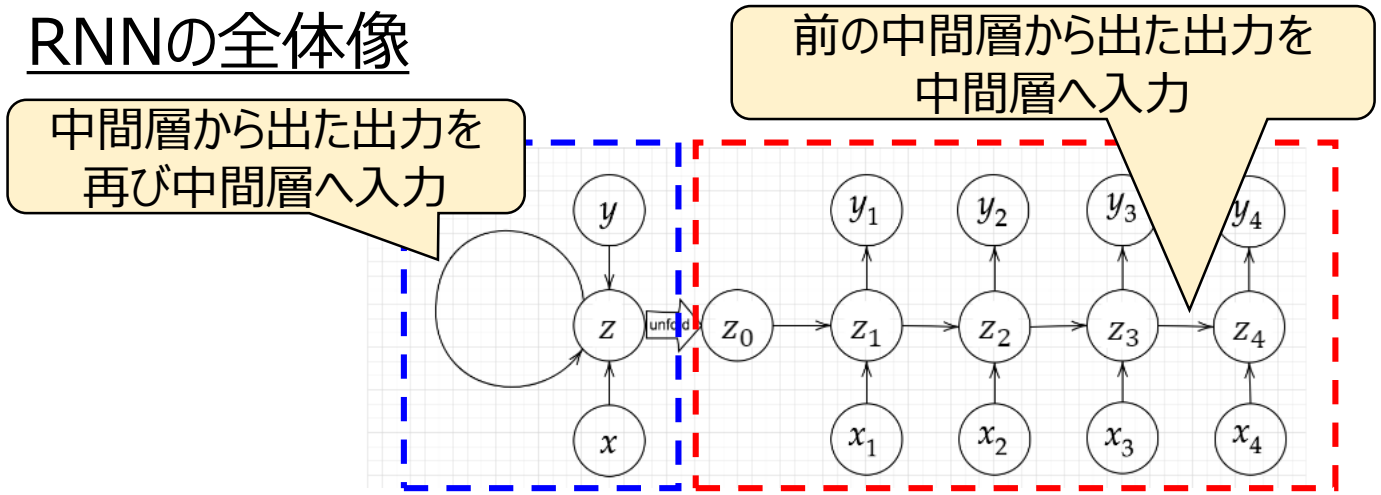
実装演習レポート【深層学習_前編】 Section1
再帰型ニューラルネットワークの概念

要点のまとめ

- RNNとは時系列データに対応可能な、ニューラルネットワークである。
  時系列データとは、時間的順序を追って一定間隔ごとに観察され、しかも相互に
  統計的依存関係が認められるようなデータの系列。
- RNNでも基本的なニューラルネットワークの構造は変わらない（入力層、中間層、出力層）。
  但し、時間的な繋がりを考慮する。

RNNの全体像



左と右は同じことを表す

RNNの数学的記述

$$u^t = W_{(in)}x^t + W\,z^{t-1} + b$$
$$z^t = f\left(W_{(in)}x^t + W\,z^{t-1} + b\right)$$
$$v^t = W_{(out)}z^t + c$$
$$y^t = g\left(W_{(out)}z^t + c\right)$$

