Generative Adversarial Nets を用いた文書分類の検証

小島　智樹† 　酒井　哲也†

† 早稲田大学基幹理工学部情報理工学科　〒 169–8555 東京都新宿区大久保 3–4–1
E-mail: †frkojima512@ruri.waseda.jp, ††tetsuyasakai@acm.org

あらまし　近年，Generative Adversarial Nets (GAN) は画像生成や画像分類，文書生成などにおいて多大な成果を挙げている．特に，画像分類においては，少数の教師データで高い正解率を達成している．本論文では，2 種類の方法を用いて，GAN による分類を文書データに用いることを試みた．1 つ目の方法は，まず word2vec を用いて文書の各単語をベクトル化する．次に，それを結合して画像のような形式のデータにする．これを Convolutional Neural Network に入力し，通常のクラス分類学習を行う．このような従来の方法に加えて，GAN によって生成された偽データを使い，データが実データか偽データかを判別させる真偽学習を行った．2 つ目の方法は，Word2vec を用いて文書の各単語をベクトル化した後，その平均をとることで各文書をベクトル表現とした．そのデータを用いて Neural Net に通常のクラス分類学習をさせた．更に，GAN によって生成された偽データを使うことで，データが実データか偽データかを判別させる真偽学習を行った．本論文では，これら 2 つの方法の評価実験を通して，GAN の文書分類への応用可能性について検討する．

キーワード　GAN, CNN, 自然言語処理, 文書分類, NLP

1. 導　　入

Generative Adversarial Nets (GAN) は，Goodfellow ら [2] が考案した，機械学習のフレームワークである．生成のためのネットワークである Generator と分類のためのネットワークである Discriminator から成り，互いに対立的に学習させ，より精巧な画像を生成させる．このフレームワークは，元は画像の生成のためのものであったが，後に画像の分類，及び文生成などにおいて大いなる成果を挙げている．

この中で，画像の分類に関しては Odena らによる研究 [4] や Goodfellow らにおける研究 [11] がある．これらの研究によれば，少数の教師データの場合において GAN を用いることで，従来のモデルよりも高い精度で分類ができることが確認されている．このような少数の教師データによる高い精度の分類が文書にも応用可能であれば，専門性の高い文書など，教師データを多数用意することが難しい文書の分類に応用でき，有用であると考えられる．

本論文においては，2 種類の文書分類のモデルをもとに，GAN の文書分類への有用性を検証した．

いずれのモデルにおいても，文書データの各単語を Word2vec を利用してベクトル化した．Word2vec は Mikolov らが考案した，単語をベクトル表現に変換する方法である [16]．この方法では，Continuous Bag Of Words (CBOW) や Skip-gram などの，文中のある単語，及びその周辺単語を用いて単語の埋め込み表現を獲得するアルゴリズムを利用する．これによって，従来の bag of words (bow) などと違い，単語の意味を保持したままのベクトル表現を得ることが可能となる．

1 つ目の提案手法は，Convolutional Neural Network (CNN) を Discriminator 部分に用い，文書を画像のように扱ったモデルである．今回，文書データを画像のように扱うために，文書データの各単語を Word2vec を用いてベクトル表現とした．このベクトル表現を複数組み合わせ，文書を画像のように扱う．Discriminator の部分には，Kim らが考案した CNN と Word2vec を利用したモデル [3] を参考にしたものを使用する．このモデルは，文書の各単語を一つ一つ見ていくのではなく，幾つかの単語をまとめて見ることで，より文書の意味を正確に把握するモデルである．

2 つ目は，通常の Neural Network (NN) を Discriminator 部分に用いる方法である．文書の各単語を Word2vec 表現に直し，その平均をとり文書をベクトル化した．この方法を用いることで，データの次元が大きくなりすぎるという 1 つ目の手法の弱点が克服できる．

