

外乱のある環境での分散深層学習の性能評価

山梨祥平[†] 大島聡史[‡] 永井亨[‡] 片桐孝洋[‡]

名古屋大学 工学部電気電子・情報工学科[†] 名古屋大学 情報基盤センター[‡]

1. はじめに

深層学習は従来の機械学習手法と比べて高い認識精度を示したことで近年注目を集めている。現状、深層学習で十分な認識精度を得るためには、大量のデータを用いた訓練処理を長時間にわたって行う必要があるため、GPU クラウドなどを用いて複数の GPU に計算を分散させることで学習を高速化させる分散深層学習が盛んに行われている。しかし、クラウドは基本的に利用者によって共有されるため、利用者の想定しない通信が外乱となって利用者の通信を阻害する恐れがある。また、半導体の製造ばらつきや冷却具合の差により性能に個体差が生じることがある。同期型データ並列の分散深層学習ではワーカーごとに学習させた勾配を同期処理によって集約しており AllReduce 通信が頻繁に行われるため、クラウドでの外乱の影響を受けやすいと考えられる。そこで本研究では分散深層学習の学習時間に通信外乱がもたらす影響を調べるために、NCCL ベンチマークを通信外乱と見立てて CNN を実行し、その学習時間と AllReduce 通信の通信時間について評価を行った。

2. MirroredStrategy

本研究ではディープラーニングの学習を複数の GPU に分散させるために、TensorFlow2.0 で提供されている MirroredStrategy [1] を用いる。MirroredStrategy とは分散学習を行う TensorFlow の API である `tf.Distribute.Strategy` のうち、1 台のマシン上での複数 GPU を用いた同期型データ並列を実現するクラスである。MirroredStrategy では GPU デバイス全てにモデルを複製してそれぞれ学習を行い、同一の更新 (AllReduce 処理) によって互いに同期を保っている。MirroredStrategy ではこの同期通信に複数の通信アルゴリズムを選択できるが、デフォルトでは NVIDIA 社が提供する

集団通信用ライブラリである NVIDIA Collective Communications Library (NCCL) を用いている。そのため、今回の実験でも AllReduce 通信に NCCL を選択した。

3. NCCL Tests

NCCL Tests [2] は NCCL の性能と精度を測定するベンチマークである。NCCL tests では NCCL が提供する各種集団通信のメッセージサイズごとの平均動作時間、アルゴリズム帯域幅、バス帯域幅を測定する。TensorFlow の MirroredStrategy では集団通信に AllReduce が行われているため今回の実験では `ncclAllReduce` のベンチマークを使用した。また実験では通信外乱として一定の間隔で繰り返し通信が行われる状況を再現するために、NCCL Tests のソースコードを改良したものをを用いた。具体的には NCCL Tests の AllReduce に関するファイルである `all_reduce.cu` の主要アルゴリズム部分を for ループで囲うことでベンチマークが繰り返し実行されるようにした。また繰り返されるベンチマーク処理の間隔を制御するために、処理が開始する時刻 t_1 を測定して、時刻 t_1 から指定時間が経過するまで次のベンチマーク処理が行われないように実装を追加した。今回の実験では時刻 t_1 から $1.0[s]$ が経過するまで次のベンチマーク処理が行われないようにした。

4. 性能評価

4.1 実験方法

本実験ではデータセットとして Fashion-MNIST, ネットワークとして CNN を使用した。Fashion-MNIST は 10 クラスに分類された 60000 枚の訓練画像と 10000 枚のテスト画像からなるデータセットであり、各画像は 28×28 ピクセルのグレースケール画像である。本実験の CNN の実装は 2 層の畳み込み層と 2 層の MAX プーリング層, 1 層の全結合層と出力層からなる。実験には上記のデータセットとネットワークを用いた学習を MirroredStrategy によって分散学習させる TensorFlow チュートリアルのサンプルプログラムを使用した。また通信外乱として用意した

Performance evaluation of distributed deep learning in an environment with disturbance

[†] Shohei Yamanashi, Electrical and Electronic Engineering and Information Engineering, Nagoya University

[‡] Satoshi Ohshima, Toru Nagai, Takahiro Katagiri, Information Technology Center, Nagoya University

NCCL Tests は正確には通信だけではなくわずかな計算も行っているため、その影響も考える必要がある。そこで通信の影響のみを調査するために NCCL Tests 内で AllReduce 通信を呼び出す `ncclallreduce()` をコメントアウトしたものを用意した (以降 `ncclallreduce()` をコメントアウトした NCCL Tests を計算外乱, NCCL Tests を通信外乱とする)。これにより、計算外乱を走らせた状態で CNN を実行した場合と通信外乱を走らせた状態で CNN を実行した場合を比較することで通信の影響を調査できると考えた。また本実験では AllReduce 通信の計測に NVIDIA 社が提供する NVIDIA Visual Profiler を用いた。

