卒業論文

多層ニューラルネットワークの
パラメータ初期化手法に関する研究

03-120584　**中丸 智貴**

指導教員　合原一幸 教授

2016 **年** 2 **月**

東京大学工学部計数工学科数理情報工学コース

概要

近年深い階層を持つ人工ニューラルネットワーク (ディープニューラルネットワーク) が多くの成果とともに再注目され, ディープニューラルネットワークの学習手法 (ディープラーニング) に関する研究が改めて活発になっている. 特に多層ニューラルネットワークでは学習に誤差逆伝播法が用いられるが, この手法による学習は初期パラメータに非常に影響を受けやすいことも知られている. 本研究ではまず, 近年利用されることの多い活性化関数 ReLU を用いた多層ニューラルネットワークにおいて Xavier Initialization を改善した. また, 今後使用される機会が増えると予想される高次元数のニューラルネットワークについて Xavier Initialization の修正を考え, 複素数ニューロンモデルから成る多層ニューラルネットワークでその効果の確認を行った.

キーワード　深層学習, Xavier Initialization, 複素数ニューラルネットワーク

目次

第 1 章

序論

1.1 人工ニューラルネットワークへの再注目

人工ニューラルネットワークは脳における神経細胞網の仕組みを模した非常に簡素なモデルであり, 学習させることで画像分類などのタスクを行うことができる. その中でも特に深い階層を持った人工ニューラルネットワーク (ディープニューラルネットワーク) を用いた機械学習は近年多くの成果を上げ再注目されている. 再注目の要因は様々であるが, 2012 年に開催された ImageNet Large Scale Visual Recognition Challenge (ILSVRC 2012) の画像分類タスクにおいて, ディープニューラルネットワークを用いたチームが人工ニューラルネットワーク以外の機械学習を用いたチームを大きく上回るスコアを記録した [12] ことは人工ニューラルネットワークの研究者以外の人にも衝撃を与えた.

人工ニューラルネットワークの応用が広がるとともに取り扱うデータの種類も多種多様になってきている. 対象とするデータが持つ性質によっては通常使われる実数の人工ニューラルネットワークは必ずしも適しておらず, 例えば風の速度と向きを取り扱う場合は複素数ニューラルネットワーク [4], 幾何変換を取り扱う場合は四元数ニューラルネットワーク [9] などを利用することでより良い結果を得られるとされている. しかしながら現時点では実数以外のディープニューラルネットワークに関する研究は, 実数のものと比べると盛んとは言えない.

1.2 ディープニューラルネットワークの学習

実数以外の場合も含め多層ニューラルネットワークでは, その学習に誤差逆伝播法 [1][15] が用いられる. しかしながらディープニューラルネットワークの場合, その深い構造のために誤差逆伝播法による学習は必ずしも期待通りに進まない. 入力層に近いほど伝播する誤差が指数的に減衰してしまい, 入力層に近い側のパラメータの更新が期待するほど進まなくなってしまう勾配消失問題 [8] などが理由としてあげられている. またディープニューラルネットワークには非常に多くのパラメータが存在し, さらにそれらが出力に対して複雑に作用することから, 勾配降下法を用いる誤差逆伝播法で良い性能を持つパラメータを任意の初期パラメータから探索することは一般に困難である.

このようにディープニューラルネットワークを誤差逆伝播法によって学習するには様々な困難が存在するものの, 適切な初期パラメータから学習を開始することで, 誤差逆伝播法によるパラメータ更新でも十分学習が進められるということが Hinton らによって発見されている [6]. 同文献において Hinton らの考案した事前学習と呼ばれる手法では, まず本来目的とするディープニューラルネットワークの学習の前にオートエンコーダ [7] を用いた学習を行う. そしてそのオートエンコーダの学習で得られたパラメータを, 本来目的としたディープニューラルネットワークの初期パラメータとして利用するものである.

1.3　ディープニューラルネットワークのパラメータ初期化手法

事前学習は強力な手法であり広く用いられている一方で, 目的とした学習の前に別の学習を行うという性質上, 非常に多くの計算機リソースが必要になってしまうという問題が挙げられる. また, 事前学習に用いるオートエンコーダは目的とするディープニューラルネットワークと比べて浅いものを用いるなどして本来の目的のディープニューラルネットワークより学習しやすくすることが意図されているが, 隠れ層が 1 層の場合においても初期パラメータによって学習の進みに少なからず差がある [11] ということも知られている. そのため, 事前学習を速やかに行うためのオートエンコーダの初期パラメータをどう設定するかという問題も存在する.

一方 Glorot らは, ネットワーク中の人工ニューロン数によって決まるある範囲の一様分布に従って初期パラメータを生成する手法 (Xavier Initialization) によっても, 学習を大きく改善できるという実験結果を示した [3]. Xavier Initialization は計算機リソースをほとんど必要とせず, 事前学習のように別の問題が新たに出てくるということはない.

しかしながら Glorot らの研究では, 活性化関数に Rectified Linear Unit (ReLU) を用いた場合について実験を行っておらず, Xavier Initialization の導出にあたって活性化関数に対して仮定されている性質も ReLU では成り立たない. ReLU はその特性からディープニューラルネットワークに適していると考えられおり [13], 実際に成果を上げている人工ニューラルネットワークでも頻繁に利用されている活性化関数である.

He らはこの Xavier Initialization の問題について, Glorot らが導出において行ったものと同様の考察を ReLU について行うことで, Xavier Initialization の改善 (以下 He Initialization と呼ぶ) を提案している [5]. 同文献によると He Initialization を用いることでディープニューラルネットワークの学習を大幅に改善できたとしている.

1.4　本研究の目的

今後さらにディープニューラルネットワークへの注目が高まり, 応用先も増えることが考えられる. 前述の通り, ディープニューラルネットワークの学習を進めるために事前学習は広く使われているが, これには非常に多くの計算機リソースを要する. そのため Xavier Initialization や He Initialization のような軽量な手法は今後さらにディープニューラルネットワークの応用を進める上でも重要になると予想される.

そこで本研究ではまず始めに, 活性化関数に ReLU を用いた多層ニューラルネットワークにおいて Xavier Initialization を改善できるかについて考察し, その結果得られた手法と He Initialization を比較する実験を行った. またそこで得た結果への考察から, He Initialization のさらなる改善を検証する実験を行った. さらに Xavier Initialization と同様の形式の初期化手法が高次元数のニューラルネットワークでも考えられるという予想から, 高次元数のニューラルネットワークでの Xavier Initialization の修正を考え, 複素数の多層ニューラルネットワークでその効果の検証する実験を行った.

第 2 章

先行研究

2.1 多層ニューラルネットワークの学習

2.1.1 順伝播による出力計算

隠れ層が N 層 (入力層, 出力層含めて全 $N+2$ 層) である多層ニューラルネットワークを考える. 第 n 層の第 k 番目と第 $n+1$ 層の第 l 番目の人工ニューロン間の結合係数を w_{kl}^{n}, 第 n 層の第 k 番目の人工ニューロンのバイアス値を b_{k}^{n}, 第 n 層の活性化関数を f_n と表し, 第 n 層の第 k 番目の人工ニューロンへの入力値を i_{k}^{n}, 出力値を o_{k}^{n} とする (図 2.1). 入力層の第 k 番目の人工ニューロンへの入力が x_k であるとすると, 各入出力値は下式のように入力層側から出力層側へと伝播させることで得られる.

$$i_{l}^{n} = \begin{cases} x_l & n = 0 \\ \sum_{k'} w_{k'l}^{n-1} o_{k'}^{n-1} & n = 1, 2, ..., N+1, \end{cases} \tag{2.1}$$

$$o_{l}^{n} = \begin{cases} i_{l}^{n} & n = 0 \\ f_n(i_{l}^{n} + b_{l}^{n}) & n = 1, 2, ..., N+1. \end{cases} \tag{2.2}$$

但し出力層の活性化関数 f_{N+1} に, 次のように定義されるソフトマックス関数を用いることもある.

$$o_{l}^{N+1} = \frac{\exp(i_{l}^{N+1} + b_{l}^{N+1})}{\sum_{k'} \exp(i_{k'}^{N+1} + b_{k'}^{N+1})}. \tag{2.3}$$

これは一つの出力値 o_{l}^{N+1} を求めるために, 出力層の他の人工ニューロンの入力値とバイアス値を用いることから式 2.2 とは異なる形式であるが, 入力層側から出力層への伝播によって出力値が求められる点は共通している.

2.1.2 誤差逆伝播法によるパラメータ更新

入力層の人工ニューロンへの各入力値 x_k を並べたベクトルを $\mathbf{x}$, 出力層の人工ニューロンの各出力値 o_{k}^{N+1} を並べたベクトルを $\mathbf{o}$ と表す. 教師あり学習である誤差逆伝播法では, ある $\mathbf{x}$ に対してのネットワークの期待出力 t_k (教師信号) が与えられていることが想定されており,

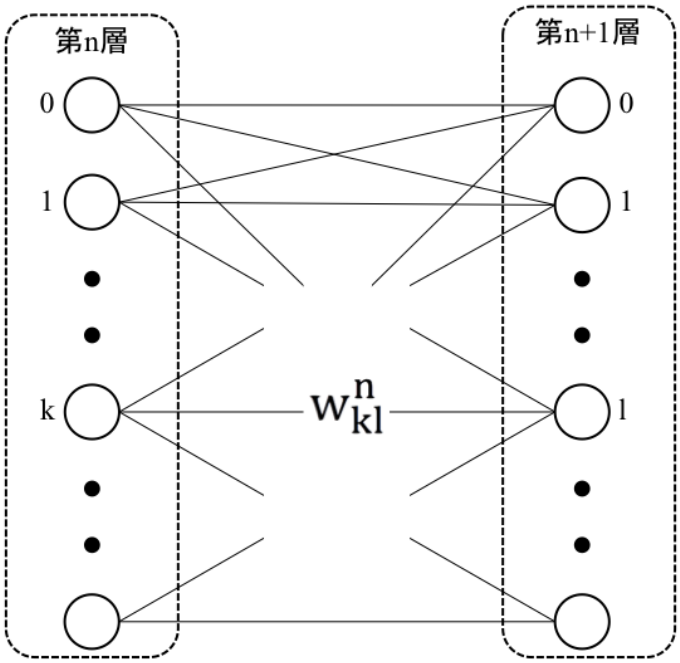

図 2.1. 多層ニューラルネットワーク

これらを並べたベクトルを $\mathbf{t}$ とする．このときネットワークの出力と教師信号の間の距離 e は誤差関数 E として

$$e = E(\mathbf{o}; \mathbf{t}) \tag{2.4}$$

と定められる．誤差関数はいくつか考案されているが, その中でももっと基本的なものは二乗誤差関数

$$\begin{aligned} E(\mathbf{o}; \mathbf{t}) &= \frac{1}{2}\|\mathbf{o} - \mathbf{t}\|^2, \\ &= \frac{1}{2}\sum_{k'}(o^{N+1}_{k'} - t_{k'})^2 \end{aligned} \tag{2.5}$$

である．また交差エントロピー関数

$$\begin{aligned} E(\mathbf{o}; \mathbf{t}) &= -\mathbf{t}^T \log \mathbf{o}, \\ &= -\sum_{k'} t_{k'} \log o^{N+1}_{k'} \end{aligned} \tag{2.6}$$

も頻繁に利用される．

誤差逆伝播法による学習では, 式 2.4 の e を最小化するようなパラメータ w^n_{kl} と b^n_k を得るために勾配降下法を用いる．一般に各 w^n_{kl}, b^n_k についての E の導関数を直接求めることは困難であるが, 出力層側から入力層側への誤差の伝播を考えることで比較的容易に各勾配を得るための式を得る．

$$\frac{\partial E}{\partial b^n_k} = \begin{cases} f'_n(i^n_k + b^n_k)\sum_{l'} w^n_{kl'}\frac{\partial E}{\partial b^{n+1}_{l'}} & n = 1, 2, ..., N \\ f'_n(i^n_k + b^n_k)\frac{\partial E}{\partial o^n_k} & n = N+1, \end{cases} \tag{2.7}$$

$$\frac{\partial E}{\partial w_{kl}^n} = o_k^n \frac{\partial E}{\partial b_l^{n+1}}. \tag{2.8}$$

但し出力層にソフトマックス関数を用いる場合には, 式 2.7 の $n = N + 1$ の部分は

$$\frac{\partial E}{\partial b_k^{N+1}} = \sum_{l'} \frac{\partial o_{l'}^{N+1}}{\partial b_k^{N+1}} \frac{\partial E}{\partial o_{l'}^{N+1}} \tag{2.9}$$

となることに注意する.

