

DeepL Proに登録して、プレゼン資料を編集しましょう  
詳しくは[www.DeepL.com/pro](https://www.deepl.com/pro?cta=edit-document)をご覧ください。

通信効率の良い分散学習の実現

Lazily Aggregated Quantized Gradients

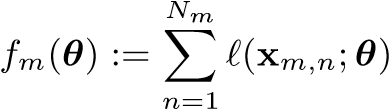
|  |  |
| --- | --- |
| Jun Sun†さん  浙江大学  Hangzhou, China 310027 sunjun16sj@gmail.com | Tianyi Chen†  レンセラー・ポリテクニック・インスティテュート（Rensselaer Polytechnic Institute  Troy, New York 12180 chent18@rpi.edu |
| ゲオルギオス・B・ジアンナキス  ミネソタ大学ツインシティー校  Minneapolis, MN 55455 georgios@umn.edu | Zaiyue Yang  南部科学技術大学（Southern U.S. of Science and Technology  Shenzhen, China 518055 yangzy3@sustc.edu.cn |

アブストラクト

本論文では、適応的に勾配通信を圧縮する、分散機械学習のための新しい集約された勾配アプローチを開発した。これは、計算された勾配をまず量子*化し、*次に古くなった勾配を再利用することで、情報量の少ない量子化された勾配の通信*をスキップ*するというものである。量子化とスキップの結果、ワーカー・サーバー間の通信が「怠惰」になり、Lazily Aggregated Quantized gradient（以後、LAQと略す）という用語が正当化されます。我々のLAQは、送信*ビット数と*通信*ラウンド*数の両方の通信オーバーヘッドを大幅に削減しながら、強凸の場合の勾配降下法と同じ線形収束率を達成できることが証明されています。実データを用いた実験により、既存の勾配法や確率的勾配法に基づくアルゴリズムと比較して、通信量を大幅に削減できることが実証されています。

# はじめに

膨大な数のモバイルデバイスを考慮すると，クラウドコンピューティングを介して集中的に機械学習を行うことは，かなりの通信オーバーヘッドを発生させ，深刻なプライバシー問題を引き起こすことになります．今日では、将来の機械学習タスクは、クラウド・センターだけでなく、ネットワーク・エッジ、すなわちデバイスから始めて実行しなければならないというのが、広く受け入れられています[17, 19]。一般に，分散学習タスクは，次のような形式の最適化問題として定式化されます．

 *p* min*✓* X *fm*(✓) with(1)

m2

ここで、*exm,n ✓* 2Rは学習すべきパラメータ、ワーカー*m*における-番目のデータベクトル（例えば、特徴量とラベル）、｜M| = *M*のMはサーバーのセットを表し、*Nmはn*を表す数です。

(1)では、`(**x**; ✓)は、*✓*と**x**に関連する損失を表し、*fm*(✓)は、*✓*と作業者*m*のすべてのデータに対応する集約された損失を表します。

Pm2M *mは*、総合的な損失関数として*f*（✓）＝*f*（✓）を定義します。

一般的に採用されているワーカーとサーバーのセットアップでは、サーバーはワーカーからローカルな勾配を収集し、以下のような勾配降下法（GD）の反復を用いてパラメータを更新します。

GDのイテレーション *✓k*+1 *= ✓k ↵ Xrfm ✓k* (2)

m2

Jun SunとTianyi Chenも同様の貢献をしています。

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

ここで、*em ✓k mは*反復時のパラメータ値を示す*k*は集約された勾配である。データサンプルがworker,*k*に分散している場合、*↵*はステップサイズであり、*rf(✓k* )=

P 2*Mrf* (✓)  *mk*

各ワーカーは、対応するローカルグラジエント *rf* (✓ ) を計算し、それをサーバーにアップロードします。すべてのローカルグラジェントが収集されたときにのみ、サーバーは完全なグラジェントを取得し、パラメータを更新することができます。しかし、(2)を実装するためには、サーバーは*すべての*ワーカーと通信して新しいグラデーションを取得する必要があります。しかし，いくつかの設定では，通信は計算よりもはるかに低速である．そのため，ワーカーの数が増えると，ワーカーとサーバの通信がボトルネックになります[10]．これは、高次元のパラメータを持ち、それに応じて大規模な勾配を持つ、一般的な深層学習ベースの学習モデルを組み込む際に、より困難になります。

1.1 先行技術

通信効率の良い分散学習法は、最近人気を集めている[10, 22]。一般的な手法は、単純な勾配更新に基づいており、勾配の*量子化*や*スパース化など、*通信を節約するために勾配を圧縮するという重要なアイデアを中心としている。量子化。量子化は，通信中に浮動小数点数を表すビット数を制限することで，勾配を圧縮することを目的としており，無線センサーネットワークを用いたいくつかのエンジニアリングタスクへの適用に成功している[21]．分散型機械学習の文脈では，1ビットのバイナリ量子化法が開発されている[5, 24]．複数ビットの量子化方式は[2, 18]で研究されており，調整可能な量子化レベルにより，iterationごとの通信コストと収束率の間のトレードオフを制御する柔軟性を与えることができる．誤差補正[32]、分散低減量子化[34]、3項ベクトルへの量子化[31]、勾配差の量子化[20]などがあります。スパース化。スパース化とは，ある閾値を超える大きさの勾配座標のみを送信することである[27]．経験的には，99%の勾配を除去しても所望の精度を得ることができる[1]．情報の損失を避けるために，小さな勾配成分を蓄積し，十分に大きくなってから適用する．蓄積された勾配は，スパース化されたストキャスティック(S)GDの反復の分散を減少させる[12, 26]．最近の研究[3]を除いて、決定論的スパース化法は、その素晴らしい経験的性能が認められていますが、性能解析の保証はありません。しかし、ランダム化された対応策は、いわゆるunbiased sparsificationと呼ばれ、収束を保証するために開発されています[28, 30]。

量子化とスパース化は同時に用いられることもある[9, 13, 14]。しかし、これらはいずれも(S)GDの更新にノイズをもたらすため、一般的に収束性が悪化する。凸性の強い損失を持つ問題では、勾配圧縮アルゴリズムは最適解の近傍に収束するか、または副次的な速度で収束する。例外として、[18]では、量子化勾配ベースのアプローチで初めて線形収束率が確立された。しかし、[18]では、1回の通信に必要なビット数を減らすことにのみ焦点を当てており、総ラウンド数を減らすことには焦点を当てていません。

とはいえ、例えば*p*次元の*✓*やその勾配などのメッセージを交換するためには、他のレイテンシ

(通信リンクの開始、キューイング、メッセージの伝搬)は、少なくともメッセージサイズに依存した伝送レイテンシーに匹敵します[23]。このため，通信ラウンド数を削減する動機となっており，場合によってはラウンドあたりのビット数よりも削減する必要があります．

前述の勾配圧縮方式とは異なり、通信ラウンド数の削減を目的とした通信効率の良い方式は、高次情報の活用[25, 36]、周期的集約[19, 33, 35]、そして最近では適応的集約[6, 7, 11, 29]によって開発されている。しかし、望ましい収束特性を犠牲にすることなく、通信ビットとラウンドを同時に節約することができるかどうかは、まだ解決されていない。本論文は、この問題を解決することを目的としている。

