CI-004

リッチフローに基づく全結合ニューラルネットワークの圧縮符号化

Compression of Fully Connected Neural Network Models by Ricci Flow

都竹 千尋 *　高橋 桂太 *　藤井 俊彰 *
Chihiro Tsutake　Keita Takahashi　Toshiaki Fujii

1 はじめに

深層ニューラルネットワーク（Deep Neural Network，DNN）が爆発的に普及し，画像認識や画像圧縮など幅広い分野で応用されている．一般に，DNN は階層構造を持つグラフによって表現される．また，エッジの重み及びノードのバイアスを学習によって最適化することで，入力値からタスクに応じた出力値を推論する．現代の DNN は，$O(10^9)$ 個もの重みとバイアスを持つことがあり，これは情報量に換算すると約 8 GB にも上る．このように，DNN が持つ情報量は極めて膨大であり，その圧縮符号化が不可欠である．

一般に，DNN が持つ情報量の大部分は重みが占めており，その圧縮符号化は 4 種類の手法に大別できる [1]．Weight Sharing は最も単純な手法であり，層間で重みを共有することで，冗長な重みの伝送をスキップする．Network Pruning は，学習時に微小な重みを零に丸めることで，重みのエントロピーを抑える．グラフ信号処理に基づく代数的な手法は，グラフの隣接行列を特異値分解し，重みよりも遥かに情報量が少ない主成分グラフ及び少数の特異値を伝送する．Knowledge Distillation は，膨大な数の重みを持つ教師モデルを学習した後に，そのモデルの推論結果を用いて小規模なモデルを学習し，重みの数を大幅に削減する．

本稿では，上述した 4 種類の手法とは大きく異なり，微分幾何学の数理に基づく新たな圧縮符号化手法を提案する．特に，ポアンカレ予想を解決に導いたリッチフロー [2]，及びペレルマンによって導入されたリッチフローの手術 [3] に着目し，リッチフローの圧縮符号化への応用の可能性を追求する．次章で述べるように，リッチフローはリーマン多様体に対する微分方程式として定義され，曲率の時間発展と特異点の除去によって，多様体を単純な部分多様体に分解する．多様体の離散表現であるグラフに対してもリッチフローの適用は可能であり [4]，グラフを単純な部分グラフに分解できる．提案手法では，リッチフローを DNN のグラフに適用する．これにより，DNN の入力から出力までの計算経路を部分グラフとして概形的に抽出できる．また，時間パラメータに応じて，部分グラフの数値精度を適応的に変更することで，効率的な符号化を実現する．

リッチフローによる DNN のグラフ分解は，有意性の高い/低い重み毎にコミュニティを形成する性質があり，推論における DNN のアクティビティを可視化することができる．この性質は DNN の圧縮符号化の目的に限定されず，DNN を理解するという意味で重要である．今後数十年にわたって，画像工学の研究分野は，DNN を礎として飛躍的な発展が成し遂げられると予想される．しかし，画像復元や画像認識など，具体的な問題に対するアルゴリズムの構築は飽和状態を迎えると予想される．これは，我々が DNN を応用する際に，DNN が画像をどのように捉えているかを十分に理解できておらず，DNN のポテンシャルを引き出せていないためである．このように，今後の画像工学では DNN のホワイトボックス化は必須であり，本稿はリッチフローによって得られる部分グラフを分析し，DNN の理解も同時に目指すものである．

2 リッチフロー・手術

本章では，多様体とグラフに対する二つのリッチフローの関係を明確にするために，連続系のリッチフローを概観した後に，離散系のリッチフローを定義する．以下では，多様体の次元やグラフの構造を限定することがある．これによって，一般性を失うかもしれないが，リッチフローの幾何学的な振る舞いがイメージしやすくなる．

2.1 連続系

リッチフローは，後述する計量テンソル g を備えた微分可能多様体 M，すなわちリーマン多様体 (M, g) に対して定義される [2]．しばらくの間，リーマン多様体の次元を 2 に限定する．例えば，球面やトーラス面を思い浮かべてもらいたい．

