Vol.2019-MPS-123 No.46
Vol.2019-BIO-58 No.46
2019/6/19

継続学習の安定性向上のためのGenerative Replayの改善

村田 健悟[1,a)]　豊田 哲也[2]　大原 剛三[2]

概要：1 つの学習モデルで同種の異なるタスクを継続的に学習する継続学習の実現は，汎用人工知能の達成において不可欠な要件の一つである．しかしながら，近年，様々な分野で著しい成績を残しているニューラルネットワークモデルを用いた場合には，新規タスクの学習により学習済みタスクを忘却してしまう，破滅的忘却と呼ばれる問題が生じることが知られており，これまでにその回避手法が幾つか提案されている．その中の 1 つの Generative Replay は，過去タスクのデータを生成する生成モデルを導入することで増加し続ける膨大な訓練データを保持し続けなくてもよいという利点をもつ反面，生成モデル自体に生じる忘却，生成サンプルの偏りによる学習継続時の性能低下という問題をもつ．そこで本研究では，生成モデル学習時の損失関数，およびサンプル生成法を改良することで，その問題を軽減することを試みる．また，ベンチマークデータを利用した実験を通して，学習した生成モデルおよび分類モデルの精度向上を示す．

Improving Stability of Continual Learning by Generative Replay

1. はじめに

1 つの学習モデルを用い，同種の異なるタスクを継続的に学習する継続学習の実現は，汎用人工知能の達成において不可欠な要件の一つである [1]．しかしながら，ニューラルネットワークモデルによる継続学習の実現は，破滅的忘却と呼ばれる，新規タスクの学習により学習済みタスクを忘却してしまう現象により，非常に困難であることが知られている [2]．

破滅的忘却は，新規タスク学習時に過去タスクの全データを用いることで，完全に回避することが可能である．しかしながら，この方法は，タスクの増加とともに保持データ数が膨大なものとなるため，現実的であるとはいえない．そこで，これまでに，過去タスクのデータを保持することなく，または，少量のデータのみを保持することで，破滅的忘却の回避を目論むモデルが幾つか提案されてきた [1,3–7]．その中の 1 つである Generative Replay [6] は，過去タスクのデータを生成する生成モデルを導入し，新規タスクの学習時に過去タスクを表す生成サンプルを使用する手法である．同手法は，過去タスクのデータを一切保持する必要がないという利点をもつ反面，生成モデル自体に生じる忘却，生成サンプルの偏りによる学習継続時の性能低下という問題をもつ．

本稿では，Generative Replay について，上記問題を軽減し，継続学習の安定性を向上させる手法を提案する．具体的には，損失関数の忘却度合いに対する滑らかさ，および Variational Autoencoder（VAE）[8] モデルにおける潜在変数への制約に着目し，同モデルに対する損失関数を改良する．また，クラスに対する潜在変数の条件付き分布を導入することで，偏りの少ないサンプル生成を実現する．さらに，これらの改良を行った提案手法に対し，ベンチマークデータを利用した実験を行い，生成モデルおよび分類モデルの精度向上を示す．

以下，2 章で関連研究について紹介し，3 章で Replay-through-Feedback（RtF）と呼ばれる関連手法の概要を説明する．4 章で提案手法の詳細について述べ，5 章で評価実験の報告および提案手法の有効性を議論する．6 章でまとめを述べる．

2. 関連研究

近年，破滅的忘却を回避し継続学習を実現する様々なニューラルネットワークモデルが提案されている．その多くは分類問題を対象としたモデルであり，これらは大きく，

[1] 青山学院大学大学院理工学研究科
Graduate School of Science and Engineering, Aoyama Gakuin University, Kanagawa, 252-5258 Japan
[2] 青山学院大学理工学部
College of Science and Engineering, Aoyama Gakuin University, Kanagawa, 252-5258 Japan
[a)] c5619156@aoyama.jp

正則化による手法 [1,3]，過去タスクの一部データを利用する手法 [4,5]，Generative Replay による手法 [6,7] の 3 つに分類することができる．

正則化による手法は，過去タスクを解くために重要なパラメータを変更しないように，新規タスクを学習する方法である．具体的には，過去タスクに対する各パラメータの “重要度” を計算し，重要なパラメータの変動に大きな罰則を与える正則化項を導入する．重要度の計算方法は様々なものが提案されており，Elastic Weight Consolidation（EWC）[1] では，フィッシャー情報量行列を利用し，Synaptic Intelligence（SI）[3] では，学習時の勾配を利用する．しかしながら，これら正則化による手法は，正則化項のハイパーパラメータ調整が不可欠であり，また，推論時にタスク情報が与えられない場合，忘却の回避が困難になることが報告されている [9]．

過去タスクの一部データを利用する手法としては，Incremental Classifier and Representation Learning（iCaRL）[4] や，Gradient Efficient Memory（GEM）[5] が挙げられる．iCaRL は，保持した過去タスクデータを訓練に使用し忘却を防ぐ．一方，GEM は，保持した過去タスクデータの勾配情報を利用し，破滅的忘却を回避する．また，保持する過去タスクデータの選択方法としては，各クラスごとに同一数のサンプルをランダムに選択する方法 [5] や，学習サンプルを得るごとに動的に保持サンプルを変化させる方法 [10] が存在する．これら過去タスクの一部データを利用する手法は，保持するサンプル数により大きく精度が変化し [5]，また，データの保持が可能でないと適用できないという問題を抱えている．

Generative Replay による手法は，過去タスクに属するデータを学習した生成モデルを導入し，新規タスクの学習時に過去タスクを表す生成サンプルを使用する方法である．生成サンプルの教師ラベルには，過去タスクを学習した分類器の出力を用いる．Deep Generative Replay（DGR）[6] は，Wasserstein GAN（WGAN）[11,12] を生成モデルとしサンプルを生成する．しかしながら，同手法には，分類モデルと生成モデルをそれぞれ訓練する必要があるため，計算コストが高いという問題がある．これに対して，Replay-through-Feedback（RtF）[7] では，生成モデルを分類モデルに統合することでその問題を解決した．具体的には，生成モデルとして Variational Autoencoder（VAE）[8] を使用し，エンコーダ部の最終隠れ層にクラス分類モジュールを追加した構造をとる．しかしながら，これらの手法は生成モデル自体に発生する忘却を十分に回避できないという問題を抱えている．