1.2 私たちの貢献

我々のアプローチを紹介する前に、1つのサーバーと*M人の*ワーカーを持つ(1)の単純な設定で、一般的な量子化(Q)GD法[24]-[20]の正統な形式を再検討します。

QGDのイテレーション *✓k*+1 *= ✓k ↵* X *Qm ✓k* (3) m2M

ここで、*Qm ✓k*は、局所勾配*rfm(✓k*)を粗く近似した量子化勾配です。正確な量子化スキームはアルゴリズムによって異なりますが、一般的に*Qm ✓kの送信は、量子*化された勾配{*Qm ✓k* }を送信するよりも少ないビット数で済みますので、サーバーはパラメータ*rfm(✓k*)を更新することができます。しかし、GDと同様に、すべてのローカル*✓の*ときにのみ。

このような状況の中、本論文では、特定のラウンドで通信*を省略できる量子*化勾配革新法（QGDと同様に単純）を打ち出しています。具体的には、サーバーからワーカーへのダウンリンク通信を同時に行うことができる（例えば、*✓kを*ブロードキャストする）のとは対照的に、サーバーは他のワーカーからの干渉を避けるためにワーカーの勾配を順次受信しなければならず、余計な待ち時間が発生してしまう。このため、ここではワーカーからサーバーへのアップリンク通信（アップロードとも呼ぶ）の回数を減らすことに焦点を当てています。我々のアルゴリズムであるLAQ（Lazily Aggregated Quantized gradient descent）は、（3）に似ており、次のように与えられます。

LAQの反復 *✓k*+1 *= ✓k ↵ rk* with *rk*=*rk* 1+ *Xk Qkm* (4) m2M

ここで、*rk*は反復*k*におけるパラメータ変化を要約した近似集約勾配であり、現在のイテレート*✓k*と古いコピー*✓*ˆ*km* 1における*fm*の2つの量子化された勾配の差である。後で紹介する適切な選択基準により、*Mkは*反復*k*においてローカル*Qkm*がアップロードされている作業者のサブセットを表し、パラメータの反復は*✓*ˆ、および*✓*ˆで与えられます。すべての*ワーカーインク*1(3)に新たな量子化勾配を要求する代わりに、以前に集約された勾配でワーカーを改良して*rk*を得るというトリックです。単純な修正により、残りのワーカーからの古い勾配を再利用しながら、QGD'*sk*からイテレーションごとの通信ラウンドを縮小します。つまり，サーバーに保存されている*therk* 1からの新しい勾配のみを使用して，この*選択されたM*をLAQの

|*Mk*|。本稿では、1ラウンドの通信は1人の作業者のアップロードを意味します。既存の量子化スキームと比較して、LAQはまず、勾配革新（現在の勾配と以前の量子化された勾配の差）を量子化し、その後、勾配通信をスキップします。LAQは、損失関数の強凸の仮定の下で、GDと同じ線形収束を達成することを厳密に証明します。また、数値実験により、本手法が通信ビット数とラウンド数の両面で既存の手法よりも優れていることを示します。

記法*`*1 .太字の小文字は列ベクトルを表し，*i* **kx**; 一方，k2とbakcxkは下向きの丸めを表す1は*`*2ノルムと

**x**]は，*a*の**x**の*i*番目のエントリを表し，｜-｜は，集合またはベクトルのカーディナリティを表します。

# LAQ: Lazily aggregated quantized gradient

通信のオーバーヘッドを削減するために，2つの相補的なステージをアルゴリズムデザインに組み込んでいます。1）勾配革新に基づく量子化と，2）勾配革新に基づくアップロードまたはアグリゲーション（Lazily Aggregated Quantized gradient：LAQ）です。前者はアップロードあたりのビット数を削減し，後者はアップロード数を削減することで，簡潔な通信を実現しています。このセクションでは、私たちの2段階のデザインの原理を説明します。

2.1 グラディエント・イノベーションに基づく量子化 *[Qm*( *km* 1)]*i* [ *fm*( *k*)]*i* [*Qm*( *k*)]*i*

量子化により、通信時に勾配ベクトルを表現するビット数が制限されます。仮に

勾配の各座標の量子化に*b*ビットを使用 2 *Rmk Rmk*

[

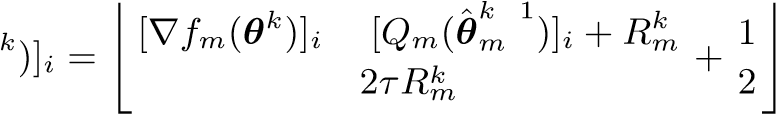
)

ベクトルは、量子化演算子を示すQとは対照的に、一般的なコンピュータのように32ビットの量子化を行います。 図1:量子化の例 (*b* = 3)

これは，勾配*rfm(✓k*)と前の量子化に依存します．勾配は、一様に離散化されたグリッドの最も近い点に投影することで、要素ごとにquan-*p*-dimensional tizedされます。このグリッドは

をする超立方体である。である。

このように量子化の粒度を定義すると、勾配革新作業者*m*の座標ごとに*b*ビットで次のように量子化することができる。

[*qm (✓ ,* *i* = 1*,--- ,p* (5)

は、[0*,*2*b,1]*以内の整数であるため、bビットで符号化できる。1]以内の整数であるため、*b*ビットで符号化できることになります。なお、分子に加えることで[*qm(✓k*)]*i*の非負性が保証され、(5)に1/2を加えることで最も近い点に丸めることが保証されます。したがって、作業者*m*における量子*化された*勾配革新は、（**1** := [1*,--- ,*1]> )

 : 送信と*qm(✓k*) (6)

これは、元の32*p*ビットの代わりに、32+*bp*ビット（*qm(✓k*)のビットの32ビット）で送信することができる。メモリに保存されている古いグラデーションと、先験的に知られている*Friedrich*があれば、*Qkm*を受信した後、サーバーは量子化されたグラデーションを次のように復元することができます

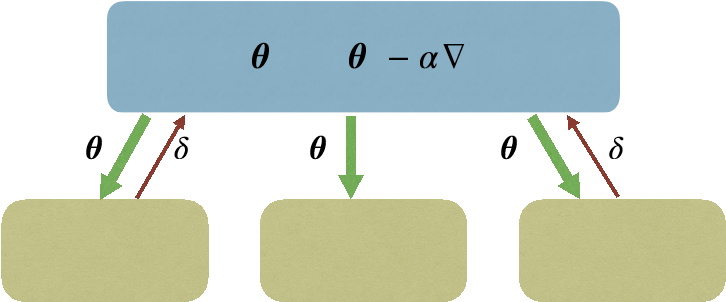
図1は、グラデーションの1つの座標を*b* = 3ビットで量子化する例です。元の値を3ビットで量子化すると、23＝8個の値が得られ、それぞれの値は自分自身を中心とした長さ2*⌧Rmkの*範囲をカバーしています。*km* := *rfm(✓k*) *Qm(✓k*)を局所的な量子化誤差とすると、量子化誤差は各値がカバーする範囲の長さの半分以下であることがわかります。