まず，計量テンソル g を定義する．M 上のある点 p を原点とする接ベクトル空間を T_pM として，その基底を $\boldsymbol{e}_1, \boldsymbol{e}_2$ とする[*1]．T_pM のある二次元ベクトル $\boldsymbol{v} = v_1\boldsymbol{e}_1 + v_2\boldsymbol{e}_2$ に対して，内積を

$$\langle \boldsymbol{v}, \boldsymbol{v} \rangle = \boldsymbol{e}_1^{\mathrm{t}}\boldsymbol{e}_1 v_1^2 + \boldsymbol{e}_1^{\mathrm{t}}\boldsymbol{e}_2 v_1 v_2 + \boldsymbol{e}_2^{\mathrm{t}}\boldsymbol{e}_1 v_2 v_1 + \boldsymbol{e}_2^{\mathrm{t}}\boldsymbol{e}_2 v_2^2 \tag{1}$$

と定義したとき，計量テンソル $g \in \mathbb{R}^{2\times 2}$ の成分は $g_{i,j} = \boldsymbol{e}_i^{\mathrm{t}}\boldsymbol{e}_j$ である．$\boldsymbol{e}_1, \boldsymbol{e}_2$ が正規直交基底の場合，g は単位行列となり，$T_pM = \mathbb{R}^2$，かつ式 (1) は三平方の定理に帰着される．つまり，計量テンソル g とは，接ベクトル空間 T_pM とユークリッド空間 $\mathbb{R}^2$ の乖離度を測る尺度である．同様に，三平方の定理からのずれを表す．

次に，リッチ曲率テンソルを定義する．T_pM の基底を $\boldsymbol{e}_1, \boldsymbol{e}_2$ として，ここでは正規直交性 $\boldsymbol{e}_i^{\mathrm{t}}\boldsymbol{e}_j = \delta_{i,j}$ を仮定する．これらの基底に直交する法平面から定まる M の断面曲率（ガウス曲率）を K と記述すると，$\boldsymbol{e}_1$ 方向のリッチ曲率テンソルは，下記の二次形式を満たす実行列 $R \in \mathbb{R}^{2\times 2}$ のことである．

$$K = \langle R\boldsymbol{e}_1, \boldsymbol{e}_1 \rangle \tag{2}$$

なお，図 1 に示すように，$K > 0$，$K = 0$，$K < 0$ は各々，p の近傍で (a) 楕円幾何，(b) ユークリッド幾何，及び (c) 双曲幾何が成立することを意味する．つまり，M の幾何的性質はリッチ曲率テンソル R によって特徴づけられる．

最後に，リッチフローは時間発展型微分方程式

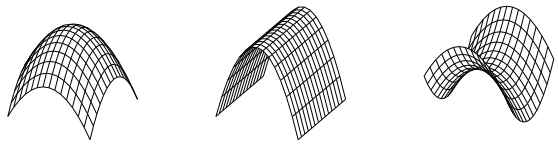
$$\frac{dg_{i,j}}{dt} = -2R_{i,j} \tag{3}$$

によって定義される．$R_{i,j} \approx -\frac{1}{2}\Delta g_{i,j}$ と近似することができるため [2]，式 (3) は放物型微分方程式として解釈できる．従って，式 (3) の解の生成過程は，同タイプの微分方程式である熱拡散方程式と同様の振る舞いを示す．具体的には，R が正定値（負定値）行列の場合，すなわち p の近傍で楕円幾何（双曲幾何）が成立するとき，M が縮退（膨張）する方向に $g_{i,j}$ が修正される．