実際のパラメータの更新は, 式 2.7, 2.8 によって得られた各勾配と学習係数 ζ, η $(0 < \zeta, \eta)$ を用いて以下のように表される.

$$w_{kl}^n \leftarrow w_{kl}^n - \zeta \frac{\partial E(\mathbf{o}; \mathbf{t})}{\partial w_{kl}^n}, \tag{2.10}$$

$$b_k^n \leftarrow b_k^n - \eta \frac{\partial E(\mathbf{o}; \mathbf{t})}{\partial b_k^n}. \tag{2.11}$$

学習係数 ζ, η は学習を通して固定された値を用いるのが最も基本的な方法であるが, これを学習の状況に応じて変更する AdaGrad[2] なども広く利用されている. t 回目の更新の際に w_{kl}^n, b_k^n を更新するために用いる学習係数をそれぞれ ζ_{kl}^{nt}, η_k^{nt} とし, t 回目の学習の w_{kl}^n, b_k^n についての勾配をそれぞれ u_{kl}^{nt}, v_k^{nt} とする. このとき AdaGrad を用いると,

$$\zeta_{kl}^{nt} = \frac{\zeta}{\sqrt{\sum_{t'} u_{kl}^{nt'}}} u_{kl}^{nt}, \tag{2.12}$$

$$\eta_k^{nt} = \frac{\eta}{\sqrt{\sum_{t'} v_k^{nt'}}} v_k^{nt} \tag{2.13}$$

と各学習係数が定められる.

2.1.3　確率的勾配降下法

前節で求めた誤差逆伝播法によるパラメータ更新の式 2.10, 2.11 は, 1 つの入力 $\mathbf{x}$ に対しての出力 $\mathbf{o}$ が教師信号 $\mathbf{t}$ に近づくような方向へのパラメータ更新を表す. 一般的な応用においては複数の入力を取り扱うことになるが, この場合はバッチ学習と呼ばれる方法を用いる. これは入力それぞれに対して式 2.7, 2.8 により勾配を算出し, それら勾配の平均をパラメータ更新の勾配として式 2.10, 2.11 に用いるというものである. つまり, M 個の入力 $\mathbf{x}_1, \mathbf{x}_1, ..., \mathbf{x}_M$ があり, それに対する出力と教師信号が $\mathbf{o}_1, \mathbf{o}_1, ..., \mathbf{o}_M$ と $\mathbf{t}_1, \mathbf{t}_1, ..., \mathbf{t}_M$ であるとして,

$$w_{kl}^n \leftarrow w_{kl}^n - \frac{\zeta}{M} \sum_{m'} \frac{\partial E(\mathbf{o}_{\mathbf{m}'}; \mathbf{t}_{\mathbf{m}'})}{\partial w_{kl}^n}, \tag{2.14}$$

$$b_k^n \leftarrow b_k^n - \frac{\eta}{M} \sum_{m'} \frac{\partial E(\mathbf{o}_{\mathbf{m}'}; \mathbf{t}_{\mathbf{m}'})}{\partial b_k^n} \tag{2.15}$$

というパラメータ更新を行う.

しかしながら現実には全データに対して式 2.14, 2.15 の計算を行うことは困難である. そのため応用においては, 全データからランダムに取り出した一部のみに対してバッチ学習 (ミニバッチ学習) を繰り返してデータ全体について最適なパラメータを探索していく. このような手法は確率的勾配降下法と呼ばれ, 人工ニューラルネットワークの学習の場合にはほぼ必ず利用される.

2.2 実数以外を取り扱うニューラルネットワーク

2.2.1 統一的表現

通常人工ニューラルネットワークでは各人工ニューロンへの入力と出力は 1 つの実数値の場合を考えることが多い. しかしながら本質的に複素数によって情報が表されるもの (信号にフーリエ変換を施したものなど) は, 各入出力が 1 つの実数値のニューラルネットワークでは期待するような結果を得にくい. そのため入出力, 結合係数, バイアス値を複素数として考える複素数ニューラルネットワークが考案されている. また同様の拡張を四元数についても行うことで四元数ニューラルネットワークなども考案されている.

しかしながらこのような実数以外のニューラルネットワークの学習を考えるとき, 複素数や四元数上の積演算の規則は実数上の積演算の規則とは異なるため, 式 2.7 から式 2.11 の実数に対する誤差逆伝播法の表式をそのまま適用することはできない. よって複素数, 四元数などのそれぞれについての誤差逆伝播法の計算方法を考える必要があり, 多くの研究ではそれらが個別に示されていることが多い.

一方小林らの研究 [10] では, これらの様々な高次元数のニューラルネットワークの統一的な行列演算による表現を提案している. 同研究においては隠れ層が 1 層のニューラルネットワークについて誤差逆伝播法を考え計算方法を導出しているが, 以下ではこれを拡張し, 隠れ層が N 層である場合 (入力層, 出力層含めて全 $N+2$ 層) についての結果を示す.

ネットワークへの入力値, 各人工ニューロンの結合係数, バイアス値が P 次元数であるとする (複素数なら $P=2$ などとなる). 第 n 層の第 k 番目と第 $n+1$ 層の第 l 番目の人工ニューロンの間の結合係数の第 p 次元目の値を w^n_{klp}, 第 n 層の第 k 番目の人工ニューロンのバイアス値の第 p 次元目の値を b^n_{kp} と表し, 第 n 層の第 k 番目の人工ニューロンの入力値の第 p 次元目を i^n_{kp}, 出力値の第 p 次元目を o^n_{kp} とする. 入力層の第 k 番目の人工ニューロンへの入力の第 p 次元目を x_{kp} と表すと, 各入出力値は下式によって得られる.

$$i^n_{lp} = \begin{cases} x_{lp} & n=0 \\ \sum_{k',q',r'} w^{n-1}_{k'lq'} o^{n-1}_{k'r'} s_{pq'r'} & n=1,2,....,N+1, \end{cases} \tag{2.16}$$

$$o^n_{lp} = \begin{cases} i^n_{lp} & n=0 \\ f_n(i^n_{lp}+b^n_{lp}) & n=1,2,...,N+1. \end{cases} \tag{2.17}$$

s_{pqr} は高次元数同士の積の定義によって定まる値であり, $0 \leq p,q,r < P$, $s_{pqr} \in \{-1,0,1\}$ である. 例えば複素数 ($P=2$) のニューラルネットワークでは, 各 s_{pqr} は表 2.1, 2.2 のようになる.

表 2.1. 複素数での s_{0qr}

	$r=0$	$r=1$
$q=0$	1	0
$q=1$	0	-1

表 2.2. 複素数での s_{1qr}

	$r=0$	$r=1$
$q=0$	0	1
$q=1$	1	0

表 2.3. 四元数での s_{0qr}

	$r=0$	$r=1$	$r=2$	$r=3$
$q=0$	1	0	0	0
$q=1$	0	-1	0	0
$q=2$	0	0	-1	0
$q=3$	0	0	0	-1

表 2.4. 四元数での s_{1qr}

	$r=0$	$r=1$	$r=2$	$r=3$
$q=0$	0	1	0	0
$q=1$	1	0	0	0
$q=2$	0	0	0	1
$q=3$	0	0	-1	0

表 2.5. 四元数での s_{2qr}

	$r=0$	$r=1$	$r=2$	$r=3$
$q=0$	0	0	1	0
$q=1$	0	0	0	-1
$q=2$	1	0	0	0
$q=3$	0	1	0	0

表 2.6. 四元数での s_{3qr}

	$r=0$	$r=1$	$r=2$	$r=3$
$q=0$	0	0	0	1
$q=1$	0	0	1	0
$q=2$	0	-1	0	0
$q=3$	1	0	0	0

実際に表 2.1, 2.2 の s_{pqr} を用いて式 2.16 の $n \neq 0$ の部分を整理すると,

$$i_{l0}^{n} = \sum_{k'} w_{k'l0}^{n-1} o_{k'0}^{n-1} - w_{k'l1}^{n-1} o_{k'1}^{n-1}, \tag{2.18}$$

$$i_{l1}^{n} = \sum_{k'} w_{k'l0}^{n-1} o_{k'1}^{n-1} + w_{k'l1}^{n-1} o_{k'0}^{n-1} \tag{2.19}$$

となり, 各高次元数の第 0 次元目を実数部分, 第 1 次元目を虚数部分と見れば複素数の積を計算していると見ることができる.

また四元数 ($P=4$) での各 s_{pqr} は表 2.3 から表 2.6 のようになる. 表 2.3 から表 2.6 の s_{pqr} を用いて式 2.16 の $n \neq 0$ の部分を整理すれば

$$i_{l0}^{n} = \sum_{k'} w_{k'l0}^{n-1} o_{k'0}^{n-1} - w_{k'l1}^{n-1} o_{k'1}^{n-1} - w_{k'l2}^{n-1} o_{k'2}^{n-1} - w_{k'l3}^{n-1} o_{k'3}^{n-1}, \tag{2.20}$$

$$i_{l1}^{n} = \sum_{k'} w_{k'l0}^{n-1} o_{k'1}^{n-1} + w_{k'l1}^{n-1} o_{k'0}^{n-1} + w_{k'l2}^{n-1} o_{k'3}^{n-1} - w_{k'l3}^{n-1} o_{k'2}^{n-1}, \tag{2.21}$$

$$i_{l2}^{n} = \sum_{k'} w_{k'l0}^{n-1} o_{k'2}^{n-1} - w_{k'l1}^{n-1} o_{k'3}^{n-1} + w_{k'l2}^{n-1} o_{k'0}^{n-1} + w_{k'l3}^{n-1} o_{k'1}^{n-1}, \tag{2.22}$$

$$i_{l3}^{n} = \sum_{k'} w_{k'l0}^{n-1} o_{k'3}^{n-1} + w_{k'l1}^{n-1} o_{k'2}^{n-1} - w_{k'l2}^{n-1} o_{k'1}^{n-1} + w_{k'l3}^{n-1} o_{k'0}^{n-1} \tag{2.23}$$

となり, 複素数の場合と同様に, 四元数の積を計算していることが確かめられる.

2.2.2 誤差逆伝播法

式 2.16, 2.17 に現れる各値は実数であるため, これらを用いることで高次元数での誤差逆伝播法による勾配計算は実数で導出する場合と同様にして得ることができる. その結果は次のようにまとめられる.

$$\frac{\partial E}{\partial b^n_{kr}} = \begin{cases} f'_n(i^n_{kr} + b^n_{kr}) \sum_{l',p',q'} w^n_{kl'q'} s_{p'q'r} \frac{\partial E}{\partial b^{n+1}_{l'p'}} & n = 1, 2, ..., N \\ f'_n(i^n_{kr} + b^n_{kr}) \frac{\partial E}{\partial o^n_{kr}} & n = N + 1, \end{cases} \tag{2.24}$$

$$\frac{\partial E}{\partial w^n_{klq}} = \sum_{p',r'} o^n_{kr'} s_{p'qr'} \frac{\partial E}{\partial b^{n+1}_{lp'}}. \tag{2.25}$$

パラメータ更新方法や確率的勾配降下法も実数において考えた方法と同様に考えることができる.

