ニューラルネットの軽量化のためのフィードバック量子化

Feedback Quantization for Data Compaction of Neural Networks

研究代表者　大阪大学大学院工学研究科 准教授　南 裕樹 *
Yuki Minami

This study focuses on the quantization problem of connection weights of neural networks. We first proposed a class of quantizers, called feedback quantizer, for the quantization of neural networks. The performance of the proposed quantizer depends on the design of the error diffusion filter. Thus, we developed a systematic design method of the error diffusion filter based on features of learning data, which is used for the learning of neural networks. In the proposed design method, satisfactory error diffusion filters are given by solving a kind of traveling salesman problems.

要旨

本研究では，学習済みのニューラルネットをできるだけ推論精度を劣化させないように軽量化することを目的としている．近年，ディープニューラルネットが注目されているが，そこで問題となるのが，ニューラルネットのサイズである．一般に，ニューラルネットのユニット間の結合重み係数は浮動小数点数で表されるため，その個数が爆発的に増大することによって，ハードウェア資源の限られたデバイスへの実装が困難になる．

この問題に対して，本研究では，結合重み係数を離散値に変換する量子化器として「フィードバック量子化器」を提案した．これは，ある場所で生じた量子化誤差をまだ量子化していない場所に拡散させることで，重要な情報をできる限り保存するというものである．

また，提案するフィードバック量子化器の性能は，量子化誤差をどこに拡散させるかに依存する．そこで，量子化誤差の拡散場所を決定する問題に注目し，その問題を巡回セールスマン問題に帰着することで，システマティックに拡散場所を決定する方法を提案した．

1．まえがき

ニューラルネットは，入力データと出力データの関係を重み付き線形和演算と非線形演算の組合せによって表現する．線形和演算に必要な結合重み係数は入出力データを用いて学習するが，一般に結合重みは実数値であり，計算機上では浮動小数点数で表される．さらに，近年注目されているディープニューラルネットの場合には，結合重みの個数は膨大となる．そのため，携帯端末のようなハードウェア資源（メモリ容量）の限られたデバイスに学習済みのディープニューラルネットを実装することが困難になる．

この問題に対して，ニューラルネットの軽量化が検討されている．ニューラルネットの軽量化には，二つのアプローチがあり，ひとつは，不要な結合やユニットを削除する枝刈り（pruning）[1]，もうひとつは，結合重み係数の量子化である[2, 3]．本研究では，量子化アプローチに注目するが，量子化の方法もさまざまある．たとえば，学習済みの実数値の結合重みを量子化するか，学習過程に量子化を組み込むことで離散値の結合重みを直接学習するか，といった量子化を適用するフェーズの違いや，単純な四捨五入型の量子化を用いるか，しきい値をランダムに変更する量子化を用いるかなど，使用する量子化器の種類の違いなどがある．

そもそも結合重み係数を量子化する問題の難しさは，連続値から離散値に変換する際に生じる量子化誤差によって，学習で獲得された重要な入出力関係が失われることである．本研究では，この難しさを克服するために，制御・信号処理分野で開発されてきた「動的量子化器」[4] をベースとした新しいタイプの量子化器を提案する．これは，ある場所で生じた量子化誤差をまだ量子化していない場所に拡散させることで，重要な情報をできる限り保存するというものである．本研究では，そのアイデアをニューラルネットに応用したフィードバック量子化手法を提案する．

従来の単純な四捨五入型量子化器などは，各結合重みを個別に量子化するフィードフォワードタイプの手法であるのに対し，提案手法は，各結合重み係数の量子化によって生じる量子化誤差を結合重み係数間で共有しながら量子化するフィード

＊ 助成決定時所属　大阪大学大学院工学研究科 講師

バックタイプの手法となる．また，先行研究 [2, 3] では，ニューラルネットの学習プロセスの中に量子化を組み込み，離散値の結合重み係数を獲得する方法が試みられているが，実数値の結合重みを学習する通常の方法に比べて学習が難しくなる．これに対して，提案手法は，量子化器そのものをスマートにすることで，量子化誤差によるニューラルネットの性能劣化をできる限り小さくするものである．つまり，ニューラルネットの再学習が必要ないため，学習済みのニューラルネットの軽量化を行うことができる．もちろん，先行研究の方法と組合せて学習プロセスの中に組込むこともできる．

