

連合学習における学習時間短縮のための 学習量と通信圧縮率の統合的最適化

穴田 穂乃香^{1,2} 欧 亜非² 高前田 伸也^{1,2}

概要: 多様な計算性能・通信帯域を持つデバイスが参加する連合学習 (Federated Learning) では、一部の低速なクライアントが全体の学習時間を律速することが大きな課題である。その有効な解決策として、サーバによる集約までの時間に制約を設け、各クライアントが時間制約内で完了可能な学習量を計算・通信性能に応じて適応的に決定する手法が提案されている。一方で、通信時間が学習時間に対して無視できない環境では、通信圧縮率の選択によって時間制約内で可能な学習量が大きく左右される。そのため、時間制約内で精度を最大化するには学習量と通信圧縮率の統合最適化が不可欠だが、その手法は確立されていない。本稿では、学習ステップ数と通信圧縮率の統合最適化手法 TALLOP を提案する。TALLOP は、学習初期段階のミニバッチ損失の推移から将来の学習ステップにおける損失を予測する LoSE と、圧縮誤差に基づいて通信圧縮に伴う損失増加量を推定する CLEN からなる。これらによって TALLOP は、各学習ステップ数・通信圧縮率の候補に対して圧縮後モデル損失を予測して最小化することで、十分小さい計算量で学習ステップ数と通信圧縮率を決定する。実験の結果、提案手法は、学習ステップ数と通信圧縮率の片方または両方を固定した場合と比較して、より短時間で高い精度に達することを確認した。

1. はじめに

スマートフォン等のエッジデバイス上に分散蓄積された膨大なデータを、プライバシーを保護しつつ効率的に活用する手法として、連合学習 (Federated Learning) [1] が近年注目を集めている。連合学習では、クライアントが保持するデータ分布の動的な変化に伴い、グローバルモデルに課される要件も逐次変遷する。例えば、ユーザの嗜好変化に応じたキーボード予測や推薦モデルの更新、交通状況・移動傾向の変化に応じた需要予測、災害時や突発的イベント時における行動推定などでは、モデルを短時間で再適応させることが求められる。この動的な環境に追従し続けるには、一回一回の学習をいかに短時間で完了させるかが極めて重要となる。ゆえに、単なる精度向上だけでなく、より短い実経過時間で高精度を達することが連合学習で求められている。

連合学習に参加するデバイスは、利用可能な計算資源や通信環境が不均一であり、学習およびパラメータ送受信に要する遅延は個々に異なる。そのため、全参加者が一定の学習量を完了するまで待機する同期的なパラメータ集約を行うと、一部の低速デバイス (straggler) に全体の進行が律速され、結果として学習時間が大幅に増大する。この問

題を回避する一策として、集約までの時間に制約を設け、時間内に完了した参加者のみで更新を行う手法が挙げられる [2]。しかし、この方法では資源の乏しい参加者が排除され、それらが保持する固有データを活用できないという課題がある。これは特にクライアント間でのデータ分布が大きく異なる場合には著しい精度低下につながる。一方で、資源が豊富な参加者においては、制約時間に対して余剰リソースが生じるため、これを有効活用することでさらなる学習加速の余地が残されている。

そこで先行研究では、各参加者の学習量を時間制約内で精度向上を最大化するように個別かつ動的に決定する手法が提案されている [3–7]。これにより、資源の乏しい参加者は、学習ステップ数の削減によって時間制約内での寄与が可能となる。同時に、資源が豊富な参加者は、より多くの学習を実行することでシステム全体の精度向上に貢献できる。

一方で、特に通信時間が計算時間に対して無視できない環境では、通信圧縮率の選択によって、時間制約内で実行可能な学習量が大きく左右される。このような状況下では、各ラウンドに割り当てられた時間を「モデル学習」と「パラメータ送信」にいかにかに配分するかが、極めて重要な課題となる。学習に多くの時間を割けば更新量は増加するものの、短時間で送信を完了するために高い圧縮率が必要と

¹ 東京大学 The University of Tokyo

² 理化学研究所 RIKEN

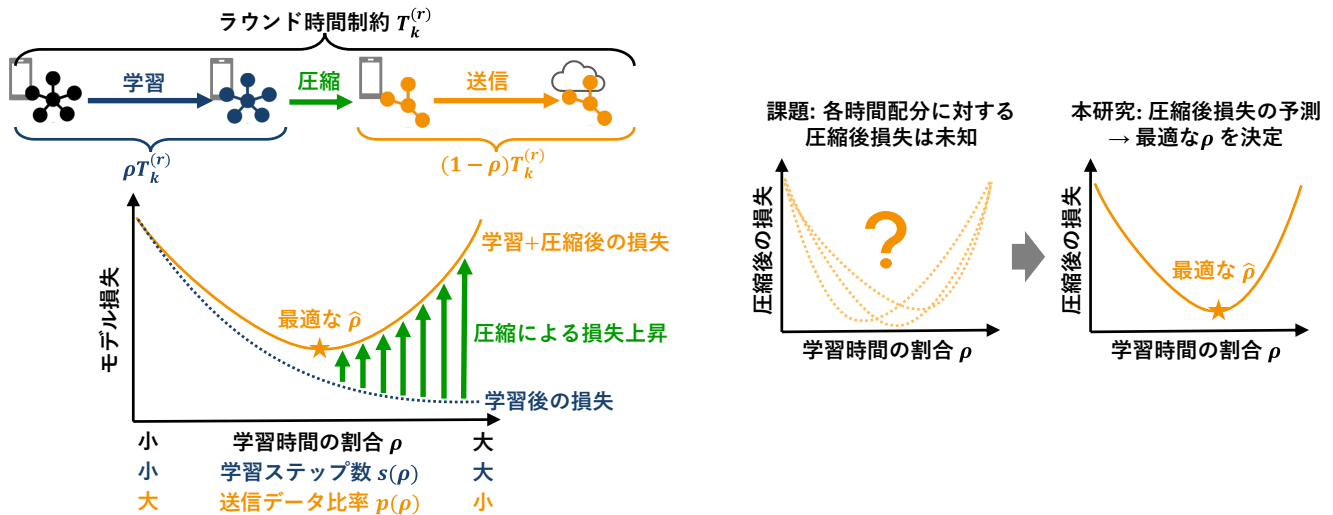


図 1: 本研究の背景と概要. 各ラウンドに時間制約が存在する連合学習環境では, ラウンド内で学習時間を長く割り当てる (学習時間比率 ρ を大きくする) と, 通信時間が不足するため高い通信圧縮率が求められ, 情報の毀損が大きくなって精度が下がる. 一方で, 通信に十分な時間を割り当てると, ローカル学習量が減少して精度が低下する. このように, 学習量と通信圧縮率の間にはトレードオフが存在し, 学習・通信の各時間配分に対して学習・通信圧縮後の損失がどのように変化するかは事前には未知である. そこで本研究では, それらの圧縮後損失を予測する手法を提案し, 各ラウンドにおける最適な学習量と通信圧縮率の決定を可能にする.