すなわち、集約された量子化誤差はk *"*kmk1 *⌧Rmk*となる。集約された量子化勾配は*」* *Q*(✓; つまり、*k*)=*PQm*(2*✓Mk*)=Qmr*(✓fk()✓*となり、*k*)の*「k」となる*。

2.2 グラディエント・イノベーションに基づくアグリゲーション

遅延グラデーション集約の考え方は、局所的に量子化された連続する2つのグラデーションの差が小さければ、冗長なグラデーションのアップロードをスキップして、サーバーで前のグラデーションを再利用しても安全であるというものです。さらに、サーバーは各ワーカーの比較的「新鮮な」グラデーションを持っていることを保証するために、いずれかのワーカーが以下のような場合、通信を強制します。

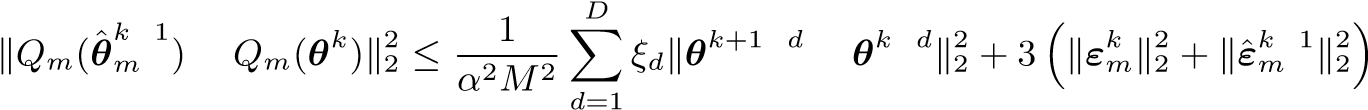
は最後の*t*¯roundsの間にアップロードされていない。を設定します。 サーバ *L*= *k* *k*

 最後に情報をアップロードしてからのイタレーション数 *tm, m* 2 M ワーカー*m* がカウントする数 *k* Q1 *k k QMk*

量子化と選択を備えた、私たちの ワークス

LAQの更新は(4)のような形で行われます。 量子化 量子化 量子化 あとは選択基準を設計するだけです。 選択 選択 選択

は，どのワーカーが量子化されたグラディエントやそのイノベーションをアップロードするかを決定します．我々は以下の図2を提案します。 図 2：LAQ通信基準による 分散学習：ワーカー*m* 2 Mは、反復*k*において、以下を満たす場合にアップロードをスキップします。

; (7a)

*tm t*¯ (7b)

はあらかじめ決められた定数、*"km "*は現在の量子化誤差、*"*ˆ "は最後にアップロードされた量子化勾配の誤差です。次節では、基準（7）の下でのLAQの収束性と通信特性を証明します。

2.3 LAQアルゴリズムの開発

要約すると，図2に示されるように，LAQは以下のように実装されます。イテレーション*k*において，サーバは学習パラメータをすべてのワーカーにブロードキャストする．各ワーカーは勾配を計算し，それを量子化して，量子化された勾配のイノベーションをアップロードする必要があるかどうかを判断そして，サーバは，選択されたワーカーからのグラジエント・イノベーションを受信した後，学習パラメータを更新する．アルゴリズムの概要はAlgorithm 2に示されています。

LAQとGDの違いを明確にするために、(4)を次のように書き直します。

✓ (8a)

m2*mkc*

*=✓k* ↵[*rf(✓k) "k* + *Xk*(*Qm(✓*ˆ*mk* 1) *Qm(✓k*))］となる。] (8b)

m2*Mc*

ここで、 , は反復*k*においてサーバとの通信をスキップする作業者のサブセットである。(2)のGDの反復に比べて、ここで採用した勾配は、量子化誤差*「k」*と勾配革新のミス「Pm2*Mkc*(*Qm(✓*ˆ*k* 1) *Qm(✓k*))]が発生します。このことから、大きい場合は、Algorithm 1 QGD アルゴリズム2 LAQ

|  |  |
| --- | --- |
| 1: | 入力：ステップサイズ *↵ > 0、*量子化ビット b。 |
| 2: | 初期化*：✓k*。 |
| 3: 4: | forServerは、*k* = 1*,*2*,--- ,K✓dokを*すべてのワーカーにブロードキャストします。 |
| 5:  6:  7: | for *m* = 1*,*2*,--- ,M* do  作業者*m*は*rfm(✓k*)と*Qm(✓k*)を計算します。  作業者*m*は(6)により*Qkm*をアップロードする。 |
| 8: | エンドフォー |
| 9:  10: | end forServerは、*Mk* = Mで（4）に従って*✓*を更新する。 |

1：入力：stepsizeと*t*¯。

2：初期化*：✓k*、および{*Qdom(✓*ˆ0*m),tm*}m2M。3：*k* = 1*,*2*,--- ,K*について

4: サーバーは、すべてのワーカーに*✓kを*ブロードキャストします。

5: for *m* = 1*,*2*,--- ,M dofm(✓k*) and *Qm(✓k*)です。

6: ワーカー*mがrm*を計算したら7: (7)がワーカーに成立する場合

8: Worker *m*は何もアップロードしません。

9: Set ✓ .

10: その他

11: Worker *m* uploads via (6).

12: Set ✓ .

13: end if 14: end for

15: サーバーは(4)に従って*✓* を更新します。

16: end for

表1：QGDとLAQの比較。

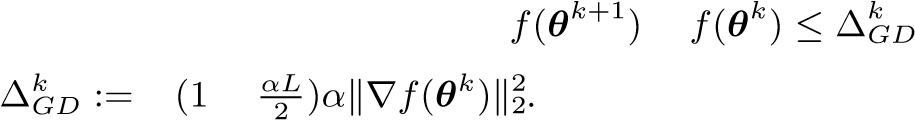
勾配を量子化するのに十分な数のビットが使用され，すべてが0に設定されるため，*Mk* := Mとなり，LAQはGDに還元されます。このように，調整はLAQの性能に直接影響します。選択基準(7)の根拠は，GDの降下量とLAQの降下量を適切に比較することにあります。下降量を比較するために，まず両アルゴリズムの1ステップ下降量を設定します．本論文のすべての結果において、以下の仮定が成り立つ。

前提1.*rf*(-)*はL-Lipschitz連続である、すなわち、定数が存在する局所勾配rfm*(-)*はLm-Lipschitz連続であり、大域勾配LmとLは以下のようになる。*

*krfm(✓*1) *rfm(✓*2)k2 Lmk*✓*1 *✓*2k2*,* 8*✓*1*, ✓*2; (9a) *krf(✓*1) *rf(✓*2)k2 Lk*✓*1 *✓*2k2*,* 8*✓*1*, ✓*2*。* (9b)

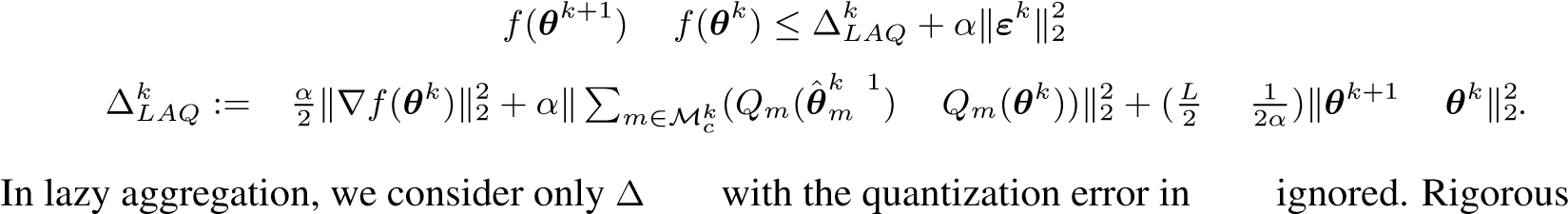
仮定1に基づき、次のレンマではGDによる目的語の降下を説明します。

Lemma 1.*勾配降下法による更新では、次のような降下法が得られます。*

(10) *ここで*

LAQの降順は，量子化と選択によってGDの降順と区別されますが，これは次のレムマで規定されています。

Lemma 2.*LAQ更新では、以下のような降下が得られます。*

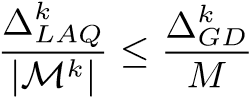
(11)