この二つのモデルについて，GAN の要素を加えることによって識別性能が向上するか検証することが本論文の目的である．

なお，従来研究において，GAN による文書の分類が有用か検証した例は，筆者が調査した限り見つけられなかった．

2. 従 来 研 究

2.1 GAN

GAN は，近年注目されている機械学習のフレームワークである．複数の乱数を入力とし，画像を出力するためのネットワークである Generator (G) と，G の作り出した画像と実際の画像を入力として，それが本物かどうかを判別するネットワークである Dicriminator (D) を，対立的に学習させることで，本物と見分けがつかないような画像を G に生成させることが目的である．数式で表すと

$$\min_G \max_D \mathbf{V}(D, G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \quad (1)$$

これが GAN で使われる損失関数である．この式で使われている文字について補足しておくと

- z は乱数を表す．
- $D(x)$ はあるデータ x を D に入力したときに，本物のデータであると識別する確率を表す．
- $G(z)$ はある乱数 z を入力したときに生成されるデータを表す．

2.2 Convolutional Neural Network

CNN は畳み込み層とプーリング層と呼ばれる，特殊なレイヤーを導入した Neural Net の一種である．データの一定のサイズの領域を，一つの特徴に圧縮することで，ある程度の大きさの領域を一つのまとまった特徴として捉えることができる．その結果，よりロバストにデータの特徴を捉えることが可能となる．

2.3 機械学習による文書分類

機械学習による文書分類には様々な手法があり，例としては以下のような手法がある．

- 文書を bag of words 表現に直し，Support Vector Machine (SVM) を用いて分類を行う [12].
- 文書の各単語を word2vec 表現に直し，それを統合したものを CNN を用いて分類を行う [3].
- Paragraph Vector や，文書の各単語の word2vec 表現の平均を用いて，文書を一つのベクトルとし NN で分類を行う [14].

今回は，上記の 2 番目と 3 番目の手法を元にし，GAN を応用した提案手法を作成した．提案手法については後述する．

2.4 GAN を用いた画像分類

GAN を用いた画像分類を扱ったモデルとして，Odena の SGAN [4] や Salimans らの方法 [11] が挙げられる．一般の画像分類のための学習は以下のような手順で行われる．

（ 1） 画像データを分類器への入力とする．
（ 2） そのデータが各クラスに属するかの確率を出力とする．
（ 3） 出力と正解とを用い，なんらかの損失関数 (e.g. binary cross entropy) で損失を計算する．
（ 4） 損失を誤差逆伝播し重みを更新する．

SGAN では，通常の画像の分類の損失に加えて，Generator が生成した画像と，実際の画像の分類を分類器に行わせ，真偽学習も行わせると言った方法が取られている．

また， Salimans ら [11] は， 分類器の構造を変えずに損失関数を工夫することにより， 通常の分類器に真偽学習も行わせることで学習量を増やし， 精度を向上させている．

上記の SGAN を用いた方法，及び Salimans らの方法による MNIST[注1] の分類は，用いる教師データが少数のときに従来の手法に比べて高い分類正解率を示している．

2.5 Word2vec

Word2vec は Mikolov らが開発した単語をベクトル化するための手法である [16]．この手法の特徴として以下が挙げられる．

- ある単語に対して，意味が近い単語と近くない単語があったとき，近い単語の方がベクトル空間上の距離が近くなる (e.g. programming という単語に対して computer と pen という単語があったときベクトル空間上で computer の方が近くなる).
- 単語同士の加算, 減算ができる (e.g. king−man=queen−woman といった計算ができる).
- 比較的小さい次元で単語が表現できる

(e.g. bow では語彙が 100 万語であったら，100 万次元のベクトルが必要である).