本稿では，まず，ニューラルネットの量子化問題を説明する．つぎに，提案手法であるフィードバック量子化を説明する．そして，フィードバック量子化では，量子化誤差をどこに拡散させるかが重要なポイントとなる．そこで，フィードバック量子化における誤差拡散フィルタの系統的な設計方法を提案する．そもそも，ニューラルネットの結合重み係数には学習に用いたデータの特徴が保存されている．そのため，量子化器の誤差拡散フィルタの構造を学習データの特徴を考慮して構築すれば，データに特化した量子化が可能となり，結果的にニューラルネットの入出力構造を保存できる可能性が高い．本研究ではこの点に着目し，学習データの特徴を利用して，誤差拡散フィルタを設計する．具体的に，まず，ニューラルネットの各ユニットに入力される学習データ同士の関係性（距離）を数値化し，それを用いてフィルタの設計問題を巡回セールスマン問題として定式化する．そして，その問題の解を数値計算によって求めることで，誤差を拡散させる場所（誤差拡散フィルタ）を決定する．

なお，以下の説明は，成果としてまとめた原稿（学術論文 [1] および国内会議論文 [2],[3]）を再構成したものであることを付記しておく．

2．ニューラルネットの量子化問題

ニューラルネットは，図 1 のように，複数のユニットが並列に配置され，ネットワーク状に結合されたものである．ℓ 層の i 番目 $(i = 1, 2, \ldots, m_\ell)$ のユニットの状態を $x_{\ell,i} \in \mathbb{R}$，出力を $y_{\ell,i} \in \mathbb{R}$ とする．そして，ℓ 層の i 番目のユニットから $\ell+1$ 層の j 番目のユニットへの結合重み係数を $w_{\ell,i}^{\ell+1,j} \in \mathbb{R}$

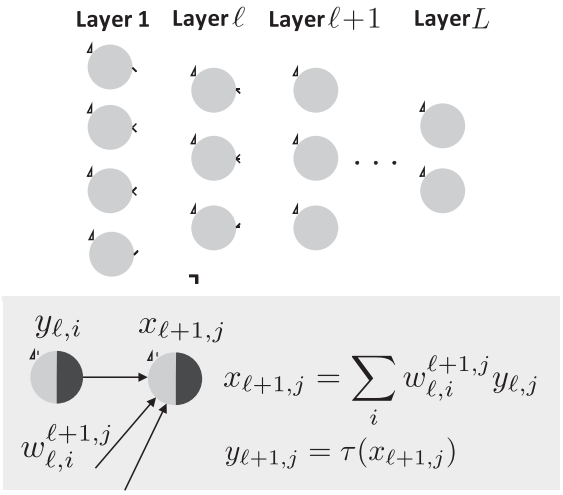

図 1: ニューラルネットとユニットモデル

とする．このとき，各ユニットのモデルは，

$$\begin{cases} x_{\ell+1,j} = \sum_{i=1}^{m_\ell} w_{\ell,i}^{\ell+1,j} y_{\ell,i} \\ y_{\ell+1,j} = \tau(x_{\ell+1,j}) \end{cases} \tag{1}$$

と表記される．ただし，$\tau : \mathbb{R} \to \mathbb{R}$ は活性化関数と呼ばれる非線形関数である．また，入力データを $U = [u_1, u_2, \ldots, u_{m_1}]^\top \in \mathbb{R}^{m_1}$ とすると 1 層目は，$y_{0,i} = u_i$ であり，L 層目の出力 $Y_L = [y_{L,1}, y_{L,2}, \ldots, y_{L,m_L}]^\top \in \mathbb{R}^{m_L}$ がニューラルネットの出力データとなる．学習時には，データセット（複数の (U, Y_L) の組）を用いて，結合重み係数 $w_{\ell,i}^{\ell+1,j}$ を決定し，推論時には，学習した $w_{\ell,i}^{\ell+1,j}$ を用いて，入力データ U から Y_L を計算する．

本研究では，ニューラルネットがすでに学習済みであるとし，その上で，そのニューラルネットの結合重み係数 $w_{\ell,i}^{\ell+1,j}$ を量子化する問題を考える．具体的に，学習済みのニューラルネットの入出力関係をできるだけ保存する，つまり推論精度をなるべく落とさないように実数値の $w_{\ell,i}^{\ell+1,j} \in \mathbb{R}$ を離散値 $v_{\ell,i}^{\ell+1,j} \in \{0, \pm d, \pm 2d, \ldots, \pm Nd\}$ におきかえる．

なお以降では，表記の簡単のために，$v_{\ell,i}^{\ell+1,j}$ などを右肩の $\ell+1$ を省略して $v_{\ell,i}^{j}$ などと表す．