*どこで*

*kLAQ* (11)

次の章では，量子化誤差を考慮したLAQの特性を示す定理を確立します。

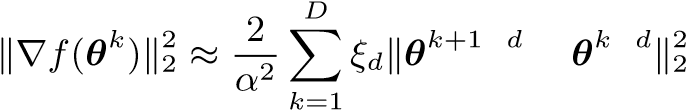
以下の部分は、数学的に厳密ではありませんが、基準(7a)の直観を示しています。遅延集約機構は，損失関数を減少させることへの貢献度を判断して，量子化された勾配革新を選択します。LAQはGDよりも通信効率が良いことが期待されます。つまり、各アップロードはより多くの降下をもたらすことになりますが、これは次のように変換されます。

*.* (12)

に相当します（補足資料の派生を参照）。

*.* (13)

しかし、各ワーカーがローカルで(73)をチェックすることは、完全に集約された勾配が得られている場合にはrreduceアップロードが行われるため、不可能です。そのため、*f(✓k*)を直接バイパスすることになりますが、これはまさに避けたいことです。さらに、以下のようにその近似値を使って計算することは意味がありません。

 (14)

ここでは定数を用いています。(74)が成立する根本的な理由は、*rf*-smoothであること。Combining*(✓k*)は、*f*(-)が*L*であるため、加重前勾配またはパラメータ差分で近似できる。

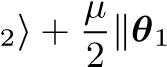
(73)と(74)から、量子化誤差を無視した通信基準(7a)が得られます。

最後に，LAQとエラーフィードバック（量子化）方式の比較を行って，このセクションを締めくくります。

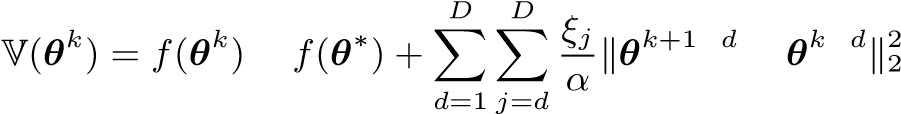
エラー・フィードバック・スキームとの比較我々のLAQアプローチは，エラー・フィードバック・スキーム（例：[3, 12, 24, 26, 27, 32]）に関連しています。どちらのアプローチも，通信量の削減（量子化，スパース化，スキップなど）によって発生したエラーや遅延イノベーションを蓄積し，次の通信ラウンドでアップロードするというものです．しかし，エラーフィードバック方式では，勾配のあるエントリの通信をスキップしても，すべての作業者と通信することになります．LAQは特定のワーカーとの通信をスキップしますが，すべての（量子化された）エントリーを通信します。この2つの方法は相互に排他的ではなく，同時に使用することができます．

# コンバージェンスとコミュニケーションの分析

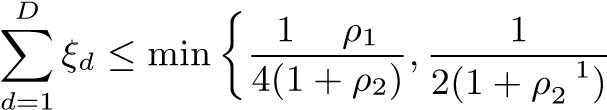
以降のLAQの収束分析は、*f*(✓)に対する以下の仮定に依存しています。仮定2.*関数f*(-)*は、μ強凸であり、例えば、次のような定数μ>0が存在する。*

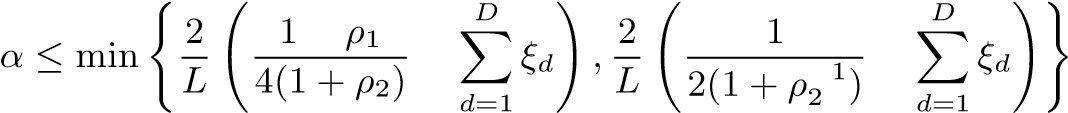
*f(✓*1) *f(✓*2) *hrf(✓*2*),✓*1 *✓ .*(15)

*✓*⇤を(1)の最適解とすると、LAQのリアプノフ関数を次のように定義します。

 (16)

リアプノフ関数V(✓)の設計は、パラメータ差分項を含む通信ルール(7a)と結合されます。直感的には、現在の反復で通信がスキップされていない場合、LAQはV(✓)の目的残差を減少させるGDのように振る舞います。特定のアップロードがスキップされている場合、LAQのルール(7a)は、V(✓)のパラメータ差に匹敵する陳腐な勾配を使用して、その下降を保証します。以下のレンマは、リアプノフ関数の進行を捉えています。レムマ 3.*仮定1と2の下で、ステップサイズ↵とパラメータ**を（任意の*0*＜⇢1＜*1*と⇢2＞0で）選択すると*

 (17a)

 (17b)

*とすると、リアプノフ関数は次のようになります。*

V*(✓k*+1) 1V*(✓k*)＋Bhk *"*kk22＋*mXk*⇣k *"*kmk22＋k *"*ˆ*km* 1k22⌘i (18)

2*Mc*

*ここで、定数*0 *< 1 <* 1*と*B *> 0は、↵と{⇠d}*に*依存している；詳細は補足資料を参照。*

タイトな分析では、(17)は複雑に見えるが、簡単な選択が可能である。例えば、*⇢*1=1/2と*⇢*2=1を選択すると、andは(17)を満たすことになる。

(18)の量子化誤差がnullであれば、Lemma 3はLyapunov関数が線形収束率を享受することを容易に示唆しています。以下では、ある条件の下では、量子化誤差を考慮してもLAQアルゴリズムが線形収束を保証できることを示します。