なり, 情報の毀損が大きくなる. 逆に, 通信に十分な時間を割いて圧縮誤差を抑えようとすれば, 肝心の学習量が不足する. 全体の学習時間を短縮するには, 環境に依存するこのトレードオフ関係を動的に予測し, 時間制約内で精度を最大化する学習量・通信圧縮率を適応的に決定する必要がある.

しかし, 先行研究では, 学習ステップ数のみを調整対象とし, 通信との統合的最適化を十分に検討していないものが多い [3–7]. また, 学習量と通信の統合最適化を扱う研究においても, クライアント間の計算資源・通信帯域の異質性には対応していない [8,9]. このため, 異質環境下において学習時間の短縮を目的とした, 学習量・通信圧縮率の統合最適化は, いまだ十分に検討されていない.

本研究では, 学習・通信の時間配分を変えたときの「学習・通信圧縮実行後のモデル損失」を予測することで, 最適な学習量および通信圧縮率を決定する手法 TALLOP (Training-communication Time ALlocation via compressed model LOss Prediction) を提案する. 具体的には, 学習初期のミニバッチ損失の推移から将来の学習ステップにおける損失を予測する LoSE (Loss-curve Smoothing and Extrapolation) と, パラメータの圧縮誤差ノルムから異なる圧縮率での圧縮による損失増加量を推定する CLEN (Compressed model Loss Estimation by error Norm) の 2 つを提案する. これらを組み合わせることで, 各クライアントは異なる時間配分に対する圧縮後損失を学習中に動的に予測し, 最小限の追加計算コストで, 自身の通信・計算環境に最適化された学習ステップ数と送信時のパラメータ圧縮率を選

択することが可能となる. さらに, 実験により, (1) 提案手法に基づく学習量・通信圧縮率の統合最適化を伴う連合学習は, それらの片方または両方を固定した場合と比べてより短い学習時間で高い精度に達すること, (2) 提案手法に伴う追加の計算オーバーヘッドは, 学習処理そのものに対して十分小さいこと, を示した.

2. 背景: 時間制約下における連合学習のワークロード最適化

連合学習において一定精度に達するまでの実時間を増大させる要因の一つが, 全体を律速する遅延デバイス (straggler) の存在である. 連合学習に参加するスマートフォンなどのエッジデバイスは, 演算性能・通信速度双方において高い多様性がある. そのため, 各ラウンドにおいて全参加者が同ステップ数を学習し同じサイズのパラメータを送信すると, ラウンドの所要時間は一部の低速デバイスに律速され, 他のデバイスは長い待機時間が生じるため, 同じ精度に達するまでにより多くの実時間を要する.

そのため, いくつかの研究では, ラウンドごとに時間制約を設けることを提案している [2]. この設定では, 時間制約内に学習・モデル更新送信を完了したクライアントのみモデルを集約し, 時間制約を超過して送られたモデル更新は無視される. しかしこれでは, 常に低速なデバイスは参加の機会が得られず, これらのデバイスが固有かつ有用なデータを持っていた場合のモデル精度が低下すると予期される.

この問題を解決するには, クライアントの計算資源や通

信環境に応じて、ラウンドの時間制約内で精度を最大化できるように、ワークロードを動的に調整する方策が有効である。これにより、資源の乏しい参加者も、学習ステップ数の削減によって時間制約内での貢献が可能となる。同時に、資源が豊富な参加者は、より多くの学習を実行することで全体の精度向上に貢献できる。

最も一般的なアプローチは、学習ステップ数の動的調整である。TimelyFL [3] や FedState [6] では、クライアントから報告された学習・通信時間に基づき、サーバ主導でラウンドごとの時間制約および学習エポック数をクライアントごとに決定する。一方で、FedBalancer [5] では単にモデル更新回数を最適化するだけでなく、クライアントは学習データのうち重要なものを重点的にサンプリングして学習している。

近年では、計算資源や通信帯域の異質性に対応するため、モデルアーキテクチャ自体を適応的に変化させる手法も注目されている。クライアントごとに異なるモデルアーキテクチャの学習を可能とした連合学習アルゴリズムとしては、サブモデル抽出による手法 [10] や知識蒸留ベースの手法 [11] などがある。AdaptiveFL [12] では、クライアントは通信帯域に応じたサイズのサブモデルを送受信し、サブモデル圧縮を意識した学習を行う。一方で ELFISH [13] では学習時間に着目し、演算回数とメモリ転送量に基づいたサブモデル選択を行っている。

しかし、これらの既存研究にはいくつかの課題が残されている。第一に、学習ステップ数と通信圧縮率の統合的な最適化が不十分である点である。通信帯域が限定的で通信時間が学習時間に対して無視できないクライアントでは、学習ステップ数に加えて通信圧縮率の選択が、同一時間内に達成可能な精度へ大きく影響する。しかし、先行研究の多くは学習ステップ数のみの調整を行い、通信との統合的最適化は十分に検討されていない [3, 5–7]。また、PyramidFL [4] では学習ステップ数に加えてアップロード時の枝刈り率を動的に調整しているが、これは更新の重要度のみから決定したものであり、時間制約のうちどのくらいを通信に費やすかという観点での最適化とは異なっている。

第二に、計算・通信資源の異質性への対応である。学習量・通信圧縮率の統合最適化を扱う研究も存在するが、これらは、異質環境下における学習時間短縮という問題設定を共有していない。AdaQuantFL [8] は、モデルパラメータの量子化が存在する仮定のもとで、総通信量に対する収束速度を最大化する量子化幅の決定方法を明らかにしているが、実時間に対する収束速度は議論されておらず、異質環境における学習時間短縮を直接目的とした手法ではない。また、FFL [9] は、各ラウンドにおける学習ステップ数と通信圧縮率の統合的な最適化を行い、収束時間短縮を目指す点で本研究と類似しているが、全クライアントの計算資

源および通信資源が同一であることを前提としており、異質環境への拡張は自明ではない。

これに対し本研究では、学習量・通信圧縮率の統合最適化を、各クライアント・各通信ラウンドに対して独立した最適化問題として定式化することで、非均質なクライアント環境においても個別クライアントに即した最適化を可能としている。すなわち、本研究の独自性は、(1) 計算資源および通信帯域に多様性を有する環境を対象とし、(2) 実経過時間に対する精度向上量改善を目的として、(3) 学習ステップ数と通信圧縮率を統合的に最適化する点にある。

3. 提案手法: TALLOP

本節では、各クライアントが各ラウンドの時間制約内において、精度を最大化するべく学習ステップ数・通信圧縮率を統合的に最適化する手法 TALLOP (training-communication Time Allocation via compressed model Loss Prediction) を提案する。まずはじめに本稿で前提とする連合学習の問題設定を定義し、学習・通信時間配分 ρ という一つの潜在変数を用いて学習ステップ数・通信圧縮率の統合最適化を定式化する。次に、学習・通信圧縮後のモデルの損失を動的に予測することで最適な学習ステップ数・通信圧縮率を決定する手順を示す。

