社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

Vision Transformer における Continual Learning

武田　麻奈†　　柳井　啓司†

† 電気通信大学 大学院情報理工学研究科 情報学専攻
E-mail: †takeda-m@mm.inf.uec.ac.jp, ††yanai@cs.uec.ac.jp

あらまし　Continual Learning では，過去に学習したタスクの知識を保持しながら，新しいデータから新しいタスクを継続的に学習することを目的としている．近年，自然言語処理で使用される Transformer をコンピュータビジョンに活用した Vision Tranformer が，画像認識タスクにおいて畳み込みニューラルネットワーク（CNN）を凌ぐ高精度を示した．しかし，Vision Transformer で Continual Learning を実現した手法はほとんどない．本論文では，CNN と Vision Transformer の両方へ適用可能な Continual Learning 手法である Mask-RKR を提案する．実験では，CNN と Vision Transformer の両方において，従来手法と比較して，少ないパラメータ数で高精度を達成することを示す．
キーワード　Continual Learning, Vision Transformer

1. はじめに

深層学習モデルでは，複数のタスクを順次与えた場合，過去に学習したタスクを新しいタスクで上書きして忘れてしまう．これを，壊滅的忘却 (catastrophic forgetting) という．Continual Learning では，壊滅的忘却に対処し，過去に学習したタスクの知識を保持しながら，新しいデータから新しいタスクを継続的に学習する．

近年，自然言語処理で使用される Transformer をコンピュータビジョンに活用した Vision Transformer [1], [2] が，画像認識タスクにおいて畳み込みニューラルネットワーク（CNN）を用いた手法を凌ぐ高精度を示している．ただし，従来の Continual Leaning 手法は一般的に CNN の畳み込み層に適用することを考慮しているため，Vision Transformer の全結合層へ適用できる手法は限られる．また，CNN と Vision Transformer の両方への有用性を示したものはない．Vision Transformer で Continual Learning を行うにあたって，Vision Transformer は CNN と比べてモデルサイズが大きいため，Continual Learning 手法を適用した際の追加のモデルサイズが大きくなる．そのため，CNN への適用を前提とする従来手法と比べて，少ないパラメータ数で壊滅的忘却を抑制する必要がある．

本論文では，CNN と Vision Transformer の両方へ適用可能な Continual Learning 手法である Mask-RKR を提案し，少ないパラメータ数で壊滅的忘却を抑制することを目的とする．

2. 関連研究

2.1 Continual Learning

実世界のデータは時間とともに常に変動しており，常に変化する分布，すなわち新しいタスクやクラスが生じる．Continual Learning は，新しいデータは学習するが過去のデータは忘れないための変化に対する剛性と適応するための可塑性という 2 つのトレードオフのバランスをとることを目的としている．この目的を達成するために，多くの研究が提案されている．リハーサル [3] は，学習済みのネットワークに新しいタスクを追加する際，新しいデータと同時に昔の学習で用いた複数のデータを合わせて学習する．蒸留 [4] は，異なる畳み込みニューラルネットワーク (CNN) 間で知識を転送するもので，大規模な CNN の学んだ知識を蒸留して小さくて軽量な CNN の学習に利用される．Elastic Weight Consolidation(EWC) [5] は，特定の重みの学習を過去のタスクにとっての重要度に応じて値が変化しないように調節する．Progressive Neural Networks [6] は，タスクを追加するたびに元のネットワークとは別に同じ構造のネットワークを並列に追加する．PackNet [7] は，あるタスクにおいて重要なパラメータをそのタスクに固有のものとし，新たなタスクの学習はそのパラメータ以外のものを用いて行う．

また，近年では，少ないパラメータ数で，一連のタスクの生成品質を維持または改善することが可能な手法が提案されている．Verma ら [8] は，Continual Learning のためのコンパクトなタスク固有の特徴マップ変換を提案した．特徴マップを変換するためにグループ単位と深さ単位の 2 つの畳み込みを利用する．Zhai ら [9] は，畳み込みフィルタを，動的なタスク固有のベースフィルタと，タスクに依存しない重み行列に因数分解することを提案した．これらの手法は，CNN の畳み込み層へ適用することを前提としており，全結合層のみを持つ Vision Transformer には使用できない．

従来手法の中でも，全結合層へ適用できて Vision Transformer へ応用可能な研究がいくつかある．Singh ら [10] は，Continual Learning において，タスクごとにネットワークの重みと中間出力を修正する手法である Rectification-based Knowledge Retention（RKR）を提案した．重みの修正では，Rectification Generator（RG）というジェネレータから生成されたパラメータを畳み込み層の重みに加えることで修正を行う．中間出力の修正では，Scaling Factor Generator （SFG）というジェネレータから生成されたパラメータを畳み込み層の出力に掛けることで修正を行