4.2 実験環境

実験には名古屋大学情報基盤センターで試験的に運用されている GPU サーバ `sx40` を使用した。表 1 に評価環境を示す。

表 1 評価環境

CPU	2 × Intel Xeon Gold 5122 (3.6GHz, 4 コア 8 スレッド, 105W TDP)
GPU	4 × NVIDIA Tesla V100 SXM2 (NVLink 対応, 300W TDP)
メモリ	384GB (12x 2Rx4 DDR4-2666 32GB)
ノード数	1 ノード
TensorFlow	2.0.0
CUDA	10.0
NCCL	2.5.6

4.3 実験結果

評価環境において、外乱がない場合と計算外乱がある場合と通信外乱がある場合で CNN を学習させて評価を行った。なお、今回の実験では 10 回の試行を行ってその平均をグラフの値とした。

図 1 は CNN 全体の学習時間による比較である。計算外乱では `ncclAllReduce` を呼び出していないのに対して、通信外乱では `ncclAllReduce` を呼び出している。そのため、計算外乱がある場合と通信外乱がある場合の CNN の学習時間の差が `ncclAllReduce` によるものと考えられる。したがって図 1 から、外乱の AllReduce 通信の影響で CNN の学習時間は約 4.5 秒増大したと考えられる。

次に、GPU 毎に 1 回の AllReduce 通信にかかった平均通信時間を図 2 に示す。図 2 についても図 1 と同様に、計算外乱の時間と通信外乱の時間の差が `ncclAllReduce` の影響による差だと考えると、外乱がない状態での CNN の AllReduce 通信の平均通信時間と比べて通信時間は GPU0 では約 9.5 倍、GPU1 では約 2.5 倍、GPU2 では約 1.9 倍、GPU3 では約 2.1 倍になった。

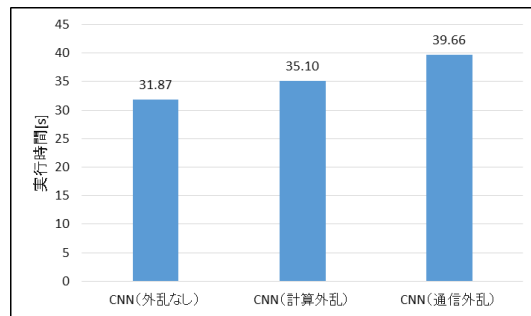


図 1 CNN の学習時間

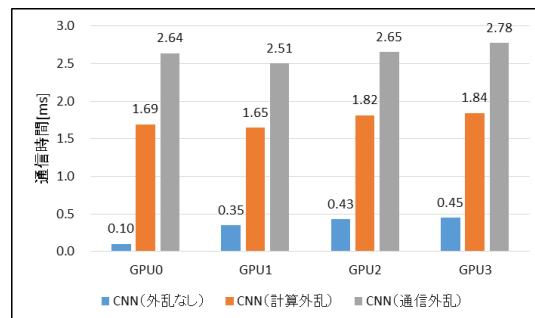


図 2 GPU 毎の 1 回の AllReduce 通信における平均通信時間

5. まとめ

NCCL ベンチマークを通信外乱として同期型データ並列の分散深層学習の実行時間を測定した結果、外乱がない場合と比べて学習時間、AllReduce 通信時間ともに増大することが確認できた。この結果から、利用者の想定しない外乱が発生するクラウドのような環境で分散深層学習の高速化を目指す場合、外乱の影響も考慮に入れる必要があると思われる。ただし今回の実験では外乱である NCCL ベンチマークの通信頻度を 1.0[s] 毎と定めたため、実際のクラウドでどの程度の通信外乱が発生しているのかを調査し、その通信頻度で検証する必要がある。また、今回の評価環境は 1 ノードであったためノード間通信が行われていない。ノード間通信を要する大規模な環境での検証が必要である。

謝辞

本研究は JSPS 科研費 JP18K19782 の助成を受けたものです。

参考文献

- [1] TensorFlow: `tf.distribute.MirroredStrategy`, https://www.tensorflow.org/api_docs/python/tf/distribute/MirroredStrategy?version=stable (accessed 2019/12/23)
- [2] NVIDIA: NCCL Tests, <https://github.com/NVIDIA/nccl-tests> (accessed 2019/12/23)