2.3 ReLU

2.3.1 概要

Rectified Linear Unit (ReLU) は, 入力値を x, 出力値 y として, 以下のように定義される活性化関数である (グラフは図 2.2).

$$y = \begin{cases} 0 & (x < 0) \\ x & (0 \leq x). \end{cases} \tag{2.26}$$

この関数は数学的には微分可能ではないが, 活性化関数の文脈において導関数は

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \begin{cases} 0 & (x < 0) \\ 1 & (0 \leq x) \end{cases} \tag{2.27}$$

と定義されている.

ReLU は $0 \leq x$ において線形関数になっている点でその他の多くの活性化関数とは異なっている. さらに ReLU の導関数が $0 \leq x$ において定数となり, 学習が進む中でも $0 \leq x$ については勾配が減衰しないため, ディープニューラルネットワークを誤差逆伝播法によって学習させるのに適していると考えられている [13]. 実際 ILSVRC 2014 において優勝したチームは活性化関数に ReLU のみを用いた畳み込みニューラルネットワークを使用しており [16], その有用性がうかがえる.

2.3.2 ReLU の改善

ReLU は $x < 0$ において定数 0 を返すため, 負方向の情報が完全に消失してしまうという特性がある. この点を改善するため, Maas らは文献 [14] において下式で表される Leaky ReLU

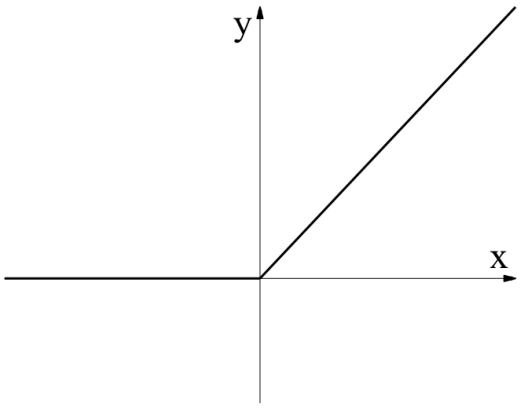

図 2.2. ReLU

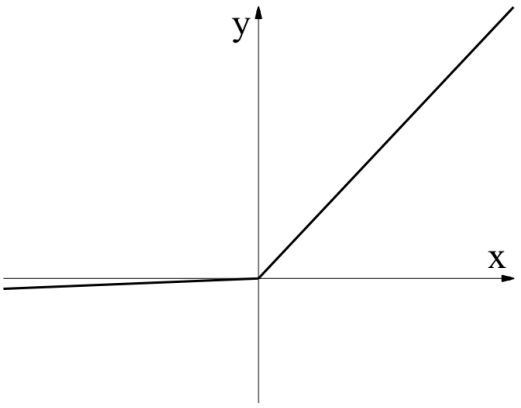

図 2.3. LReLU

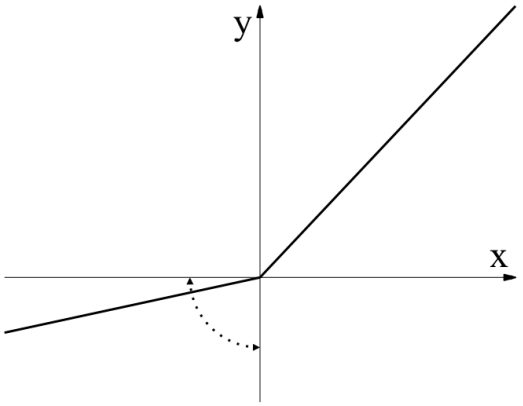

図 2.4. PReLU

(LReLU) を考案している (グラフは図 2.3).

$$y = \begin{cases} 0.01x & (x < 0) \\ x & (0 \leq x). \end{cases} \tag{2.28}$$

しかしながら同文献における実験では, LReLU を用いて得られた結果は ReLU を用いた場合に比べほとんど変化がなかったとしている.

一方 He らは LReLU の考え方をさらに発展させ以下の式で表される Parametric ReLU (PReLU)[5] を考案している (グラフは図 2.4).

$$y = \begin{cases} ax & (x < 0) \\ x & (0 \leq x). \end{cases} \tag{2.29}$$

ここで a は誤差逆伝播法による学習で最適化される値とされている. 前述のように LReLU はほとんど性能に影響を与えなかったとされたものの, He らは研究の中で PReLU を用いることで性能が大幅に改善されたとしている.

2.4 Xavier Initialization

2.4.1 概要

Glorot らは文献 [3] において, 取り扱うデータとは独立した形のパラメータ初期化手法を考えるために, 特にネットワーク初期段階での各層の入出力 i_n, o_n の分散に注目した. Glorot らはまず, 順伝播と誤差逆伝播の過程で情報の流れを維持するため (to keep information flowing) に, 任意の n, n' に対して

$$Var[i_n] = Var[i_{n'}], \tag{2.30}$$

$$Var[\frac{\partial E}{\partial b_n}] = Var[\frac{\partial E}{\partial b_{n'}}] \tag{2.31}$$

が成立することが重要であるとしている. 式 2.1, 2.2, 2.7 から,

$$\begin{aligned} Var[i_{n+1}] &= Var[\sum_{l'} w_{l'}^n o_{l'}^n], \\ &= Var[\sum_{l'} w_{l'}^n f_n(i_{l'}^n + b_{l'}^n)], \end{aligned} \tag{2.32}$$

$$Var[\frac{\partial E}{\partial b_n}] = Var[f_n'(i_n + b_n) \sum_{l'} w_{l'}^n \frac{\partial E}{\partial b_{l'}^{n+1}}] \tag{2.33}$$

であり, 第 n 層の人工ニューロン数を M_n などと表すと, 分散の加法性から,

$$Var[i_{n+1}] = M_n Var[w_n f_n(i_n + b_n)], \tag{2.34}$$

$$Var[\frac{\partial E}{\partial b_n}] = M_{n+1} Var[f_n'(i_n + b_n) w_n \frac{\partial E}{\partial b_{n+1}}] \tag{2.35}$$

となる. 前述のように w_n は各向きに伝播する値 o_n, $\frac{\partial E}{\partial b_{n+1}}$ とは独立したものを考えており, ここでは特に w_n が対称的な分布を持つとする. すると

$$Var[i_{n+1}] = M_n Var[w_n] E[\{f_n(i_n + b_n)\}^2], \tag{2.36}$$

$$Var[\frac{\partial E}{\partial b_n}] = M_{n+1} Var[w_n] E[\{f_n'(i_n + b_n) \frac{\partial E}{\partial b_{n+1}}\}^2]. \tag{2.37}$$

また, バイアス値の初期化は $b_n = 0$ と行うとすると,

$$Var[i_{n+1}] = M_n Var[w_n] E[\{f_n(i_n)\}^2], \tag{2.38}$$

$$Var[\frac{\partial E}{\partial b_n}] = M_{n+1} Var[w_n] E[\{f_n'(i_n) \frac{\partial E}{\partial b_{n+1}}\}^2]. \tag{2.39}$$

ここで f_n が線形領域にある, つまり

$$f_n(x) \approx x \tag{2.40}$$

と考えると

$$Var[i_{n+1}] = M_n Var[w_n] E[(i_n)^2], \tag{2.41}$$

$$Var[\frac{\partial E}{\partial b_n}] = M_{n+1}Var[w_n]E[(\frac{\partial E}{\partial b_{n+1}})^2]. \tag{2.42}$$

w_n が対称的な分布を持つ場合には, 式 2.1, 2.7 より, i_n, $\frac{\partial E}{\partial b_{n+1}}$ も対称的な分布を持つことに注意すれば,

$$Var[i_{n+1}] = M_n Var[w_n]Var[i_n], \tag{2.43}$$

$$Var[\frac{\partial E}{\partial b_n}] = M_{n+1}Var[w_n]Var[\frac{\partial E}{\partial b_{n+1}}]. \tag{2.44}$$

よって式 2.43, 2.44 から, 式 2.30, 2.31 を満たす条件として

$$M_n Var(w_n) = 1, \tag{2.45}$$

$$M_{n+1}Var(w_n) = 1 \tag{2.46}$$

を得る. つまり式 2.45, 2.46 を満たすように各結合係数 w_{kl}^n を生成することで式 2.30, 2.31 が満たされ, 各方向の伝播において分散が維持されることとなる. 逆にこの条件を満たさない場合には分散が層の数について指数的に増大または減少し, ディープニューラルネットワークの学習を阻害することになるとされている.

最終的に Glorot らは, 式 2.45, 2.46 の両者を近似的に満たすものとして次のような一様分布によって結合係数を初期化することを提案している (初期バイアス値は全て 0).

$$w_n \sim U(-\sqrt{\frac{6}{M_n + M_{n+1}}}, \sqrt{\frac{6}{M_n + M_{n+1}}}). \tag{2.47}$$

実際 Glorot らはこの初期化手法 (Xavier Initialization) を用いることで, 経験則的に知られていた初期化手法

$$w_n \sim U(-\sqrt{\frac{1}{M_n}}, \sqrt{\frac{1}{M_n}}) \tag{2.48}$$

に比べて学習の進み方が改善したことを報告している.

Xavier Initialization は現在利用されることの多いディープニューラルネットワークのためのライブラリ Caffe 1.0 では標準で

$$w_n \sim U(-\sqrt{\frac{3}{M_n}}, \sqrt{\frac{3}{M_n}}) \tag{2.49}$$

と定義されている (オプションとして指定することで式 2.47 の初期化も利用できる). これは式 2.45 のみを満たすものであるため, 厳密には Xavier Initialization ではないと考えられるが, ディープニューラルネットワークのライブラリでは Xavier Initialization は単に式 2.45, 2.46 の少なくとも一方を満たすものを指していることもある.

2.4.2　He Initialization

前節で示した Xavier Initialization の導出において活性化関数について用いた仮定 2.40 は, 双曲線正接関数やソフトサイン関数などの多くの活性化関数に対して成立する. しかしながら

ReLU においては $x < 0$ についてこの仮定が成り立たない．そのため He らは活性化関数が ReLU である場合には以下のようにして得られる初期化手法を用いることを提案している．

i_n, $\frac{\partial E}{\partial b_{n+1}}$ の分布の対称性に注意すれば

$$\begin{aligned} E[\{f_n(i_n)\}^2] &= \frac{1}{2}E[(i_n)^2], \\ &= \frac{1}{2}Var[i_n], \end{aligned} \tag{2.50}$$

$$\begin{aligned} E[\{f_n'(i_n)\frac{\partial E}{\partial b_{n+1}}\}^2] &= \frac{1}{2}E[\{\frac{\partial E}{\partial b_{n+1}}\}^2], \\ &= \frac{1}{2}Var[\frac{\partial E}{\partial b_{n+1}}] \end{aligned} \tag{2.51}$$

であるから式 2.38, 2.39 は

$$Var[i_{n+1}] = \frac{1}{2}M_n Var[w_n]Var[i_n], \tag{2.52}$$

$$Var[\frac{\partial E}{\partial b_n}] = \frac{1}{2}M_{n+1}Var[w_n]Var[\frac{\partial E}{\partial b_{n+1}}] \tag{2.53}$$

となる．よって ReLU を用いる場合には

$$\frac{1}{2}M_n Var(w_n) = 1, \tag{2.54}$$

$$\frac{1}{2}M_{n+1}Var(w_n) = 1 \tag{2.55}$$

という条件が導出される．He らはこれらの条件から最終的に, 以下のような正規分布による結合係数の初期化方法を提案している (He Initialization と呼ぶ).

$$w_n \sim N(0, \sqrt{\frac{2}{M_n}}). \tag{2.56}$$

文献 [5] における He らの実験では, ReLU を用いる場合には He Initialization を用いることで, Xavier Initialization に比べて大幅に学習速度が改善されたということを報告している．

第 3 章

Xavier Initialization の改善

3.1 バイアス値初期化手法の検討

Xavier Initialization, He Initialization では初期バイアス値は全て 0 と設定することが提案されている. 本研究では, 初期バイアス値が全て 0 の場合以外について考察し, He Initialization とは異なる形での, ReLU を用いる場合の Xavier Initialization の改善を試みた.