一方，画像生成などのような生成問題を対象とした継続学習モデル [13,14] は，分類問題と比べあまり提案がなされていない．Lifelong Generative Modeling（LGM）[13] は，Generative Replay により忘却を防ぎ，また，潜在変数の一部に離散分布を導入することで，偏りのないサンプル生成を実現した．また，Variational Autoencoder with Shared Embeddings（VASE）[14] は，過去モデルのパラメータを使用して忘却を防ぐことで，継続的な disentangled representation [15] の獲得を実現した．

3. RtF による学習

本章では，提案手法の基礎となる RtF [7] の学習の詳細について説明する．同手法は，継続学習を実現するニューラルネットワークモデルであり，Generative Replay により破滅的忘却を回避する．以下では，3.1 節にて本研究における問題設定，3.2 節にて RtF の概要，3.3 節にて RtF の問題点について論じる．

3.1 問題設定

以下では，各タスク T_i がデータセット D_i を持つようなタスク列 $\mathbf{T} = (T_1, T_2, \cdots, T_M)$ に対し，全てのタスク $\mathbf{T}$ を解くことのできるモデルの作成について考える．ここで，各データセット D_i は，N_i 個のサンプル $\{\boldsymbol{x}_j, y_j\}_{j=1}^{N_i}$ から構成されるとし，D_i の各サンプルが属するクラス集合を C_i で表す．なお，特にデータ集合 $\{\boldsymbol{x}_j\}_{j=1}^{N_i}$ を $\mathbf{X}_i$ で表す．また，タスク T_i の学習終了後には，データセット D_i を保持することはできない．すなわち，タスク T_i の学習時に過去タスクのデータセット $D_1, D_2, \cdots, D_{i-1}$ を参照することはできないものとする．なお，本研究では，任意の 2 タスク T_i，$T_j (i \neq j)$ に対し，クラス集合 C_i，C_j が互いに素であると仮定し，タスクに対し線形にクラス数が増加するような場合について考える．継続学習においては，一般に，クラス集合 C_i，C_j が互いに素ではなく，同一クラスのデータが別タスクに出現する場合も考えられる．しかしながら，クラス数が増加する場合における継続学習が，同一クラスのデータが増加する場合よりも，忘却を防ぐことが困難であることが報告されており [9]，本研究では前者の忘却困難性の解決に重点を置き，クラス集合に関する上記仮定を置いた．なお，RtF はクラス集合に関するこのような仮定を置いておらず，また，提案手法においても簡単な拡張により対応することが可能である．

3.2 学習の詳細

RtF は，分類モデルに生成モデル VAE [8] を統合したニューラルネットワークモデルであり，Generative Replay により忘却を防ぐ．すなわち，タスク T_i 学習時に，データセット D_i のみでなく，過去タスク $T_j (1 \leq j < i)$ のデータを表す生成サンプルを同時に用い学習を行う．以下では，RtF のネットワーク構造，学習方法，およびサンプル生成方法について説明する．

まず，RtF のネットワーク構造について説明する．RtF において生成モデルとして用いられる VAE は，エンコー

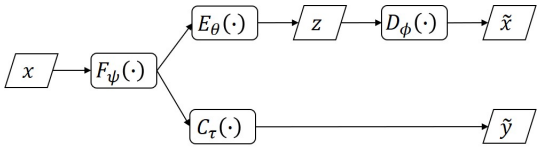

図 1 RtF および提案手法のネットワーク図
Fig. 1 Network structure of RtF and the proposed method

ダとデコーダという 2 種のモジュールから構成される．エンコーダは，入力 $\boldsymbol{x}$ を，標準正規分布 $N(\mathbf{0},\mathbf{I})$ を事前分布とする潜在変数 $\boldsymbol{z}$ に変換する役割を持ち，デコーダは潜在変数 $\boldsymbol{z}$ を元の入力 $\boldsymbol{x}$ を復元する役割を持つ．ここで，RtF ではエンコーダ部の最終隠れ層にクラス分類モジュールを結合することで，分類モデルと VAE の統合を行う．すなわち，RtF のネットワーク構造は，エンコーダとクラス分類モジュールに共通した特徴量抽出モジュール $F_{\boldsymbol{\psi}}(\cdot)$，潜在変数を出力するエンコーダモジュール $E_{\boldsymbol{\theta}}(\cdot)$，デコーダモジュール $D_{\boldsymbol{\phi}}(\cdot)$，クラス分類モジュール $C_{\boldsymbol{\tau}}(\cdot)$ の 4 モジュールにより構成される．また，以下では，特徴量抽出モジュール $F_{\boldsymbol{\psi}}(\cdot)$，クラス分類モジュール $C_{\boldsymbol{\tau}}(\cdot)$ を合わせて主モジュールと称し，特徴量抽出モジュール $F_{\boldsymbol{\psi}}(\cdot)$，エンコーダモジュール $E_{\boldsymbol{\theta}}(\cdot)$，デコーダモジュール $D_{\boldsymbol{\phi}}(\cdot)$ を合わせて VAE モジュールと称する．主モジュールは分類モデルとしての役割，VAE モジュールは生成モデルとしての役割を果たす．なお，$\boldsymbol{\psi}$，$\boldsymbol{\theta}$，$\boldsymbol{\phi}$，$\boldsymbol{\tau}$ は，それぞれパラメータを表す．また，以下でも同様の記法を用い，$\boldsymbol{\Theta}=\{\boldsymbol{\psi},\boldsymbol{\theta},\boldsymbol{\phi},\boldsymbol{\tau}\}$ により，全パラメータを表すこととする．ここで，**図 1** に RtF のネットワーク構造を表す．

次に，RtF における学習手法について説明する．上述したように，RtF は分類モデルに VAE を統合した構造をもつため，訓練に用いる損失も，主モジュールに対する損失と VAE モジュールに対する損失を統合したものとなる．すなわち，タスク T_i に属するデータ $(\boldsymbol{x},y)\in D_i$ に対する損失は，クロスエントロピー誤差 $L_{CL}(\cdot)$ と VAE に対し用いられる損失 $L_{VAE}(\cdot)$ を用い，式 (1) で表される．

$$L_{new}(\boldsymbol{\Theta};\boldsymbol{x},y)=L_{CL}(\boldsymbol{\Theta};\boldsymbol{x},y)+L_{VAE}(\boldsymbol{\Theta};\boldsymbol{x}) \quad (1)$$