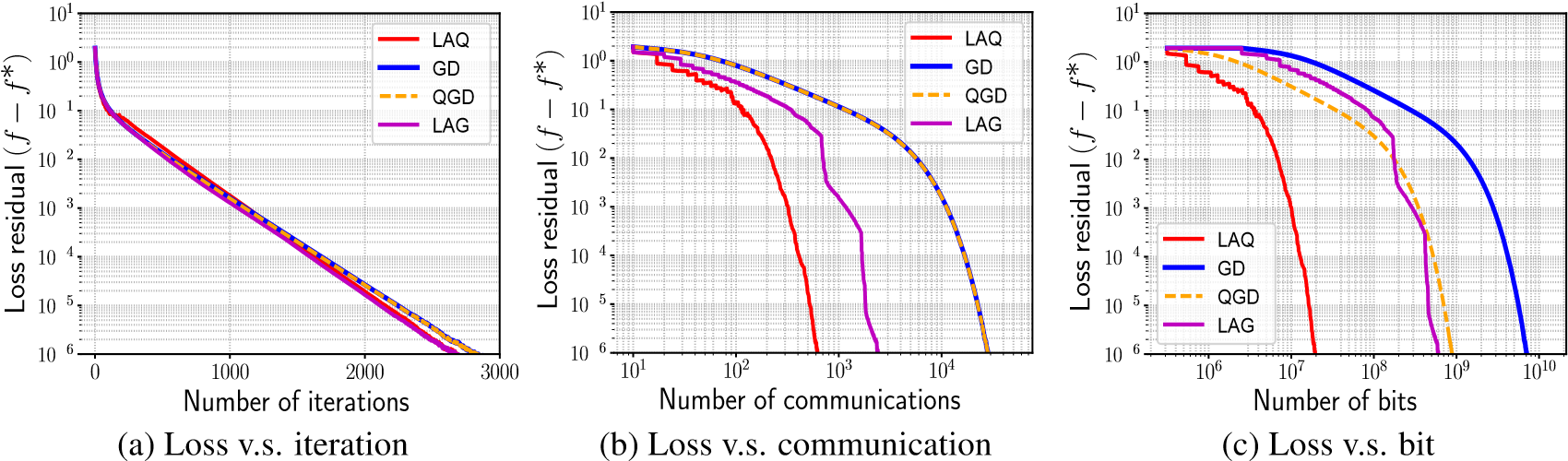


図4：損失関数の収束（ロジスティック回帰

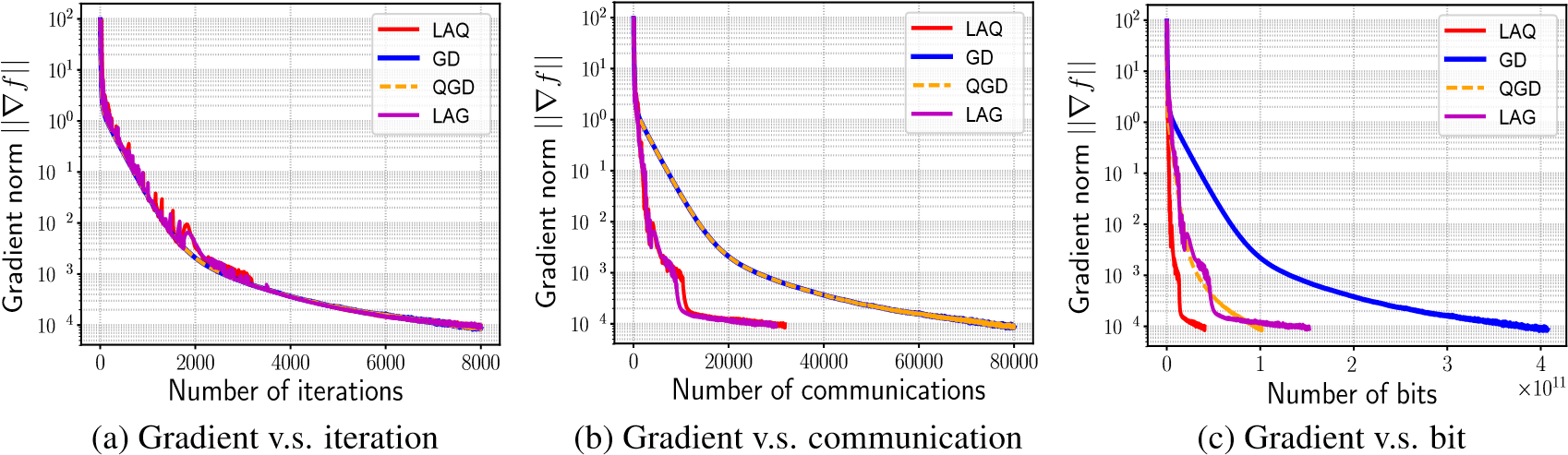


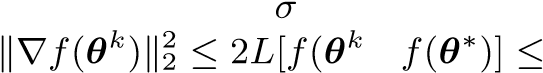
図5：勾配ノルムの収束（ニューラルネットワーク

Theorem 1.*同じ仮定と定理3のパラメータのもとでは、リアプノフ関数と量子化誤差は線形に収束する。つまり、次のような定数*2 2 (0*,*1)*が存在する。*

; (19a)

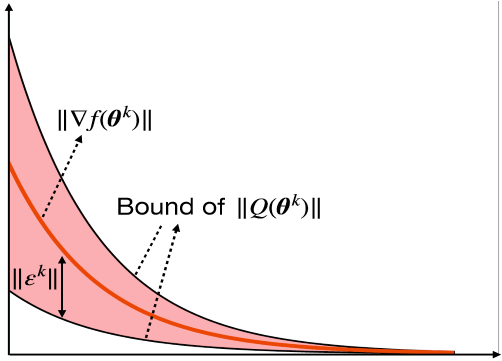
*.* (19b)

*ここで、Pは*(17)*のパラメータに依存する定数で、詳細は補足資料を参照してください。*

リアプノフ関数の定義から、テーラー*f(✓k*) *f(✓*⇤)は線形に収束する。*L-*スムースネスの結果 *inf(✓k*) *f* *(✓*⇤) V*(✓k*) 2kV0 - リスク

- は、勾配ノルムが線形に収束する。同様に、*μ*強凸性は、k*✓k* ✓⇤k22 *μ*2[*f(✓k* - k*✓k ✓*も線形収束することを意味します。

前述のLAGの解析[6]と比較して、LAQの解析は、古くなった勾配だけでなく、量子化された（不正確な）勾配も扱う必要があるため、より複雑なものとなっています。この修正は、[6]での解析の構成要素である(18)のリアプノフ関数の単調性を悪化させます。この問題を解決するために、i)古い勾配を量子化(6)で考慮する、ii)量子化誤差を新しい選択基準(7)に組み込む、という方法をとる。その結果、定理1では、量子化誤差があってもLAQが線形収束率を維持できることを示しています。これは、適切に制御された量子化誤差も直線的に収束するためです。