まず始めに He Initialization について定性的な考察を行う. 式 2.45, 2.46 と式 2.54, 2.55 から分かるように, He Initialization は Xavier Initialization で用いる分散の 2 倍の分散を持つように結合係数を初期化している. 活性化関数が ReLU である場合には負方向の入力は出力に反映されず 0 となるため情報が半減していると考えると, He Initialization は残る半数の正方向の情報を 2 倍の分散で増幅することで情報の流れを維持していると見ることができる (図 3.1, 3.2).

同様に情報の流れの観点から, 初期バイアス値が全て 0 ではない場合について, 特にその値の正負に注目して考える. b_n が負であるとき, バイアス値が 0 であった場合に次層に伝播するはずであった情報も消失することとなる. このことを考えると b_n が負であるのは有意義な方法ではないと予想される. 一方 b_n が正である場合には, バイアス値が 0 であった場合に消失するはずだった情報をかさ上げすることとなり, 結果的に次層へより多くの情報を伝播できると予想される (図 3.3). このような考察から本研究ではバイアス値が正であるものについて考えることとした. 以下では表現の短縮のため, この方法を Xavier B+ Initialization と呼ぶ.

上述の考察で得た Xavier B+ Initialization について, 各方向への伝播における分散の変化を考える. $i_n + b_n$ が正となる割合によって定まる r_n ($\frac{1}{2} \leq r_n \leq 1$) を用いると

$$\begin{aligned} E[\{f_n(i_n + b_n)\}^2] &= r_n E[(i_n + b_n)^2], \\ &= r_n\{Var[i_n] + Var[b_n] + (E[b_n])^2\}, \end{aligned} \tag{3.1}$$

$$E[\{f_n'(i_n + b_n)\frac{\partial E}{\partial b_{n+1}}\}^2] = r_n E[(\frac{\partial E}{\partial b_{n+1}})^2] \tag{3.2}$$

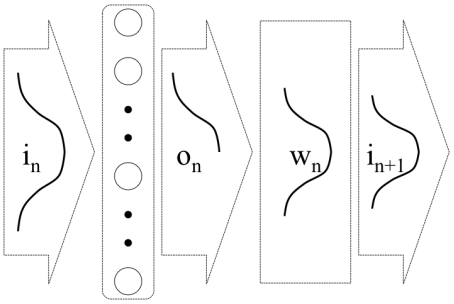

図 3.1. Xavier Initialization での分布の変遷

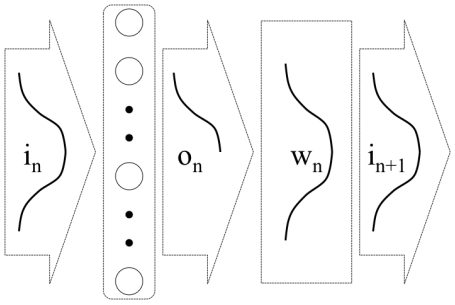

図 3.2. He Initialization での分布の変遷

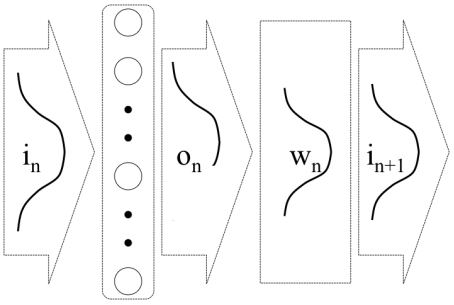

図 3.3. Xavier B+ Initialization での分布の変遷

と表せるから, 各方向への伝播における分散は

$$Var[i_{n+1}] = r_n M_n Var[w_n]\{Var[i_n] + Var[b_n] + (E[b_n])^2\}, \tag{3.3}$$

$$Var[\frac{\partial E}{\partial b_n}] = r_n M_{n+1} Var[w_n] Var[\frac{\partial E}{\partial b_{n+1}}] \tag{3.4}$$

という関係を持つ. ここで結合係数の初期化には式 2.45, 2.46 を満たすものを考えるので,

$$Var[i_{n+1}] = r_n\{Var[i_n] + Var[b_n] + (E[b_n])^2\}, \tag{3.5}$$

$$Var[\frac{\partial E}{\partial b_n}] = r_n Var[\frac{\partial E}{\partial b_{n+1}}]. \tag{3.6}$$

Xavier Initialization では $Var[b_n] = E[b_n] = 0$, $r_n = \frac{1}{2}$ であるから

$$Var[i_{n+1}] = \frac{1}{2} Var[i_n], \tag{3.7}$$

$$Var[\frac{\partial E}{\partial b_n}] = \frac{1}{2} Var[\frac{\partial E}{\partial b_{n+1}}] \tag{3.8}$$

となることと, $\frac{1}{2} \leq r_n \leq 1$ であることを考えると, Xavier B+ Initialization は活性化関数が ReLU の場合には Xavier Initialization に比べると良い方法であると予想される. 次節以降では, 上述の考察により得た Xavier B+ Initialization の効果について実験を行った.

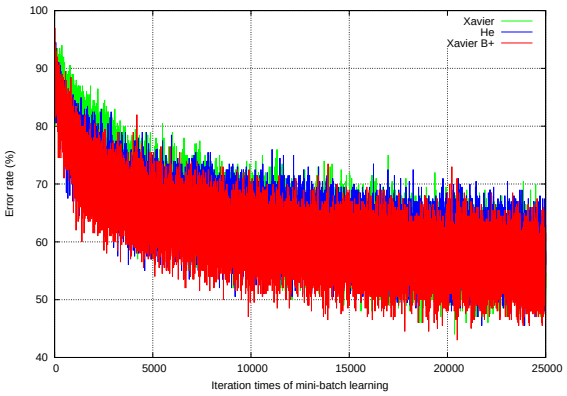

図 3.4. 移動平均を用いない場合

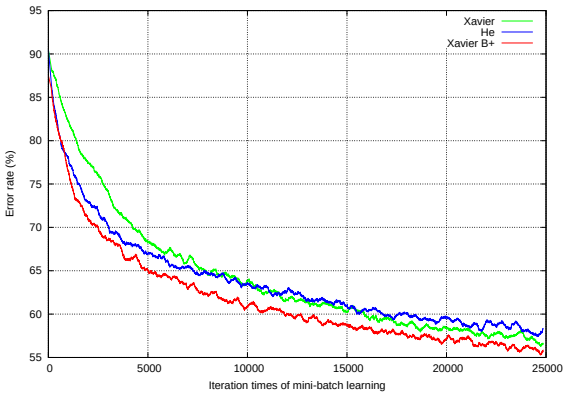

図 3.5. 移動平均を用いた場合

3.2　実験概要

Xavier B+ Initialization を検証するために, 深さと隠れ層の人工ニューロン数が異なる幾つかの多層ニューラルネットワークを用いて画像データセット CIFAR-10 に対して画像分類タスクを実行し, その誤答率を記録する実験を行った. 各実験では共通して隠れ層の活性化関数に ReLU, 出力層の活性化関数にソフトマックス関数, 誤差関数に交差エントロピー関数を使用している. また学習についても共通し, サイズ 10 のミニバッチによる確率的勾配降下法を用いている.

本実験では, 各初期化手法ごとの誤答率の変化を比較するため, 各ミニバッチ学習を完了するごとにテストデータ 10000 枚の中からからランダムに 100 枚抽出した画像の分類タスクをテストとして実行した. しかしながらこのミニバッチテストは全テストデータの 1% でしかなく, 記録される誤答率をそのままプロットすると図 3.4 のようになり, 実験の結果が読み取りにくい. そのため得られた誤答率の移動平均 (窓サイズ 300) をとったもの (図 3.5) を実験結果として次節で示す.

パラメータ初期化手法には Xavier Initialization, He Initialization, Xavier B+ Initialization の 3 つに加え, 出力層側の 5 層については Xavier B+ Initialization を用い, 残る入力層側については He Initialization を用いた混成手法 (以下表記短縮のために Joined Initialization と呼ぶ) の計 4 つを使用した. また, 本実験においては Xavier B+ Initialization を, バイアス値を一様分布に従って生成する方法

$$b_n \sim U(0, 0.5) \tag{3.9}$$

と設定している.

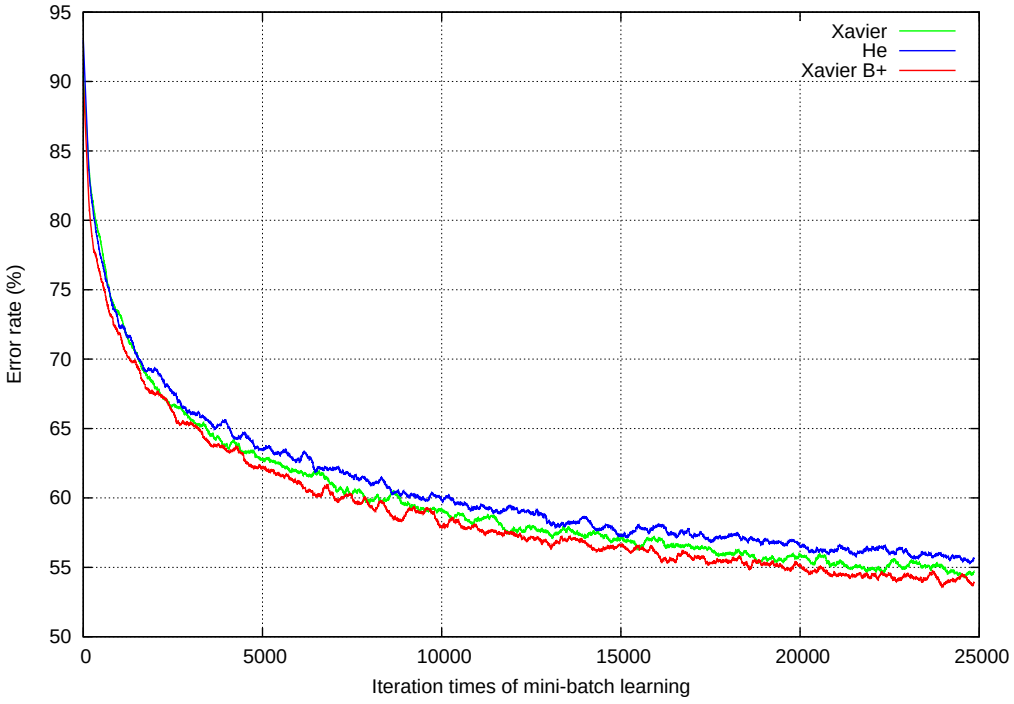

図 3.6. N=5, M=100

3.3 実験結果

実験結果は図 3.6 から 3.12 のようになった. 但し図のキャプションにおける N は隠れ層数, M は隠れ層の人工ニューロン数を表す. またキャプション中に AdaGrad と表記のないものは学習係数を 0.01 に固定して実験を行っている.

3.4 考察

3.4.1 隠れ層が 15 層以下の結果について

図 3.6 から図 3.8 が対応する実験結果となる. まず Xavier Initialization と He Initialization の違いに注目する. He Initialization は層数が多いときほど学習の進みが Xavier Initialization に比べて良いことがわかり, 特に学習回数 0 から 5000 程度の範囲においてその差は大きい. 一方で学習回数 20000 から 25000 程度の範囲では Xavier Initialization とほぼ同一か, それより劣る結果となっている.

次に Xavier B+ Initialization について注目する. Xavier B+ Initialization はいずれの実験においても Xavier Initialization を優る結果となっており, バイアス値の初期化手法を変更する方法によっても Xavier Initialization を改善できることが示された. 特に隠れ層が 15 層以下においては He Initialization にも優る結果になっていることが観察された.