また，$L_{VAE}(\cdot)$ には負の evidence lower bound（ELBO）が用いられる．負の ELBO は，VAE の訓練に通常用いられる損失関数であり，式 (2) で表される．

$$L_{VAE}(\boldsymbol{\Theta};\boldsymbol{x})=L_{complex}(\boldsymbol{\Theta};\boldsymbol{x})+L_{recon}(\boldsymbol{\Theta};\boldsymbol{x}) \quad (2)$$

$$L_{complex}(\boldsymbol{\Theta},\boldsymbol{x})=D_{KL}\left(E_{\boldsymbol{\theta}}\left(F_{\boldsymbol{\psi}}\left(\boldsymbol{x}\right)\right)||P(\boldsymbol{z})\right) \quad (3)$$

$$L_{recon}(\boldsymbol{\Theta};\boldsymbol{x})=-\mathbb{E}_{\boldsymbol{z}\sim E_{\boldsymbol{\theta}}(F_{\boldsymbol{\psi}}(\boldsymbol{x}))}\left[l_{recon}\left(D_{\boldsymbol{\phi}}(\boldsymbol{z}),\boldsymbol{x}\right)\right] \quad (4)$$

ここで，式 (2) の第 1 項 $L_{complex}(\cdot)$ は式 (3) により与えられ，エンコーダの出力と潜在変数の事前分布 $P(\boldsymbol{z})$ との KL ダイバージェンスを用いることで，エンコーダの出力を事前分布 $P(\boldsymbol{z})$ に近づかせる役割を持つ．また，式 (4) で与えられる第 2 項 $L_{recon}(\cdot)$ は，VAE の出力を入力 $\boldsymbol{x}$ に近づかせる役割を持つ．なお，$l_{recon}(\cdot)$ にはバイナリクロスエントロピー誤差や二乗誤差などが用いられる．

RtF における忘却回避措置は，Generative Replay により行われる．Generative Replay は，タスク T_i 学習時に，同タスクのデータに対する学習（式 (1)）と同時に，過去タスクのデータを表す生成サンプル $(\hat{\boldsymbol{x}},\hat{y})$ を用いた学習を行うことで，忘却を防ぐ手法である．具体的には，タスク $T_i(i>1)$ に対し，タスク T_i のデータに対する損失 $L_{new}(\cdot)$ と，生成サンプルに対する損失 $L_{old}(\cdot)$ を用いた損失関数 $L_i(\cdot)$ により学習が行われる．ここで，タスク $T_i(i>1)$ に対する損失関数 $L_i(\cdot)$ を式 (5) に示す．

$$L_i(\boldsymbol{\Theta};D_i',\hat{D})=\frac{1}{i}\left(\frac{1}{len(D_i')}\sum_{(\boldsymbol{x},y)\in D_i'}L_{new}(\boldsymbol{\Theta};\boldsymbol{x},y)\right)+\frac{i-1}{i}\left(\frac{1}{len(\hat{D})}\sum_{(\hat{\boldsymbol{x}},\hat{y})\in\hat{D}}L_{old}(\boldsymbol{\Theta};\hat{\boldsymbol{x}},\hat{y})\right) \quad (5)$$

また，1 番目のタスク T_1 学習時は過去タスクが存在しないため，損失関数 $L_1(\cdot)$ は $L_{new}(\cdot)$ によってのみ定義され，式 (6) で表される．

$$L_1(\boldsymbol{\Theta};D_1')=\frac{1}{len(D_1')}\sum_{(\boldsymbol{x},y)\in D_1'}L_{new}(\boldsymbol{\Theta};\boldsymbol{x},y) \quad (6)$$

ここで，D_i' は学習中タスク T_i のデータセット D_i からサンプリングされたミニバッチであり，$\hat{D}$ は生成サンプル集合を表す．なお，L_{new}，L_{old} の係数 $\frac{1}{i}$，$\frac{i-1}{i}$ は，新規タスクおよび過去タスクに属するクラスの不均衡性を緩和するために導入されている．また，生成サンプルに対する損失 $L_{old}(\cdot)$ は，式 (7) に示すように，主モジュールに対する損失として distillation loss [16] を用いる事を除き，$L_{new}(\cdot)$ と同様である．

$$L_{old}(\boldsymbol{\Theta};\hat{\boldsymbol{x}},\hat{y})=L_{DL}(\boldsymbol{\Theta};\hat{\boldsymbol{x}},\hat{y})+L_{VAE}(\boldsymbol{\Theta};\hat{\boldsymbol{x}}) \quad (7)$$

なお，$L_{DL}(\cdot)$ は distilation loss を表し，温度付きソフトマックス関数によって定義される．

最後に，過去タスクのデータを表すサンプル $(\hat{\boldsymbol{x}},\hat{y})$ の生成について説明する．同生成は，過去パラメータ $\hat{\boldsymbol{\Theta}}=\{\hat{\boldsymbol{\psi}},\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\phi}},\hat{\boldsymbol{\tau}}\}$ および潜在変数の事前分布 $P(\boldsymbol{z})$ を用いて行われる．なお，過去パラメータ $\hat{\boldsymbol{\Theta}}$ は，各タスクの学習終了時に保存されるモデルパラメータである．過去タスクのデータは，式 (3) を 1 つの項とする損失関数を用い学習済みであるため，その潜在変数の分布は事前分布 $P(\boldsymbol{z})$ の周辺となる．すなわち，事前分布 $P(\boldsymbol{z})$ に従い潜在変数 $\boldsymbol{z}$ を生成することで，過去タスクに属するデータの潜在変数を生成することが可能となる．また，このように生成された潜在変数 $\boldsymbol{z}$ に対し，デコーダ $D_{\hat{\boldsymbol{\phi}}}(\cdot)$ は対応するデータ $\hat{\boldsymbol{x}}$

Algorithm 1 RtF におけるサンプル生成

Require: $\hat{\boldsymbol{\Theta}} = \{\hat{\boldsymbol{\psi}}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\phi}}, \hat{\boldsymbol{\tau}}\}$ // old model parameters
  $\boldsymbol{z} \sim P(\boldsymbol{z})$ // sample $\boldsymbol{z}$ from the prior distribution $P(\boldsymbol{z})$
  $\hat{\boldsymbol{x}} \leftarrow D_{\hat{\boldsymbol{\phi}}}(\boldsymbol{z})$ // generate $\hat{\boldsymbol{x}}$
  $\hat{y} \leftarrow C_{\hat{\boldsymbol{\tau}}}\left(F_{\hat{\boldsymbol{\psi}}}(\hat{\boldsymbol{x}})\right)$ // generate $\hat{y}$
  return $(\hat{\boldsymbol{x}}, \hat{y})$