3.1 問題設定

連合学習に参加するクライアント全体の集合を $\mathcal{N} := \{1, 2, \dots, N\}$ とする。各クライアント $k \in \mathcal{N}$ は、固有の学習データセット \mathcal{D}_k を保有する。連合学習では、各ラウンドにおいて同期的にグローバルモデルパラメータを更新する。ただし、低速クライアントによる律速を防ぐため、各ラウンド r には時間制約 $T^{(r)}$ を設け、制約時間を超過して到着した更新は集約対象から除外する。

ラウンド r 開始時点のグローバルモデルパラメータを $\theta^{(r)}$ とする。中央サーバは各ラウンドの開始時に、参加クライアント K 台をランダムに選択し、 $\theta^{(r)}$ を配布する。選択されたクライアント k は、バッチサイズ B のミニバッチ学習により、モデルを $s_k^{(r)}$ ステップ更新し、学習後パラメータ $\theta_k^{(r)}$ を得る。その後、学習前後の差分を圧縮し、圧縮後更新量を中央サーバへ送信する。更新量の圧縮法としては量子化 [14] や枝刈り [15] など種々の手法が存在するが、本研究では単純な疎化を採用する。すなわち、差分ベクトルのうち絶対値が大きい順に割合 $p_k^{(r)}$ の要素のみを送信するものとする。中央サーバは受信した更新を集約し、次ラウンドのグローバルモデルを生成する。また、送信するパラメータは1要素当たり 32 bit であると仮定する。

本研究では、ラウンド開始時のモデルダウンロードは通信圧縮せずに行うと仮定する。また、学習ステップ数 $s_k^{(r)}$ および送信データ比率 $p_k^{(r)}$ を制御する潜在変数として、ダウンロード後に残ったラウンド時間 $T_k^{(r)}$ のうち、学習に

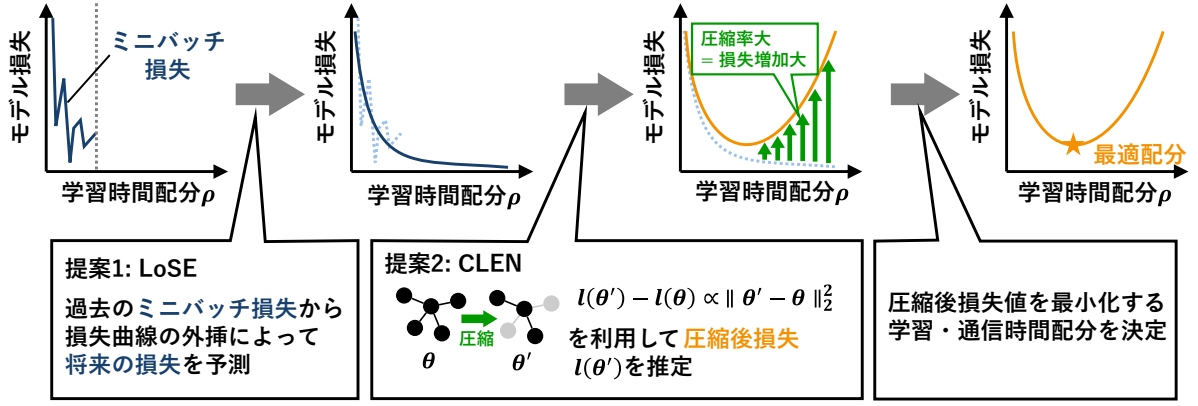


図 2: 提案手法 TALLOP の概要. まず, 学習初期に観測されるミニバッチ損失の推移に基づき, 損失曲線を外挿することで, 将来の学習ステップにおける損失を予測する (LoSE). 次に, パラメータ圧縮によって生じる誤差ノルムに基づいて, 各学習ステップ数まで学習した場合に, 残り時間内で通信可能な圧縮率を適用した際の損失増加量を推定する (CLEN). 最後に, LoSE と CLEN によって得られた各時間配分での圧縮後損失を比較し, その値を最小化する時間配分 ρ を決定する.

割り当てる割合を表す

$$\rho_k^{(r)} \in [0, 1] \quad (1)$$

を考える. クライアントは, 学習に割り当てられた時間 $\rho_k^{(r)} T_k^{(r)}$ で学習可能な最大ステップ数 $s_k^{(r)}$ を学習し, 残りの時間 $(1 - \rho_k^{(r)}) T_k^{(r)}$ で送信可能な最小の送信データ比率 $p_k^{(r)}$ で圧縮したモデルパラメータを送信すると仮定する. すなわち, $\rho_k^{(r)}$ を定めることで $s_k^{(r)}$ および $p_k^{(r)}$ は自動的に定まる. $\rho_k^{(r)}$ が大きいほど学習時間は長くなるが送信データ比率は小さくなり, 一方で小さいほど高精度な送信が可能だが学習時間は短くなるというトレードオフが存在する.

本研究の目的は, 各ラウンド・各クライアントに対し, 目標とする精度に達するまでの実時間を最小化するような $\rho_k^{(r)}$ の決定法を確立し, それによって $s_k^{(r)}$ および $p_k^{(r)}$ を決定することである. ただし, 本問題はラウンド単位・クライアント単位で独立に扱えるため, 本稿では, ラウンド r においてクライアント k が達成する損失を最小化する問題として定式化する. 以下では, まずは時間配分 $\rho_k^{(r)}$ を定めたときに学習ステップ数 $s_k^{(r)}$ および送信データ比率 $p_k^{(r)}$ がどのように定まるかを定式化したのち, 本研究において最小化すべき目的関数を $\rho_k^{(r)}$ の関数として定式化する. 以下では, クライアント k およびラウンド r が固定されているものとし, 煩雑さを避けるため必要に応じて添字 k と上付き添字 (r) を省略する.

3.1.1 時間配分 ρ と学習ステップ数 s の関係

クライアント k の計算性能を F FLOPS, ダウンロード帯域およびアップロード帯域をそれぞれ β_{down}, β_{up} bps とする. また, モデルパラメータ数を $|\theta|$, 1 サンプル学習あたりの浮動小数点演算回数を M , バッチサイズを B とする. なお, 本研究では学習の所要時間は浮動小数点演算回数に比例するという仮定をし, メモリ帯域など他の制約条

件は無視するものとする.

まず, モデルダウンロード時間は

$$T_k^{(down)} = \frac{32|\theta|}{\beta_{down}} \quad (2)$$

である. ダウンロード後に残る時間

$$T_k^{(r)} = T_k^{(r)} - T_k^{(down)} \quad (3)$$

を, 学習とアップロードで分配する.

学習・通信の時間配分を ρ とするとき, 学習に割り当てられる時間は $\rho T_k^{(r)}$ である. ローカル学習 1 ステップあたりの計算時間は $\frac{MB}{F}$ と近似できるため, 実行可能な学習ステップ数は

$$s(\rho) = \left\lfloor \frac{F}{MB} \rho T_k^{(r)} \right\rfloor \quad (4)$$

となる.

3.1.2 時間配分 ρ と送信データ比率 p の関係

学習後に残る時間 $(1 - \rho) T_k^{(r)}$ をアップロードに利用する.