3．結合重み係数のフィードバック量子化

3．1　従来手法

先行研究で用いられている二つの量子化手法 [2, 3] を説明する．

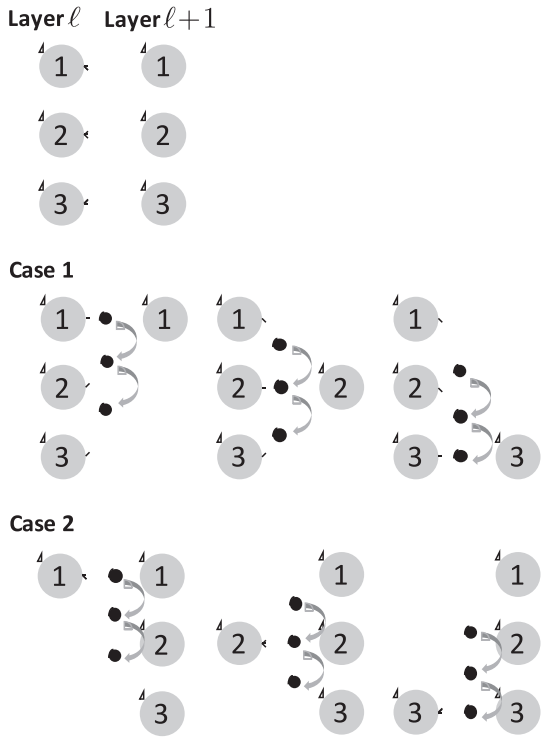

図 2: 量子化誤差の拡散例

まず一つ目は，

$$q: v^j_{\ell,i} = \begin{cases} \mathrm{sgn}(w^j_{\ell,i}) \left\lfloor \frac{|w^j_{\ell,i}|}{d} + \frac{1}{2} \right\rfloor d & (|w^j_{\ell,i}| \le Nd) \\ \mathrm{sgn}(w^j_{\ell,i}) Nd & (|w^j_{\ell,i}| > Nd) \end{cases} \tag{2}$$

で与えられる単純な四捨五入型の量子化である．ただし，$\mathrm{sgn}(\cdot)$ は符号関数，$\lfloor \cdot \rfloor$ は床関数，$d \in \mathbb{R}_+$ は量子化幅，$2N+1$ $(N \in \mathbb{N})$ は量子化の階調数である．たとえば，$d := 1$，$N := 1$ のとき，実数値を -1，0，1 の 3 値に丸めることになる．

二つ目は，ディザ量子化と呼ばれるもので，

$$Q_{\mathrm{dither}}: v^j_{\ell,i} = q(w^j_{\ell,i} + \eta(i)) \tag{3}$$

のように，結合重み係数に一様乱数 $\eta(i) \in [-d/2,\ d/2]$ を足してから単純な量子化を行う．たとえば，量子化幅が $d = 1$ で，結合重みが $0.2,\ 0.3, -0.1,\ 0.4, -0.2$ の場合，単純な四捨五入型量子化では，0, 0, 0, 0, 0 となるのに対し，ディザ量子化では，1, 0, 1, 0, 0 のような結果が得られる（乱数によって結果は異なる）．このように，単純な四捨五入型量子化では，各結合重みの量子化によってネットワークの構造が失われやすいが，ディザ量子化では，構造が比較的失われにくくなる．

3．2　フィードバック量子化器

ディザ量子化によるニューラルネットの軽量化は効果的であるが，確率的な手法であるため，常に同じ性能を実現することが困難である．そこで本研究では，ディザ量子化と同等以上の性能を実現する確定的な量子化手法として，制御・信号処理分野で研究されている動的量子化 [4] の考え方を応用した方法を提案する．動的量子化は，ある時刻で生じた量子化誤差をつぎの時刻の入力に足しあわせてから量子化するものである．提案手法はこの考えを踏襲し，ある結合重みの量子化で生じた量子化誤差を別の結合重みに反映させながら，量子化を行なう．

提案するフィードバック量子化器は，次式で与えられる．

$$Q_{\mathrm{ed}}: \begin{cases} \xi^j_{\ell,i}[t+1] = w^j_{\ell,i} + \displaystyle\sum_{(k_1,k_2)\in\mathcal{N}^j_{\ell,i}} z^{k_2}_{\ell,k_1}[t] \\ z^j_{\ell,i}[t] = q(\xi^j_{\ell,i}[t]) - \xi^j_{\ell,i}[t] \\ v^j_{\ell,i} = q(\xi^j_{\ell,i}[T]) \end{cases} \tag{4}$$