う．Piggyback [11] は，ベースモデルの重みに対して学習した重みマスクを適用して出力を変換することで，高精度で多くのタスクを学習する手法である．本論文では，Transformer へ応用可能であるという点から，提案手法に RKR と Piggyback を採用する．

2.2 Vision Transformer

Transformer は自然言語処理における機械翻訳に初めて導入され [12]，今では一般的な手法となっている．オリジナルの Transformer はエンコーダとデコーダの層で構成されていたが，BERT [13] に始まり，後の Transformer は同一のエンコーダブロックを連続して使用するようになった．その後，ViT [1] は，画素のパッチをトークンとすることで，Transformer をコンピュータビジョンに応用することを提案した．大量のデータでモデルを事前に学習し，複数の中型または小型の画像認識ベンチマークに転送することで ViT は CNN の手法より高精度を示した．近年では，ViT のアーキテクチャや学習方法を改良した，DeiT [14], CaiT [15], ConVit [16], Swin Transformer [2] など手法が提案された．Swin Transformer [2] は，シフトウィンドウを用いて階層的に計算することで，言語からビジョンへの Transformer の適応の課題を解決した手法である，具体的には，視覚的実体のスケールの大きな変化や，テキスト中の単語と比較した画像中のピクセルの高い解像度などに対処した．本論文では，提案手法を ViT と Swin Transformer へ適用することで，Vision Transformer に適用可能であることを示す．

2.3 Vision Transformer における Continual Learning

近年，Vision Transformer に特化した Continual Learning 手法がいくつか提案されている．DyTox [17] は，Transformer アーキテクチャに基づいた，Continual Learning のための手法である．全てのタスク間で初期レイヤを共有し，タスクに特化した埋め込みを生成するために，特別なタスク固有のトークンを使用する．Learning to Prompt for Continual Learning (L2P) [18] は，自然言語処理分野における新しい Continual Learning 手法であるプロンプト学習 [19] から発想を得た手法である．これらの手法は，Vision Transformer に適用できる点で提案手法と同じである．ただし，Vision Transformer に特化した手法のため CNN に適用できない点で提案手法と異なる．

3. 手　　法

本論文では，Vision Transformer における Continual Learning を行う手法として，Mask-RKR を提案する．Mask-RKR は，ベースとなる RKR [10] に対して Piggyback [11] を適用することで，RKR の特性を残しつつ，タスク数が増えることによる追加のパラメータ数を抑えることが可能である．

3.1 RKR によるタスクへの適応

Mask-RKR は，RKR をベースとして用いて，タスクごとにネットワークの重みと中間出力を修正する．[10] と同様に，RG というジェネレータから生成されたパラメータを畳み込み層の重みに加えることで，重みの修正を行う．また，SFG というジェネレータから生成されたパラメータを畳み込み層の出力に掛けることで，中間出力の修正を行う．これにより，複数タスクへモデルを適応させる．

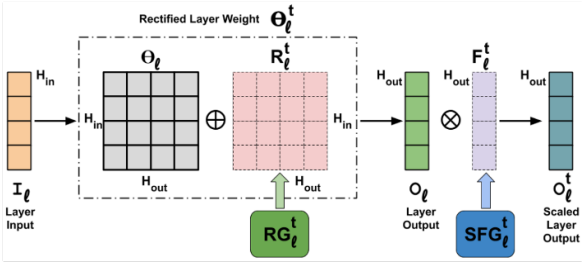

図 1　Mask-RKR における RKR を用いたタスクへの適応の概要 ([10] から引用)

本論文では，RKR を Vision Transformer の最終の出力層を除く全ての全結合層に対して適用する．それぞれの層に対して，RG による重みの修正と SFG による中間出力の修正を行う．これにより，タスクごとにモデルが最適化され，壊滅的忘却を回避する．Vision Transformer に含まれる全結合層における RKR を**図** 1 に示す．タスク t の l 番目のレイヤの場合，RG_l^t は，各タスクに適応した全結合層の重み θ_l^t を生成する．そして，θ_l^t を入力 I_l に適用することで出力 O_l を生成する．SFG_l^t は，スケーリング係数 F_l^t を生成し，O_l に適用してスケーリングされた出力 O_l^t を生成する．