- RNNの逆伝播はBPTTという手法を使用する。
  RNNにおいてのパラメータ調整方法の一種。誤差逆伝播の一種。

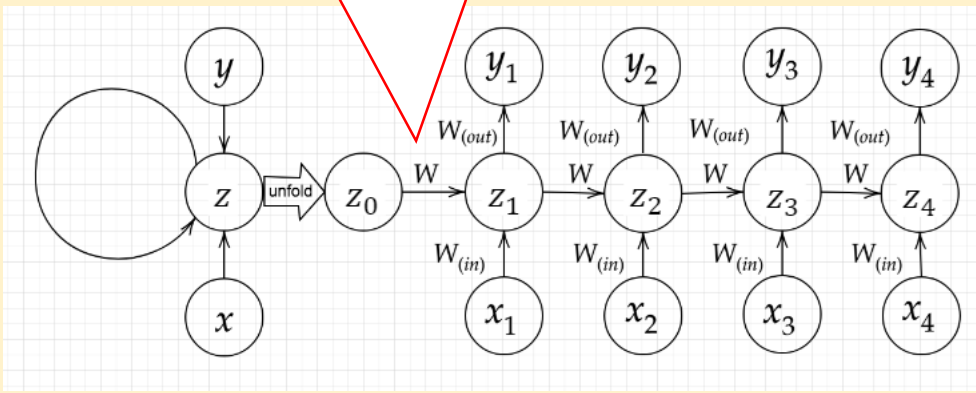
実装演習レポート【深層学習_前編】 Section1

再帰型ニューラルネットワークの概念

確認テスト

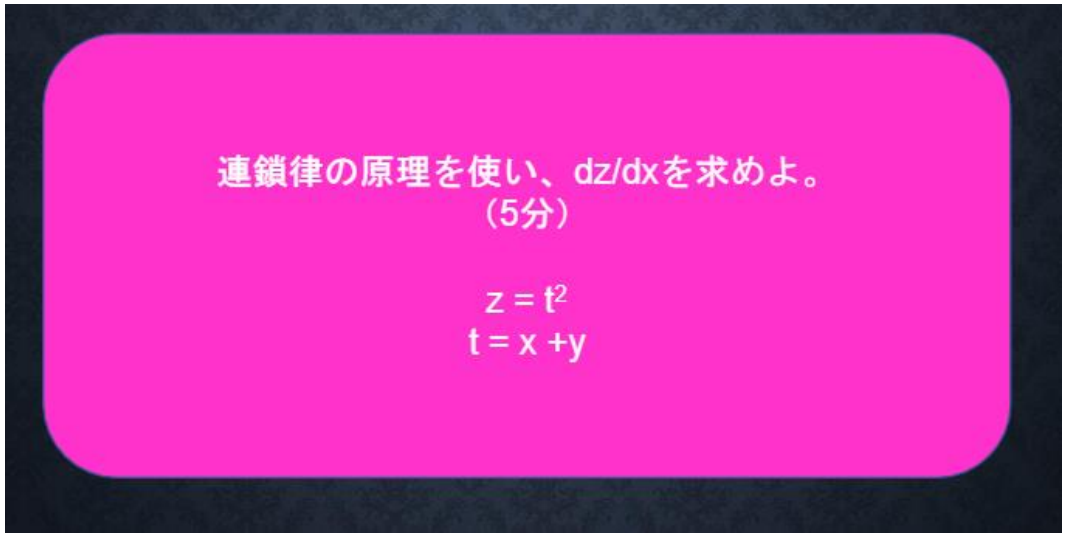
RNNのネットワークには大きくわけて3つの重みがある。1つは入力から現在の中間層を定義する際にかけられる重み、1つは中間層から出力を定義する際にかけられる重みである。
残り1つの重みについて説明せよ。
（3分）

前の中間層からの重み



実装演習レポート【深層学習_前編】 Section1

確認テスト　　再帰型ニューラルネットワークの概念



$$\frac{dz}{dx} = \frac{dz}{dt}\frac{dt}{dx}$$

$$= 2t \times 1$$

$$= 2t$$

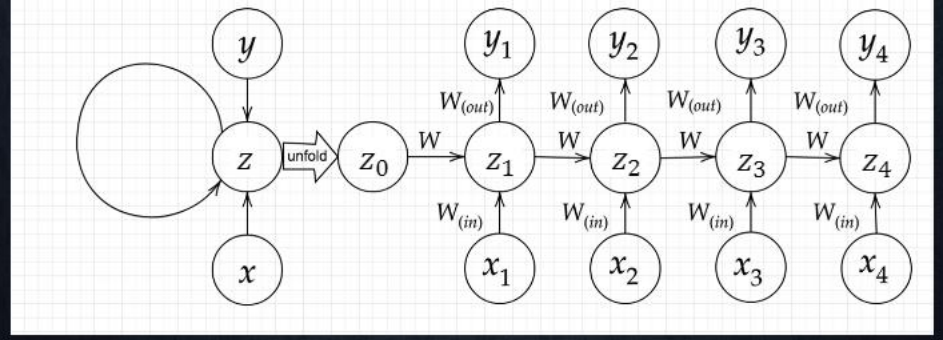
$$= \mathbf{2(x+y)}$$

実装演習レポート【深層学習_前編】 Section1

確認テスト

再帰型ニューラルネットワークの概念

下図のy_1をx・s_0・s_1・W_{in}・W・W_{out}を用いて数式で表せ。
※バイアスは任意の文字で定義せよ。
※また中間層の出力にシグモイド関数g(x)を作用させよ。
（7分）



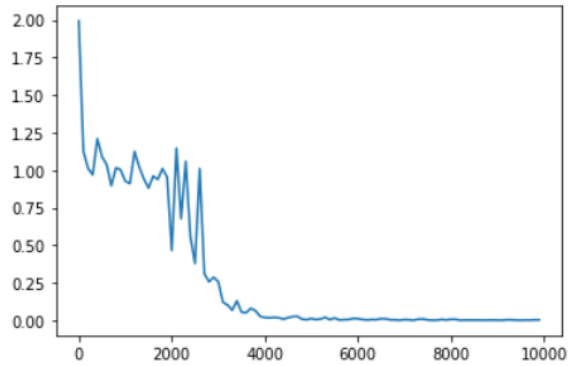
$$y_1 = g(W_{out} \cdot s_1 + C)$$
$$s_1 = f(W_{in} \cdot x_1 + W \cdot s_0 + B)$$