更新量の圧縮法としては種々の手法が存在するが, 本研究では単純な疎化を採用する. すなわち, 差分ベクトルのうち絶対値が大きい順に割合 p の要素のみを送信するものとする.

$p < 1$ の場合, 更新量は疎ベクトルとして送信し, 非ゼロ要素の値とインデックスの組を転送する. インデックス表現に必要なビット幅は

$$8 \cdot \left\lceil \frac{\log_2(|\theta|)}{8} \right\rceil \quad (5)$$

である. したがって送信データ量は

$$\left(32 + 8 \left\lceil \frac{\log_2(|\theta|)}{8} \right\rceil \right) p |\theta| \quad (6)$$

となる. よってアップロード時間は

$$T_{up}(p) := \begin{cases} \frac{(32+8 \lceil \frac{\log_2(|\theta|)}{8} \rceil) p |\theta|}{\beta_{up}} & \text{if } p < 1 \\ \frac{32|\theta|}{\beta_{up}} & \text{if } p = 1 \end{cases} \quad (7)$$

と表される。

与えられた ρ に対し、制約時間内で送信可能な最大非ゼロ要素比率を $p(\rho)$ とする。すなわち、

$$p(\rho) = \max \left\{ p \in [0, 1] \mid T_{up}(p) \leq (1 - \rho) T_k^{(r)} \right\} \quad (8)$$

である。

3.1.3 最適化の目的関数

モデル θ について、データ集合 \mathcal{D} で学習した際の損失値を $l(\theta; \mathcal{D})$ と表す。また、以下では $l(\cdot; \mathcal{D}_k)$ を単に $l(\cdot)$ と表す。

$\theta^{(r)}$ をラウンド r 開始時のパラメータとして、クライアント k が s ステップ学習した後のモデルを

$$\theta_k^{(r)}(s) = \theta^{(r)} - \eta \sum_{t=0}^{s-1} \nabla l(\theta_k^{(r)}(t); \mathcal{B}_t) \quad (9)$$

とする。ただし、 \mathcal{B}_t を t ステップ目のミニバッチとする。さらに、その差分に対して非ゼロ比率 p の疎化を施したモデルを

$$\theta_k^{(r)}(s, p) = \theta^{(r)} + S(\theta_k^{(r)}(s) - \theta^{(r)}; p) \quad (10)$$

とする。ここで、 $S(\cdot; p)$ は、絶対値が大きい順に割合 p の要素のみを保持する疎化演算子である。このとき、時間配分 ρ に対するラウンド終了時損失を

$$l_k^{(r)}(\rho) = l(\theta_k^{(r)}(s_k^{(r)}(\rho), p_k^{(r)}(\rho))) \quad (11)$$

と定義する。

本研究の目的は、各ラウンド r および各クライアント k に対し

$$\operatorname{argmin}_{\rho \in [0, 1]} l_k^{(r)}(\rho) \quad (12)$$

を求めることである。 $\rho = 0$ では学習を行わず、 ρ が過大であれば通信時間不足により送信更新が極端に疎になる。したがって、一般にはその中間に最適な ρ が存在する。

3.2 TALLOP の構成

式 (12) を解いて最適な ρ を直接求めることには、以下の困難がある。

- $l_k^{(r)}(\rho)$ を厳密に評価するには、複数の候補 ρ に対して学習・圧縮後モデルの損失を逐次計算する必要がある。正確な損失の計算には十分なサンプル数に対して推論を実行する必要がある、これを複数の ρ に対して行うことは追加計算時間が大きく、時間効率の改善という本研究の目的に反する。
- 学習が $s(\rho_0)$ ステップまで進んでいる時点では、将来の学習ステップ $s(\rho) > s(\rho_0)$ に対応する $l_k^{(r)}(\rho)$ は未

知であり、現時点で最適学習停止時刻を厳密に判断できない。

そこで本研究では、現在および未来の時刻における $l_k^{(r)}(\rho)$ を予測することで、最適な ρ を決定する手法 TALLOP を提案する。本手法は以下の 2 要素から構成される。

- **LoSE (Loss-curve Smoothing and Extrapolation)**: ミニバッチ損失系列を平滑化・外挿し、将来ステップの未圧縮損失 $l(\theta_k^{(r)}(s))$ を予測する。
- **CLEN (Compressed model Loss Estimation by error Norm)**: 圧縮によるパラメータ誤差ノルムから、圧縮後損失 $l(\theta_k^{(r)}(s, p))$ を推定する。

これらを組み合わせることで、任意の ρ に対する圧縮後損失の予測値

$$\tilde{l}_k^{(r)}(\rho) \approx l_k^{(r)}(\rho) \quad (13)$$

を構成し、三値探索によって

$$\hat{\rho}_k^{(r)} = \operatorname{argmin}_{\rho \in [0, 1]} \tilde{l}_k^{(r)}(\rho) \quad (14)$$

が求められる。最終的に、得られた $\hat{\rho}_k^{(r)}$ に対応する $s(\hat{\rho}_k^{(r)})$ ステップまで学習を継続し、非ゼロ比率 $p(\hat{\rho}_k^{(r)})$ により圧縮して更新を送信する。

以下では、まず各要素技術を述べ、その後に全体アルゴリズムを示す。

3.2.1 LoSE: 損失曲線の平滑化と外挿

任意のステップ数 s に対する $l(\theta_k^{(r)}(s))$ の推定法を示す。未到達ステップに対して損失を直接得ることは不可能であり、到達済みステップすべてに対し評価データで損失測定を行うことも高コストである。

そこで、 s_0 番目の学習ステップ完了時点において得られたミニバッチ損失の履歴

$$\left\{ l(\theta_k^{(r)}(s); \mathcal{B}_s) \mid s \in [1, s_0] \right\} \quad (15)$$

を用いて推定を行う。ただし、観測値をそのまま用いると、バッチ依存ノイズが大きく、将来時刻 $s > s_0$ の値も得られない。

本研究では真の損失曲線を、単調減少かつ漸近的に収束する関数形としてモデル化する。具体的には、損失曲線を

$$f_{loss}(s; l_{inf}, a, b, c) = l_{inf} + \frac{a}{(s + b)^c} \quad (16)$$

で表し、観測されたミニバッチ損失系列に対して最も整合的となるようにパラメータ l_{inf}, a, b, c を推定する。ここで、 l_{inf} は収束時の損失値、 a は初期ステップ付近の特異性を緩和するシフト項、 b は初期誤差の大きさ、 c は収束速度を表す係数である。これにより、観測ノイズを含む損失系列を平滑化するとともに、未観測ステップに対する外挿値

$$\tilde{l}(\theta_k^{(r)}(s)) := f_{loss}(s; \hat{l}_{inf}, \hat{a}, \hat{b}, \hat{c}) \approx l(\theta_k^{(r)}(s)) \quad (17)$$