この手法を実現するための方法として，CBOW と Skip-gram が挙げられる．CBOW は，ある単語の周辺単語が与えられたとき，ある単語が，なんであるかについての確率を求める NN を用いて，単語の埋め込み表現を獲得する方法である．例として play と well と言う周辺単語を与えられたときを考える．この図の場合

$$play = (w_{p1}, w_{p2}) \quad (2)$$

の形で埋め込み表現が表される．各 w は文脈として自然な eat apple の確率は高く，eat cola という不自然な文脈の確率を低くするように決定される．各確率を数式で表すと（注2）

$$P(play\ tennis\ well) = softmax((w_{p1}+w_{w1})w_{t1} + (w_{p2}+w_{w2})w_{t2}) \quad (3)$$

$$P(play\ apple\ well) = softmax((w_{p1}+w_{w1})w_{a1} + (w_{p2}+w_{w2})w_{a2}) \quad (4)$$

となる．上の自然な文脈の場合に対応する式の値は大きく，下の不自然な文脈の場合に対応する式の値を小さくするように，各 w を学習していく．ここで w_{w1}と w_{w2}は *well* の，w_{t1}と w_{t2} は *tennis* の，w_{p1}と w_{p2}は *play* の埋め込み表現の要素である．

一方，Skip-gram はある単語を入力として，周辺単語を出力とする NN の重みを計算することによって，単語の埋め込み表現を獲得する方法である．例として，図のように，eat という単語の埋め込み表現を計算することを考える．この図の場合

$$eat = (w_{e1}, w_{e2}) \quad (5)$$

の形で埋め込み表現が表される．各 w は文脈として自然な eat apple の確率は高く，eat cola という不自然な文脈の確率を低くするような w であることが望ましい．ここで w_{a1}と w_{a2} は *apple* の w_{c1}と w_{c2}は *cola* の埋め込み表現の要素である．各確率を数式で表すと

$$P(eat\ apple) = softmax(w_{e1}w_{a1} + w_{e2}w_{a2}) \quad (6)$$

（ 注1）: 0〜9 の手書き文字のデータセット [6].

（ 注2）: softmax 関数は活性化関数の一つである．NN の出力が $l_1 l_2 \ldots l_n$とあるとき i 番目のニューロンからの出力は $softmax(l_i) = \frac{e^{l_i}}{e^{l_1}+e^{l_2}+\ldots+e^{l_n}}$と表すことができる．

$$P\,(eat\ cola) = softmax\,(w_{e1}w_{c1} + w_{e2}w_{c2}) \tag{7}$$

となる．上の自然な文脈の場合に対応する式の値は大きく，下の不自然な文脈の場合に対応する式の値を小さくするように，各 w を学習していく．

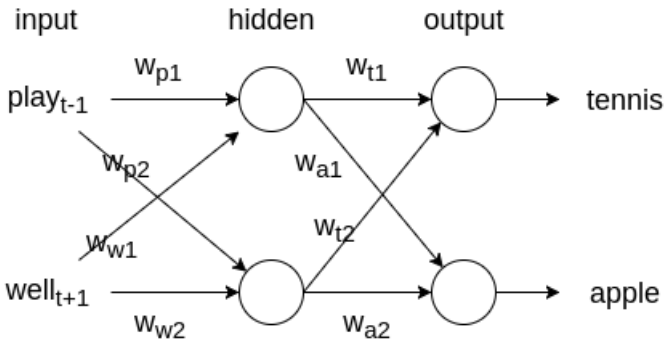

図 1　CBOW

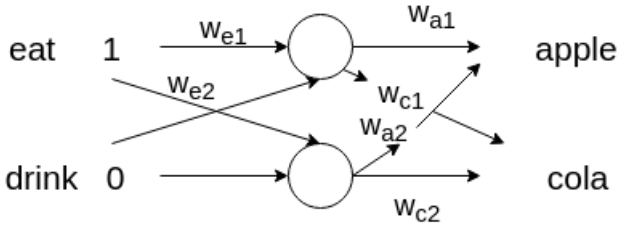

図 2　skip-gram

3. 実装したモデル

本項では，まず各モデルに共通している，Word2Vec の学習方法，及び文書データの前処理について論ずる．その後，SGAN の研究 [4]，CNN と Word2vec を用いて文書分類を行った研究 [3]，Salimans らの研究 [11] これら 3 つの研究をもとに作成した，2 種類のモデルについて説明する．

3.1 Word2Vec の学習

Word2vec の学習には Google が公開している学習済みモデル [15] を用いた．このモデルの学習には Google News dataset の一部が用いられている．

3.2 文書データの前処理

この章では，各データセットに対して施した前処理について述べる．これらの処理によって，語彙数を減らすことができ，より正確な分類が可能になる．

3.2.1 レンマ化

レンマ化 (見出し語化) とは，ある語を，辞書の見出しの形に直す処理のことである．

e.g.