リッチフローを導入するにあたり，これまでは M の次元を 2 に限定した．しかし，一般の次元，とりわけ図 2(a) に示すような $\mathbb{R}^4$ に存在する 2 つの球 $\mathbb{S}^3$ とシリンダー $\mathbb{S}^2 \times \mathbb{R}$ によって構成される単連結な三次元多様体は，そのリッチフローが特異な振る舞いを示す．具体的には，シリンダーの断面曲率が部分空間 $\mathbb{S} \times \mathbb{S}$ で正であることに起因して，図 2(b) のように，微小時間発展後にシリンダーが縮退する．これをさらに時間発展させると，図 2(c) のように，シリンダーが一点に崩壊する．ペレルマンの手術 [3] とは，崩壊する寸前に多様体を切り離す数学的操作であり，図 2(d) のように特異な多様体を部分多様体に分解

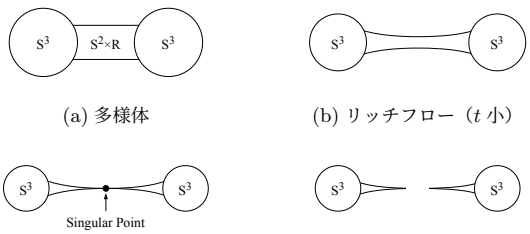
* 名古屋大学 大学院工学研究科 情報・通信工学専攻

[*1] ここでは，$\boldsymbol{e}_1, \boldsymbol{e}_2$ の実体を厳密に定義することは避けるが，一般に，正規直交性 $\boldsymbol{e}_i^{\mathrm{t}}\boldsymbol{e}_j = \delta_{i,j}$ が常に成立するとは限らないことに注意されたい．



(a) 正　(b) 零　(c) 負

図 1　多様体の曲率



(a) 多様体　(b) リッチフロー（t 小）

(c) リッチフロー（t 大）　(d) 部分多様体（手術後）

図 2　連続系のリッチフローとその手術

(a) $\mu_1(X)$　(b) $\mu_2(X)$

図 3　カップリングの例

する．これにより，ペレルマンは三次元の閉多様体が 8 種の標準的構造に分解できることを示し，その帰結としてポアンカレ予想を解決に導いた．このような，手術付きリッチフローをグラフ分解に応用したものが，離散系のリッチフローである．

2.2　離散系

まず，ワッサースタイン計量（アース・ムーバー距離）を定義する．以降では，離散距離空間 $(\mathbb{Z}, d)$ 及び離散確率変数 $X \in \{x_1, \cdots, x_n\}$ を扱う．ここで，$d : (\mathrm{Int}, \mathrm{Int}) \to \mathbb{R}$ は距離関数であり，$(z_1, z_2) \in \mathbb{Z}^2$ や $(x_1, x_2) \in X^2$ などの整数の組を正の実数に対応づけるものである．ワッサースタイン計量は，異なる確率分布の類似度を測る尺度であり，$z_i, z_j \in \mathbb{Z}$ で添字づけられた確率分布を $\mu_{z_i}(X), \mu_{z_j}(X)$ としたとき，ワッサースタイン計量は

$$W(\mu_{z_i}, \mu_{z_j}) = \min_{P \in \gamma(\mu_{z_i}, \mu_{z_j})} \sum_{k,l} d(x_k, x_l) p_{k,l} \tag{4}$$

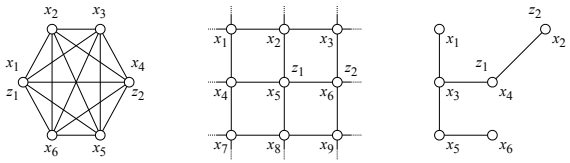
$$\gamma(\mu_{z_i}, \mu_{z_j}) = \left\{ P : \sum_l p_{k,l} = \mu_{z_i}(x_k), \sum_k p_{k,l} = \mu_{z_j}(x_l) \right\}$$