を得る。ここで $\hat{l}_{inf}, \hat{a}, \hat{b}, \hat{c}$ は推定された最適パラメータである。

3.2.2 CLEN: 圧縮誤差ノルムに基づく圧縮後モデルの損失値推定

前節により、学習前後のモデルの損失の推定値 $\tilde{l}(\theta^{(r)})$, $\tilde{l}(\theta_k^{(r)}(s))$ が得られる。次に、これらを用いて、 $l(\theta_k^{(r)}(s, p))$ を推定する。まずはじめに、学習済みモデル θ に微小摂動 $\Delta\theta$ を加えたとき、損失値の上昇量が摂動の大きさの二乗に比例することを理論的・経験的に考察する。次に、この性質を用いて $\tilde{l}(\theta^{(r)})$ と $\tilde{l}(\theta_k^{(r)}(s))$ から $l(\theta_k^{(r)}(s, p))$ を推定する手法を示す。

損失値とパラメータ摂動の関係。 学習済みモデル θ に微小摂動 $\Delta\theta$ を加えたとき、損失関数 $l(\cdot)$ が θ の近傍で2階微分可能であれば、テイラー展開より

$$l(\theta + \Delta\theta) = l(\theta) + \nabla l(\theta)^\top \Delta\theta + \frac{1}{2} \Delta\theta^\top H(\theta) \Delta\theta + o(\|\Delta\theta\|_2^2) \quad (18)$$

が成り立つ。ここで $H(\theta)$ はヘッセ行列である。学習済みモデルでは $\nabla l(\theta) \approx 0$ とみなせるため、

$$\Delta l := l(\theta + \Delta\theta) - l(\theta) \approx \frac{1}{2} \Delta\theta^\top H(\theta) \Delta\theta \quad (19)$$

となる。さらに、 $\Delta\theta$ が特定方向へ偏らない場合、平均的には

$$\Delta l \propto \|\Delta\theta\|_2^2 \quad (20)$$

と近似できる。

圧縮後損失値の推定。 クライアント k が s ステップ学習した後のモデル $\theta_k^{(r)}(s)$ に対し、圧縮による摂動を

$$\Delta\theta_k^{(r)}(s, p) = \theta_k^{(r)}(s, p) - \theta_k^{(r)}(s) \quad (21)$$

と定義する。このとき、

$$l(\theta_k^{(r)}(s, p)) - l(\theta_k^{(r)}(s)) \approx \lambda \|\Delta\theta_k^{(r)}(s, p)\|_2^2 \quad (22)$$

と近似する。比例係数 $\lambda > 0$ は、 $\theta_k^{(r)}(s, 0) = \theta^{(r)}$ より、

$$\lambda = \frac{l(\theta^{(r)}) - l(\theta_k^{(r)}(s))}{\|\theta_k^{(r)}(s) - \theta^{(r)}\|_2^2} \quad (23)$$

と推定できる。

したがって、 $l(\theta_k^{(r)}(s, p))$ は、

$$\tilde{l}(\theta_k^{(r)}(s, p)) := \tilde{l}(\theta^{(r)}) + \frac{\tilde{l}(\theta^{(r)}) - \tilde{l}(\theta_k^{(r)}(s))}{\|\theta_k^{(r)}(s) - \theta^{(r)}\|_2^2} \|\theta_k^{(r)}(s, p) - \theta^{(r)}\|_2^2 \quad (24)$$

として近似される。

この近似の妥当性については、実験的にも検証した。対象モデルとして、MobileNetV2 [16] および Vision Transformer [17] の2種類を用いた。各モデルについて、学習前のパラメータを θ_0 、CIFAR-10 [18] に対して1エポック学習後のパラメータを θ とする。さらに、式(10)に基づき、非ゼロ比率 p で疎化を施したモデルパラメータ $\theta(p)$ を

$$\theta(p) = \theta_0 + S(\theta - \theta_0; p) \quad (25)$$

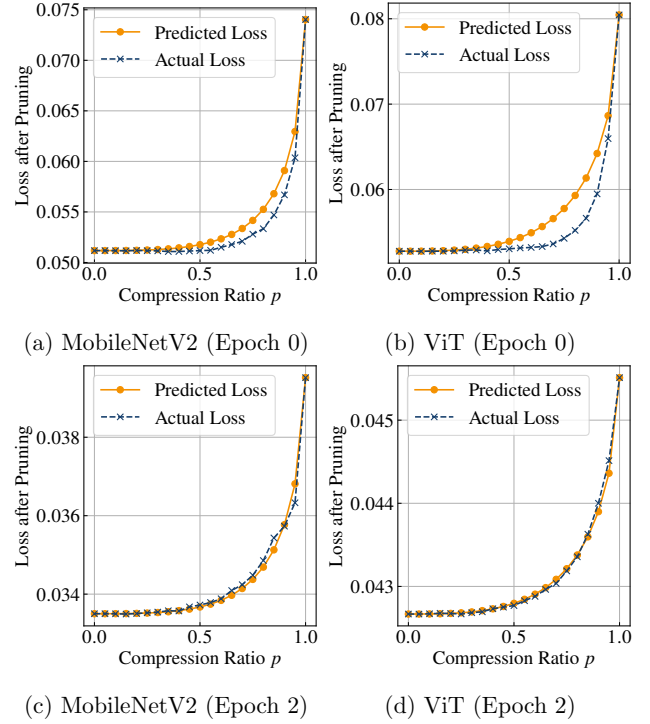


図 3: 多様な非ゼロ比率 p における疎化による圧縮後のモデルの損失について、提案手法 CLEN (*Predicted Loss*) は実際の損失 (*Actual Loss*) を正確に予測している。

と定義する。このとき、式(24)によって算出される $\theta(p)$ の予測損失と、実際に $\theta(p)$ を用いて評価した真の損失値とを比較した。また、学習初期段階だけでなく、学習途中における近似制度も確認するため、 θ_0 として初期段階のパラメータ (Epoch 0) に加え、2エポック学習後のパラメータ (Epoch 2) を用いた場合についても同様の評価を行った。

図 3 は、各非ゼロ比率 p に対して、 $\theta(p)$ の実際の損失値と、提案手法により予測された損失値とを比較した結果を示している。全体として両者は良好に一致しており、提案手法が圧縮後モデルの損失値を高い精度で推定可能であることが確認できる。一方で、特に Epoch 0 の条件では、予測損失が実際の損失値をやや上回る傾向が見られた。この乖離をさらに低減し、学習初期を含む広い条件下で予測精度を向上させることは、今後の課題である。

3.2.3 TALLOP の統合アルゴリズム

最後に、上記2手法を組み合わせ、各ラウンドにおいて最適な学習・通信時間配分 ρ を決定する。

もっとも単純な手法は、毎学習ステップ完了ごとに、それまでのミニバッチ損失の履歴から LoSE と CLEN を用いて $\tilde{l}_k^{(r)}(\rho) := \tilde{l}(\theta_k^{(r)}(s(\rho), p(\rho)))$ を構成する。この関数は $\rho \in [0, 1]$ で単峰性を有すると仮定し、 ρ に対して三分探索を適用することで、最小値を与える $\hat{\rho}$ を求める。