- apples → apple
- slept → sleep

3.2.2 stemming

steaming (語幹化) は，ある単語を語幹に直す処理である．

e.g.

- introduce → introduc
- practice → practic

3.2.3 ストップワードの除去

ストップワードとは，一般に文書の分類に有益では無いとされる，前置詞や代名詞と言った単語のことである．これを除去することで，より文書を分類しやすい形にすることができる．

3.3 Word2vec+CNN を用いたモデル (提案手法 1)

3.3.1 データの成形

このモデルではデータの形成を以下の手順で行った．

（ 1） 文書に，3.2 節で述べたような前処理を行う．

（ 2） 文書の各単語を，Word2vec を用いてベクトル化する．

（ 3） 各ベクトルを結合する．

（ 4） 1〜3 の手順をすべての文書データに行った後，各データの長さを統一するために，ゼロパディング(注3) を行う．

これにより，文書を一枚の画像のように扱うことができる．

3.3.2 Discriminator (D) のモデルの概要

D にはを参考にした CNN のモデルを用いた．具体的には以下のようなモデルである．

（ 1） 実際の文書データ，または G により作られたデータを入力とする．

（ 2） 高さが 3，幅が単語ベクトルのサイズのフィルタで畳み込みを行う．

（ 3） 畳み込んだデータに対して，フィルタサイズ 3 で max-pooling(注4) を行う．

（ 4） 全結合を行う．

なお，各層の活性化関数には relu(注5) を用いた．

3.3.3 D の学習

このモデルにおいて，D は 2 種類の損失による学習を行った．

1 つは，実際のデータを入力として行われる，クラス分類的な損失である．具体的には，モデルからの出力と正解ラベルを用いて，binary cross entropy を計算し，これを損失とした．

もう 1 つは，G によって生成されたデータと，実際のデータを用いた，真偽分類的な損失である．通常では，元のデータが n クラスであったとき，G によって作られたデータに対するラベルである，fake クラスを追加して $n+1$ クラス分類を行う必要がある．しかし，Salimans らの方法 [11] を利用することで，分類器は n クラスを行いながらも，真偽分類を行うことができる．以下にその方法を示す．

まず，2 値分類を行う分類器に，fake クラス (f クラス) を加えた 3 値分類器を考える．また，これに対して f クラス以外のクラスを総称して $real$ クラス (r クラス) とする．活性化を行う前の出力を l_0，l_1，l_f としたとき，softmax 関数を用いて活性化をすると，入力データ X があるクラス c である確率 $p\,(c)$ は

（注3）：CNN に入力するデータの長さを統一するために長さが足りない分 0 を詰める操作.

（注4）：プーリングの方法の一つでフィルタ内の最大値をとる．

（注5）：$f(x) = \max(x,\ 0)$ とする活性化関数.

$$P(c|X) = \frac{e^{l_c}}{e^{l_0} + e^{l_1} + e^{l_f}} \tag{8}$$

と表すことができる．

ここで，出力 l_c に関して $l_c \leftarrow l_c - l_f$ とする．すると，l_f は常に 0 とみなすことができるため，仮想的に f クラスに対応する出力を考えなくてすむ．この結果，仮想的に $n+1$ 番めの出力が存在するとしながら，実際は N 個の出力だけの NN とすることができる．これにより，ある画像が G から生成されたものである確率は

$$P(f|X) = \frac{1}{e^{l_0} + e^{l_1} + 1} \tag{9}$$

と書くことができる．この方法によって，一般の n クラス分類器は構造を変えず，クラス分類を行いながら，真偽分類を行うことが可能となる．

これを用いて GAN の損失関数を

$$\mathbf{V}L_D = \mathbb{E}_{x\sim p_{data}(x)}[\log(P(R|X))] - \mathbb{E}_{z\sim p_z(z)}[\log(P(f|G(z)))]. \tag{10}$$