と定義される．ここで，$P \in \mathbb{R}^{n \times n}$ はカップリング行列と呼ばれ，各要素 $p_{k,l}$ は $\mu_{z_i}(x_k)$ から $\mu_{z_i}(x_l)$ への分配量を表す．例えば，図 3 のような二つの確率分布 μ_1, μ_2 が与えられているとする．この場合，$p_{1,3} = p_{5,3} = 1/10$ として，確率 $\mu_1(x_1)$ と $\mu_1(x_5)$ を $\mu_1(x_3)$ に $1/10$ ずつ分配することで，μ_1 を μ_2 に変換できる．ワッサースタイン計量は，このような分配量 P の線型結合で確率分布間の類似度を測る．

次に，オリバー・リッチ曲率を定義する．これは，$z_i, z_j \in \mathbb{Z}$ として，

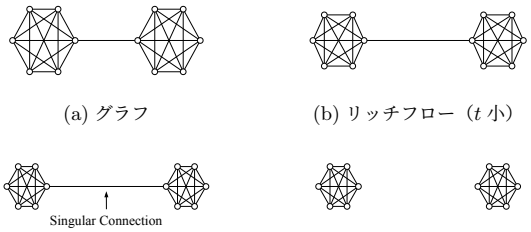
$$k_{i,j} = 1 - \frac{W(\mu_{z_i}, \mu_{z_j})}{d(z_i, z_j)} \tag{5}$$

である．ここで，図 4 のようなグラフを考える．ある二つのノードは，z_i, z_j と番号づけられている（この例では，$i = 1, j = 2$）．$\mu_{z_i}(X)$ を『ノード z_i が別のノード（確率変数）$X \in \{x_1, x_2, \cdots\}$ と接続される



(a) 正　(b) 零　(c) 負

図 4　グラフの曲率



(a) グラフ　(b) リッチフロー（t 小）

(c) リッチフロー（t 大）　(d) 部分グラフ（手術後）

図 5　離散系のリッチフローとその手術

という事象の確率分布』とすると，グラフのオリバー・リッチ曲率が定義できる．文献 [4] では，確率分布を

$$\mu_{z_i}(x_k) = \begin{cases} \alpha & \text{if } x_k = z_i \\ \frac{1-\alpha}{C} \exp(-d(x_k, z_i)) & \text{if } x_k, z_i \text{ connected} \\ 0 & \text{o/w} \end{cases} \tag{6}$$

と定義している．$d(x_k, z_i)$ はノード x_k, z_i 間の距離，$\alpha \in [0, 1]$ は分布の形状を決定するパラメータ，C は規格化定数である．隣接ノード間の距離を 1 とすると，図 4 の (a) 完全グラフ，(b) グリッドグラフ，(c) 連結パスグラフは，ノード z_1, z_2 間で正，零，負のオリバー・リッチ曲率を持つ．また，グラフの構造を曲率によって特徴づけることができる．このような性質は，オリバー・リッチ曲率がリッチ曲率の系であり [5]，多様体の形状を曲率によって特徴づけられるという性質を継承したことによるものである．

最後に，離散系のリッチフローとは，時間発展型微分方程式

$$\frac{dw_{i,j}}{dt} = -k_{i,j} w_{i,j} \tag{7}$$

のことである．ここで，$w_{i,j}$ はノード z_i, z_j 間の重みである．式 (7) は連続系のリッチフローと類似した振る舞いを示す．例えば，図 5(a) のように，正の曲率を持つ完全グラフがパスグラフで接続されている場合を考える．このグラフでは，パスグラフが負の曲率を持っており，図 5(b) のように，微小時間発展後に連結部分の重み $w_{i,j}$ が膨張する．これをさらに時間発展させると，図 5(c) のように，重みが正の無限大に発散して特異な接続となる．文献 [4] では，$w_{i,j} > \mathrm{th}$ となる接続をカット（手術）することによって，図 5(d) のように，特異なグラフを二つの部分グラフに分割する．