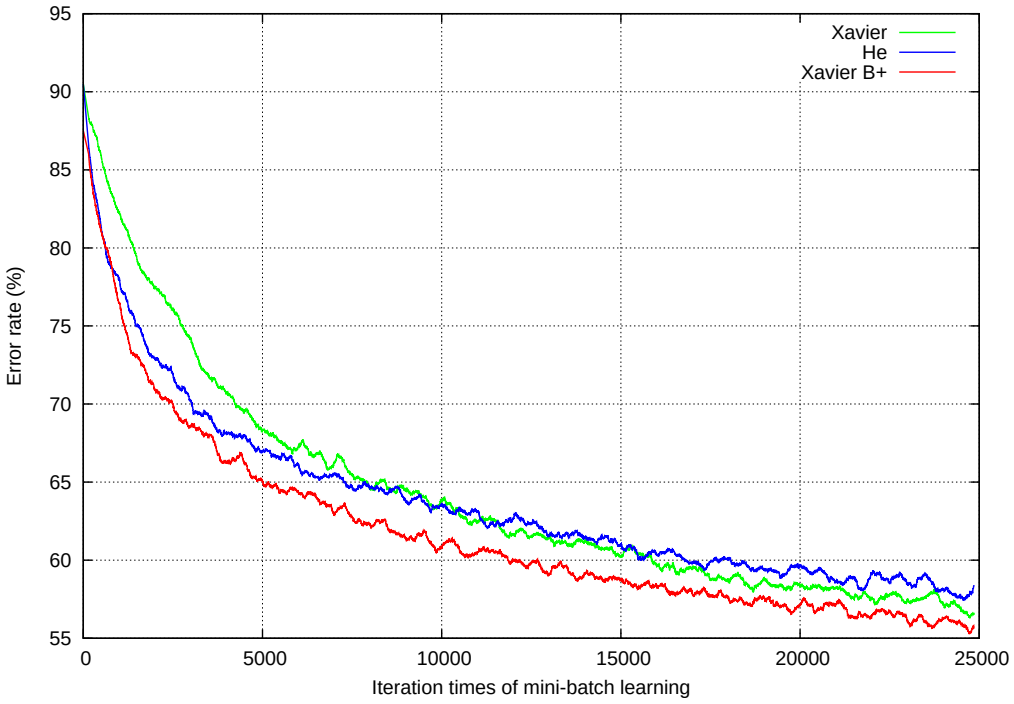

図 3.7. N=10, M=100

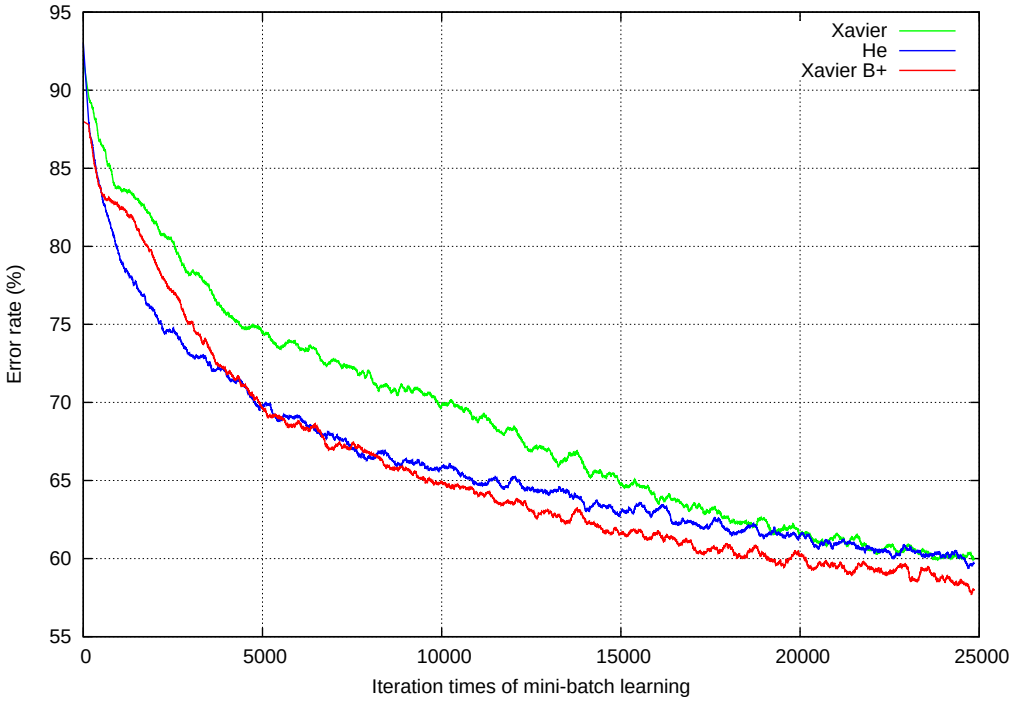

図 3.8. N=15, M=100

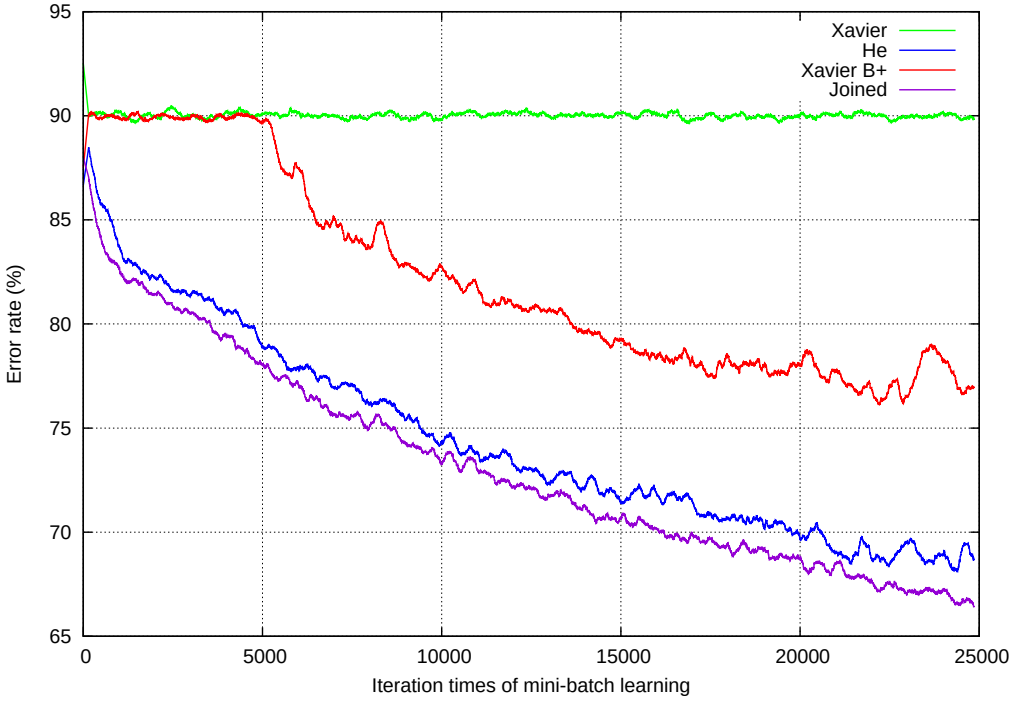

図 3.9. N=30, M=100

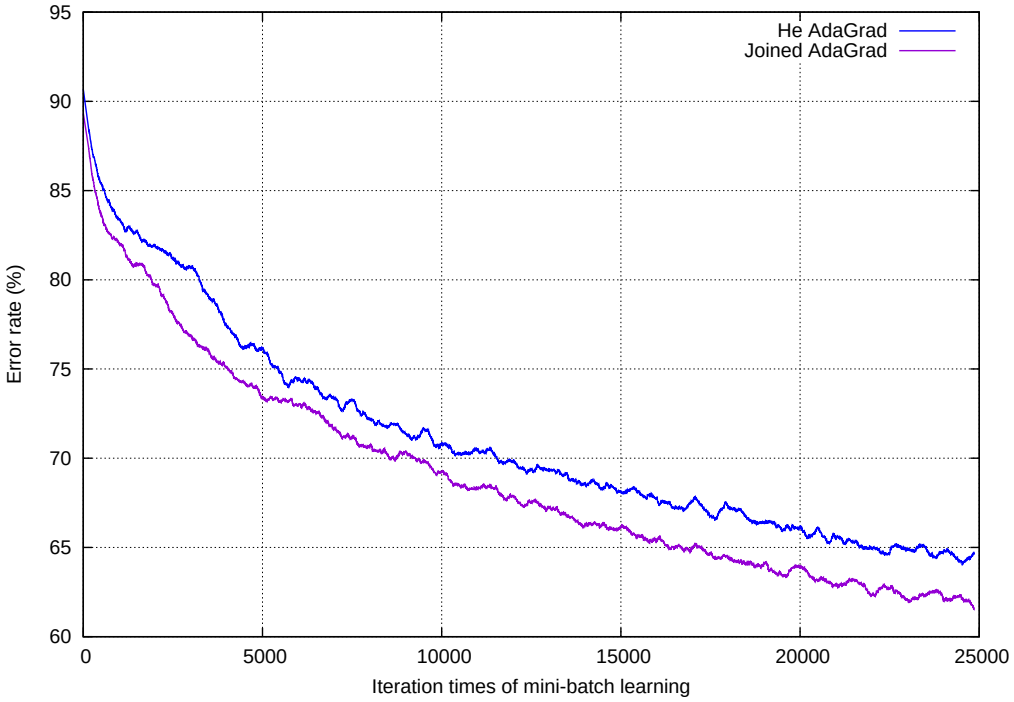

図 3.10. N=30, M=100, AdaGrad

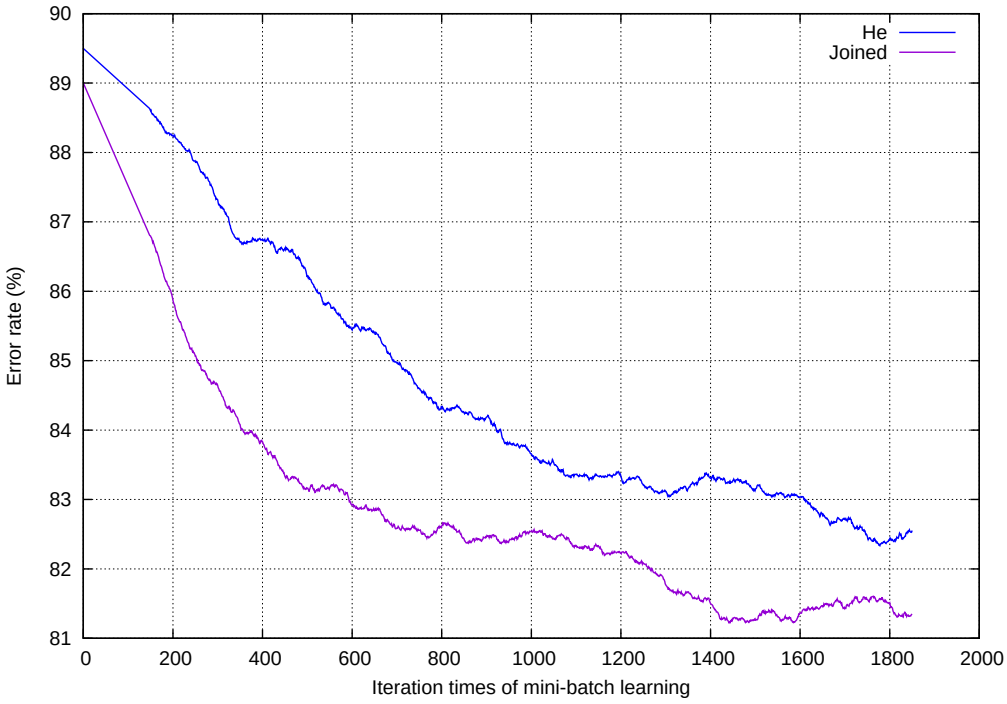

図 3.11. N=30, M=200

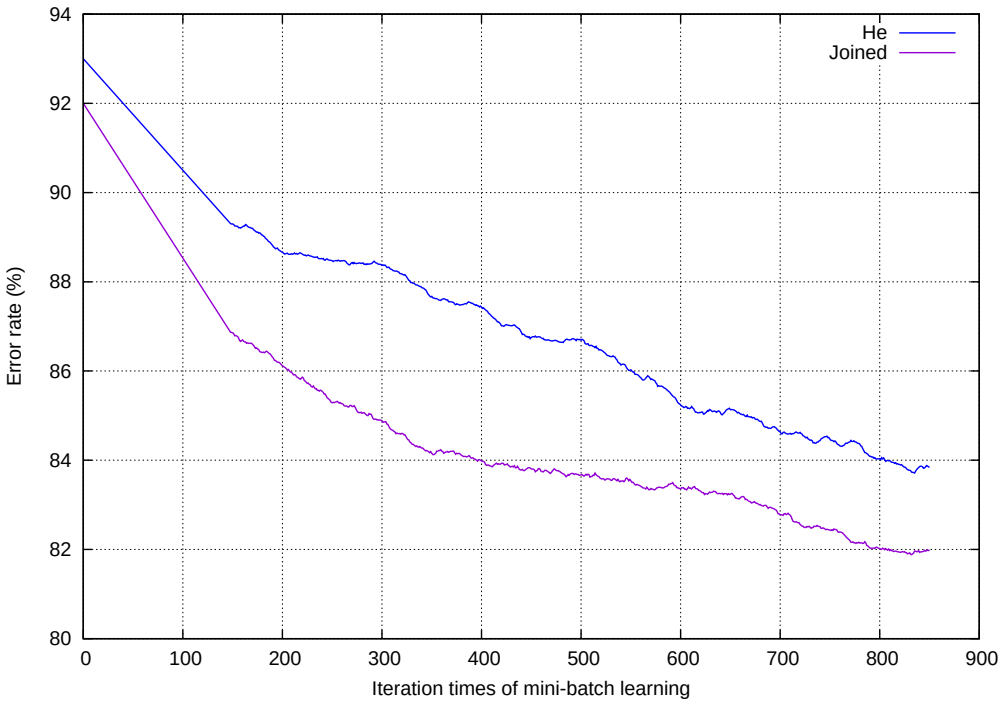

図 3.12. N=30, M=500

また He Initialization を用いた場合には初期の学習は早く進む一方, ある程度学習が進んだ段階においては学習が進みにくくなっていることも観察される. ここまでで議論していた各方向への伝播における分散の考察は学習の初期段階における状態を想定しており, 学習がある程度進んだ状態については同様の議論をそのまま適用できないと考えられ, 要因の分析は難しい. そのためある程度学習が進んだ状態でどのようなことが起こっているのか調査することは今後の課題の 1 つと考えられる.

3.4.2 隠れ層が 30 層の結果について

図 3.9 から図 3.12 が対応する実験結果となる. 但し $M = 200$, $M = 500$ については, 計算機リソースの制約上他の実験結果に比べて学習回数が少ない結果となっている.

まず図 3.9 から, He らが文献 [5] において示した結果と同様に, Xavier Initialization を用いると全く学習が進まないという状態が観察された. He Initialization はこの場合にも学習を進めることができており, その改善の効果が確認される. 一方 Xavier B+ Initialization は, He Initialization には劣るものの, 学習回数 5000 程度以降から学習を進めることができている. この結果からも Xavier Initialization が He Initialization とは異なる形で改善できると示された.

前述のように隠れ層が 30 層となる場合には, Xavier B+ Initialization が He Initialization を上回らなかったものの, 15 層以下の場合には Xavier B+ Initialization が上回る結果を得ている. そこで層数が少ない場合には Xavier B+ Initialization が優れているという予想から, 先述した Joined Initialization を考え, 同様の実験を行った.

Joined Initialization と He Initialization の比較については AdaGrad を用いた場合, 隠れ層の人工ニューロン数を増大させた場合についても実験を行うことでその効果を確認した. 図 3.9 から図 3.12 から分かるように, 本実験で行った全ての場合について Joined Initialization は He Initialization より優れた結果を示している. この結果から He Initialization はさらに改善できるのではないかと予想されるが, Xavier B+ Initialization の持つ効果について非常に定性的な分析しかできていないため, 今後さらに様々な実験と解析を行う必要があると考えられる.

第 4 章

複素数ニューラルネットワークでのパラメータ初期化手法

4.1 Xavier Initialization の修正

先述の通り, 今後様々なデータセットへの応用から高次元数のディープニューラルネットワークの需要が高まることが予想され, その際には Xavier Initialization のような軽量な初期化手法は重要になると考えられる. そこでまず, 以下では Glorot らの導出に倣って高次元数における Xavier Initialization を考える.

式 2.16 より

$$Var[i_{n+1}] = Var[\sum_{k',q',r'} w^n_{k'q'} o^n_{k'r'} s_{q'r'}]. \tag{4.1}$$

第 n 層の複素ニューロン数を M_n とすれば, 分散の加法性から

$$Var[i_{n+1}] = M_n(\sum_{q',r'} Var[w^n_{q'} o^n_{r'} s_{q'r'}]). \tag{4.2}$$

ここで $s_{q'r'}$ に注目する. $s_{q'r'} \in \{-1, 0, 1\}$ であるから

$$\begin{aligned} Var[i_{n+1}] &= M_n(\sum_{q',r',s_{q'r'}=0} Var[0] + \sum_{q',r',s_{q'r'}=1} Var[w^n_{q'} o^n_{r'}] + \sum_{q',r',s_{q'r'}=-1} Var[-w^n_{q'} o^n_{r'}]), \\ &= M_n(\sum_{q',r',s_{q'r'}\neq 0} Var[w^n_{q'} o^n_{r'}]). \end{aligned} \tag{4.3}$$

$s_{q',r'} \neq 0$ である (q', r') の組は各 p について共通して P 個ある (表 2.1, 表 2.2 など) ことを考えると,

$$Var[i_{n+1}] = PM_n Var[w_n o_n]. \tag{4.4}$$

さらにバイアス初期値は全て 0 とし, 活性化関数が線形領域にあると仮定すると

$$Var[i_{n+1}] = PM_n Var[w_n] Var[i_n]. \tag{4.5}$$