と変形でき計算することができる．

3.3.4 Generator (G)

G には通常の NN のモデルを用いた．具体的には以下のようなモデルである．

（ 1） −1〜1 の乱数を複数結合したものを，入力とする．

（ 2） 二層の中間層を経る．

（ 3） 出力として $N \times L$ 次元の出力をする (ここで N は単語ベクトルの次元数，L は最長の入力データ長)．

（ 4） これを N 個ごとに区切り L 個の単語のデータかのように扱う．

この出力を，なるべく D が本物の文書と間違えるようなデータにすることが目的である．

3.3.5 G の学習

このモデルにおいて，G の損失は，いかに作られたデータを作られたデータと認識されてしまうか，つまりは，D がいかに作られたデータが作られたデータだと見破ってしまうかで決まる．その計算は，D の学習の項で述べたように，仮想的にクラス f に相当する出力を定義することで

$$L_G = \mathbb{E}_{z\sim p_z(z)}[\log(P(f|G(z)))] \tag{11}$$

となる．

3.4 Word2vec+通常の NN を用いたモデル (提案手法 2)

3.4.1 データの成形

このモデルでは，データの形成を以下の手順で行った．

（ 1） 文書に 3.2 節で述べたような前処理を行う．

（ 2） 文書の各単語を，Word2Vec を用いてベクトル化する．

（ 3） 各ベクトルの各要素の平均をとる．

（ 4） 1〜3 の手順をすべての文書データに行う．

3.4.2 Discriminator (D)

このモデルの D には，基本的に通常の NN を使用する．但し，文書のクラスに作られたデータに対するクラスとして，fake という新たなクラスを追加する．よって，元のクラスが [positive, negative] の 2 クラス分類であったら，[positive，negative，fake] の 3 クラス分類に変更する．具体的には以下のようなモデルである．

（ 1） 成形された文書データを入力とする．

（ 2） 2 層の中間層を経る．

（ 3） 出力として，入力データが各クラスに属する確率を出力する．

なお，各層の活性化関数には relu を用い，出力層には softmax 関数を用いた．

3.4.3 D の学習

D の学習には，実際のデータと G が作成したデータを組み合わせたものを用いた．これらを，元の n 個のクラス+ fake クラスの $n+1$ クラス分類を行い，学習を行った．損失関数には binary cross entropy を用いた．

3.4.4 Generator (G)

このモデルの G には通常の NN を使用する．具体的には以下のようなモデルである．

（ 1） −1〜1 の乱数を複数結合したものを入力とする．

（ 2） 2 層の中間層を経る．

（ 3） 出力として実データと同様の次元の出力を行う，これを G が生成した偽データと扱う．

なお，各層の活性化関数には relu を，出力層の活性化関数には sigmoid 関数(注6) を用いた．

3.4.5 G の学習

G の学習には，もうひとつのモデルと同様の損失関数である $L_G = \mathbb{E}_{z\sim p_z(z)}[\log(P(f|G(z)))]$ を用いた．

4. 評価実験

4.1 モデルの実装について

今回モデルの実装のために，プログラミング言語としては Python を用い，フレームワークとしては Chainer [7] を使用した．また，実装や前処理において，複数のサイトを参考にした [9] [10].

4.2 評価に用いたデータセット

今回の研究において，提案手法を評価する際に複数のデータセットを用いた．表 4.1 に概要を示す．

表 1 データセットの概要

データセット	データの概要	クラス数	データ数	最大長	平均
MR	映画の感想のスニペット	2	10662	270	116
reuters	ロイターのニュース	46	11228	2376	146
IMDB	映画の感想	2	25000	2494	235

これらのデータセットは以下の基準で選定された

(注6)： $f(x) = \frac{1}{1+e^{-x}}$ を満たす関数.