Algorithm 2 RtF における学習の流れ

Require: $(D_1, D_2, \cdots, D_M)$ // Dataset sequence
  Initialize model parameters $\boldsymbol{\Theta}$
  for $i = 1$ **to** M **do**
    repeat
      // Sample from the current dataset.
      $D'_i \sim D_i$
      if $i = 1$ **then**
        // Update the parameters based on the loss function L_1(eq. 6) by an optimizer
        $\boldsymbol{\Theta} \leftarrow update(\nabla L_1(\boldsymbol{\Theta}; D'_i))$
      else
        // Generate datas based on the old parameters.（Algorithm 1）
        $\hat{D} \leftarrow generate(\hat{\boldsymbol{\Theta}})$
        // Update the parameters based on the loss function L_i（eq. 5） by an optimizer.
        $\boldsymbol{\Theta} \leftarrow update(\nabla L_i(\boldsymbol{\Theta}; D'_i, \hat{D}))$
      end if
    until the parameters converge
    // Save the current parameters as old parameters.
    $\hat{\boldsymbol{\Theta}} \leftarrow \boldsymbol{\Theta}$
  end for
  return $\boldsymbol{\Theta}$ // model parameters

を復元するよう訓練が行われているため，$\hat{\boldsymbol{x}}$ は過去タスクのデータを表していると考えられる．生成された $\hat{\boldsymbol{x}}$ に対応する教師データ $\hat{y}$ の生成についても，過去パラメータ $\hat{\boldsymbol{\Theta}}$ を用い行われる．ここで，$\hat{\boldsymbol{x}}$ が過去タスクのデータを表しているとすると，過去パラメータ $\hat{\boldsymbol{\Theta}}$ は同データについて訓練済みであるため，主モジュールの出力 $C_{\hat{\boldsymbol{\tau}}}\left(F_{\hat{\boldsymbol{\psi}}}(\hat{\boldsymbol{x}})\right)$ を教師データ $\hat{y}$ とみなすことが可能である．上記で示した方法を複数回行うことで，訓練に用いるサンプル集合 $\hat{D}$ を生成する．ここで，Algorithm 1 にサンプル生成の手順を示す．また，RtF の学習全体の流れを Algorithm 2 にまとめる．

3.3 問題点

本節では，RtF に存在する 2 つの問題点について述べる．1 つ目は，VAE モジュールに生じる忘却である．生成サンプルに対する損失 $L_{old}(\cdot)$（式 (7)）の第二項より，VAE モジュールに対する忘却阻止が主に $L_{VAE}(\boldsymbol{\Theta}, \hat{\boldsymbol{x}})$ により行われていることがわかる．すなわち，生成サンプルを実サンプルと同様に学習することで忘却の回避を行っているが，この方法では忘却の回避が困難であることが報告されている [17]．2 つ目は，生成サンプルの偏りである．RtF において，過去タスクを表すサンプルの生成は，潜在変数 $\boldsymbol{z}$ を事前分布 $P(\boldsymbol{z})$ からサンプリングすることで行われる．しかしながら同サンプル生成方法では，事前分布 $P(\boldsymbol{z})$ から離れた潜在変数分布を持つようなデータがサンプリングされず，生成サンプルに偏りが生じるという問題を持つ [13]．

4. 提案手法

本章では，提案手法の詳細について説明する．提案手法では，3.3 節で言及した RtF の問題点を，損失関数およびサンプル生成方法を改良することで解決し，継続学習の安定性向上を試みる．損失関数の改良においては，式 (7) の第二項で表される，VAE モジュールに対する忘却阻止項に注目し，同項を改善する．また，サンプル生成方法においては，クラス c に対する条件付き分布 $p(\boldsymbol{z}|c)$ を用いた潜在変数 $\boldsymbol{z}$ のサンプリングを行うことで偏りの緩和を試みる．なお，問題設定として 3.1 節で述べた問題と同様なものを考え，ネットワーク構造についても RtF と同様なもの（図 1）を仮定し，記法についても特に断りのない限り，3 章と同様の記法を用いる．以下では，4.1 節にて損失関数の改善方法，4.2 節にてサンプル生成の改善方法を述べる．

4.1 損失関数の改善

本節では，提案手法における損失関数，特に式 (7) の第二項で表される，VAE モジュールに対する忘却阻止項の改善方法について述べる．なお，特に記載のない限り，$\hat{\boldsymbol{x}} \in \hat{\mathbf{X}}$ を過去パラメータより生成された生成サンプルとし，$\hat{\boldsymbol{\Theta}} = \{\hat{\boldsymbol{\psi}}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\phi}}, \hat{\boldsymbol{\tau}}\}$ を，過去タスクの学習終了時に保持した過去パラメータとする．

RtF における VAE モジュールに対する忘却阻止項である $L_{VAE}(\cdot)$ は，式 (2) に示すように $L_{complex}(\cdot)$ と $L_{recon}(\cdot)$ の 2 項に分解することができる．そこでまず，式 (3) により表される $L_{complex}(\cdot)$ について考える．同項は，生成モデルの潜在変数 $\boldsymbol{z}$ の複雑性を制御する項であり，エンコーダ部の出力と事前分布 $P(\boldsymbol{z})$ との KL ダイバージェンスを用い，潜在変数 $\boldsymbol{z}$ の分布を可能な限り事前分布 $P(\boldsymbol{z})$ に近づかせる役割を持つ．一方，生成サンプル $\hat{\boldsymbol{x}}$ に対し，過去パラメータを使用したエンコーダの出力 $E_{\hat{\boldsymbol{\theta}}}(F_{\hat{\boldsymbol{\psi}}}(\hat{\boldsymbol{x}}))$ は，$\hat{\boldsymbol{x}}$ を表現するために，少なくとも必要な複雑性をもつと解釈できる．このことは，同サンプルに対する事前分布 $P(\boldsymbol{z})$ を用いた複雑性の制御が，過去パラメータを使用したエンコーダの出力 $E_{\hat{\boldsymbol{\theta}}}(F_{\hat{\boldsymbol{\psi}}}(\hat{\boldsymbol{x}}))$ による制御と比較したとき，潜在変数の複雑性に対する過度な制約であることを示唆する．そこで，過去パラメータを使用したエンコーダの出力により，生成サンプルに対する潜在変数の複雑性制御を行う．すなわち，式 (8) により同制御を行う．

$$D_{KL}\left(E_{\boldsymbol{\theta}}(F_{\boldsymbol{\psi}}(\hat{\boldsymbol{x}}))||E_{\hat{\boldsymbol{\theta}}}(F_{\hat{\boldsymbol{\psi}}}(\hat{\boldsymbol{x}}))\right) \tag{8}$$