なお，時間パラメータ t を離散化すると，時刻 $t+1$ における式 (7) の解は，有限差分法に基づいて

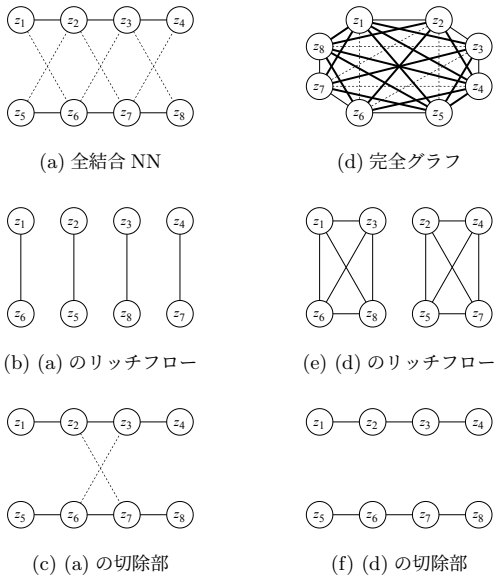
$$w_{i,j}^{[t+1]} - w_{i,j}^{[t]} = -\varepsilon k_{i,j}^{[t]} w_{i,j}^{[t]}, \quad \varepsilon > 0 \tag{8}$$

を反復計算することで得られる．

3　提案手法

3.1　概要

式 (7) によって定義されるリッチフローを DNN の圧縮符号化に応用する．本研究では，正の重み $w_{i,j}$ で接続された全結合 NN を扱う．提案手法では，リッチフローによって DNN の重みを時間発展させて，



(a) 全結合 NN　(d) 完全グラフ

(b) (a) のリッチフロー　(e) (d) のリッチフロー

(c) (a) の切除部　(f) (d) の切除部

図 6　全結合 NN と完全グラフのリッチフロー

時刻 t で手術する．また，切除した重み $w_{i,j}^{[t]}$ を原グラフの重み $w_{i,j}^{[0]}$ に置き換え，t に依存させて，重みの数値精度を適応的に変更して符号化する．また，t におけるグラフの形状（トポロジー）も符号化する．

3.2 重みの補間

提案手法では，距離関数を $d(z_i, z_j) = w_{i,j}$ と定義する．この定義のもとで，ノード z_i, z_j が接続されていない場合の重みを考える．図 6(a) は全結合 NN の例であり，実線と破線は各々 $w_{i,j} = 4$ と 0.1 を表し，接続されていないエッジの重みは $w_{i,j} = 0$ とした．このグラフの $t = 2$ におけるリッチフローは図 6(b) となり，$w_{i,j} = 0.1$ の重みでコミュニティを形成する．切除部は (c) であるが，(b) のコミュニティから漏れた重み $w_{i,j} = 0.1$ が含まれる．この理由は，以下の通りである．まず，$t = 1$ におけるリッチフローが，(a) をパスグラフ $\{z_1, z_6, z_3, z_8\}$ と $\{z_5, z_2, z_7, z_4\}$ に分解する．ここで，$k_{6,3}, k_{2,7}$ は負のオリバー・リッチ曲率を持つ．従って，$t = 2$ のリッチフローでパスグラフの重み $w_{6,3}, w_{2,7}$ が発散し，これらが手術によってカットされるため，(b) のようなグラフが得られる．以上より，z_i, z_j が接続されていない場合に $w_{i,j} = 0$ とすると，グラフの過剰細分化が起きてしまい，重みがコミュニティから漏れる可能性がある．一般に，同一の情報源に属するシンボルは，情報源拡大によって効率的な符号化が可能である．故に，コミュニティから漏れた重みに対して，再び符号化を適用するのは効率が悪い．このようなグラフが生成される原因は，z_i, z_j が接続されていない場合に $w_{i,j} = 0$ としたためである．