- 多様な文書のクラス数で実験を行う．
- 多様な文書のジャンルで実験を行う．
- 多様な文書の長さで実験を行う．

以下で，使用したデータセットについて説明を行う．

4.2.1 MR

Movie Review Data，通称 MR は映画のレビューのスニペットである [5]．各レビューには *positive* か *negative* のラベルがつけられている．

4.2.2 IMDB

IMDB は，映画のレビューにラベルを付けたものである．今回は，Python の機械学習の Keras [8] が提供している整形済みのものを使用した．各レビューには， *positive* か *negative* のラベルがつけられている．MR と違い，映画のレビューの全文であるため，文長は長くなっている．

4.2.3 Reuters

Reuters はニュースサイトであるロイターの記事データに，ラベルを付けたものである．今回は，Python の機械学習のライブラリである Keras [8] が提供している，整形済みのものを使用した．各データは，46 種類のトピックでラベル付けが行われている．

4.3 ベースライン

本論文の目的は，GAN による真偽学習の損失を加味したとき，文書の分類正解率が向上するかを検証することである．よって，今回のベースラインは，各モデルを真偽判定損失を用いることなく学習させたモデルとする．

4.4 評価指標

今回，評価指標として正解率 (accuracy) の最大値を用いた．まず，正解率を以下のように定義する．

$$\text{正解率} = \frac{\text{正しくクラス分類できたテスト用文書の数}}{\text{テスト用文書の総数}}$$

但し，ここで「正しく分類できた」とはモデルが出力した各ラベルの確率のうち最大のものが正解ラベルと一致することを示す．例えば，あるデータに対して，モデルがクラス 0，1，2 の三値分類を行うことを考える．モデルが出力した，各クラスに属する確率をそれぞれ，p_0，p_1，p_2とする．そのデータの正解ラベルが 1 であれば，$[p_0, p_1, p_2]$=[0.1,0.5,0.3] のように，p_1 の値が一番大きく出力されたとき，正しく分類されたとみなす．

上記のように定義した正解率を訓練を 2 エポック行う度に計測し，一通り訓練を行った中で，得られた正解率の最大値を用いて各モデルを比較する．

4.5 実験方法

実験は以下の手順で行われる

（ 1） データセットの P%をテストデータとして，(100-P)%を訓練データとして分割する．

（ 2） 訓練データを用いて，モデルの学習を行う．

（ 3） 2 エポックに 1 度，テストデータを用いて，提案手法のモデルとベースラインの正解率の計算を行う．

（ 4） 2，3 を繰り返し行い，その学習過程で正解率の最大値を評価指標とする．

モデルの比較に正解率の最大値を用いたのは，今回のモデルでは学習過程で正解率が上下することがあり，一定エポック後における正解率で比較を行うと，正確に両モデルの分類正解率を比較できない可能性があるからである．

今回 P の値としては 90 を用いた．これは，少数の教師データを用いて画像に対する高い分類正解率を示した先行研究と同様の現象を，文書を対象した場合に確認したかったためである．各データの繰り返しの回数は，結果の部分に記載する．

4.6 評価結果

以下が前述の 3 種類のデータセットを用いて，提案手法とベースラインを比較した結果である．表における提案手法 1，提案手法 2 はそれぞれ以前の項で紹介した CNN を用いたモデル，及び NN を用いたモデルを示す．各データに関して，ランダムに教師データと訓練データに分割し，4 回実験を行い，実験毎に正解率の最大値を記録した．