次に，式 (4) で表される $L_{recon}(\cdot)$ について考える．ここでは，新規タスクの学習開始時点に注目する．この時，忘却阻止項の値は 0 であることが望ましい．なぜなら，現

在のパラメータと保持されている過去パラメータは同一であるため，忘却が全く生じていないと解釈できるからである．しかしながら，この時，生成サンプル $\hat{\boldsymbol{x}}$ に対する損失 $L_{recon}(\boldsymbol{\Theta};\hat{\boldsymbol{x}})$ は 0 とならず，忘却阻止項が忘却の程度に対して滑らかな関数ではないことを意味する．そこで，式 (9) を用いることで，忘却が生じていない時点において，忘却の程度に対し滑らかな関数を実現する．実際，現在のパラメータと過去パラメータが同一の場合，式 (9) は 0 となる．

$$-\mathbb{E}_{\boldsymbol{z}\sim E_{\boldsymbol{\theta}}(F_{\boldsymbol{\psi}}(\hat{\boldsymbol{x}}))}\left[l_{recon}\left(D_{\boldsymbol{\phi}}(\boldsymbol{z}),D_{\hat{\boldsymbol{\phi}}}(\boldsymbol{z})\right)\right] \tag{9}$$

以上より，提案手法における VAE モジュールに関する忘却阻止項は，式 (10) によって与えられる．

$$\begin{aligned}L_{old\_VAE}(\boldsymbol{\Theta};\hat{\boldsymbol{x}}) = D_{KL}\left(E_{\boldsymbol{\theta}}(F_{\boldsymbol{\psi}}(\hat{\boldsymbol{x}}))||E_{\hat{\boldsymbol{\theta}}}(F_{\hat{\boldsymbol{\psi}}}(\hat{\boldsymbol{x}}))\right)\\ -\mathbb{E}_{\boldsymbol{z}\sim E_{\boldsymbol{\theta}}(F_{\boldsymbol{\psi}}(\hat{\boldsymbol{x}}))}\left[l_{recon}\left(D_{\boldsymbol{\phi}}(\boldsymbol{z}),D_{\hat{\boldsymbol{\phi}}}(\boldsymbol{z})\right)\right]\end{aligned} \tag{10}$$

なお，提案手法における損失関数全体は，式 (7) の第二項を式 (10) に置換したものとして与えられ，その他の変更は行わない．

4.2 サンプル生成方法の改善

本節では，提案手法におけるサンプル生成方法について述べる．提案手法では，クラス c に対する条件付き分布 $p(\boldsymbol{z}|c)$ を推定し，同分布を用い潜在変数をサンプリングすることで，偏りの少ないサンプル生成を実現する．具体的には，RtF のように事前分布 $P(\boldsymbol{z})$ を用いて潜在変数 $\boldsymbol{z}$ をサンプリングするのではなく，クラス c に対する潜在変数の条件付き分布 $p(\boldsymbol{z}|c)$ とクラス c の事前分布 $P(c)$ の積 $p(\boldsymbol{z}|c)P(c)$ を用いて潜在変数 $\boldsymbol{z}$ をサンプリングする．ここで，クラス c の事前分布 $P(c)$ として離散一様分布を用いる．なお，サンプル $(\hat{\boldsymbol{x}},\hat{y})$ の生成は，上記の方法でサンプリングした潜在変数 $\boldsymbol{z}$ を用い，RtF と同様の方法で行う．

また，条件付き分布 $p(\boldsymbol{z}|c)$ の推定を行うに際し，同分布に対し式 (11)-(13) のような仮定を置く．すなわち，クラス c によって条件付けられた潜在変数 $\boldsymbol{z}$ が正規分布に従い，平均値パラメータ $\boldsymbol{\mu}_c$ が正規分布，精度パラメータ $\boldsymbol{\beta}_c$ がガンマ分布に従うと仮定する．

$$\boldsymbol{z}|c \sim N(\boldsymbol{\mu}_c,\boldsymbol{\beta}_c^{-1}) \tag{11}$$

$$\boldsymbol{\mu}_c \sim N(\boldsymbol{m}_c,\boldsymbol{s}_c^2) \tag{12}$$

$$\boldsymbol{\beta}_c \sim Gamma(\boldsymbol{a}_c,\boldsymbol{b}_c) \tag{13}$$

分布の推定は，タスク T_i の学習終了時に，エンコーダの出力を用い行う．具体的には，平均値パラメータのハイパーパラメータ $\boldsymbol{m}_c$，$\boldsymbol{s}_c$ を最尤推定，精度パラメータのハイパーパラメータ $\boldsymbol{a}_c$，$\boldsymbol{b}_c$ をモーメント法により推定する．この時，任意のクラス $c \in C_i$ に対し，クラス c に所属するデータ $\mathbf{X}^c$ が存在する．そこで，同データセットを用い，式 (14)-(17) に従い条件付き分布 $p(\boldsymbol{z}|c)$ の推定を行う．

$$\boldsymbol{m}_c = \frac{1}{N_c}\sum_{\boldsymbol{x}\in\mathbf{X}^c} E^{\boldsymbol{\mu}}_{\boldsymbol{\theta}}(F_{\boldsymbol{\psi}}(\boldsymbol{x})) \tag{14}$$

$$\boldsymbol{s}_c^2 = \frac{1}{N_c}\sum_{\boldsymbol{x}\in\mathbf{X}^c} (E^{\boldsymbol{\mu}}_{\boldsymbol{\theta}}(F_{\boldsymbol{\psi}}(\boldsymbol{x})) - \boldsymbol{m}_c)^2 \tag{15}$$