RG による重みの修正を**式** (1) に示す．

$$\Theta_l^t = \Theta_l \oplus R_l^t \tag{1}$$

$$R_l^t = MATMUL(LM_l^t, RM_l^t) \tag{2}$$

RG では，重みを修正するパラメータ R が学習され，タスク t のレイヤ l の重みに R_l^t がそれぞれ加算される．ここで，Θ_l はレイヤ l の重み，Θ_l^t はタスク t のレイヤ l の修正された重み，$\oplus$ は要素毎の和である．また，RG ではパラメータ数を削減する工夫として，R_l^t の低ランク近似を行う．これを**式** (2) に示す．そのため，RG ではサイズが小さい 2 つの行列 LM_l^t と RM_l^t を学習し，これら 2 つの行列の積により重み修正のパラメータ R_l^t を生成する．畳み込み層では，サイズ $(W_f * C_{in}) \times K$ の行列 LM_l^t とサイズ $K \times (H_f * C_{out})$ の行列 RM_l^t から，サイズ $W_f \times H_f \times C_{in} \times C_{out}$ のフィルタ R_l^t を生成する．ここで，W_f,H_f,C_{in},C_{out} はそれぞれ畳み込みフィルタの幅，高さ，入力チャネル，出力チャネルである．全結合層では，サイズ $H_{in} \times K$ の行列 LM_l^t とサイズ $K \times H_{out}$ の行列 RM_l^t から，サイズ $H_{in} \times H_{out}$ のフィルタ R_l^t を生成する．ここで，H_{in},H_{out} はそれぞれ全結合層の入力と出力サイズである．MATMUL は行列積を表す．K は低ランク近似のランクである．また，$*$ はスカラー乗算を表し，$K << (W_f * C_{in})$，$K << (H_f * C_{out})$ かつ $K << H_{in}$，$K << H_{out}$ である．

SFG による中間出力の修正を**式** (3) に示す．

$$O_l^t = O_l \odot F_l^t \tag{3}$$

SFG では，中間出力を修正するパラメータ F が学習され，タスク t のレイヤ l の中間出力に F_l^t が乗算される．ここで，O_l はレイヤ l の出力，O_l^t はタスク t のレイヤ l の修正された出力，

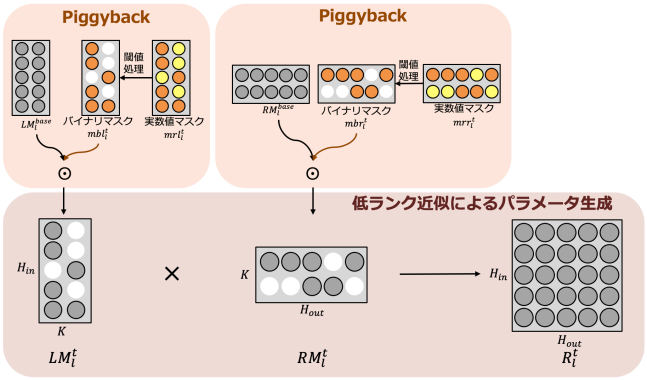

図 2　Mask-RKR における RG の概要

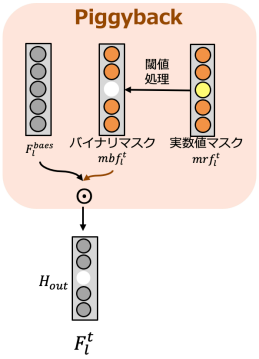

図 3　Mask-RKR における SFG の概要

⊙ は要素毎の積である．SFG では，F_l^t を学習する．畳み込み層では F_l^t のサイズは C_{out}，全結合層では H_{out} である．RG と比べ，SFG では生成するパラメータ数が少ないため行列の低ランク近似は不要である．

3.2　Piggyback によるパラメータ削減

CNN と比べて Vision Transformer はパラメータ数が多くなるため，Continual Learning 手法を適用したときに，タスクごとに増加するパラメータ数が多くなる．RKR はパラメータ数を抑えるために低ランク近似によるパラメータ生成を行うなどの工夫を行なっている．それに加えて Mask-RKR では，Piggyback を用いることでさらなるパラメータ数の削減を行う．

Piggyback は，ベースモデルの重みに対して学習した重みマスクを適用して出力を変換する手法である．Mask-RKR は，直接重みに対して Piggyback を適用するのではなく，RKR のパラメータに適用する．具体的には，RG の低ランク近似された 2 つのパラメータ LM と RM，SFG のパラメータ F に適用する．マスクの学習は，実数値の重みのセットを保持し，決定論的閾値関数を通過させてバイナリマスクを取得し，それを既存の重みに適用することで実現される．誤差逆伝播により実数値の重みを更新することで，タスクに適したバイナリマスクが学習される．タスクごとに異なるバイナリマスクを学習し，RG と SFG のパラメータに要素ごとに適用することで，最小限の追加パラメータで，同じベースネットワークを複数のタスクに再利用することができる．本論文では，大規模データセットの ImageNet-1k で学習したモデルをベースモデルとして使用する．