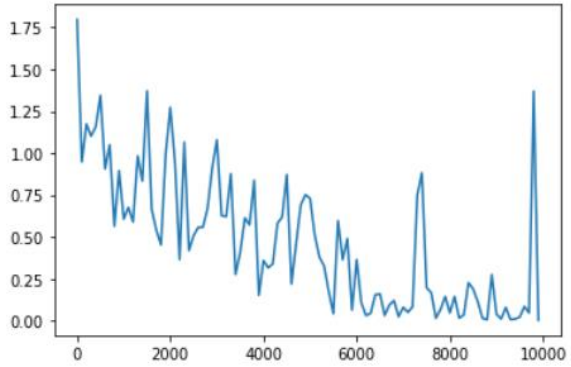
実装演習レポート【深層学習_前編】 Section1
再帰型ニューラルネットワークの概念

実装演習結果(3_1_simple_RNN_after.ipynb)

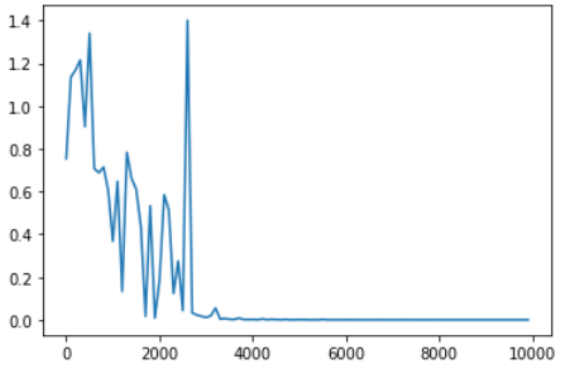
[try] weight_init_stdやlearning_rate, hidden_layer_sizeを変更してみよう

weight_init_std	1	**2**	1	1
learning_rate	0.1	0.1	**0.5**	0.1
Hidden_layer_size	16	16	16	**32**

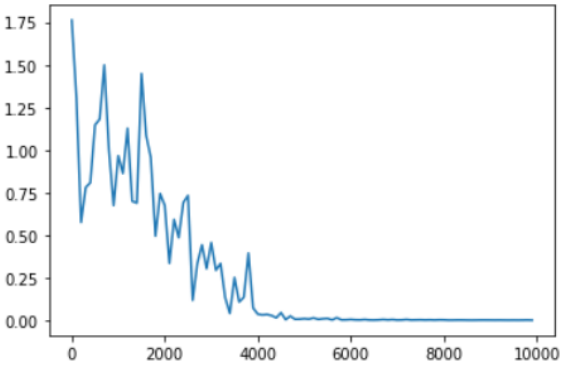




学習が安定しない



若干、早めに収束



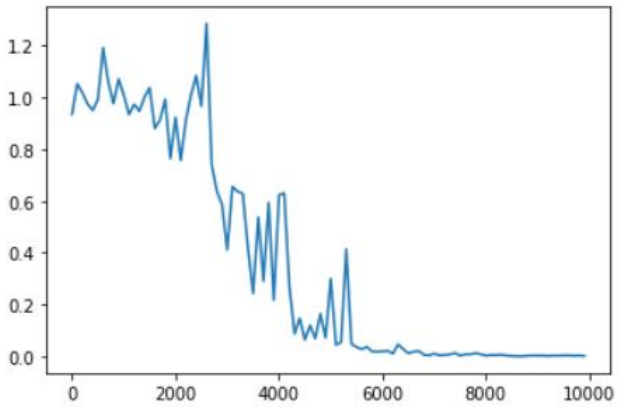
大きな変化なし
少し、収束傾向異なる

実装演習レポート【深層学習_前編】 Section1
再帰型ニューラルネットワークの概念

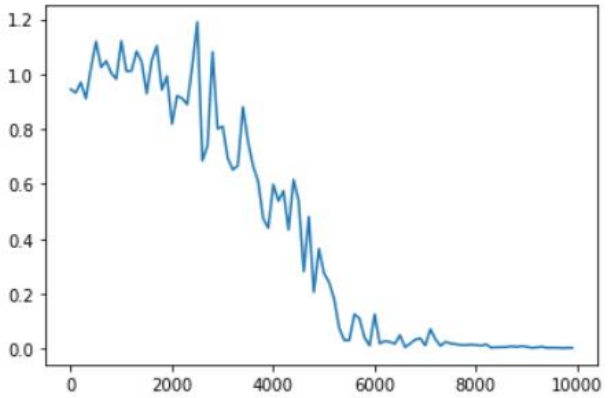
実装演習結果(3_1_simple_RNN_after.ipynb)