ただし、上記推定処理を毎学習ステップ後に実行すると追加計算コストが無視できない。そこで本研究では、適切なタイミングで推定処理を1回のみ実行する。非圧縮のモ

デルパラメータをアップロードすることができる最大の学習ステップ数を s_0 とする。すなわち,

$$s_0 = \max_{s \in \mathbb{Z}_{\geq 0}} s \quad \text{s.t.} \quad T_{\text{train}}(s) + T_{\text{up}}(1) \leq T^{(r)} \quad (26)$$

とする。学習後の損失値が学習ステップ数に対して単調減少すると仮定すると、学習ステップ数が s_0 以下であれば常に非圧縮のモデルパラメータを送信できるため、圧縮による精度劣化が起こらない。すなわち、 $s_0 = s(\rho_0)$ たる ρ_0 を考えると、 $l_k^{(r)}(\rho)$ は $\rho < \rho_0$ で単調減少することが保証される。したがって、 $l_k^{(r)}$ は ρ_0 以降で最小となるため、本研究では $s = s_0$ に到達した時点で一連の推定を実施し、最適な時間配分 $\rho(\geq \rho_0)$ を決定する。これによって、推定処理が最適な学習ステップ数を過ぎてから実行されることを避けつつも、ミニバッチ損失をより多く集めることで損失曲線の外挿をより正確に行うことができる。

さらに、式 (24) にも軽微な変更を加える。この計算式に基づいて異なる s について $\tilde{l}(\theta_k^{(r)}(s, p))$ を計算するには $\theta_k^{(r)}(s)$ および $\theta_k^{(r)}(s, p)$ が既知である必要があるが、これは未来の s については未知であるうえ、過去の s についてもすべての学習ステップに対応するモデルパラメータを保持しておくことは非現実的である。そこで、これらの代わりに、推定処理実行時点でのパラメータ $\theta_k^{(r)}(s_0)$ および $\theta_k^{(r)}(s_0, p)$ によって代用して

$$\tilde{l}'(\theta_k^{(r)}(s, p)) := \tilde{l}(\theta^{(r)}) + \frac{\tilde{l}(\theta^{(r)}) - \tilde{l}(\theta_k^{(r)}(s))}{\|\theta_k^{(r)}(s_0) - \theta^{(r)}\|_2^2} \|\theta_k^{(r)}(s_0, p) - \theta^{(r)}\|_2^2 \quad (27)$$

と定義される \tilde{l}' を用いる。

これらの変更を含め、TALLOP による学習ステップ数 s および送信データ比率 p の決定手順全体を Algorithm 1 に示す。

4. 実験と考察

次に、提案手法の有効性を実証するための評価を行う。まず、提案手法 TALLOP に基づいた学習量・通信圧縮率の統合的な最適化を伴った連合学習が、その片方または双方を固定した場合よりも短い実時間で十分な精度に達することを示す。次に、演算回数の見積もりを通じて、提案手法が行う時間配分がもたらす計算オーバーヘッドが学習コストに比べて十分小さく、無視できるものであることを示す。

4.1 学習実時間と達成精度の関係の評価

次に、提案手法 TALLOP を用いることで、既存の連合学習手法や固定的なモデル圧縮よりも、学習の実時間に対する精度向上の効率 (実時間効率) が大幅に改善されることを実験によって示す。画像分類タスクを対象とし、モデルアーキテクチャは MobileNetV2 [16] を採用した。データセットには CIFAR-10 [18] を用い、総クライアント数を

Algorithm 1 TALLOP による学習ステップ数 s および送信データ比率 p の決定手順

```

1: function PREDICTCOMPRESSEDLOSS( $\rho$ )
2:    $\tilde{l}(\theta^{(r)}) \leftarrow f_{\text{loss}}(0; \hat{l}_{\text{inf}}, \hat{a}, \hat{b}, \hat{c})$ 
3:    $\tilde{l}(\theta_k^{(r)}(s)) \leftarrow f_{\text{loss}}(s; \hat{l}_{\text{inf}}, \hat{a}, \hat{b}, \hat{c})$ 
4:    $\theta_k^{(r)}(s_0, p(\rho)) \leftarrow \theta^{(r)} + S(\theta_k^{(r)}(s_0) - \theta^{(r)}; p(\rho))$ 
5:   Calculate  $\tilde{l}'(\theta_k^{(r)}(s(\rho), p(\rho)))$  based on Eq. (27)
6:   return  $\tilde{l}'(\theta_k^{(r)}(s(\rho), p(\rho)))$ 
7: end function

8: function TALLOP( $\mathcal{L}_{\text{history}}$ )
9:    $\hat{l}_{\text{inf}}, \hat{a}, \hat{b}, \hat{c} \leftarrow \text{Fit past minibatch losses } \mathcal{L}_{\text{history}} \text{ with Eq. (16)}$ 
10:   $\rho_{\min} \leftarrow 0, \quad \rho_{\max} \leftarrow 1$ 
11:  while  $\rho_{\max} - \rho_{\min} > \varepsilon$  do
12:     $\rho_1 \leftarrow \frac{2\rho_{\min} + \rho_{\max}}{3}$ 
13:     $\rho_2 \leftarrow \frac{\rho_{\min} + 2\rho_{\max}}{3}$ 
14:     $l_1 \leftarrow \text{PREDICTCOMPRESSEDLOSS}(\rho_1)$ 
15:     $l_2 \leftarrow \text{PREDICTCOMPRESSEDLOSS}(\rho_2)$ 
16:    if  $l_1 < l_2$  then
17:       $\rho_{\max} \leftarrow \rho_2$ 
18:    else
19:       $\rho_{\min} \leftarrow \rho_2$ 
20:    end if
21:  end while
22:  return  $s(\rho_{\min}), p(\rho_{\min})$ 
23: end function

24: function TRAINANDUPLOAD
25:   $\theta^{(r)} \leftarrow \text{Receive global model from the server}$ 
26:   $s \leftarrow 1$ 
27:   $\mathcal{L}_{\text{history}} \leftarrow \emptyset$ 
28:   $\hat{s} \leftarrow \infty, \hat{p} \leftarrow 0$ 
29:  while true do
30:     $l_{\text{batch}} \leftarrow l(\theta_k^{(r)}(s-1); \mathcal{B}_s)$ 
31:     $\theta_k^{(r)}(s) \leftarrow \theta^{(r)} - \eta \nabla l_{\text{batch}}$ 
32:    Add  $(s, l_{\text{batch}})$  to  $\mathcal{L}_{\text{history}}$ 
33:    if  $s = s_0$  then
34:       $\hat{s}, \hat{p} \leftarrow \text{TALLOP}(\mathcal{L}_{\text{history}})$ 
35:    end if
36:    if  $s = \hat{s}$  then
37:      Transmit  $S(\theta_k^{(r)}(\hat{s}) - \theta^{(r)}; \hat{p})$  to the server
38:      return
39:    end if
40:     $s \leftarrow s + 1$ 
41:  end while
42: end function

```

表 1: 実験におけるクライアントの計算速度・通信帯域の分布. m および σ は対数正規分布の平均および対数標準偏差を表す.