Mask-RKR の RG におけるマスク学習の手順を，**図** 2 に示す．RG では，タスク t のレイヤ l の重みに対して重みを修正するパラメータ R_l^t がそれぞれ加算される．R_l^t は，パラメータ削減のために低ランク近似され，2 つの行列 LM_l^t と RM_l^t に分解して学習される．ここで，畳み込み層では $R_l^t \in \mathbb{R}^{W_f \times H_f \times C_{in} \times C_{out}}$，$LM_l^t \in \mathbb{R}^{(W_f * C_{in}) \times K}$，$RM_l^t \in \mathbb{R}^{K \times (H_f * C_{out})}$，全結合層では $R_l^t \in \mathbb{R}^{H_{in} \times H_{out}}$，$LM_l^t \in \mathbb{R}^{H_{in} \times K}$，$RM_l^t \in \mathbb{R}^{K \times H_{out}}$ となる．まず，ImageNet-1k でベースとなるモデルを学習する．ImageNet-1k では，マスクは学習せずに LM_l^{base} と RM_l^{base} のみ学習する．次に，タスク t では，ImageNet-1k で学習されたベースとなるパラメータ LM_l^{base} と RM_l^{base} は固定したまま，マスクのみを学習する．マスクは実数値で学習される．実数値マスクは LM と RM それぞれで学習し，LM の実数値マスク mrl_l^t は LM_l^1 と同じサイズ，RM の実数値マスク mrr_l^t は RM_l^1 と同じサイズである．それぞれの実数値マスクを [11] と同様に，**式** (4) で与えられるハードバイナリ閾値関数に通すことにより，閾値化されたマスク行列 mbl_l^t と mbr_l^t を得る．ここで，mr は実数値マスク，mb はバイナリマスク，τ は選択された閾値である．バイナリマスクでは，閾値化されたマスク行列の特定の値 mb_{ji} が 0 か 1 かによって，ベースとなるパラメータを活性化したり消したりする．

$$mb_{ji} = \begin{cases} 1, \text{ if } mr_{ji} \geq \tau \\ 0, \text{ otherwise} \end{cases} \tag{4}$$

閾値処理によって生成されたバイナリマスクを用いると，タスク $t > 1$ において，**式** (2) で表される RG による重みの修正は**式** (5) に置き換えられる．ここで，⊙ は要素ごとの積またはマスキングを示す．

$$\begin{aligned} LM_l^t &= LM_l^{base} \odot mbl_l^t \\ RM_l^t &= RM_l^{base} \odot mbr_l^t \\ R_l^t &= MATMUL((LM_l^{base} \odot mbl_l^t) \\ &\quad , (RM_l^{base} \odot mbr_l^t)) \end{aligned} \tag{5}$$

次に Mask-RKR の SFG におけるマスク学習の手順を，**図** 3 に示す．SFG では，タスク t のレイヤ l の中間出力に $F_l^t \in \mathbb{R}^{C_{out}}$ が乗算される．ここで，畳み込み層では $F_l^t \in \mathbb{R}^{C_{out}}$，全結合層では $F_l^t \in \mathbb{R}^{H_{out}}$ である．まず，SFG でも RG と同様に，ImageNet-1k でベースとなるネットワークを学習する．ImageNet-1k では，マスクは学習せずに F_l^{base} のみ学習する．次に，タスク t では，ImageNet-1k で学習されたベースとなるパラメータ F_l^{base} は固定したまま，実数値マスクのみを学習する．F_l^t の実数値マスク mrf_l^t は，F_l^1 と同じサイズである．それぞれの実数値マスクを [11] と同様に，**式** (4) で与えられるハードバイナリ閾値関数に通すことにより，閾値化されたマスク行列 mbf_l^t を得る．

与えられたタスクのためのマスクを学習した後，実数値マスクの重みは不要となる．実数値マスクは破棄され，バイナリマスクのみを保存する．典型的なニューラルネットワークのパラメータは，（PyTorch の実装も含めて）32 ビットの float 値で表現される．バイナリマスクはパラメータごとに 1 ビットしか必要とせず，タスクごとの追加パラメータ数はおおよそベースパラメータの 1/32 になる．