そこで，z_i, z_j が接続されていない場合に重みの補間を行う．具体的には，ダイクストラ法 [6] によって z_i, z_j の最短距離を求め，その値を $w_{i,j}$ とする．これによって，図 6(a) の全結合 NN は，(d) のような完全グラフに変換される．ここで，太線はダイクストラ法によって補間されたエッジを表す．図 6(e) は (d) のリッチフローであり，(b) と同様に，$w_{i,j} = 0.1$ でコミュニティを形成するが，全てのノード間で正のオリバー・リッチ曲率を持つ．従って，(e) の時間発展は縮小方向に向かい，十分大きな手術定数 th を用いることで，$t \geq 2$ においてリッチフローが細分化されることはない．また，(d) の切除部は，(f) のように $w_{i,j} = 4$ のみで構成されるパスグラフとなる．

3.3 重みの数値精度

一般に，重みの数値精度が DNN の推論精度に与える影響は少なく，入力から出力までの連結性を担保することが重要である [7]．図 6(f) に示すように，切除部のグラフはある程度の連結性が担保されており，推論の計算経路を抽出できる点が特徴である．提案手法では，切除部の重みを時間パラメータ t に依存させて，適応的に数値精度を変更することで，連結性を保ちつつ推論精度の低下を抑える．なお，3.1 節で述べたように，符号化対象は切除した重み $w_{i,j}^{[t]}$ に対応する原グラフの重み $w_{i,j}^{[0]}$，及びグラフのトポロジーであることを再喝しておく．

図 6(a) ような全結合 NN は，正解データセット D を用いて学習されており，学習時のロスは L であったと仮定する．数値精度の変更後に達成したいロスを L'，リッチフローによる最大時間発展回数を T とする．また，

$$\Delta L = \frac{L' - L}{T + 1} \tag{9}$$

と定義する．まず，$t = 1$ に関して，データセット D を学習済みの DNN に入力して，ロスの増加が ΔL 以下となるような，重みの小数部のビット精度を全探索する．また，ロスの増加が ΔL 以下となる最小のビット長で重みの小数部を丸める．次に，一般の t に関して，時間 $t - 1$ までの重みの数値精度を固定して，$t = 1$ と同様に，時間 t における重みの最小ビット長を求める．また，重みの小数部分を丸める．最後に，$t = T$ における符号化が済んでいない部分グラフ，すなわち最後のリッチフローで得られたグラフを，一般の t と同様の方法で符号化する．なお，丸めた重みは Float 精度で表現され，これを Lempel-Ziv-Markov chain Algorithm（LZMA）で圧縮する．また，全てのグラフは，t に依存させて独立に符号化する．復号器は，全ての部分グラフを統合することで DNN のグラフを得る．

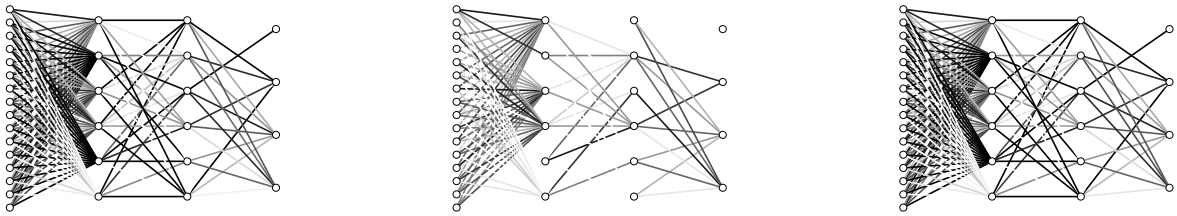
4 実験

ここでは，サイズが 4×4 である雑音パッチを入力として，雑音の確率密度関数を認識する問題を考える．DNN の構築には，TensorFlow v2.5.0 を用いる．確率密度関数として，一様分布 $U(0, 1)$ と 3 種のベータ分布 $\beta(0.5, 0.5), \beta(3.0, 1.0), \beta(1.0, 3.0)$ を考え，各分布に対応する雑音パッチを 10,000 個生成した．DNN は入力層，Flatten 層，全結合層 ×2，及び出力層を持っており，全結合層のノード数は 6 である．また，全結合層と出力層の活性化関数は各々 ReLU と Softmax である．学習には $40,000 \times 4 \times 4$ のテンソルを入力し，出力層は『入力雑音がどの確率密度関数に属するか』を確率推論する．ロス関数は Sparse Categorical Cross Entropy であり，オプティマイザには Adam，エポック数は 30 とした．学習後の認識率 L は 0.737 であり，図 7(a) のようなグラフが得られた[*2]．これを Float 精度で表現し，LZMA 圧縮すると 1438 bytes となる．

