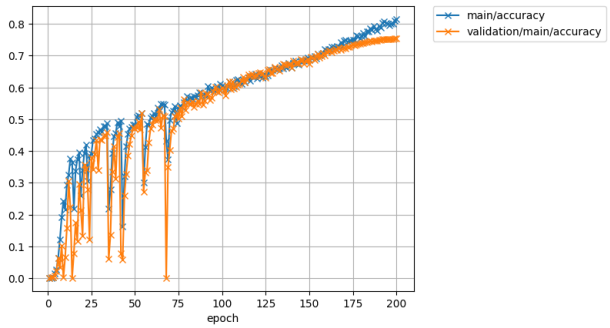
(a) 16 ノード, $k = 64$
正解率: 76.7%



(b) 32 ノード, $k = 128$
正解率: 77.3%

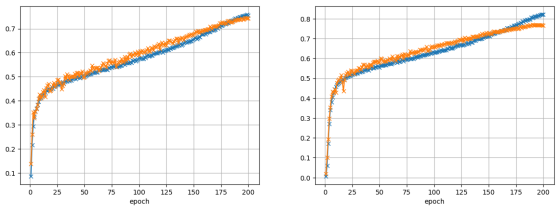


(c) 64 ノード, $k = 256$
正解率: 77.2%



(d) 128 ノード, $k = 512$
正解率: 75.4%

図 3: ImageNet/Darknet53 の学習曲線



(a) ImageNet/Darknet19　(b) ImageNet/Darknet53

図 4: Darknet19 と Darknet53 の比較 ($k = 64$)

以上により，Goyal らの Linear Scaling 法は，実装が容易でありながら，一定のバッチサイズ以下であれば，異なる学習モデル，比較的小規模な学習データに対しても適用可能な手法であることが示された．今後は FP16 への対応や，異なる学習率を設定する LARS (Layer-wise Adaptive Rate Scaling)[15] に対応した実装と評価を行いたい．

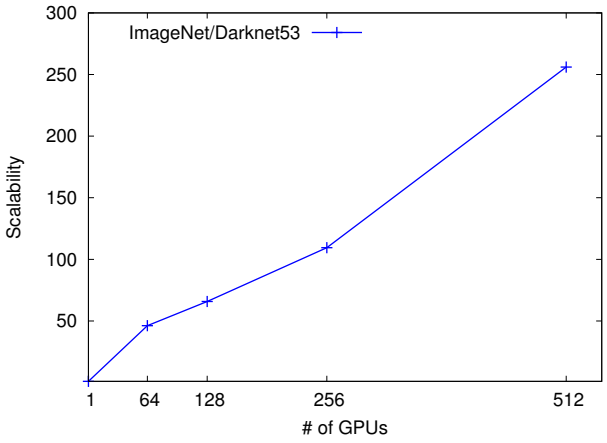


図 5: ImageNet/Darknet53 のスケーラビリティ

参考文献

[1] Redmon.J and Farhadi, A.: YOLOv3: An Incremental Improvement, *CoRR*, Vol. abs/1804.02767 (online), available from ⟨http://arxiv.org/abs/1804.02767⟩ (2018).

[2] Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M. and Tang, P. T.: On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima, *International Conference on Learning Representations*, (online), available from ⟨https://openreview.net/forum?id=H1oyRlYgg⟩ (2017).

[3] Goyal, P., Dollár, P., Girshick, R. B., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y. and He, K.: Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, *CoRR*, Vol. abs/1706.02677 (online), available from ⟨http://arxiv.org/abs/1706.02677⟩ (2017).

[4] Mikami, H., Suganuma, H., U.-Chupala, P., Tanaka, Y. and Kageyama, Y.: ImageNet/ResNet-50 Training in 224 Seconds, *CoRR*, Vol. abs/1811.05233 (online), available from ⟨https://arxiv.org/abs/1811.05233⟩ (2018).