表 1　実験 3 で使用したデータセット

データセット	概要	クラス数	訓練データ	テストデータ
D. Textures	テクスチャとパターン	47	1,880	1,880
GTSRB	ドイツの道路標識	4	31,367	7,842
SVHN	数字画像	10	47,217	26,040
UCF101	アクション	101	7,629	1,908
VGG-Flowers	花	102	1,020	1,020

4. 実　　験

4.1 実験概要

本論文では，ベースラインとの比較実験とアブレーション実験を行った．ベースラインとの比較実験では，提案手法の性能を検証するために，3 つのモデルへ様々な Continual Learning 手法を適用してその性能を比較した．クラス数やドメインが異なるデータセットを使用して 3 つの Continual Learning 設定で実験を行った．

- 実験 1：CIFAR-100 を用いた実験
- 実験 2：ImageNet-1k を用いた実験
- 実験 3：異なるドメインのデータセットを用いた実験

アブレーション実験では，提案手法である Mask-RKR の構成要素が及ぼす影響を検証した．

実験 1 とアブレーション実験では，動植物や機器，乗り物など 100 クラスが含まれるデータセットである CIFAR-100 を使用した．Continual Learning 設定で実験を行うために，CIFAR-100 をそれぞれ 10 のクラスを持つ 10 タスクに分割した．画像サイズは 32×32 である．訓練データに 50,000 枚，テストデータに 10,000 枚を使用した．実験 2 では，1,000 クラスが含まれる大規模なデータセットである ImageNet-1k を使用した．Continual Learning 設定で実験を行うために，ImageNet-1k をそれぞれ 100 のクラスを持つ 10 タスクに分割した．画像サイズは 224×224 である．訓練データに 1,232,167 枚，テストデータに 49,000 枚を使用した．実験 3 では，Visual Decathlon（VD）[20] ベンチマークから異なるドメインの 5 つのデータセットを使用した．VD ベンチマークとは，10 個の異なる視覚領域を同時に解決する能力を評価するベンチマークである．使用したデータセットを**表** 1 に示す．画像サイズは全てのデータセットで 72×72 である．

本実験では，ResNet-18 [21]，ViT [1]，Swin Transformer [2] の 3 つのモデルを使用した．ResNet-18 は CNN，ViT と Swin Transformer は Vision Transformer のアーキテクチャをベースとしている．Swin Transformer は入力する画像サイズに対応したモデルを使用する必要があるため，実験 1，実験 3 とアブレーション実験ではモデルサイズが小さい Swin-Tiny モデル，実験 2 では一般的な Swin Transformer のモデルである Swin-Base モデルを使用した．実験で使用するモデルは全て，ImageNet-1k で事前学習済のモデルを使用した．

実験では，ベースラインとして Transformer で実装可能な従来の Continual Learning 手法を使用して，提案手法の性能を比較した．「Single」は，各タスクを個別のモデルで学習したものである．「Multi Head」は，最終出力層のみをタスクごとに入れ替えて学習させた手法である．「RKR」[10] は，タスクごとにネットワークの重みと中間出力を修正する手法である．全ての実験において，低ランク近似のランク K は，最もパラメータ数が少ない $K = 2$ を使用した．「Piggyback」[11] は，学習した重みマスクを適用して出力を変換する手法である．ハードバイナリ閾値関数の閾値 τ は, [11] と同様に 5e-3 とした．「Ours」は，提案手法の Mask-RKR である．ベースとなる RKR に Piggyback を適用することでパラメータ数を抑えた手法である．Piggyback のハードバイナリ閾値関数の閾値 τ は,「Piggyback」と同様に 5e-3 とした．低ランク近似のランク K は 2 つの場合で実験を行なった．1 つ目は $K = 2$ としてパラメータ数を最も抑えた「Ours(K=2)」，2 つ目は K の値を増やしてパラメータ数を「RKR」と同じにした「Ours K+」である．実験で使用した全ての Mask-RKR のベースパラメータは，ImageNet-1k で事前学習したものを使用した．

表 2　CIFAR-100 を用いた実験の結果（実験 1）

手法	平均精度			パラメータ数 [M]		
	ResNet-18	ViT	Swin	ResNet-18	ViT	Swin
Single	0.833	0.857	0.876	111.72	856.59	11.98
Multi Head	0.727	0.791	0.768	11.22	85.73	1.22
RKR(K=2)	0.794	0.843	0.858	11.74	89.88	1.43
Piggyback	**0.804**	0.838	**0.875**	14.71	112.27	1.56
Ours(K=2)	0.781	0.840	0.841	11.28	86.26	1.24
Ours K+	0.796	**0.845**	0.858	11.74	89.87	1.43