$$\boldsymbol{a}_c = \frac{\boldsymbol{\mu}_\beta^2}{\boldsymbol{\sigma}_\beta^2} \tag{16}$$

$$\boldsymbol{b}_c = \frac{\boldsymbol{\mu}_\beta}{\boldsymbol{\sigma}_\beta^2} \tag{17}$$

$$\boldsymbol{\mu}_\beta = \frac{1}{N_c}\sum_{\boldsymbol{x}\in\mathbf{X}^c} E^{\boldsymbol{\beta}}_{\boldsymbol{\theta}}(F_{\boldsymbol{\psi}}(\boldsymbol{x})) \tag{18}$$

$$\boldsymbol{\sigma}_\beta^2 = \frac{1}{N_c}\sum_{\boldsymbol{x}\in\mathbf{X}^c} (E^{\boldsymbol{\beta}}_{\boldsymbol{\theta}}(F_{\boldsymbol{\psi}}(\boldsymbol{x})) - \boldsymbol{\mu}_\beta)^2 \tag{19}$$

ここで，N_c はデータ $\mathbf{X}^c$ に属するデータ数，$E^{\boldsymbol{\mu}}_{\boldsymbol{\theta}}(\cdot)$，$E^{\boldsymbol{\beta}}_{\boldsymbol{\theta}}(\cdot)$ はそれぞれ，エンコーダの最終層のうち，平均値パラメータ，精度パラメータを出力する部分を表す．

また，タスク T_i を含む訓練を行うことにより，過去タスク $T_j(1 \le j \le i-1)$ に所属するデータセットの潜在分布が，条件付き分布を推定した時点から変化する．そのため，任意のクラス $\hat{c} \in C_j$ に対し，すでに推定した条件付き分布 $p(\boldsymbol{z}|\hat{c})$ を更新する必要がある．しかしながら，クラス $\hat{c}$ に属するデータは存在しないため，生成データを用い条件付き分布の更新を行う．データの生成は，推定済みの条件付き分布 $p(\boldsymbol{z}|\hat{c})$ を用い潜在変数 $\boldsymbol{z}$ をサンプリングし，同潜在変数 $\boldsymbol{z}$ を使用し RtF におけるデータ生成と同様の方法で行う．同手順によって生成されたデータ $\mathbf{X}^{\hat{c}}$ による条件付き分布 $p(\boldsymbol{z}|\hat{c})$ の更新は，式 (14)-(17) により行う．すなわち，更新前の条件付き分布を事前分布とせず，生成データ $\mathbf{X}^{\hat{c}}$ のみに基づいて更新する．これは，タスク T_i の学習により，潜在分布が変化し得るため，更新前の条件付き分布は事前分布となり得ないと考えられるためである．

5. 評価実験

本研究では，ベンチマークデータを用い，生成モデル，分類モデルの双方に対する提案手法の評価を行った．

5.1 データセット

本研究では，Split MNIST [3] と Split Fashion MNIST [13] の 2 つのデータセットを用いた．以下に，それぞれの概要を示す．

- Split MNIST : MNIST [18] の各クラス，またはクラス集合を 1 タスクと解釈する．MNIST は，手書き数字画像データセットであり，0∼9 の各数字のグレースケール画像により構成される．また，各数字がクラスに対応した 10 クラスのデータセットであり，各クラスごとに約 6,000 枚の学習用データと約 1,000 枚の評価用データを含む．

- Split Fashion MNIST．Split MNIST と同様，Fashion MNIST [19] の各クラス，またはクラス集合を 1 タスクと解釈する．Fashion MNIST は，10 クラスからなる衣料品グレースケール画像データセットであり，各クラスはズボン，ドレスといった衣料品カテゴリである．また，各クラスごとに 6,000 毎の学習用データと 1,000 枚の評価用データを含む．

なお，生成モデルに対する実験では，各クラスを 1 タスクと解釈した 10 タスクの問題，分類モデルに対する実験では 2 クラスを 1 タスクと解釈した 5 タスクの問題を扱った．ここで，クラスの提示順についてはラベル番号の若い順とし，分類モデルに対する実験ではクラス集合を $\{0,1\},\{2,3\},\cdots,\{8,9\}$ の順に提示した．また，評価時に所属タスクに関する情報を与えない，class-incremental learning [9] にて評価を行った．

5.2 比較手法

提案手法に対し，以下に示す手法を用い比較を行った．

- superior：全タスクのデータを用い訓練を行う手法．評価値の上限値を与えるものとして解釈できる．
- EWC [1]：正則化により破滅的忘却を防ぐ手法．
- LGM [13]：VAE に対し，離散潜在分布を導入することで忘却を防ぐ手法．
- GR：生成モデルに対し，Generative Replay を行うことで忘却を防ぐ手法．
- DGR-GAN [6]：生成モデルとして WGAN を用い，Generative Replay により忘却を防ぐ手法．
- DGR-VAE [7]：生成モデルとして VAE を用い，Generative Replay により忘却を防ぐ手法．
- RtF [7]：分類モデルに VAE を統合したモデルを用いた手法．

なお，LGM および GR は生成モデルに対する継続学習手法であるため，生成モデルに対する実験のみ行った．また，EWC についても，分類モデルの class-incremental learning では忘却を防ぐことができないことが報告されている [9] ため，生成モデルに対する実験のみ行った．

5.3 実験設定

公正な比較を行うため，生成モデルに対する実験では，すべて同一のネットワークを用いた．すなわち，VAE を実験対象の生成モデルとし，エンコーダ，デコーダの中間層がそれぞれ 1 層の MLP を使用した．なお，中間層のユニット数は 400，潜在次元数は 14 である．また，$l_{recon}(\cdot)$ については，バイナリクロスエントロピー誤差を用い，最適化関数として Adam [20] を用い各タスクごとに 1,000 回の学習を行った．なお，Adam のハイパーパラメータはデフォルトパラメータを用いた．また，バッチサイズについては，128 を使用し，Generative Replay を用いる手法については同数の生成サンプルを使用した．なお，これらの実験設定は，文献 [13] を参考にした．