表 2　各データ，及び各モデルにおける正解率の最大値

データセット (エポック数)	モデル	1 回目	2 回目	3 回目	4 回目
MR (50)	ベースライン	0.6491	0.6499	0.6403	0.6522
	提案手法 1	0.6397	0.6320	0.6318	0.6278
reuters (200)	ベースライン	0.6264	0.6353	0.6326	0.6068
	提案手法 1	0.6366	0.6340	0.6246	0.6153
IMDB (200)	ベースライン	0.86032	0.84808	0.84752	0.84572
	提案手法 2	0.86044	0.84612	0.84688	0.84648

4.7 統計的検定

本項では，ベースラインと提案手法における，正解率の最大値に有意差があるか，符号検定(注7) を行った結果を示す．検定は各データセットについて，以下の手順で行った

（ 1） あるデータセットをテストデータと訓練データに 9:1 の割合でランダムに分割．

（ 2） 提案手法とベースラインについてそれぞれ，(1) のデータを用いて，正解率の最大値を計測．

（ 3） 正解率の最大値を記録したときの，各データに対する正解，不正解を記録する．

（ 4） 1 から 3 を 4 回繰り返し行う．

（ 5） 提案手法とベースライン各々について，4 回分の正解率の最大値をとった際の各データに対する正解，不正解が得られた．これを用いて，提案手法とベースラインの正解率の最大値について有意水準 αは 0.05，帰無仮説 H_0は「各モデルの正解率の最大値に有意差は無い」とする 符号検定を行う．

検定の結果を表 3 に示す．表の ○は帰無仮説が棄却されたことを，× は帰無仮説が棄却されなかったことを示す．この結果 MR を用いた実験の一部では有意差が確認できたが，残りの実験では有意差は確認できなかった．

（注7）: 酒井 (2015) は「…いずれの手法が優れているかという情報しか得られない場合がある。…このような場合には符号検定を用いることができる。」(p.152) と述べている [17].

表 3　各データ，及び各モデルにおける正解率の最大値

データセット (サンプルサイズ)	モデル	1 回目	2 回目	3 回目	4 回目
MR (4796)	p 値	0.1852	0.0101	0.2351	0.0005
	帰無仮説	×	○	×	○
reuters (2246)	p 値	0.0884	0.8740	0.2025	0.2195
	帰無仮説	×	×	×	×
IMDB (25000)	p 値	0.9473	0.1476	0.6508	0.6170
	帰無仮説	×	×	×	×

4.8 考　　察

先行研究において，MNIST の分類に GAN を用いることで，多大な分類正解率の向上が見られた．しかし，今回検証した，文書の分類に GAN を用いるによる，分類正解率の向上はほぼ見られなかった．その理由の一つは，ラベルとそれに対するデータの多様性にあると考えられる．

図 3　MNIST において 9 のラベルが付けられたデータ [6]

図 3 は Keras [8] がデータセットとして提供している，MNIST [6] において，9 のラベルがつけられたデータである．このように崩れている，歪んでいる等の差はあれどラベル 9 に対応するものは 9 が書かれている画像に限られた．そのため，GAN が fake データを作る際に，模倣しやすいと考えられる．

一方，以下に示す 2 つの MR のデータは，共にラベル positive が付けられたものである

1: ‘if you sometimes like to go to the movies to have fun , wasabi is a good place to start .’

2: ‘this is a film well worth seeing , talking and singing heads and all .’

この二つのデータは，同じ positive ラベルが付けられてはいるが，全く違う単語が使われている．このように，あるラベルに対する特徴が多岐に渡るため，GAN が fake データを作る際に模倣をすることが難しいと考えられる．これが原因で，GAN の学習が上手く行かなかったことが，今回の実験において正解率向上が見られなかった原因の一つであると考えられる．