表 3　ImageNet-1k を用いた実験の結果（実験 2）

手法	平均精度			パラメータ数 [M]		
	ResNet-18	ViT	Swin	ResNet-18	ViT	Swin
Single	0.678	0.888	0.902	112.18	858.76	868.46
Multi Head	0.523	0.871	0.887	11.68	86.57	87.77
RKR(K=2)	0.545	**0.885**	0.892	12.20	90.71	92.34
Piggyback	0.440	0.881	0.805	15.17	113.11	113.94
Ours(K=2)	0.557	0.879	0.870	11.75	87.10	88.35
Ours K+	**0.582**	**0.885**	**0.894**	12.43	90.71	92.30

表 4　異なるドメインのデータセットを用いた実験の結果（実験 3）

手法	平均精度			パラメータ数 [M]		
	ResNet-18	ViT	Swin	ResNet-18	ViT	Swin
Single	0.776	0.816	0.842	111.91	857.39	594.62
Multi Head	0.567	0.625	0.682	11.32	85.89	59.59
RKR(K=2)	0.714	0.791	**0.840**	11.58	87.97	61.49
Piggyback	**0.723**	**0.809**	0.839	13.07	99.16	68.75
Ours(K=2)	0.695	0.775	0.824	11.38	86.36	60.02
Ours K+	0.720	0.778	0.831	11.52	87.67	61.39

全ての実験において，オプティマイザは，初期学習率 3e-2，モーメンタム 0.9 の SGD を使用した．スケジューラは，linear warmup と cosine decay を設定した．損失関数は，Cross Entropy Loss を使用した．バッチサイズは 100 である．評価指標には，学習した全タスクの精度の平均を使用した．

4.2 ベースラインとの比較実験

ベースラインとの比較実験では，提案手法の性能を検証するために，3 つのモデルへベースラインと提案手法を適用してそ

の性能を比較した．

実験 1 では，CIFAR-100 を 10 クラスを持つ 10 タスクに分割して実験を行った．分類クラス数が 10 であり，それぞれのタスクが似たドメインであるという，比較的簡単な Continual Learning 設定での提案手法の精度を検証する．実験結果を**表** 2 に示す．ここで，表中の太字は，最も精度が高い値を示している．「Ours K+」の K の値は，ResNet-18 で 17，ViT で 19，Swin Transformer で 18 を使用した．実験の結果，全てのモデルにおいて「RKR」，「Piggyback」，「Ours」が全体的に高い精度を示した．また，ResNet-18 と Swin Transformer では「Piggyback」，ViT では「Ours K+」が最高精度を示した．「Multi Head」は，タスクごとに最終層を用意するだけなので追加パラメータ数はかなり少なく抑えられている．しかし，精度はその他の手法に劣っており，最終層を入れ替えただけでは，モデルをタスクに上手く適用できないことがわかる．「RKR」と「Piggyback」は，パラメータ数を抑えつつ高い精度を示している．どちらも近い精度を示しているが，パラメータ数の増加は「RKR」の方が抑えることができている．「Ours(K=2)」は，「RKR」と「Piggyback」と比べて，同等の精度を示しつつパラメータ数の増加が小さい．Mask-RKR は，重みフィルタと同じサイズのマスクを用意する必要がある Piggyback と比べ，低ランク近似されたパラメータと同じサイズのマスクを用意するだけで良いので Piggyback よりパラメータ数を削減できる．また，タスク固有のパラメータをそれぞれ用意する必要がある RKR と比べ，Mask-RKR はベースパラメータに対してタスクごとに固有のバイナリマスクを適用してパラメータを生成するので，必要な追加パラメータ数が 1/36 に削減できる．「Ours(K=2)」は，パラメータ数を抑える代わりに，基本的に「RKR」と「Piggyback」より精度は低くなってしまう．ただし，Mask-RKR は RKR をベースとしているため，RKR と同様に低ランク近似のランク K を増加させることで，パラメータ数が増加する代わりに精度を上げることができる．「Ours K+」は「RKR」と同じパラメータ数でありながら「RKR」よりも高い精度を示した．このことから，Mask-RKR は RKR と同じパラメータ数を使用すると，RKR よりも優れた精度を達成できるといえる．以上の結果から，Mask-RKR はパラメータ数の増加を最小限に抑えつつ高い精度を達成できることがわかった．