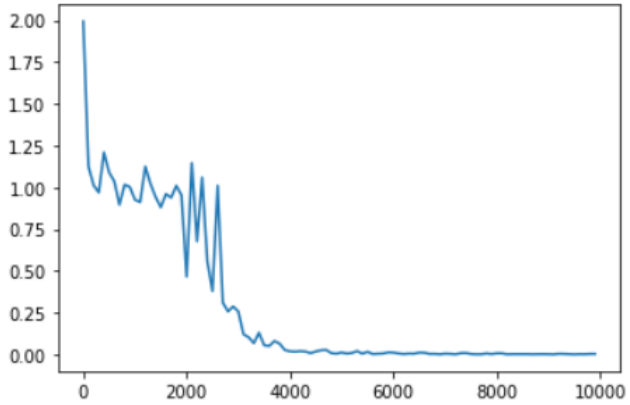
[try] 重みの初期化方法を変更してみよう

Xavier



He



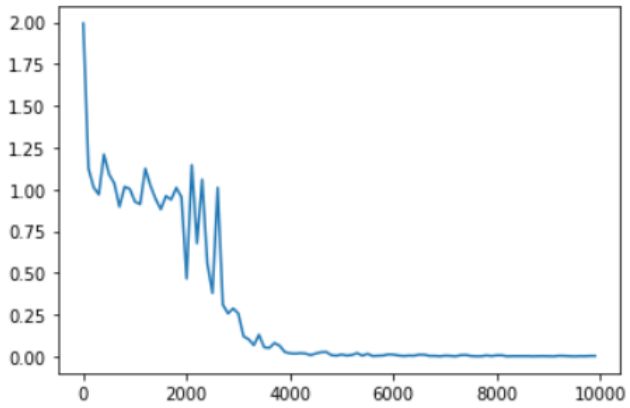


Xavier，He共に初期の損失は小さいが、
減少の割合が小さくなる。

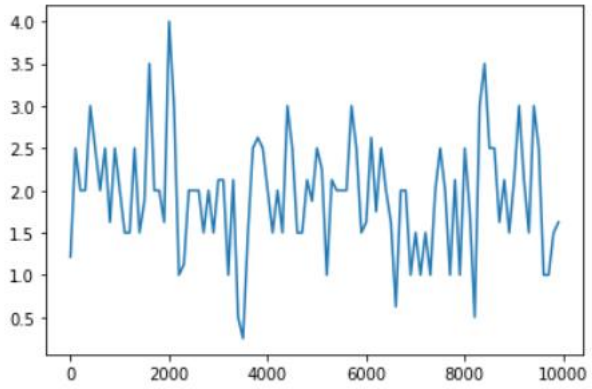
実装演習レポート【深層学習_前編】 Section1
再帰型ニューラルネットワークの概念

実装演習結果(3_1_simple_RNN_after.ipynb)

[try] 中間層の活性化関数を変更してみよう

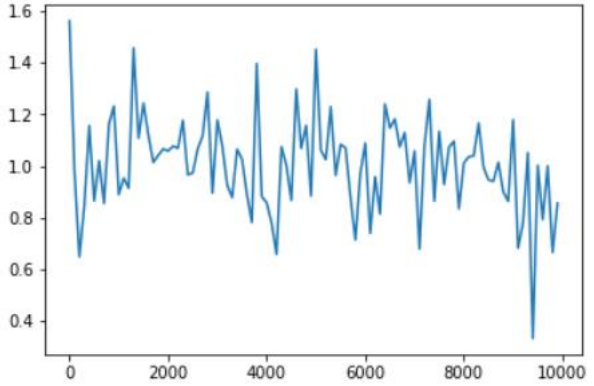


Relu



学習が進まない。

tanh



学習が進まない。

実装演習レポート【深層学習_前編】 Section1
再帰型ニューラルネットワークの概念

参考図書レポート

再帰型ニューラルネットワーク（Recurrent Neural Network：RNN）
とは、ニューラルネットワークを拡張して時系列データを扱えるようにしたもの。

RNNの学習でも出力値と正解値の誤差Eを計算し、それが小さくなるように
勾配降下法を使って重みwをチューニングしていく。
RNNでは、中間層の時間経過を考慮しなければならないので、そのままでは
誤差逆伝播法を適用できない。そこで、時間経過を表現できるように改良。
よく使用されるのが、BPTT(backpropagation through time)。
RNNのネットワークを中間層出力を介して時間方向に展開する方法。
それによって、RNNを時間経過を含めて1つの大きなニューラルネットワークと
みなすことにより、誤差逆伝播法を適用することが可能となる。

