AI 勉強会（第 3 回）

木更津工業高等専門学校
大枝真一

2017 年 11 月 13 日

1　はじめに

第 1 回 AI 勉強会では，ニューラルネットワークの基本である前向き計算と学習法である BP 法の説明を行った．第 2 回 AI 勉強会では，学習の効率化とコーディングについて説明した．

第 3 回 AI 勉強会では，Deep Learning に関する話題について解説を行う．今回の勉強会では主にスライドを用いて説明する．

2　第 2 回の復習

2.1　ニューラルネットワークのコーディング

前向き計算，Back Propagation 法による重みの修正は数式をそのままコーディングするだけでよい．ただし，慣れるまではスカラー値で考え，手計算の結果と一致するかひとつひとつ検証しながらコーディングすることを推奨する．

プログラムテストで，検証結果がうまくいかない場合の原因は，以下の 3 点が考えられる．

1. 数理モデルが間違っている．
2. プログラムにバグがある．
3. 初期パラメータが適切でない．

新しいニューラルネットワークのモデルを考案した場合，1 について考慮する必要がある．しかし，すでに発表されているモデルを追試する場合，数理モデルに間違いがあることは一般的には考えられない．そのため，1 に関して考慮する必要はない．ただし，複雑なモデルなどで理解の間違いの可能性はあるかもしれない．

2 については，プログラミングスキルに依存する．自らの手で，ゼロからニューラルネットワークを構築するなどの経験があると，バグフィックスまでの期間短縮が可能となる．

3 については，ニューラルネットワークの学習動作について深い理解が必要となる．経験はもち

ろん必要であるが，学習過程の挙動と数理モデルとの関係を洞察することで，初期パラメータを適切に設定できるようになる．2 と 3 の区別はなかなかつかないことが多いため，「ハマる」となかなかやっかいである．ニューラルネットワークのフレームワークが各種開発されているが，使いこなすためにはある程度時間がかかる．

結局は，自らの力で作成した経験がないと難しいのではないかと考えている．

2.2 学習の効率化

ニューラルネットワークの学習は非常に時間がかかる．学習効率化の方法として，ミニバッチ法，モーメント法など多くの手法が提案されている．

2.3 過学習と汎化性能

トレーニングデータで学習を行って，テストデータで評価を行う．目的はテストデータに対する正答率を上げることであるため，トレーニングデータで学習しすぎると，過学習が生じてテストデータに対する汎化性能が低下する．解消する方法として early stoppping と呼ばれる方法がある．

2.4 ベクトルと行列による表現

ニューラルネットワークで行っていることは積和計算であるため，ベクトルと行列を用いれば簡潔に表現できる．Python の numpy を使えば，ベクトルと行列のままコーディングができるため，簡潔に記述できるだけでなく，計算速度の面でもメリットがある．

2.5 サンプルプログラム

以下の URL にサンプルプログラムがある．

サンプルプログラム

```
https://github.com/crotsu/Study_AI
```

3 Deep Learning

ニューラルネットワークを多層・大規模化したものを Deep Learning と呼ぶ．基本動作は従来のニューラルネットワークと同じである．しかし，多層・大規模化することにより次のような問題点が生じていた．

1. 計算時間
2. 過学習

3. 勾配消失問題

3.1 計算の効率化

学習精度を上げるためには，大量のトレーニングデータが必要となる．近年，ビッグデータという単語がバズワードになるほどデータ収集が容易となった．一方，データ量が多くなったことで多くの計算時間が必要となった．これを解決したのも技術の進歩であり，高速な GPU が開発されたことにより並列計算が可能となり，大幅な計算時間の短縮が実現されている．

さらに，学習の効率化のためにいくつかの方法が提案されている．

1. データの正規化
2. データ拡張
3. アンサンブル学習
4. サンプル順序
5. 重みの初期化
6. モーメント法
7. 学習係数の決め方

データの正規化は，特徴ベクトルの成分ごとの平均・分散を揃えて学習をしやすくすることである．データ拡張は，もともとあるデータを水増しして増やすことである．例えば，画像を学習したい場合，トレーニング用の画像を微小な回転・移動を加えたり，ノイズを加えたりして，トレーニング画像を水増しする．アンサンブル学習は，複数のネットワークの出力の平均を最終的な出力とするものである．同じネットワーク構造でも初期重みによって学習結果が異なる．そこで初期重みの異なる複数のネットワークの出力平均をとることで，学習精度を向上させるものである．サンプル順序とは，データ数が多い場合，まだ未学習のデータを優先的に割り当てる操作を行うことである．重みを初期化とは，初期重みの適切に与えることである．通常はランダムな値を与える．モーメント法は前回の修正量を今回の修正量に加える方法である．学習係数の決め方については様々な方法が提案されており，AdaGrad, Adam などが有名である．

3.2 過学習

学習を行えば行うほど，過学習が生じる．これを解決する方法として，以下の 3 つの方法がある．

1. Early Stopping
2. ドロップアウト
3. 正則化

Early Stopping は，トレーニング中にテストデータを与えてネットワークを評価し，テストデータに対する誤差が増大したときに学習を打ち切る方法である．ドロップアウトは，学習中にニュー

ロンを動作させない方法である．学習中にランダムにニューロンが動作しないため，学習効率が落ちる．これが過学習の抑制に繋がる．実際に用いるときは全結合とする．正則化は回帰分析でのリッジ回帰と同じである．リッジ回帰は，要はパラメータがある範囲を超えないように，目的関数にロス項を追加する．これにより，パラメータがスパースになることが期待できる．ニューラルネットワークにおいても同様に，重み減衰を行うことでネットワークのスパース性を期待する手法である．

3.3 勾配消失問題

前向き計算では，ニューロンの出力を計算するときに非線形関数を用いている．したがって，前向き計算は非線形計算である．応答関数としてシグモイド関数を用いている場合，値域が $[0, 1]$ に制限される．しかし，Back Propagation では，積和の計算しか行っていないため線形計算である．つまり，重みが 1.0 よりも大きければ，修正量どんどんおおきくなり発散する．また，重みが 1.0 より小さければ，修正量は小さくなり消失してしまう．

そこで，多層のニューラルネットワークを 3 層ごとに学習する方法が考案された．Autoencoder（自己符号化器）と呼ばれるネットワーク構造の学習法である．3 層のニューラルネットワークとして，入力がそのまま出力されるように学習を行う．そして，4 層目を追加する．入力-中間層の重みはそのまま固定して学習せず，今度は中間層の出力がそのまま出力されるように学習を行う．これを当初の多層のニューラルネットワーク全体におよぶまで繰り返す．このようにして，初期重みを与えて，最後に分類をするための出力層を追加して，全体を fine tuning する．この方法により，従来では考えられないほど，深い (deep) ニューラルネットワークでも，学習が可能となった．