

2025 年度 アジャイル レポート課題

24G1133 安尾 晃貴

はじめに

0.1 実験の目的

現代社会において IoT 機器の普及は著しく、様々なセンサーから大量のデータが生成されている [1]. これらのデータは記録や解析、遠隔伝送といった用途で広く活用される一方で、ストレージ容量の制約が現実上の課題となっている [2]. 特に Arduino R4 WiFi (SRAM 32KB, フラッシュメモリ 256KB) のような組み込みマイコンボードでは、大規模なデータやディープラーニングを扱う上で容量が制約となる.

本研究において、カラーセンサーの補正を目的としたニューラルネットワークを Arduino R4 WiFi に実装を試みた際に、メモリ不足の問題が発生した. 具体的には、入力層 3 ノードは、隠れ層 70 ノード × 2 層、出力層 3 ノードの 3 層ニューラルネットワークの全結合層を実装しようとしたところ、重み行列だけで $(3 \times 70 + 70 \times 70 + 70 \times 3) \times 4$ バイト = 21,280 バイトを消費する. さらにバイアス項 $(70 + 70 + 3) \times 4$ バイト = 572 バイト、活性化関数の中間出力やその他の変数を含めると、総メモリ使用量は約 22KB 以上となり、SRAM 容量 (32KB) の約 7 割を占有してしまう. 加えて、プログラム本体や WiFi 通信用のライブラリ、スタック領域なども考慮すると、実際に利用可能なメモリは更に限られ、プログラムが正常に動作しない事態に陥った.

したがって、限られたメモリ資源の中でニューラルネットワークモデルを実装するためには、重み行列を効率的に圧縮する技術が不可欠である. 特にニューラルネットワークでは、学習の過程で多くの重みがほぼ 0 となり、重み行列が疎行列として表されることが多い [3]. 疎行列とは大部分の要素が 0 である行列であるため、0 以外の値とその位置情報のみを保持することで保存すべきデータ量を大幅に削減でき、メモリ効率の向上が期待できる [4].

本実験では代表的なデータ圧縮方式を調査し、その性能を比較・評価することで、圧縮率と復元精度の両立を目指す. 具体的には、疎データに適した行列表現 (CSR 形式) や値の量子化を組合せて実装し、圧縮後のデータサイズと復元時の MSE (平均二乗誤差)、さらに圧縮によるメモリ削減率を定量的に測定する. これにより、Arduino R4 WiFi 等の組み込み環境においても実用可能なニューラルネットワークモデルの実装に必要なデータ圧縮手法の有効性を明らかにし、IoT デバイス上でより大きな AI の実装に向けた知見を得ることを目的とする.

0.2 実験の理論

本実験で検討する 3 つの圧縮方式について理論的な背景と選定理由を述べる.

まず、本研究ではスペース化の導入について検討する. スペース化とは、データ中の重要度の低い要素を選択的に 0 にして疎な表現へ変換する処理を指す. ニューラルネットワークにおいては、学習により得られた重み行列の中には出力への寄与が小さい要素が多数存在することが知られており [3]、これらを 0 に設定することでモデルの表現能力を大きく損なうことなく疎行列化が可能である.

スペース化の利点は、0 でない要素の割合を減らすことで、保存すべきデータ量を大きく削減できる点にある. 密行列では多くの 0 を含んでいても全ての要素を保持する必要があるが、疎行列であれば 0 でない要素とその位置だけを記録すればよく、メモリ使用量を効率的に抑えられる. また、CSR 形式などを用いることで、計算面でも高速化が期待できる. しかし、スペース化によって一部の情報が失われるため、過度に 0 でない要素を減らすと推論精度が低下する可能性がある.

本研究では、疎行列の格納方式として CSR 形式の導入を検討する。CSR は、各行における 0 でない要素の値とその列インデックスを配列として保持し、さらに行の先頭位置を示すポインタ配列を併せて管理する表現方式である。この形式を用いることで、実際に計算に関与する 0 でない要素のみを格納でき、行列をそのまま保存する場合と比較してメモリ使用量を大幅に削減できる。特にニューラルネットワークの重み行列は、スパース化によって行方向に多数の 0 の要素が発生することが多い [3]。そのため、本研究で扱う重み行列に CSR 形式を適用することで、メモリ削減効果が期待できる。

本研究では、値領域の削減手法として量子化を導入を検討する。量子化は、浮動小数点値を低ビット幅の整数にし、1 つの重みを表現するために必要なビット数を削減する手法である。これにより、重み行列全体の保存容量を効率的に抑えることができる。本研究では、スパース化に加えて量子化を併用することで、重み行列の表現効率をより一層高め、高い圧縮効果を実現することを目的とする。

0.3 実験方法

本実験では、複数の圧縮手法をまとめて適用できるように実装する。全体の手順は、スパース化、CSR 形式への変換、量子化の 3 段階から構成される。各段階の実装方法を以下に示す。

0.3.1 スパース化処理

入力データである浮動小数点の重み行列 W に対してスパース化処理を行う。絶対値が閾値 τ 未満の要素を 0 に置換し、0 でない要素数を削減する。

$$W_{ij} = \begin{cases} W_{ij}, & \text{if } |W_{ij}| \geq \tau \\ 0, & \text{if } |W_{ij}| < \tau \end{cases} \quad (1)$$