提案手法の有効性を確認するために，先の DNN 構造を用いて Network Pruning との比較を行う．1 章で述べたように，この方法は学習時にネットワークの微小な重みを零に丸めることで，有意な重みの数を低減し，認識精度の低下を許容しつつ符号量を抑える．TensorFlow においては零の割合（Final Sparsity）を指定できるため，本実験では 30 % で DNN を学習した．このとき，エポック数は 30 とした．学習した重みの小数部を 5 bit に丸め，Float 精度で LZMA 圧縮すると 742 bytes になる．学習後の認識率は 0.715 であり，図 7(b) のようなグラフが得られた．

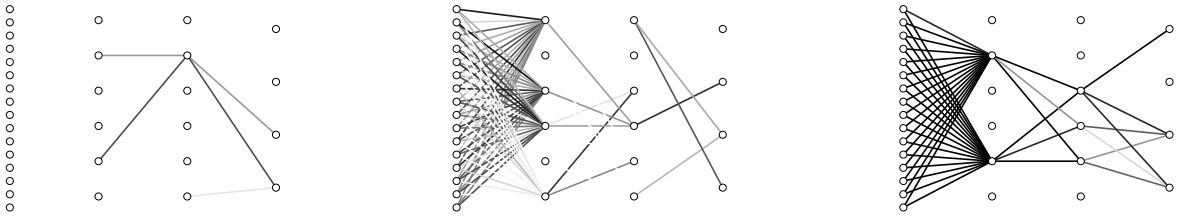
提案手法においては，図 7(a) に対して，手術付きリッチフローを $T = 5$ 回適用する．このとき，手術定数 th は時間 t に依存させ，具体的には t における重みの最大値 ×0.95 とした．また，Network Pruning と同等の認識精度となるように，$L' = 0.715$ とした．これにより，ΔL は $(0.737 - 0.715)/6 = 0.0037$ となり，時刻 t における数値精度の変更による認識精度の低下は 0.0037 にスケジューリングされる．図 8(a)–(e) は $t = 1, \cdots, 5$ におけるリッチフローの切除部であり，(f) は $t = 5$ のリッチフローである．これらを全て統合すると，図 7(a) を再