実装演習レポート【深層学習_前編】 Section2 LSTM

要点のまとめ

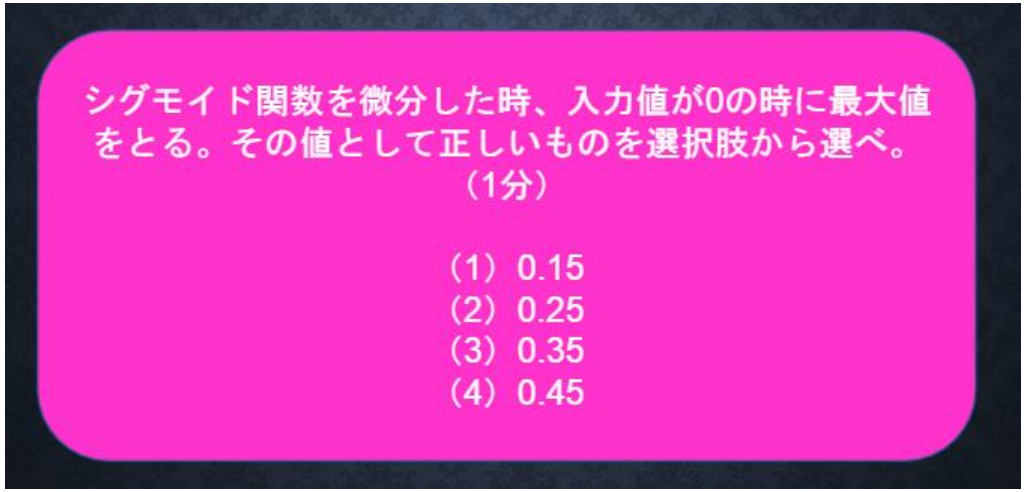
- RNNの課題。時系列を遡れば遡るほど、勾配が消失していく。
  長い系列の学習が困難。
  （勾配消失問題のとは逆に、勾配が指数関数的に大きくなる勾配爆発問題もある。）
  この勾配消失を解決するために構造を変化させたものがLSTM。

  従来の記憶と学習の両方を持った中間層では勾配消失問題がおきてしまう。
  LSTMではCECという記憶することのみに特化した中間層を持つ。そして、CECの
  周りに学習機能を配置する。具体的には、入力ゲートと出力ゲートの２つ。
  入力ゲートは記憶したい方法（今回の入力と前回の入力をどれくらい覚えるか）を
  CECに伝える。
  CECが記憶した情報をどういう風に出力すればよいかを出力ゲートがCECに伝える。

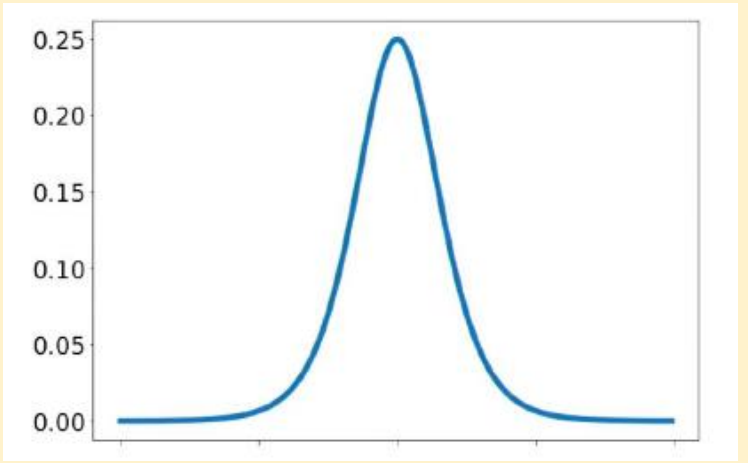
  CECの課題。過去の情報を保管し続けている。要らなくなった情報を削除できない。
  これを解稀するために忘却ゲートを配置する（要らなくなったデータを無くす）。