ただし，$t \in \{0\} \cup \mathbb{N}$ は繰り返しステップ，$T \in \mathbb{N}$ は最大ステップ数，$\xi^j_{\ell,i} \in \mathbb{R}$ は状態（初期値は $\xi^j_{\ell,i}[0] = 0$）である．また，$\mathcal{N}^j_{\ell,i}$ は量子化誤差の伝搬場所を決めるものであり，自由に決めることができる設計パラメータである．たとえば，図 2 の Case 1 のような誤差伝搬を考えた場合，$\mathcal{N}^1_{\ell,1} = \emptyset$，$\mathcal{N}^1_{\ell,2} = \{(1,1)\}$，$\mathcal{N}^1_{\ell,3} = \{(2,1)\}$，$\mathcal{N}^2_{\ell,1} = \emptyset$，$\mathcal{N}^2_{\ell,2} = \{(1,2)\}$，$\mathcal{N}^2_{\ell,3} = \{(2,2)\}$，$\mathcal{N}^3_{\ell,1} = \emptyset$，$\mathcal{N}^3_{\ell,2} = \{(1,3)\}$，$\mathcal{N}^3_{\ell,3} = \{(2,3)\}$ となる．ただし，$\emptyset$ は空集合を表す．

式 (4) を用いた量子化では，T 回反復計算をすることで，各結合重み係数の量子化値 $v^j_{\ell,i}$ が得られる．$z^j_{\ell,i}[t]$ が量子化誤差に対応しており，これを $\mathcal{N}^j_{\ell,i}$ で指定される場所に拡散する．

3．3　数値実験

ここでは，4 つの量子化手法：式 (2) の q，式 (3) の Q_{dither}，式 (4) の Q_{ed} (Case1)，Q_{ed} (Case2) を用いて結合重み係数の 3 値化（$d := 1$, $N := 1$）を行う．

実験で用いるデータセットは手書き文字の MNIST[5] である．60,000 サンプルの訓練セットと 10,000 サンプルのテストセットから成る画像のデータセットで，各サンプルは 28 × 28 グレースケール画像である．ネットワーク構造は，中間層は全結合の 3 層とし，ユニット数は 784–1024–1024–1024–10，活性化関数は ReLU，optimizer は Adam，誤差関数は SoftmaxCrossEntropy とする．そして，訓練データセットを用いて 100 epoch の

表 1: MNIST データセットでの正解率

q	Q_{dither}	Q_{ed} (Case 1)	Q_{ed} (Case 2)
9.7	30.2	**47.2**	19.1

表 2: Fashion-MNIST データセットでの正解率

q	Q_{dither}	Q_{ed} (Case 1)	Q_{ed} (Case 2)
10.2	30.3	**35.7**	13.2

学習によってニューラルネットの全ての実数結合重み係数を決定した．テストデータに対する正解率は，98.6%であった．

結合重み係数を量子化したときのテストデータに対する正解率（%）を調べた．その結果を表 1 に示す．結果を見ると，単純に量子化する q よりも誤差拡散型量子化を行った Q_{eq} の方が正解率が高いことがわかる．また，量子化誤差を拡散させる場所を変えることで結果が異なることもわかる．とくに，Case 1 と Case 2 では，Case 1 の方が高い正解率となった．詳細な解析は今後の課題であるが，ユニットモデルが式 (1) であるため，Case 1 のようにすることで，あるユニットの量子化誤差を隣のユニットに反映することができ，それにより量子化誤差の影響が出にくくなっていると考えられる．さらに，ディザ量子化も提案手法と同等の良い結果であることがわかる．しかし，確率的な手法であるため，常に同じ性能を実現することが困難である．実際，表 1 の値は 100 回実験を行ったときの平均値であり，最大値は 42.8，最小値は 18.6，標準偏差は 4.82 であり，ばらつきがある．この点からも，確定的な手法である提案手法の有用性が確認できる．

つぎに，他の数値例として，Fashion-MNIST[6] を用いて実験を行った．Fashion-MNIST は，手書き文字の MNIST と同じ規格のデータセットとなっている．

上記と同じ設定で学習を行ったところ，テストデータに対する正解率は 90.2%であった．このとき，結合重み係数を量子化したときのテストデータに対する正解率（%）を調べた．その結果を表 2 に示す．結果を見ると，上記の結果と同様の結果が得られていることがわかる．なお，ディザ量子化の結果は 100 回実験したときの平均値であり，最大値は 47.8，最小値は 16.9，標準偏差は 6.52 であった．