0.3.2 CSR 形式への変換

CSR 方式の実装は以下の手順で行う。まず、圧縮対象となる学習済みの重みを取得し、疎行列化のしきい値を設定する。取得したテンソルは NumPy 配列に変換し、配列内の絶対値がしきい値未満の要素は 0 に置換する。これにより、計算上無視できる微小な値を除去し、疎行列化の効果を高めることができる。

次に、しきい値処理後の配列を CSR 形式に変換する。CSR 形式は、0 でない要素を格納する data 配列、列インデックスを示す indices 配列、行ごとの先頭位置を示す indptr 配列、及び行列サイズを示す shape 配列で構成する。

最後に、これらの配列を C++ で利用可能な形に出力することで、組み込みシステムに組み込むことが可能となる。上記の手順をフローチャートで示すと以下のようになる。

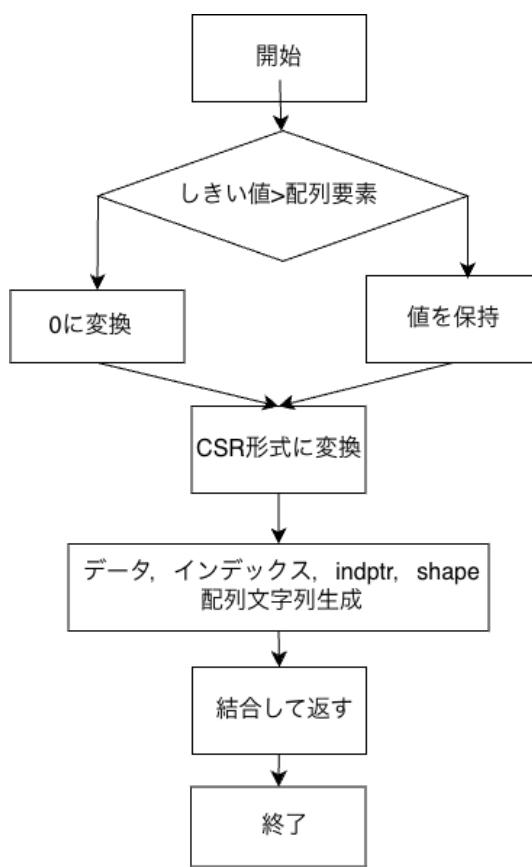


図 1 CSR 形式変換のフローチャート

0.3.3 量子化

量子化は、まず層内の重みの中で絶対値が最大の値を求め、これを用いてスケールを計算する。スケールは int8 の最大値 127 に対応させるように設定し、これにより元の float 値の比率を保持したまま整数化できる。次に各重みをこのスケールで割り、小数点以下を四捨五入することで int8 に変換する。こうして得られた int8 の値は C++ 側でスケールを掛けることで元の float 値に近い形に復元できる。

0.4 評価指標

本実験では、圧縮手法の性能を復元精度と圧縮性能の二つの観点から評価する。復元精度は平均二乗誤差(以下 MSE)を用いて定量的に評価し、圧縮性能は圧縮率を指標として測定する。

まず、復元精度の評価では、復元データ \hat{x}_i と元データ x_i の差に基づき、次式で定義される MSE を算出する。

$$MSE = \frac{1}{N} \sum_{i=1}^N (x_i - \hat{x}_i)^2$$

ここで、 N はデータの全要素数を表す。MSE は値が小さいほど元データと復元データの一一致度が高く、復元精度が良好であることを示す。各条件下で複数回の実験を行い、その平均値および標準偏差を求めて統計的に評価する。

次に、圧縮性能の評価では、元データのサイズ S_{orig} と圧縮後のサイズ S_{comp} を測定し、圧縮率を次式で定義する。この値が大きいほど高い圧縮性能を示すことになる。

$$\text{圧縮率} = \frac{S_{\text{orig}}}{S_{\text{comp}}}$$

これらの指標を用いて、各圧縮方式およびパラメータ設定に対する性能を比較する。特に、スパース化の閾値や量子化のビット幅を変化させた際の MSE および圧縮率の変化を分析し、圧縮効率と復元精度の相互関係を評価する。すべての測定は同一環境下で実施し、実験理論で述べた各手法の組み合わせを試すことで、それぞれの圧縮方式の特徴と有効性を明らかにすることを目的とする。

参考文献

- [1] 総務省, 『令和 3 年版情報通信白書』, 「第 1 部 特集 デジタルで支える暮らしと経済」, 2021 年.
<https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r03/html/nd105220.html>, (参照 2025 年 10 月 24 日).
- [2] IEEE Access, “A Survey on Resource Management in IoT Operating Systems”, 2018 年. <https://ieeexplore.ieee.org/document/8300305/>, (参照 2025 年 10 月 24 日).
- [3] 東芝レビュー, “(論文タイトル)”, Vol.74 No.4, pp. (ページ番号), 2019 年 7 月.
https://www.global.toshiba/content/dam/toshiba/migration/corp/techReviewAssets/tech/review/2019/04/74_04pdf/f01.pdf, (参照 2025 年 10 月 24 日).
- [4] IEEE Access, “A Survey on Resource Management in IoT Operating Systems,” 2018. <https://arxiv.org/abs/1602.01528>, (参照 2025 年 10 月 24 日).