実装演習レポート【深層学習_前編】 Section2 LSTM

確認テスト



(2) 0.25



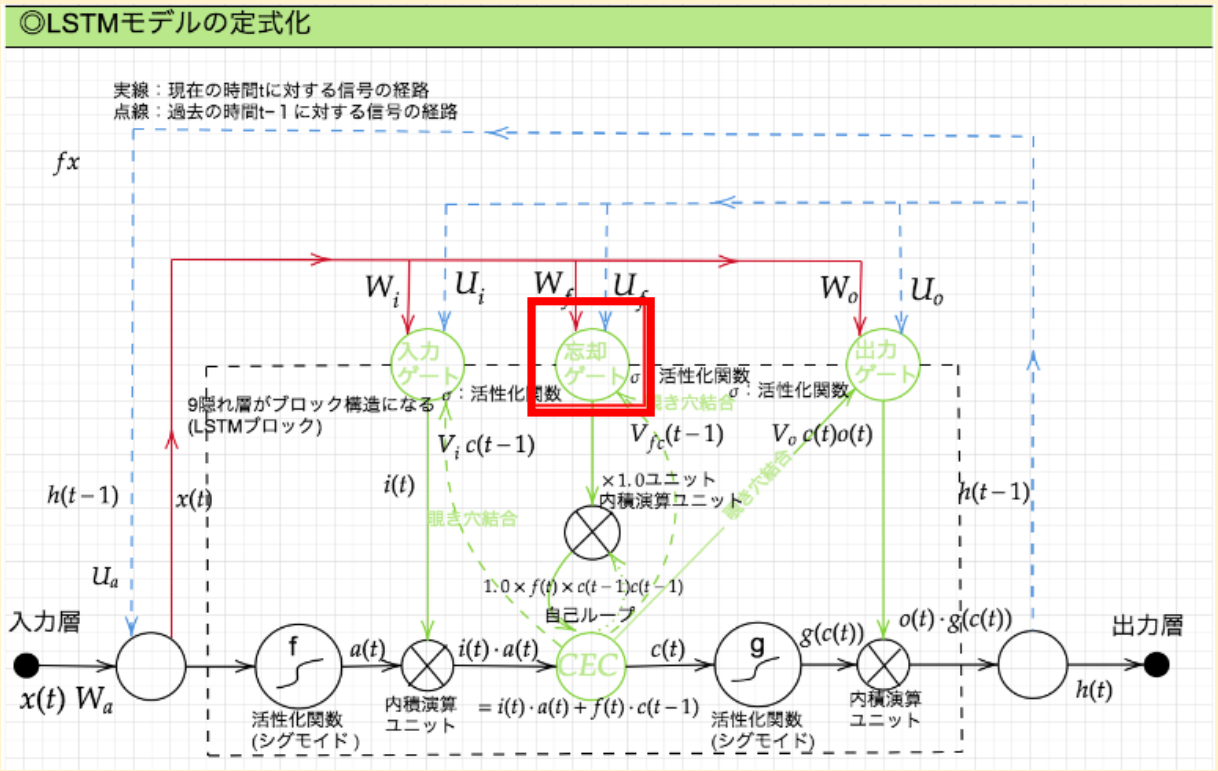
実装演習レポート【深層学習_前編】 Section2 LSTM

確認テスト

以下の文章をLSTMに入力し空欄に当てはまる単語を予測したいとする。
文中の「とても」という言葉は空欄の予測において
なくなっても影響を及ぼさないと考えられる。
このような場合、どのゲートが作用すると考えられるか。

「映画おもしろかったね。ところで、とてもお腹が空いたから何か＿＿＿。」
（3分）

忘却ゲート



実装演習レポート【深層学習_前編】 Section2 LSTM

参考図書レポート

・RNNの課題としては、勾配消失問題がある。誤差を逆伝播するときに、
過去に遡るにつれて勾配が消えてしまうという問題が発生する。
この問題を解決する為に考えられたのが、LSTM。
LSTMはLSTMブロックと呼ばれる機構を導入し、時系列情報をうまく
ネットワーク内に保持することを可能としている。

LSTMは主に2つのブロックから成り立っている。
①：誤差を内部にとどまらせる為のセル、勾配消失を防ぐ（CEC）
②：必要な情報を必要なタイミングで保持・消却させるためのゲート
（入力ゲート、出力ゲート、忘却ゲート）

実装演習レポート【深層学習_前編】 Section3 GRU

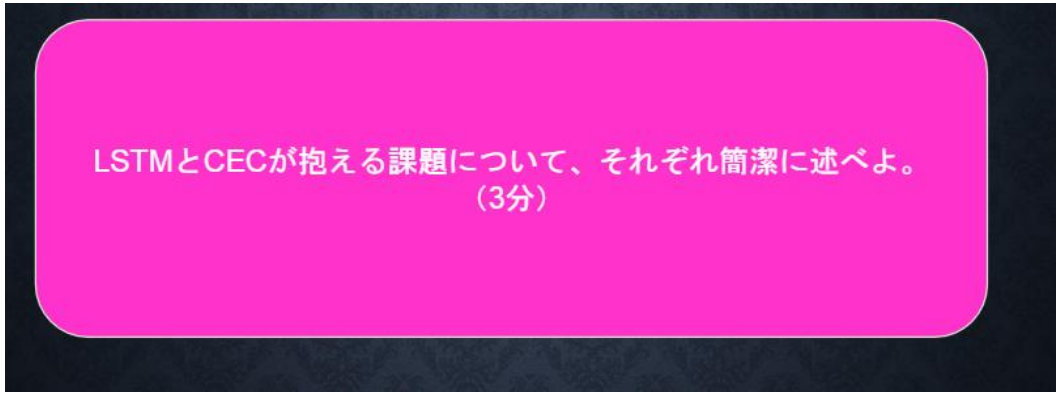
要点のまとめ

- GRUはLSTMの改良版。
  LSTMでは、パラメータが多く計算負荷が大きかった。
  しかし、GRUでは、そのパラメータを大幅に削減し、精度は同等またはそれ以上が望めるようになった構造。計算負荷が小さくなった。