命題1仮定*1の下で、⇠*1 *⇠*2 *--- ⇠Dを満たす定数を選び**、dm, m* 2 Mを*次のように定義すると、このようになります。*

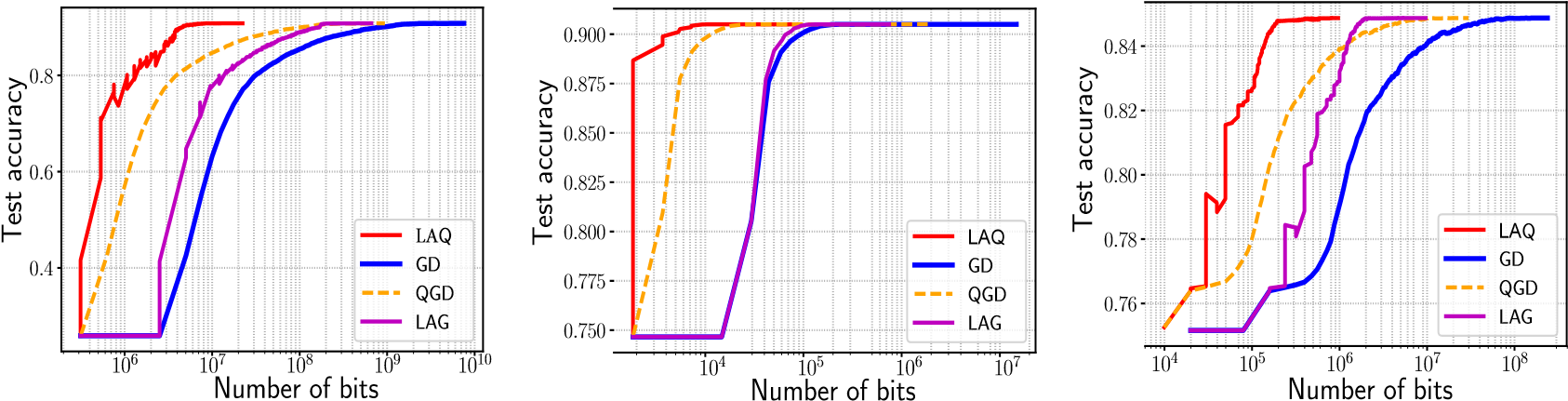
*dm* := *maxd d*|*L*2*m ⇠d*/(32*M*2*D), d* 2 {1*,*2*,--- ,D*}.(20)*とすると、ワーカーmはk番目の反復まで、サーバーと最大でk*/(*dm* + 1)の*通信を行う。*

この命題は、ローカル損失関数の滑らかさが、ローカルワーカーのコミュニケーション強度を決定することを意味します。 反復計算の結果、ローカルワーカーの通信強度が決まる。 図3：勾配ノルムの減衰(a) MNIST (b) ijcnn1 (c) covtype

図6：3つの異なるデータセットでのテスト確度

# 数値実験と結論

実用的な機械学習問題において，我々の性能分析を検証し，通信量の削減効果を確認するために，強凸である正則化ロジスティック回帰と，非凸であるニューラルネットワークに対するアルゴリズムの性能を評価した．使用するデータセットはMNIST [15]で，*M* = 10人のワーカーに一様に分散している．実験では，次のように設定した．

LAQをベンチマークするために、勾配ベースのアルゴリズムと*D* = 10*, ⇠*1 *= ⇠*2 *= --- ,⇠D* = 0*.*8*/D*, *t*¯= 100という2つのクラスのアルゴリズムと比較しました。補足資料の詳細な設定を参照してください。 ミニバッチ・ストキャスティック・勾配ベース・アルゴリズム - 以下の2つのテストに対応しています。

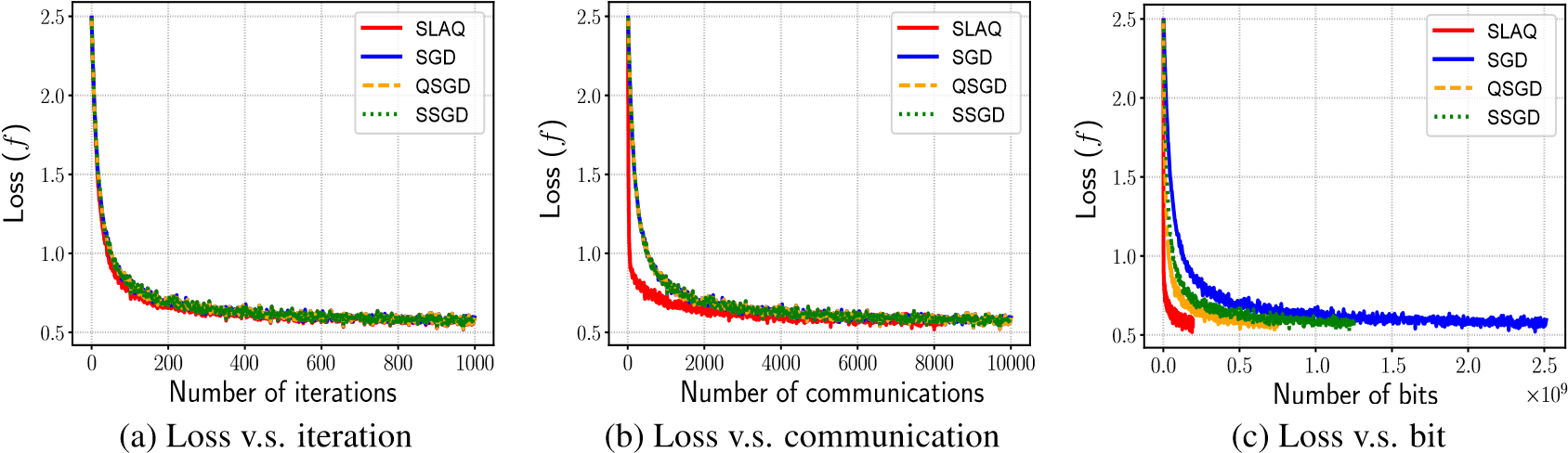


図7：損失関数の収束（ロジスティック回帰

勾配ベースのテストGD, QGD [18], lazily aggregated gradient (LAG) [6] を検討します。座標あたりのビット数は、ロジスティック回帰では *b* = 3、ニューラルネットワークでは 8 とした。ステップサイズはどちらのアルゴリズムも ↵ = 0*.*02 とした。図4は、ロジスティック回帰タスクの目的収束を示している。図4(a)は、定理1を検証しており、例えば、強く凸な損失関数の下での線形収束率を示しています。図4(b)に示すように、LAQは、我々の選択ルールのおかげで、GDやQGDよりも少ない通信ラウンド数で済みますが、勾配量子化のためにLAGよりも多くのラウンド数を必要とします。しかし，図4(c)に示すように，LAQの総送信ビット数はLAGのそれよりも大幅に少ない。ニューラルネットワークモデルの場合

図5は、勾配ノルムの収束を示しており、LAQも競争力のある性能を示しています。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| アルゴリズム | イテレーション # | コミュニケーション # | Bit # | 精度 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| LAQ | ロジスティック | **2673** | **620** | **1***.***95**⇥**107** | **0***.***9082** |
| ニューラルネットワーク | **8000** | **31845** | **4***.***05**⇥**1010** | **0***.***9433** |
| GD | ロジスティック | 2820 | 28200 | 7*.*08 ⇥ 109 | 0*.*9082 |
| ニューラルネットワーク | 8000 | 80000 | 4*.*07 ⇥ 1011 | 0*.*9433 |
| QGD | ロジスティック | 2805 | 28050 | 8*.*81 ⇥ 108 | 0*.*9082 |
| ニューラルネットワーク | 8000 | 80000 | 1*.*02 ⇥ 1011 | 0*.*9433 |
| LAG | ロジスティック | 2659 | 2382 | 5*.*98 ⇥ 108 | 0*.*9082 |
| ニューラルネットワーク | 8000 | 29916 | 1*.*521011 | 0*.*9433 |

⇥

表2：勾配ベースのアルゴリズムの比較。ロジスティック回帰では，損失残差が10 6に達した時点ですべてのアルゴリズムが終了し，ニューラルネットワークでは，すべてのアルゴリズムが固定の反復回数で実行される．