同様にして逆伝播について考えると,

$$Var[\frac{\partial E}{\partial b_n}] = PM_{n+1}Var[w_n]Var[\frac{\partial E}{\partial b_{n+1}}] \tag{4.6}$$

を得る.

最後に式 4.5, 式 4.6 の条件に合わせ, 実数における Xavier Initialization (式 2.47) を P 次元数に合わせて修正すると,

$$w_n \sim U(-\sqrt{\frac{6}{P(M_n + M_{n+1})}}, \sqrt{\frac{6}{P(M_n + M_{n+1})}}). \tag{4.7}$$

以下では本実験に合わせて式 4.7 で $P = 2$ とした

$$w_n \sim U(-\sqrt{\frac{3}{M_n + M_{n+1}}}, \sqrt{\frac{3}{M_n + M_{n+1}}}) \tag{4.8}$$

を Complex Xavier Initialization と呼ぶことにする. これに合わせ, 前章で実験を行った Xavier B+ Initialization の結合係数の初期化に Complex Xavier Initialization を用いるものを Complex Xavier B+ Initialization と呼ぶことにする. ただし本実験でも初期バイアス値の生成に式 3.9 を用いている.

4.2 He Initialization の修正

前節における Xavier Initialization の修正と同様に考えることで活性化関数に ReLU を用いる場合の初期化手法である He Initialization (式 2.56) は次のように修正される.

$$w_n \sim N(0, \sqrt{\frac{2}{PM_n}}). \tag{4.9}$$

以下では Xavier Initializtion の場合と同様に, 式 4.9 で $P = 2$ とした

$$w_n \sim N(0, \sqrt{\frac{1}{M_n}}) \tag{4.10}$$

を Complex He Initialization と呼ぶことにする. さらにこれに合わせ, 前章で実験を行った Joined Initialization の入力層側, 出力層側をそれぞれ Complex He Initialization, Complex Xavier B+ Initialization としたものを Complex Joined Initialization と呼ぶことにする.

4.3 実験概要

P 次元数における Xavier Initialization, He Initialization の修正 (式 4.7, 4.9) が妥当であるか検証するため, 複素ニューロンモデルから成る多層ニューラルネットワークを用いて, 画像データセット MNIST の各画像をフーリエ変換したデータについて分類タスクを行った. 各実験では共通して出力層の活性化関数にソフトマックス関数, 誤差関数に交差エントロピー関

数を用い, 学習はサイズ 10 のミニバッチによる確率的勾配降下法によって行った. また各ミニバッチ学習が完了するたびにサイズ 100 のミニバッチテストを実行し, 誤答率の記録を行った.

本実験では先述した目的を複素数ニューラルネットワークにおいて検証するため, 隠れ層の活性化関数が双曲線正接関数の場合には初期化手法に Xavier Initialization, Complex Xavier Initialization の計 2 つを使用し, ReLU の場合には Complex Xavier Initialization, Complex He Initialization, Complex Xavier B+ Initialization, Complex Joined Initialization の計 4 つを用いた.

また本実験では行うのは画像分類タスクであるが, 複素数ニューラルネットワークでは出力も複素数となるため, このタスクにおける教師信号や, ネットワークの出力値がどのカテゴリを指しているとみなすかということは自明ではない. そこで本実験では, 今回用いる教師信号を $\mathbf{t}'$, 実数において用いられる教師信号を $\mathbf{t}$ として,

$$\mathbf{t}' = \mathbf{t} + \mathrm{i}\mathbf{t} \tag{4.11}$$

と定め, 出力値の実数成分と複素数成分の和が最大のものをネットワークの出力したカテゴリとみなした. 本実験は各初期化手法の比較を目的としているため, このように定めても大きな問題は起きないと考えられる.

4.4　実験結果

実験結果は図 4.1 から図 4.6 のようになった. 各図は前章の実験と同様に, 得られた誤答率の移動平均 (窓サイズ 50) をとったものであり, 図のキャプションにおける N は隠れ層数, M は隠れ層の複素ニューロン数, f は使用した隠れ層の活性化関数を表す. また図のキャプションに AdaGrad と表記のないものは学習係数を 0.01 に固定して実験を行っている.

4.5　考察

4.5.1　活性化関数が双曲線正接関数の結果について

図 4.1 から図 4.4 より分かる通り, 隠れ層の活性化関数が双曲線正接関数を用いた場合には Complex Xavier Initialization が Xavier Initialization よりも優れた結果となり, 高次元数に合わせた修正式 4.7 は適切な修正であったと考えられる. また深いほど Xavier Initialization と Complex Xavier Initialization の間に差が表れていることも図 4.1 から図 4.3 より読み取ることができ, 今後需要が増えると考えられるディープニューラルネットワークの初期化手法として非常に有用であると考えられる.