	計算速度 [GFLOPS]				通信帯域 [Mbps]			
	m	σ	最小	最大	m	σ	最小	最大
設定 1	20	0.5	5.48	61.64	80	1.0	3.83	491.25
設定 2	200	0.5	54.8	616.4	80	1.0	3.83	491.25

100 とする. 各クライアントのデータ分布には $\alpha = 0.5$ の Dirichlet 分布を適用し, Non-IID 性を再現する. なお, 各ラウンドにおいては, 全クライアントの中からランダムに選択された 20 クライアントのみが学習に参加する.

参加者の計算資源や通信帯域については, 対数正規分布で分布すると仮定し, 平均 m と対数標準偏差 σ に対して

$$X = \exp\left(\ln m - \frac{\sigma^2}{2} + \sigma Z\right), \quad Z \sim \mathcal{N}(0, 1) \quad (28)$$

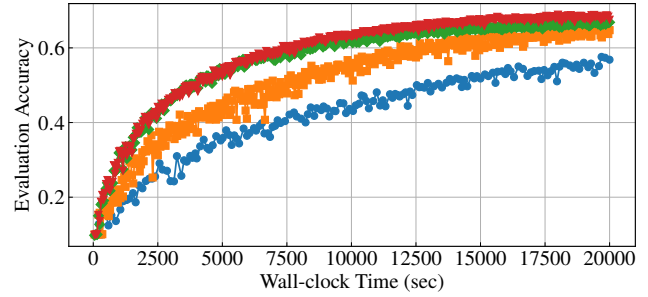
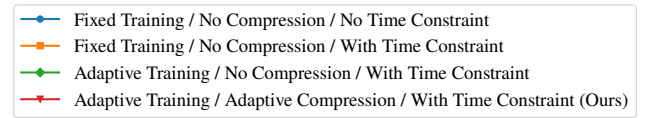
として生成した. 計算速度が通信速度に対して比較的遅いパターン, 速いパターンの 2 通りを用意した. それぞれの設定における具体的な分布は表 1 にまとめた.

本実験では, 以下の 4 つの手法について, 学習開始からの経過実時間に対する精度改善量を比較した.

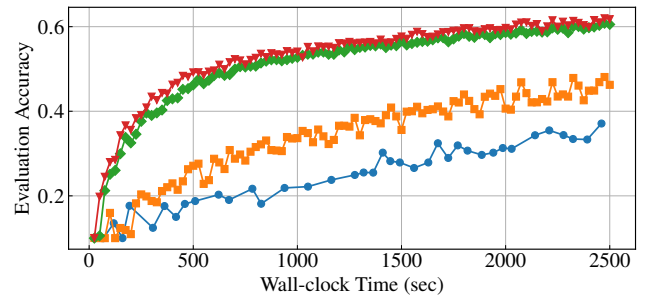
- (i) **固定学習量・通信圧縮なし・時間制約なし**: 全クライアントが一貫して 1 エポックの学習を行い, 非圧縮のモデルパラメータを送信する. ラウンドの終了時間は, 最も所要時間の長いクライアントに依存する.
- (ii) **固定学習量・通信圧縮なし・時間制約あり**: 全クライアントが一貫して 1 エポックの学習を行い, 非圧縮のモデルパラメータを送信するが, 低速クライアントによる遅延を防ぐためにラウンドに一律の制限時間を設ける. 制限時間を超過したモデルのアップロードは破棄される.
- (iii) **適応的学習量・通信圧縮なし・時間制約あり**: 各クライアントはラウンドの制限時間内で精度を最大化するように学習ステップ数を動的に調整するが, 通信圧縮は一貫して行わない.
- (iv) **提案手法 TALLOP**: 提案手法 TALLOP に基づき, ラウンドの制限時間内で精度を最大化するように, 学習量と通信圧縮率を統合的に最適化する.

図 4 は, これら 4 手法について, 経過実時間に対するグローバルモデル精度の推移を示したものである. ラウンド時間制約を設けない場合 (青・円), 低速クライアントが律速となるため, 1 ラウンドあたりの所要時間が長く, 実時間ベースでは学習完了までに多くの時間を要している. これに対し, 1 ラウンドあたりの時間制約を導入した場合 (橙・正方形), この問題は大きく緩和されている.

一方, クライアントごとにワークロードを動的に調整し



(a) 設定 1: 計算が低速



(b) 設定 2: 計算が高速

図 4: 各手法における経過実時間と精度の関係. 提案手法による学習量・通信圧縮率の統合最適化 (赤・三角) により, それらを固定した場合 (青・円, 橙・正方形) や, 学習量のみを最適化した場合 (緑・菱形) と比較して, より短時間で高精度に到達することが確認された.

た場合, 高速なクライアントにはより多くの学習ステップを割り当て, 低速なクライアントにも実行可能な範囲で学習機会を与えられる. その結果, 固定的なワークロードを用いた場合と比較して, 1 ラウンドあたりの精度向上量が大きく増加している (赤・三角, 緑・菱形). さらに, 通信圧縮を行わず学習量のみを最適化した場合と比較しても, 提案手法により学習量と通信圧縮率を統合最適化したほうが, 一定精度へ到達するまでの時間が短いことが確認できる.

さらに, 2 つのクライアント設定を比較すると, 計算性能が高い設定 2 のほうが, 手法間の学習速度差がより顕著に現れている. これは, 計算時間が短い環境では, 同一ラウンド時間内に実行可能な学習ステップ数が増加し, ワークロード最適化によって高速クライアントの余剰計算資源をより有効に活用できるためである. 加えて, 計算時間に対して通信時間の占める割合が相対的に大きくなるため, 通信圧縮による時間短縮効果も現れやすい. したがって, 本提案手法は, 通信時間が全体性能の律速となりやすい環境において, 特に有効であると考えられる.

4.2 提案手法の計算オーバーヘッド解析

次に、提案手法 TALLOP の導入によって追加的に生じる計算オーバーヘッドについて、Algorithm 1 に基づき演算回数の観点から解析を行う。本節では、最適化アルゴリズムに要する追加計算量が、通常のモデル学習処理に要する計算量と比較して十分に軽微であることを定量的に示す。

TALLOP では、時間配分 $\rho \in [0, 1]$ を変数として $\text{PREDICTCOMPRESSEDLOSS}(\rho)$ を評価し、その値に基づいて三値探索を行う。探索区間の終了条件を区間幅 ε とすると、 $\text{PREDICTCOMPRESSEDLOSS}(\rho)$ の評価回数は $O(\log(1/\varepsilon))$ である。

次に、 $\text{PREDICTCOMPRESSEDLOSS}(\rho)$ 1 回あたりの計算量を見積もる。支配的な処理は、(i) スパース化後パラメータ $\theta_k^{(r)}(s_0, p(\rho))$ の生成と (ii) 元のパラメータ $\theta^{(r)}$ との ℓ_2 距離計算の 2 つである。モデルの総パラメータ数を $P = |\theta|$ とすると、(i) では重要度に基づく選択またはソート処理が必要となるため、 $O(P \log P)$ となる。また、(ii) の距離計算は全要素に対する走査であるため、 $O(P)$ である。