分類モデルに対する実験では，以下のように，分類に用いるネットワーク構造が全て同様となるようなネットワークを用いた．

- superior：中間層が 1 層，ユニット数が 400 の MLP を用いた．
- DGR-GAN：分類器は superior と同様の構造をとる．WGAN については，critic，generator ともに中間層 1 層の MLP を用いた．ここで，潜在次元数は 100 とした．
- DGR-VAE：VAE として，エンコーダ，デコーダともに中間層 1 層の MLP を用いた．また，分類器の構造及び潜在次元数は DGR-GAN と同様である．
- RtF：DGR-VAE で使用する VAE のエンコーダ最終層に，クラス分類層を追加した構造を用いた．

また，class-incemental learning を対象とするため，全タスクに対しすべて同じ出力層を使用する single-headed な構造 [21] を用いた．その他の実験設定については，各タスクごとに 2,000 回の学習を行ったことを除き，生成モデルに対する実験と同様である．また，DGR-GAN，DGR-VAE については，生成モデルに対しても同様に各タスクごとに 2,000 回の学習を行った．なお，これらの実験設定は，文献 [7] を参考にした．

5.4 評価方法

生成モデル，分類モデルともに，全タスク終了時点における，各タスクに対する評価指標の平均値により評価を行った．なお，生成モデルに対しては ELBO，分類モデルに対しては Accuracy を評価指標として用いた．両評価指標ともに，高い値がより良い精度であることを示す．ここで，式 (20) に ELBO の定義を示す．

$$\mathrm{ELBO} = \frac{1}{len(\mathbf{X})} \sum_{\boldsymbol{x}\in\mathbf{X}} \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})] - D_{KL}\left(q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})||P(\boldsymbol{z})\right) \tag{20}$$

なお，$\mathbf{X}$ が評価用データ，$q_{\boldsymbol{\theta}}(\cdot)$ がエンコーダ，$p_{\boldsymbol{\phi}}(\cdot)$ がデコーダを示す．なお，Accuracy は，予測結果のうち正解ラベルと同一の予測を行った割合により定義する．また，別シードを用い，生成モデルに対しては 5 回，分類モデルに対しては 10 回実験を行い，これらの結果の平均を評価に用いた．

5.5 実験結果と考察

まず，生成モデルに対する結果を**表 1** 及び**図 2** に示す．なお，紙面の都合上 Split Fashion MNIST に対する結果は省略し，Split MNIST に対する結果のみを示す．ここで，表 1 は評価結果，図 2 は各タスクに対する評価値の推移を

Vol.2019-MPS-123 No.46
Vol.2019-BIO-58 No.46
2019/6/19

表 1　MNIST に対する各生成モデルの ELBO（5 回平均）
Table 1　ELBO of Generative Models on MNIST

method	ELBO
superior	-93.37
EWC	-140.26
LGM	-167.67
GR	-111.94
Proposed	**-104.66**

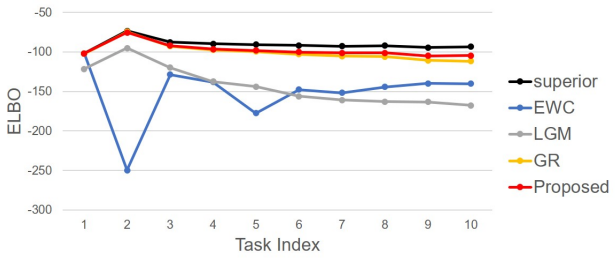

図 2　各生成モデルに対する ELBO の推移（MNIST）
Fig. 2　The Transition of ELBO of Generative Models for a Task Sequence on MNIST(average over 5 times)

示す．表 1 より，提案手法が他の比較手法よりも，良い精度を示していることがわかる．また，図 2 より，5 タスク目までは GR と提案手法の差がほぼ生じないものの，タスクが増加するにつれ徐々にその差が増大していることがわかる．ここで，GR と提案手法の差が徐々に大きくなっていく理由としては，5 タスク目までは GR が生成するサンプルが十分に過去タスクのデータを表しているのに対し，6 タスク目以降では生成サンプルが過去タスクのデータと乖離し，忘却が大きく進んでいくためと考えられる．一方，提案手法では，損失関数の改善による忘却の緩和および条件付き潜在分布を用いたサンプル生成により，生成サンプルと過去タスクのデータとの差を微小に保つことで，タスク数の増加に伴う忘却を緩和していると考えられる．また，Split Fashion MNIST においても同様の結果となることを確認した．

次に，分類モデルに対する結果を**表 2**，**表 3**，**図 3**，**図 4** に示す．ここで，表 2，表 3 はそれぞれ，MNIST，Fashion MNIST に対する評価結果を示す．また，図 3，図 4 はそれぞれ，MNIST，Fashion MNIST に対する評価値の推移を示す．表 2，表 3 より，両データセットにおいて，提案手法が他の比較手法よりも，良い精度を示していることがわかる．また，図 3 より，MNIST を用いた実験では，4 タスク目までは提案手法と RtF に大きな差は生じていないものの，5 タスク目で大きく差が生じていることがわかる．上記のような精度の差の発生は，生成モデルに対する結果と同様，生成サンプルと過去タスクのデータとの差に起因するものであると考えられる．一方，図 4 より，Fashion MNIST を用いた実験

表 2　MNIST に対する各分類モデルの Accuracy（10 回平均）
Table 2　Accuracy of Classification Models on MNIST

method	Accuracy
superior	98.12
DGR-GAN	90.87
DGR-VAE	91.07
RtF	92.20
Proposed	**94.88**

表 3　Fashion MNIST に対する各分類モデルの Accuracy（10 回平均）
Table 3　Accuracy of Classification Models on Fashion MNIST

method	Accuracy
superior	89.27
DGR-GAN	69.78
DGR-VAE	73.36
RtF	70.02
Proposed	**77.72**

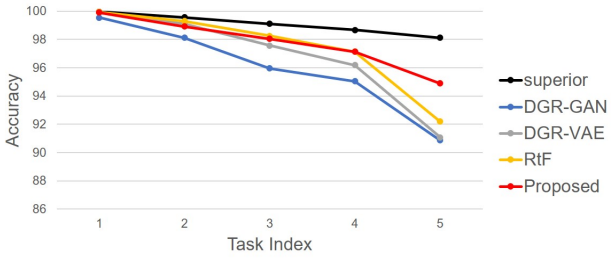

図 3　各分類モデルに対する結果の推移（MNIST）
Fig. 3　The Transition of Accuracy of Classification Models for a Task Sequence on MNIST(average over 10 times)