  LSTMと異なる点は、CECが無くなったこと。入力ゲート、出力ゲート、忘却ゲートの代わりにリセットゲート、更新ゲートが配置されている。

実装演習レポート【深層学習_前編】 Section3 GRU

確認テスト

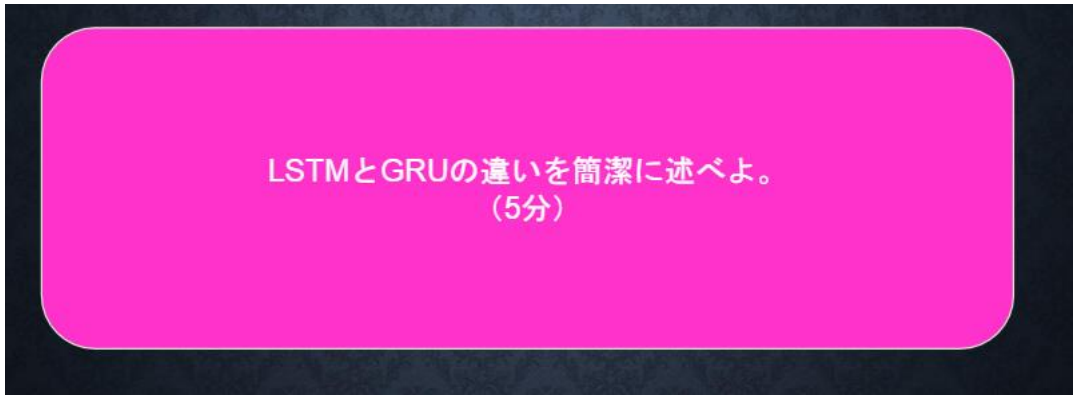


LSTMではパラメータが多く、計算負荷が大きい。

CECは情報を記憶していく機能のみ、学習機能は無い。

実装演習レポート【深層学習_前編】 Section3 GRU

確認テスト



LSTMにはCECがあるが、GRUにはない。

LSTMには入力ゲート、出力ゲート、忘却ゲートの構成だが、GRUはリセットゲートと更新ゲートの構成。

LSTMはパラメータが多く計算負荷が大きいが、GRUはパラメータ数が少なく計算負荷が小さい。

実装演習レポート【深層学習_前編】 Section3 GRU

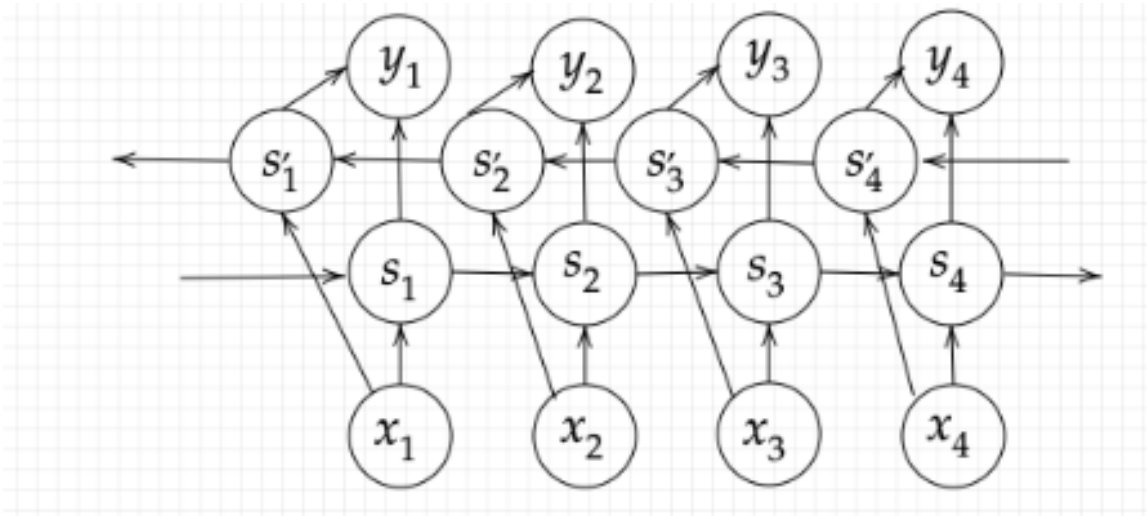
参考図書レポート

- GRUはLSTMの計算コストが大きいという課題に対して、２つのアプローチで改善。
  ① 状態ベクトル数を減らす（時間依存の状態数を減らす）
  LSTMではCECとhの２つが状態を保持していたが、これを１つにまとめる。
  これにより、記憶セルという構造は無くなる。
  ② ゲートコントローラ数を減らす
  LSTMでは入力ゲート、忘却ゲート、出力ゲートに１つずつのコントローラが
  必要でした。そこで、忘却ゲートと入力ゲートの操作を１つのコントローラで操作
  するように変更する。
  記憶の忘却はゲートコントローラの値を１から差し引いたものを乗算することで、
  記憶の更新はゲートコントローラの値を１時刻前の状態ベクトルと乗算した後に
  加算することで実現されている。
  すなわち新たな記憶を保持するときは過去の記憶は忘れるように連動させることで
  構造を簡略化させています。

実装演習レポート【深層学習_前編】 Section4 双方向RNN

要点のまとめ

- 双方向RNNは過去の情報だけでなく、未来の情報を加味することで、
  精度を向上させるためのモデル
  文章の推敲や、機械翻訳等に使用される。



実装演習レポート【深層学習_前編】 Section4 双方向RNN

参考図書レポート

・双方向RNN(Bidirectional RNN)とは、中間層の出力を未来への順伝播と過去への逆伝播の両方向に伝播するネットワークである。
BRNNでは、学習時に過去と未来の情報の入力を必要とすることから、運用時も過去から未来までのすべての情報を入力してはじめて予測できるようになる。
そのため、BRNNの応用範囲が限定される。