以上より、TALLOP 全体の追加計算量は、1 ラウンドあたり $O(P \log P \log \frac{1}{\varepsilon})$ と見積もられる。ここで例として、MobileNetV2 を用いる場合を考え、具体的な演算回数を見積もる。MobileNetV2 を用いる場合、パラメータ数は $P \approx 2 \times 10^6$ である。また、 $\log \frac{1}{\varepsilon} \approx 10$ とすると、TALLOP の追加計算量は 1 ラウンドあたり概ね 10^8 から 10^9 程度のオーダーとなる。一方、MobileNetV2 の学習時に必要な積和演算回数は、1 サンプルあたり約 9×10^8 である。したがって、TALLOP による追加計算は、多く見積もっても数サンプル分の学習計算量に相当する。通常の連合学習では、各ラウンドで多数サンプルに対するローカル学習を実行するため、この追加オーバーヘッドが総学習時間に与える影響は僅少であるといえる。

5. まとめ

本稿では、時間制約付きの連合学習における学習量・通信圧縮率のトレードオフ関係に着目し、将来的な学習・圧縮後のモデル損失の予測に基づいて、最適な学習ステップ数および通信圧縮率を決定する手法 TALLOP を提案した。TALLOP では、学習初期のミニバッチ損失の推移から将来の学習ステップにおける損失を予測し (LoSE)、さらに各学習ステップにおいて、残り時間内で送信可能なパラメータ圧縮率により圧縮した場合のモデル損失を予測する (CLEN) ことで、学習・圧縮後の損失を最小化する学習ステップ数と通信圧縮率の組み合わせを決定する。さらに、実験により、(1) 提案手法に基づく学習量・通信圧縮率の統合最適化を伴う連合学習は、それらの片方または両方を固定した場合と比較して、より短い学習時間で高い精度に到達すること、(2) 提案手法に伴う追加の計算オーバーヘッドは、学習処理そのものに対して十分小さいこと、を

示した。

本研究では、ラウンドの時間制約が所与である状況を想定したが、実際にはこの制約自体の設定が学習の収束速度に多大な影響を及ぼすと考えられる。したがって、システム全体の状況に応じた最適なラウンド時間制約の動的決定手法の確立が、今後の重要な課題である。さらに、計算・通信速度についても、同一クライアント内での動的な状態変化や、サーバ側での通信輻輳を考慮した、より複雑な実環境モデル下での検証が求められる。

謝辞 本研究の一部は JST CREST JPMJCR21D2 の支援による。

参考文献

- [1] McMahan, B., Moore, E., Ramage, D., Hampson, S. and Arcas, B. A. y.: Communication-Efficient Learning of Deep Networks from Decentralized Data, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1273–1282 (2017).
- [2] Lee, J., Ko, H. and Pack, S.: Adaptive Deadline Determination for Mobile Device Selection in Federated Learning, *IEEE Transactions on Vehicular Technology*, Vol. 71, No. 3, pp. 3367–3371 (online), DOI: 10.1109/TVT.2021.3136308 (2022).
- [3] Zhang, T., Gao, L., Lee, S., Zhang, M. and Avestimehr, S.: TimelyFL: Heterogeneity-aware Asynchronous Federated Learning with Adaptive Partial Training, *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 5064–5073 (online), DOI: 10.1109/CVPRW59228.2023.00535 (2023).
- [4] Li, C., Zeng, X., Zhang, M. and Cao, Z.: PyramidFL: a fine-grained client selection framework for efficient federated learning, *Proceedings of the 28th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pp. 158–171 (online), DOI: 10.1145/3495243.3517017 (2022).
- [5] Shin, J., Li, Y., Liu, Y. and Lee, S.-J.: FedBalancer: data and pace control for efficient federated learning on heterogeneous clients, *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, pp. 436–449 (online), DOI: 10.1145/3498361.3538917 (2022).
- [6] Zhang, D., Sun, W., Zheng, Z.-A., Chen, W. and He, S.: Adaptive device sampling and deadline determination for cloud-based heterogeneous federated learning, *Journal of Cloud Computing*, Vol. 12, No. 1, p. 153 (online), DOI: 10.1186/s13677-023-00515-6 (2023).
- [7] Narmadha, K. and Varalakshmi, P.: FedEff: efficient federated learning with optimal local epochs for heterogeneous clients, *Scientific Reports*, Vol. 15, No. 1, p. 38860 (online), DOI: 10.1038/s41598-025-22672-1 (2025).
- [8] Jhunjhunwala, D., Gadhikar, A., Joshi, G. and Eldar, Y. C.: Adaptive Quantization of Model Updates for Communication-Efficient Federated Learning, *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3110–3114 (online), DOI: 10.1109/ICASSP39728.2021.9413697 (2021).
- [9] Nori, M. K., Yun, S. and Kim, I.-M.: Fast Federated Learning by Balancing Communication Trade-Offs, *IEEE Transactions on Communications*,

- Vol. 69, No. 8, pp. 5168–5182 (online), DOI: 10.1109/TCOMM.2021.3083316 (2021).
- [10] Diao, E., Ding, J. and Tarokh, V.: HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients, *arXiv preprint arXiv:2010.01264*, No. arXiv:2010.01264 (online), DOI: 10.48550/arXiv.2010.01264 (2021).
- [11] Li, D. and Wang, J.: FedMD: Heterogenous Federated Learning via Model Distillation, *arXiv preprint arXiv:1910.03581*, No. arXiv:1910.03581 (online), DOI: 10.48550/arXiv.1910.03581 (2019).
- [12] Liu, G., Lin, W., Huang, T., Shi, F., Wu, W. and Shen, L.: AdaptiveFL: Communication-Adaptive Federated Learning Under Dynamic Bandwidth, *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 36, No. 9, pp. 17199–17211 (online), DOI: 10.1109/TNNLS.2025.3560656 (2025).
- [13] Xu, Z., Yang, Z., Xiong, J., Yang, J. and Chen, X.: *ELFISH: Resource-Aware Federated Learning on Heterogeneous Edge Devices* (2019).
- [14] Wen, W., Xu, C., Yan, F., Wu, C., Wang, Y., Chen, Y. and Li, H.: TernGrad: Ternary Gradients to Reduce Communication in Distributed Deep Learning, *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 30 (2017).
- [15] Li, A., Sun, J., Wang, B., Duan, L., Li, S., Chen, Y. and Li, H.: LotteryFL: Personalized and Communication-Efficient Federated Learning with Lottery Ticket Hypothesis on Non-IID Datasets, *arXiv preprint arXiv:2008.03371*, No. arXiv:2008.03371 (online), DOI: 10.48550/arXiv.2008.03371 (2020).
- [16] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and Chen, L.-C.: MobileNetV2: Inverted Residuals and Linear Bottlenecks, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4510–4520 (2018).
- [17] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J. and Houtsby, N.: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, *arXiv preprint arXiv:2010.11929* (2020).
- [18] Krizhevsky, A. and Hinton, G.: Learning Multiple Layers of Features from Tiny Images (2009).