また，文書ベクトルの多様性も GAN の学習を難しくしていると考えられる．上の 2 文の positive のラベルの要因となっているとされる，単語の fun と worth に着目する．単語の品詞は，それぞれ名詞と形容詞である．Word2vec によるベクトル化において，似た意味の単語は似たベクトルとなる傾向があるが，その単語のベクトル化で用いられるのは，単語の前後の単語である．しかしながら，上の 2 文においてラベル付けの要因となっている単語は，それぞれ別の品詞である．そのため，一般に前後の単語は大きく異なっている．このように，ラベリングの要因となる単語は様々な品詞に及び，必ずしもベクトルが類似したものにはならないと考えられる．よって，単語ベクトルの和である文書を表すベクトルも，例え同じラベルが付けられていても，類似したものになるとは限らない．このように，同じラベルの文書でありながらベクトル表現上は全く異なったものになりうることも，GAN の学習を難しくし，分類正解率の向上を阻む原因と考えられる．

さらに，今回実装したモデルの Generator の単純さも問題になっている可能性がある．先行研究において，画像生成や分類の GAN の Generator には，画像の生成に適している CNN が使われている．一方，今回 Generator として一般の NN を利用したが，これでは文の生成という複雑な目的には十分ではなく，精巧な fake が作れなかった可能性がある．よって，Generator を，文書生成に有効であるとされている Long Short Term Memory (LSTM)[注8] 等に変更することで，より精巧な fake を作成することができ，これにより分類正解率が向上する可能性がある．

5. 結　　論

本論文では，様々な分野で多大な成果を挙げている GAN の文書分類への応用について検証した．そのために，文書を Word2Vec を用いてベクトル化を行い，CNN や NN を用いた通常のクラス分類に加え，GAN を用いた真偽判定学習を使ったモデルを作成した．実験の結果，GAN を用いることによる文書分類正解率の向上はほぼ確認できなかった．その原因として，ラベルに対するデータの多様性，ラベリングの要因となる単語の品詞の多様性，モデルの構造，等が考えられる．今後のモデルの改善案としては，LSTM など文書生成に特化したモデルを Generator に用いることが考えられる．これにより，より精巧な fake が作成でき，分類正解率が向上する可能性がある．

文　　献

[1] Richard Socher and Alex Perelygin and Jean Wu and Jason Chuang and Christopher Manning and Andrew Ng and Christopher Potts: Parsing With Compositional Vector Grammars,EMNLP,2013.

[2] Ian J. Goodfellow,Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio : Generative Adversarial Nets，arXiv:1406.2661 [stat.ML]，2014.

[3] Yoon Kim:Convolutional Neural Networks for Sentence Classification，EMNLP，p.1746-1751，2014.

[4] Augustus Odena:Semi-Supervised Learning with Generative Adversarial Networks,Data Efficient Machine Learning workshop at ICML 2016 ,2016.

[5] MovieReviewData
http://www.cs.cornell.edu/people/pabo/movie-review-data/

[6] THE MNIST DATABASE of handwritten digits
http://yann.lecun.com/exdb/mnist/

[7] chainer
https://github.com/chainer/chainer

[8] keras.datasets
https://github.com/keras-team/keras/tree/master/keras/datasets

[9] improved-gan
https://github.com/musyoku/improved-gan

（注8）：セルと呼ばれる過去の情報を記憶しておく部分によって，長い文書であっても始めの方の情報を保持できる機械学習のモデル．文書分類や文書生成に用いられる．

[10] Python,NLTK で自然言語処理
http://haya14busa.com/python-nltk-natural-language-processing

[11] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen: Improved Techniques for Training GANs,arXiv:1606.03498 [cs.LG],2016.
[12] T Joachims:Text categorization with support vector machines: Learning with many relevant features,ECML-98,1998.

[13] Minqing Hu and Bing Liu:Mining and summarizing customer reviews,Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD-04), 2004.
[14] Quoc Le,Tomas Mikolov:Distributed Representations of Sentences and Documents,ICML,2014.
[15] GoogleNews-vectors-negative300.bin.gz
https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit

[16] Tomas Mikolov，Ilya Sutskever，Kai Chen，Greg Corrado，Jeffrey Dean，Distributed Representations of Words and Phrases and their Compositionality，2013.
[17] 酒井哲也. 情報アクセス評価方法論: 検索エンジンの進歩のために. コロナ社，2015.