二つの実験結果より，提案手法の有用性が確認

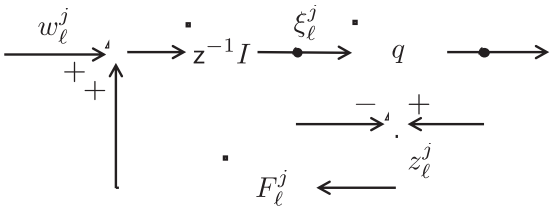

図 3: フィードバック量子化器の構造

できた．その一方，誤差拡散の方法には自由度がある．今回は二通りの方法を検証したが，データセットの特徴を考慮して拡散場所を決定することで，さらに性能劣化を小さくできる可能性がある．次章ではこの点について考える．

4．量子化誤差の拡散場所の設計

式 (4) の Q_{ed} は，各結合重み係数を量子化するものであるが，以下のように，それらをまとめて書くことができる．まず，$\xi_\ell^j := [\xi_{\ell,1}^j\ \xi_{\ell,2}^j\ \cdots\ \xi_{\ell,m_\ell}^j]^\top \in \mathbb{R}^{m_\ell}$，$w_\ell^j := [w_{\ell,1}^j\ w_{\ell,2}^j\ \cdots\ w_{\ell,m_\ell}^j]^\top \in \mathbb{R}^{m_\ell}$，$z_\ell^j := [z_{\ell,1}^j\ z_{\ell,2}^j\ \cdots\ z_{\ell,m_\ell}^j]^\top \in \mathbb{R}^{m_\ell}$，$v_\ell^j := [v_{\ell,1}^j\ v_{\ell,2}^j\ \cdots\ v_{\ell,m_\ell}^j]^\top \in \{0, \pm d, \pm 2d, \ldots, \pm Nd\}^{m_\ell}$ と定義する．そして，簡単のため，図 2 の Case 1 ような量子化誤差の誤差拡散方法を考える．つまり，$\ell+1$ 層の j 番目のユニットに結合されているエッジ上で誤差拡散を行うとする．すると，式 (4) は，

$$\begin{cases} \xi_\ell^j(t+1) = w_\ell^j + F_\ell^j z_\ell^j(t) \\ z_\ell^j(t) = q(\xi_\ell^j(t)) - \xi_\ell^j(t) \qquad (j = 1, 2, \ldots, m_{\ell+1}) \\ v_\ell^j = \xi_\ell^j(T) \end{cases} \tag{5}$$

となる．とくに，図 2 の Case 1 の場合は，

$$F_\ell^j = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad (j = 1, 2, 3) \tag{6}$$

となる．これをブロック線図で表したものが，図 3 である．z^{-1} はシフト作用素を表す．量子化誤差 $z_\ell^j \in \mathbb{R}^{m_\ell}$ をフィードバックする構造になっていることがわかる．そして，F_ℓ^j の選択が量子化の性能に影響を与える．F_ℓ^j を適切に設計すれば，大幅に性能が向上する可能性がある．本研究では，$F_\ell^j \in \mathbb{R}^{m_\ell \times m_\ell}$ $(j = 1, 2, \ldots, m_{\ell+1},\ \ell = 1, 2, \ldots, L-1)$ を誤差拡散フィルタと呼び，これを設計することを考える．

4．1　学習データの特徴を利用した誤差拡散フィルタの設計

本研究では，誤差拡散フィルタ $F_\ell^j \in \mathbb{R}^{m_\ell \times m_\ell}$ $(j = 1, 2, \ldots, m_{\ell+1},\ \ell = 1, 2, \ldots, L-1)$ を設計する．F_ℓ^j は $m_\ell \times m_\ell$ 次元であり，それが $m_{\ell+1}$ 個ある．ニューラルネットの層が L 層まであることと，各層のユニットの個数 m_ℓ が数百から数千になることを踏まえると， 超大規模な設計問題になることがわかる．本研究では，図 2 の Case 1 ように 1 箇所ずつ誤差を拡散することにし，

- ニューラルネットを構築する際に用いた学習データを利用して，各層のユニット間の関係性を把握し数値化する（ノルムで測る）
- ユニット間の関係性をもとに重み付きグラフを生成し，誤差拡散フィルタの設計問題を巡回セールスマン問題に帰着する（イメージ図を 図 4 に示す

ことによって，誤差拡散フィルタを設計する．上記のユニット間の関係性を把握することは以下の狙いがある．図 2 の Case 1 のような構造を考えると，あるユニットの結合重み係数の量子化誤差を近隣のユニットの結合重みに反映することになる．一方，ユニットに入力されるデータが近ければ，それらのユニットは推論時に似たような役割を担うことが予想される．そのため，誤差を反映させる場所を入力データの関係性が近いところにすることで，全体として，量子化誤差の影響が出にくくなることが期待できる．