図 4.7 は, AdaGrad を用いるかどうか以外は共通の条件であるときの結果 (図 4.3 と図 4.5) を一つにまとめたものである. 図 4.7 を見ると, Complex Xavier Initialization によって初期化することで AdaGrad を 用いた Xavier Initialization と同程度の結果を最終的に得られている. 図 4.7 の結果はミニバッチ学習回数が 300 回のものであるため, 300 回以上については AdaGrad を用いた Xavier Initialization が優ると予想されるものの, 少なくとも複素数の

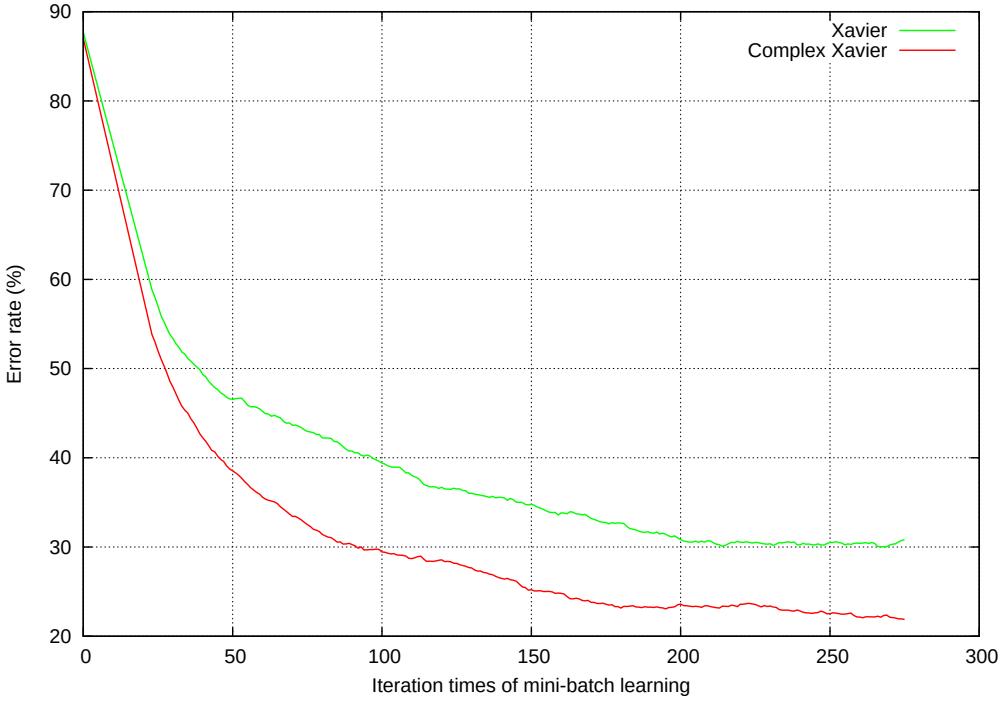

図 4.1. $N = 10$, M=100, f=双曲線正接関数

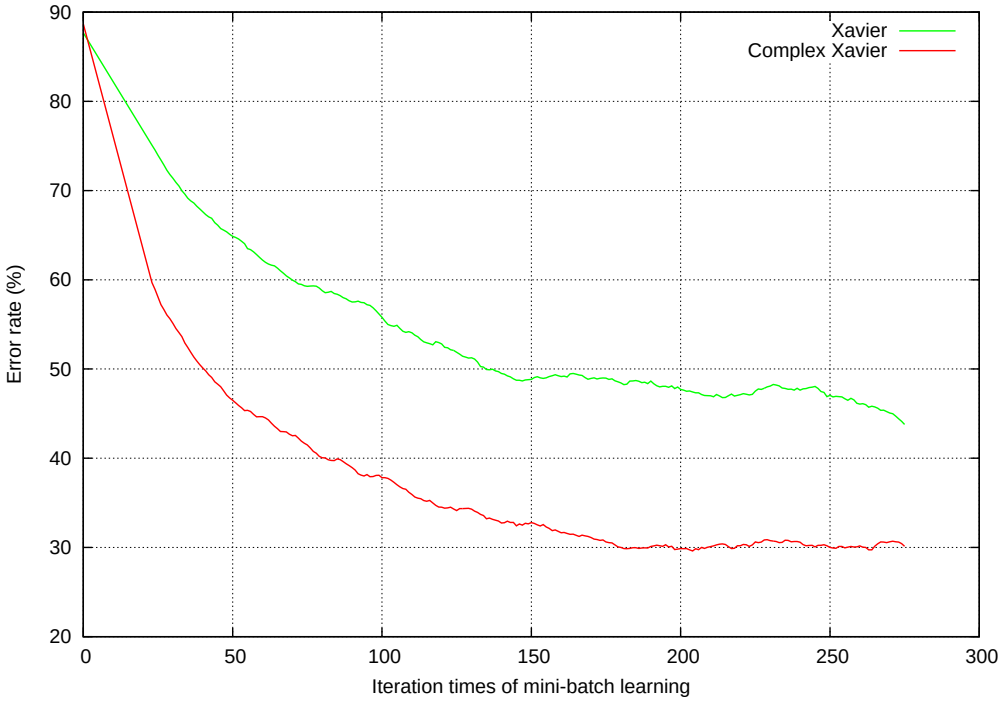

図 4.2. $N = 20$, M=100, f=双曲線正接関数

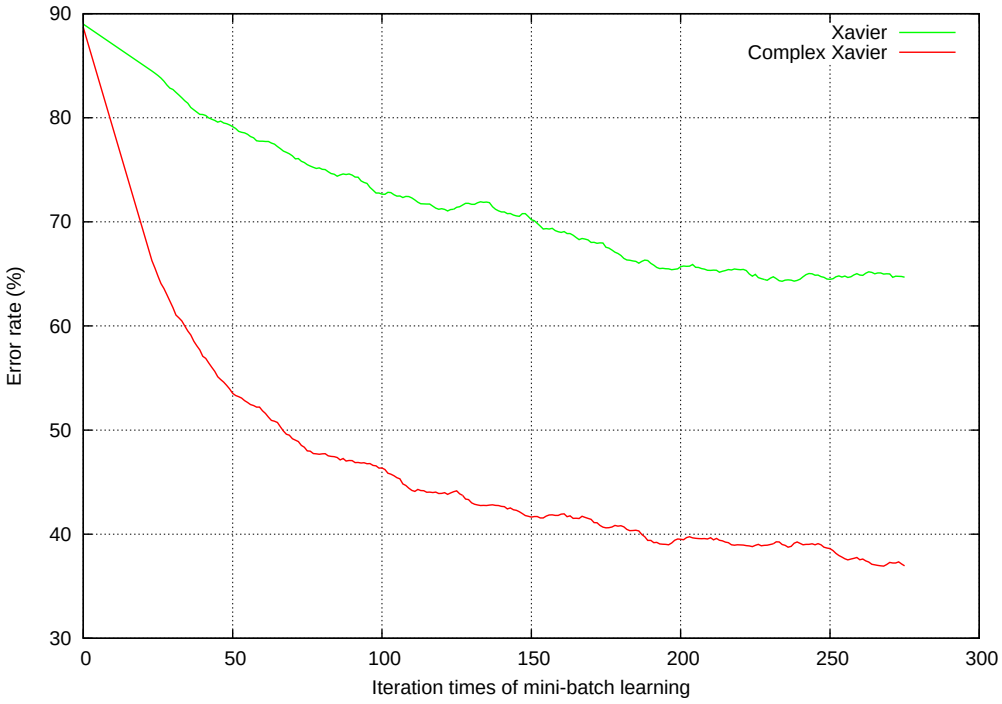

図 4.3. $N = 30$, M=100, f=双曲線正接関数

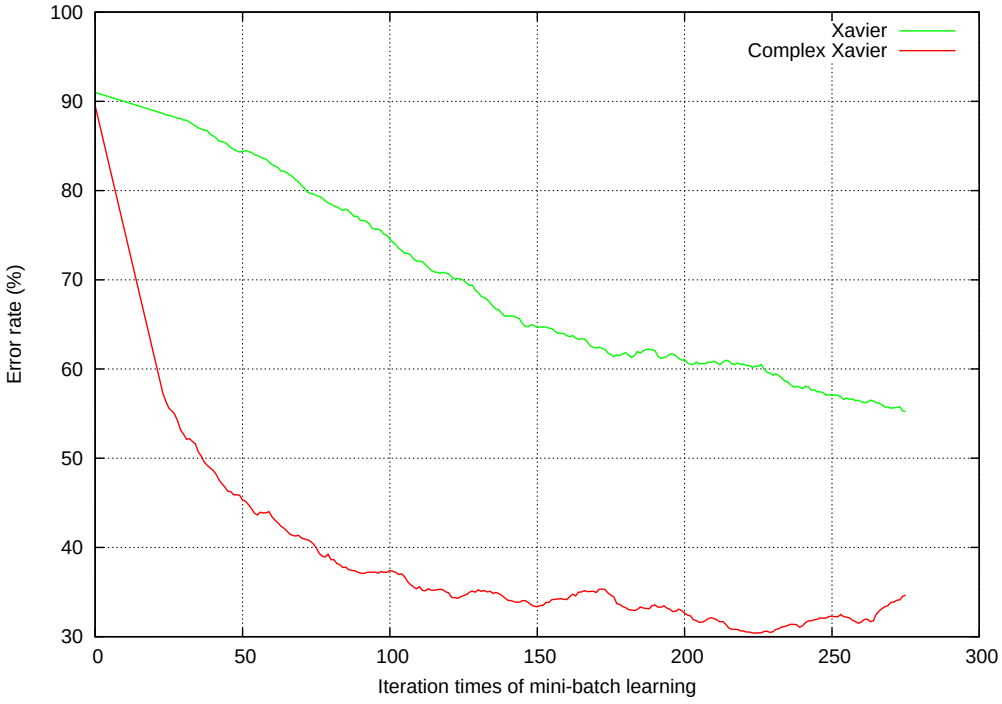

図 4.4. $N = 30$, M=200, f=双曲線正接関数

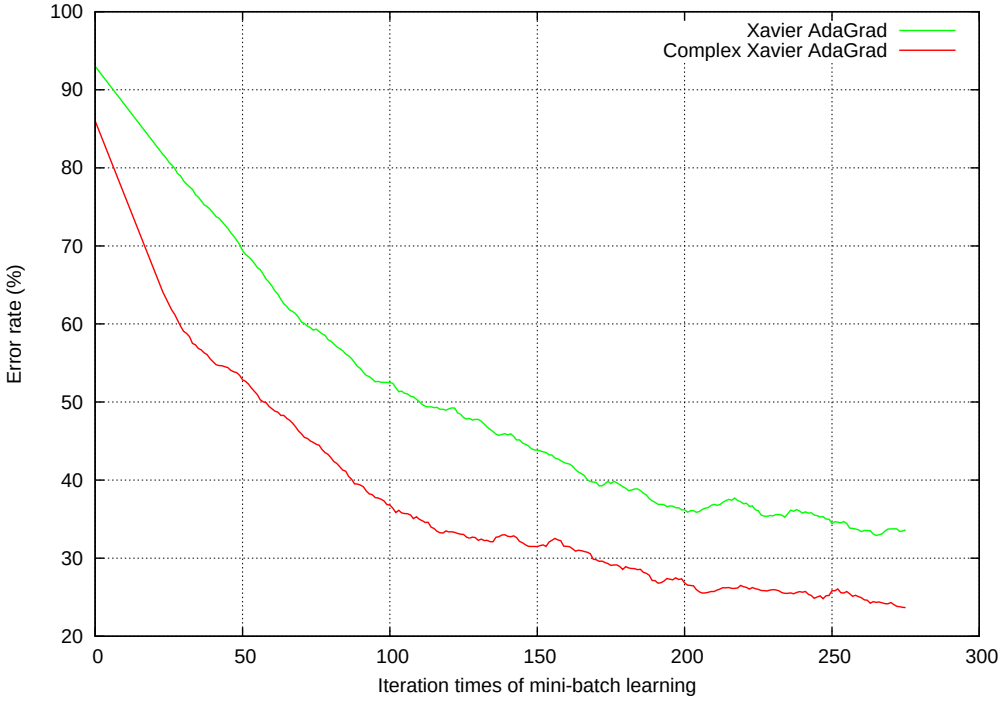

図 4.5. $N = 30$, M=100, f=双曲線正接関数, AdaGrad

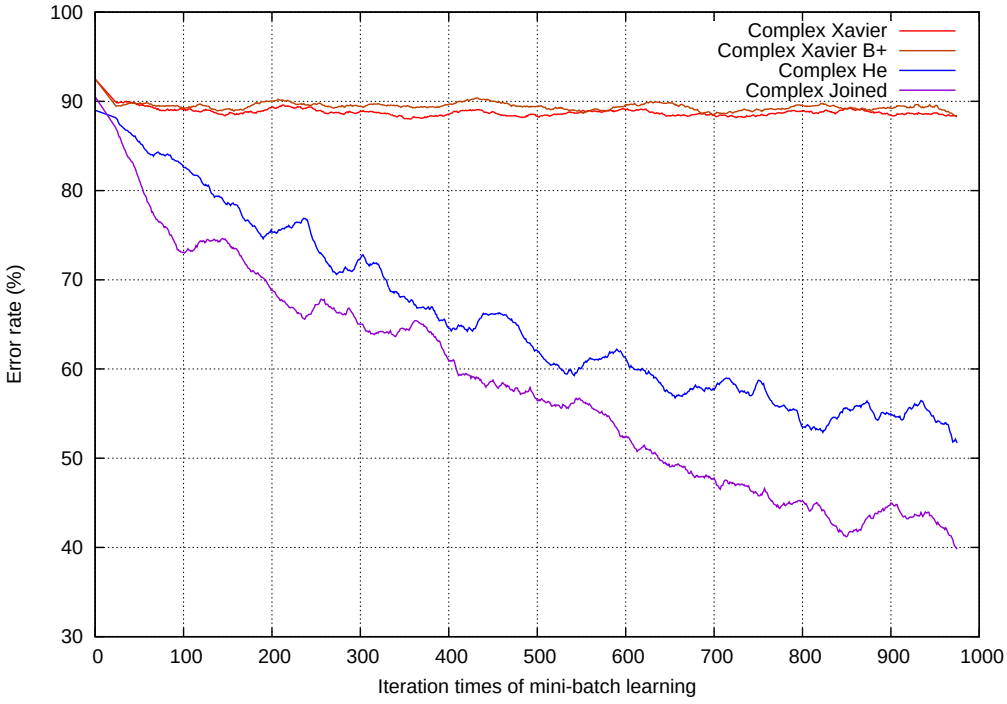

図 4.6. $N = 30$, M=100, f=ReLU

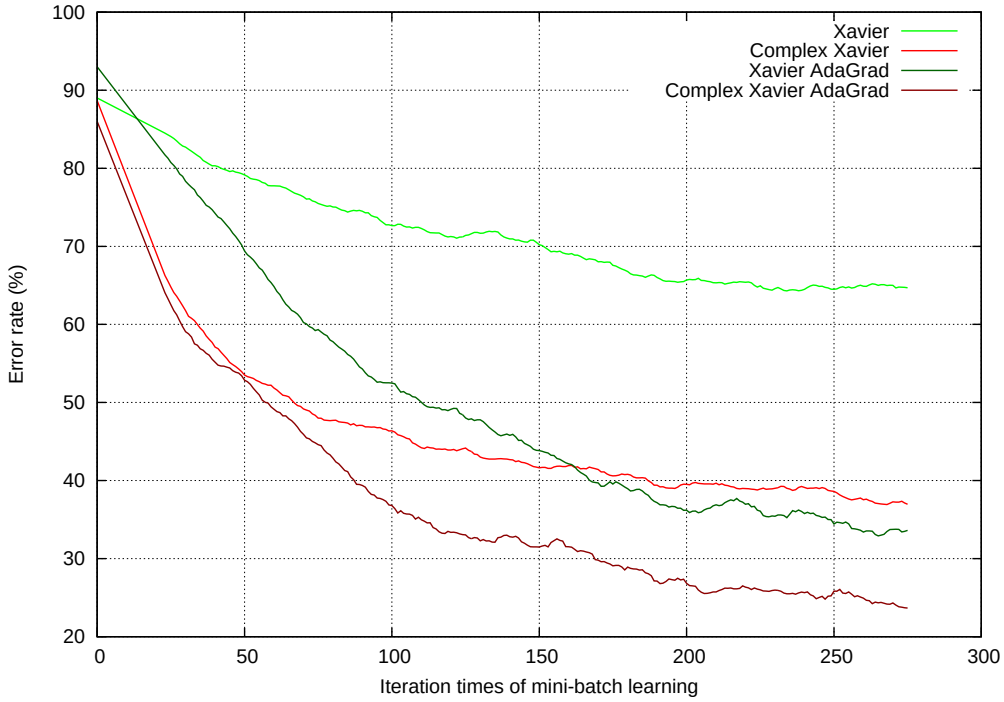

図 4.7. 図 4.3 と図 4.5 をまとめた図

ニューラルネットワークにおいても初期パラメータの設定手法がその後の学習で大きな差を生むことが確認される.

4.5.2　活性化関数が ReLU の結果について

隠れ層の活性化関数を ReLU とした場合の結果の図 4.6 では, まず Complex Xavier Initialization による初期化では学習を進めることができていないことが観察される. その一方 Complex He Initialization は学習を進めることができており, 前章で行った実数での実験結果 (図 3.9) と同様の結果となっている. このことから He Initialization の考え方は複素数のニューラルネットワークにおいても有効であるということが示される. さらに高次元数に合わせた He Initialization の修正式 4.9 は適切な修正であったと考えられる.

次に Complex Xavier B+ Initialization について見ると, 実数での実験結果の図 3.9 とは異なり, 実験を行った範囲では学習を進めることができていない. しかしながら Complex Joined Xavier Initialization は Complex He Initialization よりも優れた結果を出していることに加え, 前章で行った実数での実験結果においてもミニバッチ学習回数が 5000 回程度までは学習が進んでいないことを考えると, ミニバッチ学習回数を増やして実験を行うことで Complex Xavier Initialization と Complex Xavier B+ Initialization の間に差が現れることが予想される.

Complex Joined Xavier Initialization について観察すると, この方法が隠れ層が 30 層の

ディープニューラルネットワークにおいて最も優れた初期化方法になっていることが分かる. 本実験では計算機リソースの制約からミニバッチ学習を 1000 回までしか行うことができなかったものの, 前章で行った実数での実験 (図 3.9) と同様の結果となっており, ミニバッチ学習回数が 1000 回以上においても Complex He Initialization よりも優れた結果となることが予想される.

4.5.3 今後の課題

本実験では適切なデータセットの不足から, 高次元数の一つとして $P = 2$ の複素数についてのみでしか実験を行うことができていない. 四元数のニューラルネットワークにおいても式 4.7, 4.9 の導出は同様にできるため, 式 4.7, 4.9 の効果は期待できると考えられるが, 今後四元数として表現されることに適したデータセットを用いて実験的に確認することは重要な課題であると言える.

第 5 章

結論

本研究ではまず活性化関数に ReLU を用いた多層ニューラルネットワークについて, 定性的な考察から He Initialization とは異なる形で Xavier Initialization の改善 (Xavier B+ Initialization) を行った. また He Initialization と Xavier B+ Initialization の比較実験の結果を考察することで, He Initialization をさらに改善できる (Joined Initialization) 可能性を示すことができた.

しかしながら本研究で考えた Xavier Initialization の改善には He らが導出したような明快な数式を与えることができていない. それゆえ本研究中の実験で用いた Xavier B+ Initialization の数値設定 (式 3.9) などが最適であるのか, 本研究で実験した構造以外の人工ニューラルネットワークでどのような結果になるのかなど, 多くの疑問が残る形になっており, 今後さらに様々な検証実験, 理論研究を行う必要があると考えられる.

また本研究では, 高次元数のニューラルネットワークでの Xavier Initialization の修正を理論的に導出し, 実験によりこの修正の妥当性を確認した. 実験的には複素数についてのみで検証を行っているものの, 四元数でも本研究の修正提案は有効である可能性が高いと考えられる.

本研究で行った複素数のニューラルネットワークに関する研究はパラメータ初期化手法についてのみであるが, そこで得られた結果は実数で行った場合と類似したものとなっている. このことは既存の実数のディープニューラルネットワークに関する研究成果が高次元数においても同様に利用できる可能性を示唆していると考えられる. 本研究では既に実数として存在している画像データセット MNIST の各画像をフーリエ変換によって複素数で表すことで実験を行ったが, 応用においては得られているデータが複素数などの高次元数であることも少なくない. 特にこのような高次元数データセットに対して, ディープニューラルネットワークの利用がさらに広がることが期待される.