実験 2 では，ImageNet-1k を 100 クラスを持つ 10 タスクに分割して実験を行った．それぞれのタスクが似たドメインであるが，分類クラス数は 100 クラスであり，実験 1 よりも難しい Continual Learning 設定での提案手法の精度を検証する．「Ours K+」の K の値は，ResNet-18 で 3，ViT で 18，Swin Transformer で 19 を使用した．実験結果を**表** 3 に示す．実験の結果，実験 1 と同様に全てのモデルにおいて「RKR」，「Piggyback」，「Ours」が全体的に高い精度を示した．「Ours(K=2)」は，「RKR」と「Piggyback」と比べて，同等の精度を示しつつパラメータ数の増加が小さい．また，「Ours K+」は，「RKR」と同じパラメータ数でありながら「RKR」よりも高い精度を示した．このことから，実験 1 よりも難しい設定の Continual Learning 設定でも，Mask-RKR はパラメータ数の増加を最小限に抑えつつ高い精度を達成できることがわかった．

表 5　Piggyback の有用性の検証結果

RG	SFG	平均精度			パラメータ数 [M]		
w/ PB	w/ PB	ResNet-18	ViT	Swin	ResNet-18	ViT	Swin
×	×	**0.794**	0.843	**0.858**	11.74	89.88	1.43
✓	×	0.780	0.844	0.846	11.33	87.07	1.28
×	✓	**0.794**	**0.845**	**0.858**	11.69	89.15	1.40
✓	✓	0.781	0.840	0.841	11.28	86.26	1.24

表 6　Piggyback の適用場所の検証結果

手法	平均精度			パラメータ数 [M]		
	ResNet-18	ViT	Swin	ResNet-18	ViT	Swin
RG w/o PB	0.794	**0.845**	**0.858**	11.69	89.15	1.40
R w/ PB	**0.805**	**0.845**	0.847	14.41	110.05	1.55
LMRM w/ PB	0.781	0.840	0.841	11.28	86.26	1.24

実験 3 では，異なるドメインを持つ D. Textures，GTSRB，SVHN, UCF101, VGG-Flower を順に学習して実験を行う．「Ours K+」の K の値は，ResNet-18 で 9，ViT で 11，Swin Transformer で 10 を使用した．実験結果を**表** 4 に示す．実験の結果，実験 1，実験 2 と同様に全てのモデルにおいて「RKR」，「Piggyback」，「Ours」が全体的に高い精度を示した．ただし，実験 1 や実験 2 と比べて，ViT と Swin Transformer を使用した場合，「Ours K+」は「RKR」以上の精度が得られなかった．タスク固有のパラメータをそのまま生成する RKR と比べて，Mask-RKR はベースパラメータへマスクを適用してタスク固有のパラメータを生成する．そのため，ベースパラメータから遠い距離にあるドメインに柔軟に対応できない可能性があることが原因だと考えられる．このことから，異なるドメインのデータセットの Continual Learning 設定では，Mask-RKR は RKR と比べると精度が低下することがわかった．

4.3　アブレーション実験

アブレーション実験では，Mask-RKR の構成要素が異なる場合の，精度とパラメータ数を比較した．アブレーション実験では，Mask-RKR のハードバイナリ閾値関数の閾値 τ =5e-3，低ランク近似のランク $K = 2$ で実験を行った．

まずは，Mask-RKR の RG と SFG のそれぞれに Piggyback を適用する場合としない場合を比較することで，Piggyback の有用性を検証した．実験結果を**表** 5 に示す．「RG w/ PB」と「SFG w/ PB」はそれぞれ，RG と SFG に Piggyback が適用されているかどうかを表す．ここで，RG と SFG の両方に Piggyback を適用しない場合は RKR と同じである．実験結果から，RG と SFG の両方に Piggyback を適用しないモデル（RKR）と，SFG にのみ Piggyback を適用するモデルが最も高い精度を示すことがわかった．これは，タスク固有のパラメータをそのまま生成する場合と比べ，Piggyback はベースパラメータへマスクを適用してタスク固有パラメータを生成するため，生成されるパラメータに制約が生じてしまうことが原因と考えられる．また，SFG にのみ Piggyback を適用するモデルの精度が高い理由としては，SFG は RG と比べてパラメータの修正に与える影響が小さいことが原因と言える [10]．そのため，SFG へ Piggyback を