以下では，まず，ニューラルネットの 1 層目の結合重みの量子化のための誤差拡散フィルタの設計について述べ，そのあと，ℓ 層目の量子化のためのフィルタ設計を説明する．

1 層目の結合重みの量子化：学習データとして，m_1 次元のベクトル U が n セットあるとし，それらを $U^1,\ U^2, \ldots, U^n$ と表記する．そして，各学習データの i 番目の要素を取り出して並べたベクトルを $\theta_{1,i}$ と書く．つまり，$\theta_{1,i}$ は n 次元のベクトルとなり，それが，m_1 セット生成される．$\theta_{1,i}$ は，ニューラルネットの 1 層目の i 番目のユニットへの入力を集めたものである．

このとき，各ベクトル間の距離を 1 ノルムを用いて測る．

$$a_{1,ij} = \|\theta_{1,i} - \theta_{1,j}\|_1 \tag{7}$$

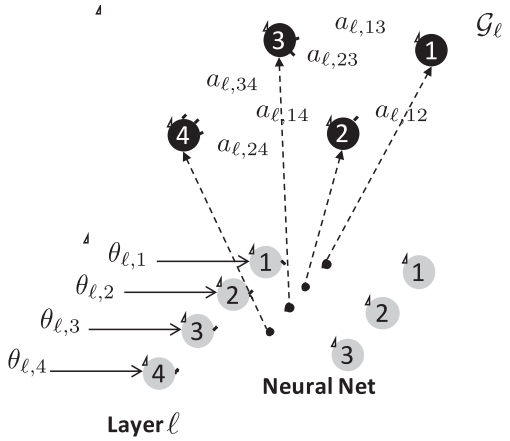

図 4: 量子化誤差の誤差拡散問題と巡回セールスマン問題の関係

ただし，$a_{1,ii} = 0$，$a_{1,ij} = a_{1,ji}$ である．$a_{1,ij}$ は，i 番目のユニットに入力される学習データと，j 番目のユニットに入力される学習データの近さを表している．$a_{1,ij}$ の値が小さいほど，学習データが似ているということである．

m_1 個の頂点の集合 $\mathcal{V}_1 := \{1, 2, \ldots, m_1\}$ から構成される完全グラフ $\mathcal{G}_1(\mathcal{V}_1, \mathcal{E}_1)$ および，各辺 $e_{1,ij} = (i, j) \in \mathcal{E}_1$ に対するコスト $a_{1,ij}$ が与えられているとする（図 4）．このとき，すべての点を 1 回ずつ経由する閉回路での辺上のコストの合計を最小にするもの（ハミルトン閉路）を求める．具体的に，解を頂点集合 $\mathcal{V}_1$ の順列 $\{\sigma_1(1), \sigma_1(2), \ldots, \sigma_1(m_1)\}$ で表現する．ただし，$\sigma_1(i)$ は，i 番目に通る頂点に対応している．このとき，上記の問題は，

$$\sum_{k=1}^{m_1-1} a_{1,\sigma_1(k)\sigma_1(k+1)} \tag{8}$$

を最小にする順列 $\{\sigma_1(1), \sigma_1(2), \ldots, \sigma_1(m_1)\}$ を求める問題として定式化できる．この問題は，(対称) 巡回セールスマン問題として知られており，既存の数値最適化アルゴリズムを用いて解（近似解）を得ることができる．

最後に，順列 $\{\sigma_1(1), \sigma_1(2), \ldots, \sigma_1(m_1)\}$ から誤差拡散フィルタを求める．$F_1^j := \{F_1^j(\mu, \nu)\}$ $(j = 1, 2, \ldots, m_2)$ は，

$$F_1^j(\mu, \nu) = \begin{cases} 1 & \text{if } (\mu, \nu) = (\sigma_1(k+1), \sigma_1(k)) \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

である．これと式 (5) を用いて 1 層目と 2 層目をつなぐエッジの重み係数の量子化を行う．たとえば，$m_1 = 3$ のとき，$\sigma_1(1) = 3,\ \sigma_1(2) = 1,\ \sigma_1(3) = 2$