以上のように本研究では実数以外のディープニューラルネットワークも含めて各初期化手法ごとの学習の進み方を比較し, 改善を行った. しかしながら本研究で用いた全結合型の多層ニューラルネットワークの学習によって得られた最終的な分類性能は, 事前学習をした場合や畳み込みニューラルネットワークによる場合に比べれば大きく劣るものにしかならなかった. その一方で Xavier Initialization や He Initialization は全結合型での考察から始まり, 現在畳み込みニューラルネットワークのパラメータ初期化にも大きく貢献していることなどを考える

と, 本研究での提案手法を畳み込みニューラルネットワークに適用することでさらなる改善が行える可能性があると考えられる.

謝辞

テーマの決定からその後の研究の方向性, 卒業論文の執筆にあたって注意する点, 意識すべき点などについてアドバイスをくださった合原一幸教授, 先行研究に関する情報から発表資料の改善へのアドバイス, 発表練習にお付き合い頂くなど多岐にわたるご指導くださった奥助教と藤居技術専門職員に深く感謝の意を表します.

参考文献

[1] Amari, Shunichi “A theory of adaptive pattern classifiers” IEEE Transactions on Electronic Computers, 16(3) 1967: 299–307

[2] Duchi, John, Elad Hazan, and Yoram Singer. “Adaptive subgradient methods for online learning and stochastic optimization.” The Journal of Machine Learning Research 12 2011: 2121–2159.

[3] Glorot, Xavier, and Yoshua Bengio “Understanding the difficulty of training deep feedforward neural networks” International conference on artificial intelligence and statistics 2010

[4] Goh, S. L., et al. “Complex-valued forecasting of wind profile.” Renewable Energy 31.11 (2006): 1733-1750.

[5] He, Kaiming, et al. “Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification.” arXiv preprint arXiv:1502.01852 2015.

[6] Hinton, Geoffrey E, Simon Osindero, and Yee-Whye Teh “A fast learning algorithm for deep belief nets” Neural computation 187 2006: 1527-1554

[7] Hinton, Geoffrey E, and Ruslan R Salakhutdinov “Reducing the dimensionality of data with neural networks” Science 313(5786) 2006: 504–507

[8] Hochreiter, Sepp, et al. “Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.” 2001.

[9] Isokawa, Teijiro, et al. “Quaternion neural network and its application” Knowledge-Based Intelligent Information and Engineering Systems Springer Berlin Heidelberg, 2003

[10] Kobayashi, Masaki, Jun Muramatsu, and Haruaki Yamazaki “Construction of high - dimensional neural networks by linear connections of matrices” Electronics and Communications in Japan Part III: Fundamental Electronic Science 86(11) 2003: 38–45

[11] Kolen, Hohn F, and Jordan B Pollack, Jordan B “Backpropagation is sensitive to initial conditions” Complex Systems 4 1990: 269–280

[12] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton “ImageNet classification with deep convolutional neural networks” Advances in neural information processing systems 2012

[13] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. “Deep learning.” Nature 521.7553 (2015): 436-444.

[14] Maas, Andrew L, Awni Y Hannun, and Andrew Y Ng. “Rectifier nonlinearities improve neural network acoustic models.” Proc. ICML. Vol. 30. 2013.

[15] Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams “Learning representations by back-propagating errors” Cognitive modeling 5 1988: 3

[16] Simonyan, Karen, and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition.” arXiv preprint arXiv:1409.1556 2014.