[*2] 重みを可視化するために，重みの最大値を 1 に規格化して $[0, 1]$ を 256 階調で描画している．

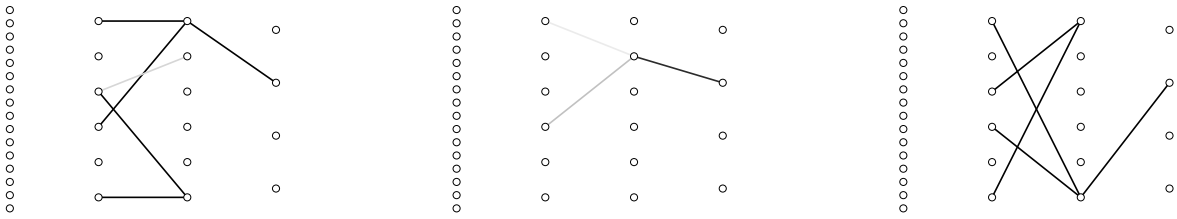


(a) 原グラフ　(b) Network Pruning　(c) 提案手法

図 7　各手法で得られた全結合 NN



(a) $t = 1$ の切除部　(b) $t = 2$ の切除部　(c) $t = 3$ の切除部



(d) $t = 4$ の切除部　(e) $t = 5$ の切除部　(f) $t = 5$ のリッチフロー

図 8　図 7(a) のリッチフローと切除部

構成できる．3.2 節の提案手法に用いて，各切除部 (a)–(e) とリッチフロー (f) の重みの数値精度を適応的に変更した．図 7(c) は数値精度変更後の重みを統合して得られたグラフであり，その符号量は 306 bytes（トポロジーを含む）であった．また，認識精度は 0.7156 であり，目標のロス $L' = 0.715$ に十分近い値となっている．以上より，提案手法は符号化前の DNN の認識精度を保ちつつ，約 80 % の圧縮に成功している．また，Network Pruning と比較すると，提案手法は同一の認識精度である一方，約 60 % の符号量を削減できており，提案手法はレート歪の意味で有効であると言える．

切除部 (a)–(e) とリッチフロー (f) の小数部分の数値精度は各々 $1, 4, 5, 2, 1, 2$ bit であった．3.3 節で述べたように，DNN の推論精度はグラフの連結性と相関がある．つまり，(b),(c) のように，切除部に入力から出力までの計算経路が含まれる場合，推論精度を保つには多くのビット数が必要となる．従って，4,5 bit という比較的多くのビット数が用いられることは，妥当な結果であると言える．一方，局所的に接続されたグラフの数値精度を下げた場合，認識精度への影響は少ない．従って，(a),(d),(e),(f) のようなグラフに対して，その符号化に少ないビット数が割り当てられる．

図 8(c) に着目すると，入力雑音が $U(0,1), \beta(3.0, 1.0), \beta(1.0, 3.0)$ に属する確率を推論する計算過程が視覚的に理解しやすい．例えば，第一全結合層の 2, 5 番目のノード，及び第二全結合層の 3, 4, 5 番目のノードに繋がる重みは，確率推論への寄与率が大きいことが目視できる．このような情報は，図 7(a),(b) から直ちに読み取ることが不可能である．このように，提案手法はネットワークの可視化及びホワイトボックス化に関しても一線を画している．

5　まとめ

本研究では，DNN の効率的な圧縮符号化を目的として，手術付きリッチフローによるグラフの細分化手法とグラフの適応的な数値精度変更手法を提案した．Network Pruning と比較して，提案手法は極めて効率的に DNN を圧縮できることが確認できた．また，リッチフローは，DNN の計算過程を視覚的に理解しやすくする意味で有効であることが確認できた．提案手法におけるリッチフローは，ワッサースタイン計量とダイクストラ法の計算が必要であるが，前者の計算量がボトルネックになっており，大規模なグラフのリッチフローが現状は計算不可能である．今後は，凸解析などを用いて，その高速化に取り組む予定である．

参考文献

[1] J. O' Neill, "An overveiw of neural network compression," *arXiv:2006.03669*, 2020.

[2] R. S. Hamilton, "The Ricci flow on surfaces," *Contemporary Mathematics*, vol. 71, pp. 237–262, 1988.

[3] G. Perelman, "Ricci flow with surgery on three-manifolds," *arXiv:math/0303109*, 2003.

[4] C.-C. Ni, Y.-Y. Lin, F. Luo, and J. Gao, "Community detection on networks with Ricci flow," *Scientific Reports*, vol. 9, no. 9984, 2019.

[5] Y. Ollivier, "Ricci curvature of Markov chains on metric spaces," *Journal of Functional Analysis*, vol. 256, no 3, pp. 810–864, 2009.

[6] E. W. Dijkstra, A note on two problems in connexion with graphs, *Numerische Mathematik*, pp. 269–271, 1959.

[7] V. Vanhoucke, A. Senior, and M. Z. Mao, "Improving the speed of neural networks on CPUs," *Deep Learning and Unsupervised Feature Learning Workshop*, 2011.