の解が得られたとすると，

$$F_1^j = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{10}$$

となり，ユニット 3 の誤差が ユニット 1 拡散され，さらに，ユニット 1 の誤差がユニット 2 に拡散される．

ℓ 層目の結合重みの量子化： 以下を $\ell = 2$ から $\ell = L-1$ まで繰り返す．

Step1 $\ell - 1$ 層までの量子化した結合重み係数を用いてニューラルネットを構築する．

Step2 **Step1** で構築したニューラルネットに学習データ U^1, $U^2, \ldots, U^n$ を入力したときの ℓ 層における各ユニットへの入力データ（$\ell - 1$ 層の出力データ）をまとめたものを $Y_{\ell-1}^1, Y_{\ell-1}^2, \ldots, Y_{\ell-1}^n$ とする．ただし，$Y_0^j = U^j$ である．

Step3 1 層目のときと同様に，i 番目の要素を取り出して並べたベクトルを $\theta_{\ell,i} \in \mathbb{R}^n$ $(i = 1, 2, \ldots, m_\ell)$ とし，

$$a_{\ell,ij} = \|\theta_{\ell,i} - \theta_{\ell,j}\|_1 \tag{11}$$

を計算する．

Step4 m_ℓ 個の頂点の集合 $\mathcal{V}_\ell := \{1, 2, \ldots, m_\ell\}$ から構成される完全グラフ $\mathcal{G}_\ell(\mathcal{V}_\ell, \mathcal{E}_\ell)$ および，各辺 $e_{\ell,ij} = (i, j) \in \mathcal{E}_\ell$ に対するコスト $a_{\ell,ij}$ が与えられている（図 4）とし，

$$\sum_{k=1}^{m_\ell - 1} a_{\ell,\sigma(k)\sigma(k+1)} \tag{12}$$

を最小にする頂点集合 $\mathcal{V}_\ell$ の順列 $\{\sigma_\ell(1), \sigma_\ell(2), \ldots, \sigma_\ell(m_\ell)\}$ を求める．

Step5 誤差拡散フィルタの係数 $F_\ell^j := \{F_\ell^j(\mu, \nu)\}$ $(j = 1, 2, \ldots, m_{\ell+1})$ を

$$F_\ell^j(\mu, \nu) = \begin{cases} 1 & \text{if } (\mu, \nu) = (\sigma_\ell(k+1), \sigma_\ell(k)) \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

と定め，式 (5) を用いて結合重みの量子化を行う．

なお，文献 [3] では，上記の方法を改良した誤差拡散フィルタの多段階設計手法を検討しているが，本稿では割愛する．

表 3: 提案手法の MNIST データセットでの性能

q	Q_{ed} (Case 1)	Q_{ed} (TSP)
9.7	41.6	**71.7**

表 4: 提案手法の Fashion-MNIST での性能

q	Q_{ed} (Case 1)	Q_{ed} (TSP)
14.2	58.6	**67.4**

４．２　数値実験

提案手法の効果を確認するために，二つの数値実験を行う．

実験で用いるデータセットは，3 章で用いた MNIST と Fashion-MNIST である．60,000 サンプルの訓練セットと 10,000 サンプルのテストセットから成る画像のデータセットで，各サンプルは 28 × 28 グレースケール画像である．ネットワーク構造は，中間層は全結合の 3 層とし，ユニット数は 784–1024–1024–1024–10，活性化関数は ReLU，optimizer は Adam，誤差関数は Softmax-CrossEntropy とする．そして，訓練セットを用いて 100 epoch の学習によってニューラルネットの全ての実数結合重み係数を決定した．テストデータに対する正解率は，98.6%であった．

結合重み係数の 3 値化（$d := 1$, $N := 1$）を行う．結合重み係数を量子化したときのテストデータに対する正解率（%）を調べた．その結果を表 3 に示す．Q_{ed} (Case1) は，ユニット番号の小さいものから大きいものに順に誤差を拡散させるものであり，Q_{ed} (TSP) は，前節の手法による結果を用いて誤差を拡散するものである．結果を見ると，単純に量子化する q よりもフィードバック量子化器 Q_{ed} の方が正解率が高いことがわかる．また，量子化誤差を拡散させる場所を変えることで結果が異なることもわかり，提案手法により学習データに特化した誤差拡散フィルタを設計した方が高い正解率となった．

つぎに，他の数値例として，Fashion-MNIST を用いて実験を行った．MNIST の場合と同じ設定で学習を行ったところ，テストデータに対する正解率は 89.6%であった．このとき，結合重み係数を量子化したときのテストデータに対する正解率（%）を調べた．その結果を 表 2 に示す．結果を見ると，上記の結果と同様の結果が得られていることがわかる．

表 5: メモリ削減結果 (MNIST)

	32 bit	# of zero	1 bit
Original	2910208	0	0
Proposed	0	2698187	212021

表 6: メモリ削減結果 (Fashion-MNIST)