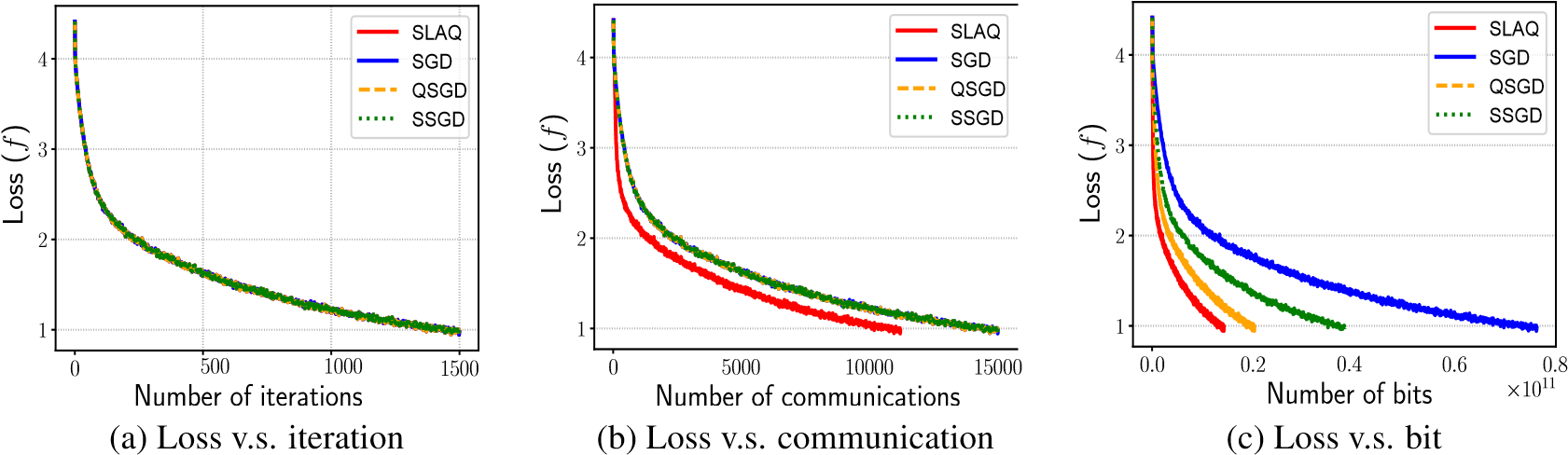


図8：損失関数の収束（ニューラルネットワーク

は，非凸問題に対してロジスティックモデルの結果と同様に，LAQ は最も少ないビット数を必要とします．表2は，所定の精度に到達するために必要な反復回数，アップロード回数，ビット数をまとめたものです．

図6は，一般的に使用されている3つのデータセット（MNIST，ijcnn1，covtype）における上記アルゴリズムのテスト精度を示しています．これらのデータセットに適用した場合，LAQは送信ビット数を削減しながらも，同じ精度を維持しています。

確率的勾配に基づくテストstochastic gradient descent (SGD)、quantized stochastic gradient descent (QSGD) [2]、sparsified stochastic gradient descent (SSGD) [30]、そしてLAQのstochastic version (SLAQと略す)をテストする。ミニバッチサイズは500、↵ = 0*.*008、座標あたりのビット数はロジスティック回帰では*b* = 3、ニューラルネットワークでは8と設定されている。図7と図8に示すように、SLAQは最も少ない通信ラウンド数とビット数で済む。今回の確率的勾配テストでは、勾配ベースのアルゴリズムと比較して、SLAQの通信削減効果はLAQほどではないものの、SLAQは最先端のアルゴリズムであるQSGDやSSGDなどを上回っています。その結果を表3にまとめました。なお，ビット数や不均一性のレベルが異なる場合の詳細な結果については，補足資料で報告している．

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| アルゴリズム | イテレーション # | コミュニケーション # | Bit # | 精度 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| SLAQ | ロジスティック | **1000** | **8255** | **1***.***94**⇥**108** | **0***.***9018** |
| ニューラルネットワーク | **1500** | **11192** | **1***.***42**⇥**1010** | **0***.***9107** |
| SGD | ロジスティック | 1000 | 10000 | 2*.*51 ⇥ 109 | 0*.*9021 |
| ニューラルネットワーク | 1500 | 15000 | 7*.*63 ⇥ 1010 | 0*.*9100 |
| QSGD | ロジスティック | 1000 | 10000 | 7*.*51 ⇥ 108 | 0*.*9021 |
| ニューラルネットワーク | 1500 | 15000 | 2*.*03 ⇥ 1010 | 0*.*9100 |
| SSGD | ロジスティック | 1000 | 10000 | 1*.*26 ⇥ 109 | 0*.*9013 |
| ニューラルネットワーク | 1500 | 15000 | 3*.*821010 | 0*.*9104 |

⇥

表3：ミニバッチストキャスティック・グラジェント・ベース・アルゴリズムの性能比較。

本論文では、通信効率の良い分散学習問題を研究し、勾配革新に基づいて、通信の量子化とスキップを同時に行うLAQを提案した。オリジナルのGD法と比較して、強く凸な損失関数に対しても線形収束率を維持している。これは、LAQが通信ビットとラウンド数の両方を大幅に削減したことによるものです。凸型正則化ロジスティック回帰や非凸型ニューラルネットワークモデルを用いた数値実験により、既存の一般的な手法に対するLAQの優位性が実証されました。謝辞

J.SunとZ.Yangのこの研究は、深圳市科学技術委員会から一部支援を受けています。

Innovations under Grant GJHZ20180411143603361, in part by Department of Science and

Technology of Guangdong Province under Grant 2018A050506003, and in part by the Natural Science Foundation of China under Grant 61873118.また、J. Sunの研究は、China Scholarship Councilの支援を受けています。G. Giannakisの研究は、NSF 1500713, and 1711471によって一部サポートされています。

# リファレンス

1. Alham Fikri Aji and Kenneth Heafield.分散型勾配降下法のためのスパースな通信

*Proc.Conf.Empi.Meth.自然言語処理。*, Copenhagen, Denmark, Sep 2017.

1. Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic.QSGD: 勾配の量子化とエンコーディングによる通信効率の良いSGD。In *Proc.Advances in Neural Info.Process.Syst.*, ページ 1709-1720, Long Beach, CA, Dec 2017.
2. Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cédric Renggli.sparsified gradient methodの収束。In *Proc.Advances in Neural Info.Process.Syst.*, ページ 5973-5983, モントリオール, カナダ, 2018年12月.
3. Yossi Arjevani and Ohad Shamir.分散凸型学習・最適化の通信量In *Proc.Advances in Neural Info.Process.Syst.*, ページ 1756-1764, Montreal, Canada, Dec 2015.
4. Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar.SignSGD: 非凸問題の圧縮最適化.In *Proc.Intl.Conf.Machine Learn.*, ページ 559-568, Stockholm, Sweden, Jul 2018.
5. Tianyi Chen, Georgios Giannakis, Tao Sun, and Wotao Yin.LAG: Lazily aggregated gradient for communication-efficient distributed learning.In *Proc.Advances in Neural Info.Process.Syst.*, ページ 5050-5060, Montreal, Canada, Dec 2018.
6. Tianyi Chen, Kaiqing Zhang, Georgios Giannakis, and Tamer Bas¸ar.Communication-Efficient Distributed Reinforcement Learning.*IEEE Trans. on Automatic Control*, submitted April 2019. *arXiv preprint:1812.03239*
7. Mert Gurbuzbalaban, Asuman Ozdaglar, and Pablo A Parrilo.インクリメンタル・アグリゲーション・グラジェント・アルゴリズムの収束率について。*SIAM Journal on Optimization*, 27(2):1035-1048, 2017.
8. Peng Jiang and Gagan Agrawal.A linear speedup analysis of distributed deep learning with sparse and quantized communication.In *Proc.Advances in Neural Info.Process.Syst.*, ページ 2525-2536, モントリオール, カナダ, 2018年12月.
9. Michael I Jordan, Jason D Lee, and Yun Yang.Communication-efficient distributed statistical inference.*J. American Statistical Association*, to appear, 2018.
10. Michael Kamp, Linara Adilova, Joachim Sicking, Fabian Hüger, Peter Schlicht, Tim Wirtz, and Stefan Wrobel.動的モデル平均化による効率的な分散型深層学習.In *Euro.Conf.Machine Learn.Knowledge Disc.Data.*, ページ 393-409, Dublin, Ireland, 2018.
11. Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi.エラーフィードバックはsignsgdと他の勾配圧縮スキームを修正する。In *Proc.Intl.Conf.Machine Learn.*, ページ 3252-3261, Long Beach, CA, Jun 2019.
12. Jakub Konecnˇ y, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh,` and Dave Bacon.Federated learning:Strategies for improvement communication efficiency. *arXiv preprint:1610.05492*, Oct 2016.
13. Jakub Konecnˇ y and Peter Richtárik.ランダム化された分布平均推定。Accuracy vs` Communication.*Frontiers in Applied Mathematics and Statistics*, 4:62, Dec 2018.
14. Yann LeCun, Corinna Cortes, and CJ Burges.Mnist手書き数字データベース。*AT&T Labs [Online].Available: http://yann. lecun. com/exdb/mnist*, 2:18, 2010.
15. Mu Li, David G Andersen, Alexander J Smola, and Kai Yu.パラメータサーバを用いた通信効率の良い分散機械学習.In *Proc.Advances in Neural Info.Process.Syst.*, pages 19-27, Montreal, Canada, Dec 2014.
16. Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu.Decentralized algorithms can outperform centralized algorithms?A case study for decentralized parallel stochastic gradient descent.In *Proc.Advances in Neural Info.Process.Syst.*, ページ 5330-5340, Long Beach, CA, Dec 2017.
17. Sindri Magnússon, Hossein Shokri-Ghadikolaei, and Na Li.On maintaining linear convergence of distributed learning and optimization under limited communication. *arXiv preprint arXiv:1902.11163*, 2019.
18. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas.分散されたデータからのディープネットワークの通信効率の良い学習In *Proc.Intl.Conf.Artificial Intell.and Stat.*, ページ 1273-1282, Fort Lauderdale, FL, April 2017.
19. Konstantin Mishchenko, Eduard Gorbunov, Martin Takác, and Peter Richtárik.圧縮された勾配差による分散ˇ学習. *arXiv preprint:1901.09269*, Jan 2019.
20. Eric J Msechu and Georgios B Giannakis.打ち切りと量子化によるWSNでの推定のためのセンサ中心のデータ削減.*IEEE Trans.Sig.Proc*. , 60(1):400-414, Jan 2011.
21. Angelia Nedic, Alex Olshevsky, and Michael Rabbat.分散型最適化におけるネットワークトポロジーと通信-´計算のトレードオフ。*Proceedings of the IEEE*, 106(5):953-976, May 2018.
22. ラリー・L・ピーターソン、ブルース・S・デイビー。*コンピュータネットワーク:A Systems Approach*.Morgan Kaufman, Burlington, MA, 2007.
23. Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu.1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns.In *Proc.Conf.Intl. Speech Comm.Assoc*. , Singapore, Sept 2014.
24. Ohad Shamir, Nati Srebro, and Tong Zhang.近似ニュートン型法を用いた通信効率の良い分散型最適化.In *Proc.Intl.Conf.Machine Learn.*ページ 1000-1008, 北京, 中国, 2014年6月.
25. Sebastian U. Stich, Jean-Baptiste Cordonnier, and Martin Jaggi.メモリを用いたスパース化SGD。In *Proc.Advances in Neural Info.Process.Syst.*, ページ 4447-4458, モントリオール, カナダ, 2018年12月.
26. 日興ストーム。コモディティGPUクラウドコンピューティングを用いたスケーラブルな分散DNN学習In *Proc.Conf.Intl.Speech Comm.Assoc*. , Dresden, Germany, Sept 2015.
27. Hongyi Wang, Scott Sievert, Shengchao Liu, Zachary Charles, Dimitris Papailiopoulos, and Stephen Wright.Atomo:アトミックスパース化による通信効率の良い学習.In *Proc.Advances in Neural Info.Process.Syst.*, ページ 9850-9861, モントリオール, カナダ, 2018年12月.
28. Jianyu Wang and Gauri Joshi.Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms. *arXiv preprint:1808.07576*, August 2018.
29. Jianqiao Wangni, Jialei Wang, Ji Liu, and Tong Zhang.通信効率の高い分散型最適化のための勾配スパース化.In *Proc.Advances in Neural Info.Process.Syst.*, ページ 1299-1309, モントリオール, カナダ, 2018年12月.
30. Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li.Terngrad:Ternary gradients to reduce communication in distributed deep learning.In *Proc.Advances in Neural Info.Process.Syst.*, ページ 1509-1519, Long Beach, CA, Dec 2017.
31. Jiaxiang Wu, Weidong Huang, Junzhou Huang, and Tong Zhang.Error compensated quantized SGD and its applications to large-scale distributed optimization. *arXiv preprint arXiv:1806.08054*, 2018.
32. Hao Yu and Rong Jin.確率的非凸最適化のための動的バッチサイズを持つ並列SGDの計算と通信の複雑さについて。In *Proc.Intl.Conf.Machine Learn.*, Long Beach, CA, Jun 2019.
33. Hantian Zhang, Jerry Li, Kaan Kara, Dan Alistarh, Ji Liu, and Ce Zhang.Zipml:エンド・ツー・エンドで低精度な線形モデルをトレーニングし、少しだけ深層学習を行う。In *Proc.Intl.Conf.Machine Learn.*, ページ 4035-4043, シドニー, オーストラリア, 2017年8月.
34. Sixin Zhang, Anna E Choromanska, and Yann LeCun.エラスティック・アベレージングSGDによる深層学習In *Proc.Advances in Neural Info.Process.Syst.*, pages 685-693, Montreal, Canada, Dec 2015.
35. Yuchen Zhang and Xiao Lin.DiSCO:自己調整可能な経験的損失のための分散最適化．In *Proc.Intl.Conf.Machine Learn.*, ページ 362-370, Lille, France, June 2015.