において，タスク数が少ない段階から提案手法と他の手法とで差が生じた理由としては，Fashion MNIST が MNIST よりも複雑な画像を対象としていることから，実データを良く表したサンプルの生成が困難であることが起因すると考えられる．比較手法は，サンプル生成をクラスによらずランダムに行っている事に対し，提案手法はクラスに依存したサンプル生成により同クラスをより良く近似したサンプル生成を行っている．提案手法では，この生成サンプルの尤もらしさにより，より複雑なデータセットに対する忘却の更なる緩和を可能にしていると考えられる．ここで，他手法と比較した場合における，提案手法の統計的有意性を示すため，両データセットの Accuracy に対する検定を行った．この検定では，RtF，DGR-VAE をそれぞれ MNIST，Fashion MNIST に対する比較対象とした．また，両結果ともに，Shapiro-Wilk 検定及び F 検定の結果から，等分散でない正規分布であると仮定できたため，Welch の t 検定により検定を行った．同検定による p 値は，MNIST，Fashion-MNIST に対しそれぞれ，$3.28e^{-8}$，$2.55e^{-6}$ となった．これは，提案手法による精度向上が両データセットに対し統計的に有意であることを示している．

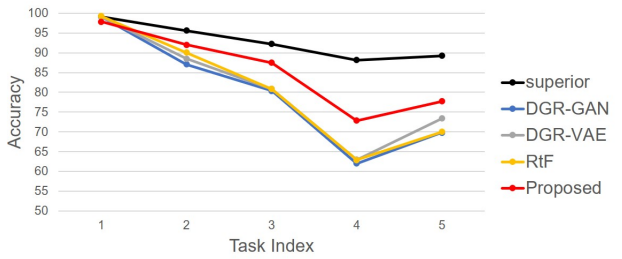

図 4　各分類モデルに対する結果の推移（Fashion MNIST）

Fig. 4　The Transition of Accuracy of Classification Models for a Task Sequence on Fashion MNIST(average over 10 times)

6. おわりに

本論文では，継続学習に対する 1 手法である Generative Replay において生じる，生成モデルの忘却と生成サンプルの偏りという問題に対し，損失関数を改良することで生成モデルに生じる忘却を緩和し，潜在変数に対するクラス条件付き分布を用いサンプルを生成することで，偏りの少ないサンプルを生成する手法を提案した．また，ベンチマークデータを用いた評価実験を通して，提案手法の精度が比較手法よりも有意に優れていることを示した．しかしながら，学習が進行した時点で精度の上限値との差が大きくなる場合が依然としてあり，継続学習の実現にはより洗練された手法が望まれる．

参考文献

[1] Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A. et al.: Overcoming catastrophic forgetting in neural networks, *Proceedings of the national academy of sciences*, Vol. 114, No. 13, pp. 3521–3526 (2017).

[2] McCloskey, M. and Cohen, N. J.: Catastrophic interference in connectionist networks: The sequential learning problem, *Psychology of learning and motivation*, Vol. 24, Elsevier, pp. 109–165 (1989).

[3] Zenke, F., Poole, B. and Ganguli, S.: Continual learning through synaptic intelligence, *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, JMLR. org, pp. 3987–3995 (2017).

[4] Rebuffi, S.-A., Kolesnikov, A., Sperl, G. and Lampert, C. H.: icarl: Incremental classifier and representation learning, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2001–2010 (2017).

[5] Lopez-Paz, D. et al.: Gradient episodic memory for continual learning, *Advances in Neural Information Processing Systems*, pp. 6467–6476 (2017).

[6] Shin, H., Lee, J. K., Kim, J. and Kim, J.: Continual learning with deep generative replay, *Advances in Neural Information Processing Systems*, pp. 2990–2999 (2017).

[7] van de Ven, G. M. and Tolias, A. S.: Generative replay with feedback connections as a general strategy for continual learning, *arXiv preprint arXiv:1809.10635* (2018).

[8] Kingma, D. P. and Welling, M.: Auto-encoding variational bayes, *arXiv preprint arXiv:1312.6114* (2013).

[9] van de Ven, G. M. and Tolias, A. S.: Three scenarios for continual learning, *arXiv preprint arXiv:1904.07734* (2019).

[10] Vitter, J. S.: Random sampling with a reservoir, *ACM Transactions on Mathematical Software (TOMS)*, Vol. 11, No. 1, pp. 37–57 (1985).

[11] Arjovsky, M., Chintala, S. and Bottou, L.: Wasserstein gan, *arXiv preprint arXiv:1701.07875* (2017).

[12] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V. and Courville, A. C.: Improved training of wasserstein gans, *Advances in Neural Information Processing Systems*, pp. 5767–5777 (2017).

[13] Ramapuram, J., Gregorova, M. and Kalousis, A.: Lifelong generative modeling, *arXiv preprint arXiv:1705.09847* (2017).

[14] Achille, A., Eccles, T., Matthey, L., Burgess, C., Watters, N., Lerchner, A. and Higgins, I.: Life-long disentangled representation learning with cross-domain latent homologies, *Advances in Neural Information Processing Systems*, pp. 9873–9883 (2018).

[15] Bengio, Y., Courville, A. and Vincent, P.: Representation learning: A review and new perspectives, *IEEE transactions on pattern analysis and machine intelligence*, Vol. 35, No. 8, pp. 1798–1828 (2013).

[16] Hinton, G., Vinyals, O. and Dean, J.: Distilling the knowledge in a neural network, *arXiv preprint arXiv:1503.02531* (2015).

[17] Lesort, T., Caselles-Dupré, H., Garcia-Ortiz, M., Stoian, A. and Filliat, D.: Generative Models from the perspective of Continual Learning, *arXiv preprint arXiv:1812.09111* (2018).

[18] LeCun, Y., Cortes, C. and Burges, C. J.: The MNIST database of handwritten digits, 1998, *URL http://yann. lecun. com/exdb/mnist*, Vol. 10, p. 34 (1998).

[19] Xiao, H., Rasul, K. and Vollgraf, R.: Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms (2017).

[20] Kingma, D. P. and Ba, J.: Adam: A method for stochastic optimization, *arXiv preprint arXiv:1412.6980* (2014).

[21] Farquhar, S. and Gal, Y.: Towards robust evaluations of continual learning, *arXiv preprint arXiv:1805.09733* (2018).