	32 bit	# of zero	1 bit
Original	2910208	0	0
Proposed	0	2552210	357998

４．３　量子化による軽量化の定量評価

上記の実験結果より，提案手法の有用性が確認できた．ここでは，量子化によってどの程度メモリ容量が削減できるかについて考察する．

量子化によって，重み係数は -1, 0, 1 の 3 値となる．0 を含んでいるため，0 の部分を削除すれば，ニューラルネットを軽量化できる．また，その他は，-1 と 1 のどちらかであるので，1 bit で表現可能である．つまり，32 bit の実数値を 1 bit の離散値で表現できる．上記の数値例において，0 の個数と 1 bit で表現できる係数の個数を調べると，表 5，表 6 のようになった．もともと，2,910,298 個の 32 bit の結合重みがあったが，表 5 のように，量子化によって，2,698,187 個の結合重みが 0 になり，1 bit で表現可能な結合重みは，212,021 個となった．つまり，2,910,298 × 32 bit のメモリ容量を 212,021 × 1 bit のメモリ容量に圧縮できたことになる．オリジナルのニューラルネットのメモリ容量を 100 % としたとき，量子化後は，0.2277 % となる．つまり，500 分の 1 の容量で 71%の正解率が得られるということである．問題依存な部分もあると予想されるが，提案手法による量子化によって，ニューラルネットを大幅に軽量化できる可能性が高いと考えられる．

５．あとがき

本研究では，ニューラルネットの軽量化のためのフィードバック量子化手法を提案した．そして，量子化誤差を拡散させる場所を決める誤差拡散フィルタの設計問題を巡回セールスマン問題に帰着させて設計する手法を提案した．さらに，二つの数値実験をとおして，提案手法の有効性を確認した．

本研究の内容に関して，つぎの 3 点を強調しておく．第一に，提案アプローチは，量子化誤差をフィードバックし整形するという，制御工学分野の知見を応用するものになっている．これにより，結合重み係数の静的な量子化問題が動的システムによる量子化誤差のフィルタリング問題として記述される．この取り組みは著者の知る限り他に類をみない．第二に，誤差拡散フィルタの次元は，ニューラルネットの規模に応じて爆発的に増加する．その大規模フィルタの設計問題を巡回セールスマン問題に帰着させることで，効率よくフィルタを設計する手法を提案している．また，学習データの特徴を利用することで，対象とするニューラルネットに特化した量子化が可能になることも特筆すべき点である．第三に，ニューラルネットの量子化によって，メモリ容量の小さいデバイスへの組み込みが可能となる．また，量子化によって多くの結合重みをゼロにできれば，スパースなモデルが構築でき，計算コストの大幅削減や入出力関係の理解にも役立つ．さらに，浮動小数点の重み係数をの整数値に変換すれば，浮動小数点演算装置（FPU）がないマイコンへの実装も可能になる．これは，ニューラルネットのさらなる普及につながる．

今後の課題としては，誤差拡散フィルタの設計論の一般化や畳み込みニューラルネットへの発展，提案手法のさまざまな実問題への応用などがあげられる．

発表論文

[1] 南裕樹，池田智裕，石川将人：誤差拡散型量子化によるニューラルネットワークの軽量化，システム制御情報学会論文誌, Vol. 32, No. 3, pp. 133–135 (2019)

[2] 南裕樹，池田智裕，石川将人：ニューラルネットの軽量化のためのノイズシェーピング量子化：学習データの特徴を利用したフィルタ設計，第 6 回制御部門マルチシンポジウム， 2G1-2 (2019/3)

[3] 南裕樹, 池田智裕, 石川将人：深層学習モデルの圧縮のためのノイズシェーピング量子化器の多段階設計，第 63 回システム制御情報学会研究発表講演会, GSe03-4 (2019/5)

参考文献

(1) R. Reed: Pruning algorithms-a survey; *IEEE Transactions on Neural Networks*, Vol. 4, No. 5, pp. 740–747 (1993)

(2) M. Courbariaux, Y. Bengio, and J.-P. David, Binaryconnect: Training deep neural networks with binary weights during propagations, *Advances in NeuralInformation Processing Systems 28*, 3123/3131 (2016)

(3) C. Zhu, S. Han, H. Mao, and W.J. Dally, Trained Ternary Quantization, *International Conference on Learning Representations 2017* (2016)

(4) 南裕樹，森田亮介：ノイズシェーピング量子化 III –制御のための動的量子化 (1)，システム/制御/情

報，**61**-6, 241/246,（2017）

(5) http://yann.lecun.com/exdb/mnist/

(6) https://github.com/zalandoresearch/fashion-mnist