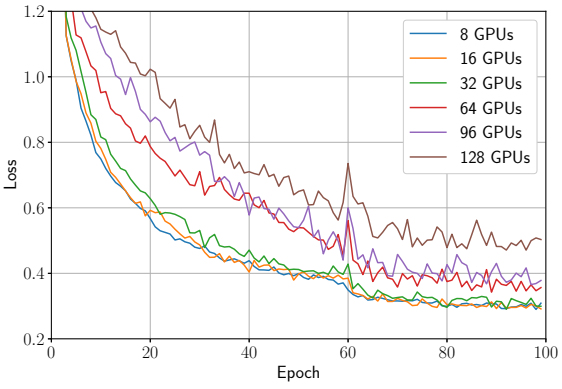


図 6: Pascal VOC/YOLOv3 の学習誤差曲線

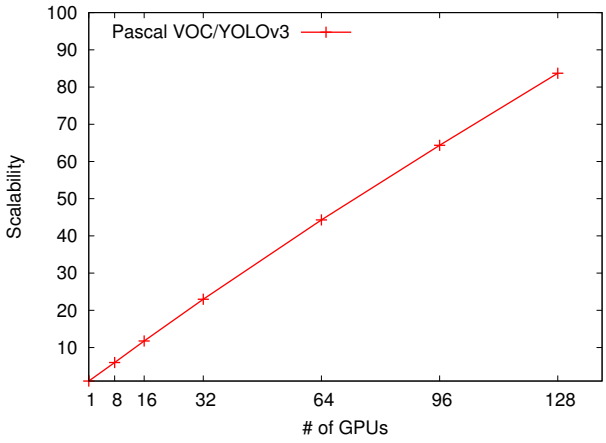


図 7: Pascal VOC/YOLOv3 のスケーラビリティ

[5] Redmon, J. and Farhadi, A.: YOLO9000: Better, Faster, Stronger, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6517–6525 (2017).

[6] 西川由理，佐藤　仁，小澤　順：一般物体検出 YOLO の分散深層学習による性能評価，研究報告ハイパフォーマンスコンピューティング（HPC），Vol. 2018-HPC-166(12), No. 1-6 (2018).

[7] 小川宏高，松岡　聡，佐藤　仁，高野了成，滝澤真一郎，谷村勇輔，三浦信一，関口智嗣：AI 橋渡しクラウド–AI Bridging Cloud Infrastructure (ABCI)–の構想，情報処理学会研究報告，Vol. 2017-HPC-160, No. 28, pp. 1–7 (2017).

[8] ABCI: https://abci.ai/.

[9] He, K., Zhang, X., Ren, S. and Sun, J.: Deep Residual Learning for Image Recognition, *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778 (online), DOI: 10.1109/CVPR.2016.90 (2016).

[10] Lin, T., Dollár, P., Girshick, R. B., He, K., Hariharan, B. and Belongie, S. J.: Feature Pyramid Networks for Object Detection, *CoRR*, Vol. abs/1612.03144 (online), available from ⟨http://arxiv.org/abs/1612.03144⟩ (2016).

[11] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. and Berg, A. C.: SSD: Single shot multibox detector, *European conference on computer vision*, Springer, pp. 21–37 (2016).

[12] Smith, S. L., Kindermans, P.-J., Ying, C. and Le, Q. V.: Don't Decay the Learning Rate, Increase the Batch Size, *International Conference on Learning Representations*, (online), available from ⟨https://openreview.net/forum?id=B1Yy1BxCZ⟩ (2018).

[13] Smith, S. L. and Le, Q. V.: A Bayesian Perspective on Generalization and Stochastic Gradient Descent, *International Conference on Learning Representations*, (online), available from ⟨https://openreview.net/forum?id=BJij4yg0Z⟩ (2018).

[14] Akiba, T., Suzuki, S. and Fukuda, K.: Extremely Large Minibatch SGD: Training ResNet-50 on ImageNet in 15 Minutes, *CoRR*, Vol. abs/1711.04325 (online), available from ⟨http://arxiv.org/abs/1711.04325⟩ (2017).

[15] You, Y., Zhang, Z., Hsieh, C.-J., Demmel, J. and Keutzer, K.: ImageNet Training in Minutes, *Proceedings of the 47th International Conference on Parallel Processing*, ICPP 2018, pp. 1:1–1:10 (online), DOI: 10.1145/3225058.3225069 (2018).

[16] Jia, X., Song, S., He, W., Wang, Y., Rong, H., Zhou, F., Xie, L., Guo, Z., Yang, Y., Yu, L., Chen, T., Hu, G., Shi, S. and Chu, X.: Highly Scalable Deep Learning Training System with Mixed-Precision: Training ImageNet in Four Minutes, *CoRR*, Vol. abs/1807.11205 (online), available from ⟨http://arxiv.org/abs/1807.11205⟩ (2018).

[17] Ying, C., Kumar, S., Chen, D., Wang, T. and Cheng, Y.: Image Classification at Supercomputer Scale, *CoRR*, Vol. abs/1811.06992 (online), available from ⟨http://arxiv.org/abs/1811.06992⟩ (2018).

[18] Everingham, M., Gool, L., Williams, C. K., Winn, J. and Zisserman, A.: The Pascal Visual Object Classes (VOC) Challenge, *Int. J. Comput. Vision*, Vol. 88, No. 2, pp. 303–338 (online), DOI: 10.1007/s11263-009-0275-4 (2010).

[19] Everingham, M., Eslami, S. M. A., Van Gool, L., Williams, C. K. I., Winn, J. and Zisserman, A.: The Pascal Visual Object Classes Challenge: A Retrospective, *International Journal of Computer Vision*, Vol. 111, No. 1, pp. 98–136 (2015).

[20] The PASCAL Visual Object Classes Homepage: http://host.robots.ox.ac.uk/pascal/VOC/.

[21] 佐藤　仁，西川由理，小澤　順：多人数追跡のための分散深層学習による高精度な検出にむけて，人工知能学会全国大会論文集，Vol. JSAI2018 (2018).