適用しても精度へ与える影響が小さい．ただし，これらのモデルは RKR と比べて削減できるパラメータ数が少ない．RG と SFG の両方に Piggyback を適用したモデルは，パラメータ数が最も少ないモデルである．2 番目にパラメータ数が少ないのは，RG にのみ Piggyback を適用したモデルであるが，RG と SFG の両方に Piggyback を適用したモデルの方が，同等の精度でパラメータ数が少ない．本実験では，Vision Transformer へ適用することを考慮し，最もパラメータ数を抑えることができる RG と SFG の両方に Piggyback を適用したモデルを使用する．

次に，Mask-RKR の RG において，重み修正パラメータ R に直接 Piggybask を適用した場合の「R w/ PB」と，LM と RM それぞれに Piggyback を適用する場合の「LMRM w/ PB」と，RG に Piggyback を適用しない場合の「RG w/o PB」を比較した．全てのモデルで，SFG には Piggyback を適用している．実験結果を**表** 6 に示す．実験の結果，「R w/ PB」は「RG w/o PB」や「LMRM w/ PB」と比べてパラメータ数が増加することが分かった．LM や RM と比べて R はパラメータ数が膨大であり，それと同じサイズのマスクを用意する必要がある「R w/ PB」はより多くのパラメータを必要とすることが原因といえる．また，精度は「LMRM w/ PB」より向上するが，「RG w/o PB」とは同程度であり，Piggyback を適用したことによる精度の向上は見られなかった．以上の結果から，パラメータ数を抑えるためには，RG では Piggyback を LM と RM に適用した方が効果的であることが分かった．

5. おわりに

本論文では，CNN と Vision Transformer の両方へ適用可能な Continual Learning 手法である Mask-RKR を提案した．Mask-RKR は，タスクごとにネットワークの重みと中間出力を修正することで様々なタスクにモデルを適用させる手法である RKR [10] と，学習した重みマスクを適用して出力を変換する Piggyback [11] を組み合わせた手法である．実験から，Mask-RKR は従来手法と比べて，パラメータ数の増加を抑えつつ高い精度を達成できることがわかった．また，RKR よりもパラメータ数を抑えることができ，パラメータ数に制限を持つ現実社会の問題に対して，より幅広く対応できる汎用性を持つといえる．

今後は，異なるドメインのデータセットを用いた Continual Learning にも対応できる柔軟性を持たせたい．具体的には，レイヤ毎に適応的に RKR のパラメータの近似度合いを調整することで，パラメータ数をおさえつつ高精度な Continual Learning の実現を目指す．

文　献

[1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proc. of International Conference on Learning Representation*, 2021.

[2] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proc.of IEEE Computer Vision and Pattern Recognition*, 2021.

[3] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, Vol. 7, No. 2, pp. 123–146, 1995.

[4] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *Proc. of Neural Information Processing Systems conference*, 2015.

[5] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proc.of the National Academy of Sciences*, 2016.

[6] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv:1606.04671*, 2016.

[7] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proc.of IEEE Computer Vision and Pattern Recognition*, 2018.

[8] Vinay Kumar Verma, Kevin J. Liang, Nikhil Mehta, Piyush Rai, and Lawrence Carin. Efficient feature transformations for discriminative and generative continual learning. In *Proc.of IEEE Computer Vision and Pattern Recognition*, 2021.

[9] Mengyao Zhai, Lei Chen, and Greg Mori. Hyper-lifelonggan: Scalable lifelong learning for image conditioned generation. In *Proc.of IEEE Computer Vision and Pattern Recognition*, 2021.

[10] Pravendra Singh, Pratik Mazumder, Piyush Rai, and Vinay P. Namboodiri. Rectification-based knowledge retention for continual learning. In *Proc.of IEEE Computer Vision and Pattern Recognition*, 2021.

[11] Arun Mallya and Svetlana Lazebnik. Piggyback: Adding multiple tasks to a single, fixed network by learning to mask. In *Proc.of European Conference on Computer Vision*, 2018.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proc. of Neural Information Processing Systems conference*, 2017.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proc. of the Conference of the North American Chapter of the Association for Computational Linguistics*, 2018.

[14] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *Proc. of International Conference on Machine Learning*, 2021.

[15] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *Proc.of IEEE International Conference on Computer Vision*, 2021.

[16] Stéphane d'Ascoli, Hugo Touvron, Matthew L. Leavitt, Ari S. Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. *arXiv:2103.10697*, 2021.

[17] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. *arXiv:2111.11326*, 2021.

[18] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. Learning to prompt for continual learning. *arXiv:2112.08654*, 2021.

[19] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv:2107.13586*, 2021.

[20] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. *In Advances in Neural Information Processing Systems*, p. 506–516.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc.of IEEE Computer Vision and Pattern Recognition*, 2016.