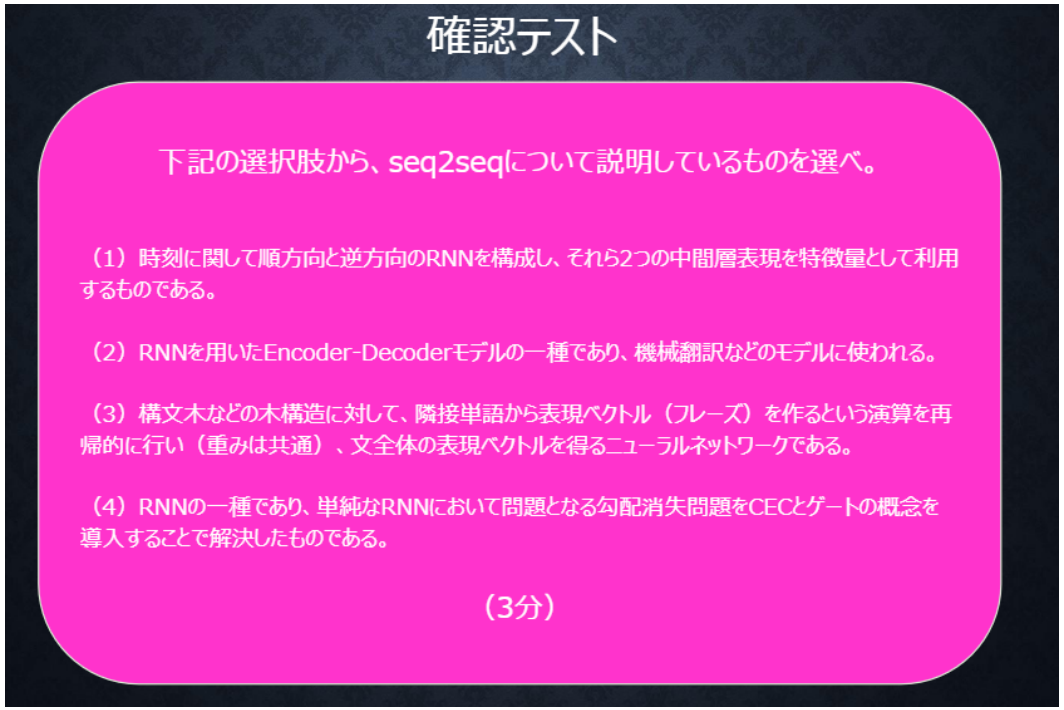
実装演習レポート【深層学習_前編】 Section5 Seq2Seq

要点のまとめ

- Seq2Seqとは２つのネットワークがドッキングされている自然言語のニューラルネットワーク。
  Encoder-Decoderモデルの一種を指します。
  （Encoderは文章の意味を入力。DecoderはEncoderから入力されたものから発信。）
  機械対話や機械翻訳などに使用されている。
  Encoderでは文章を単語等のトークン毎に分割し、トークンごとのIDに分割する。
  IDから、そのトークンを表す分散表現ベクトルに変換して、RNNへ入力。
  Decoderでは生成確率に基づいてトークンをランダムに選ぶ。それをEmbeddingして
  DecoderRNNの次の入力とする。それを繰り返して文字列に直す。

実装演習レポート【深層学習_前編】 Section5 Seq2Seq

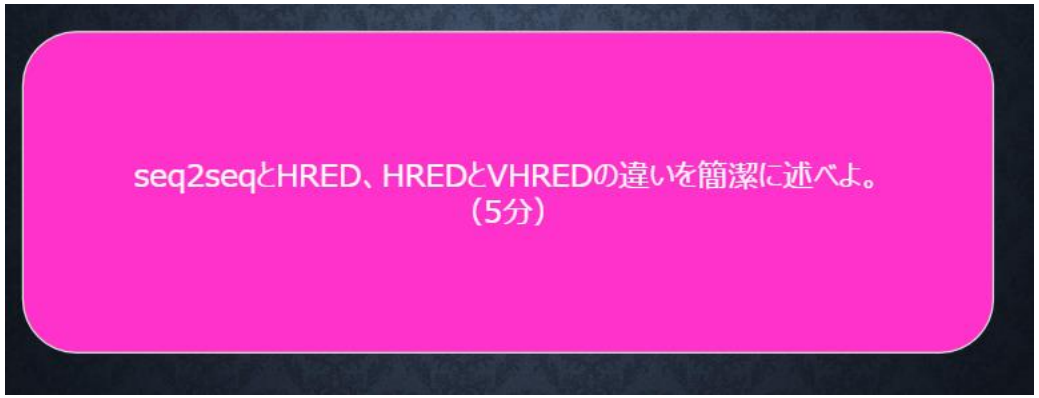
確認テスト



(2)

実装演習レポート【深層学習_前編】 Section5 Seq2Seq

確認テスト



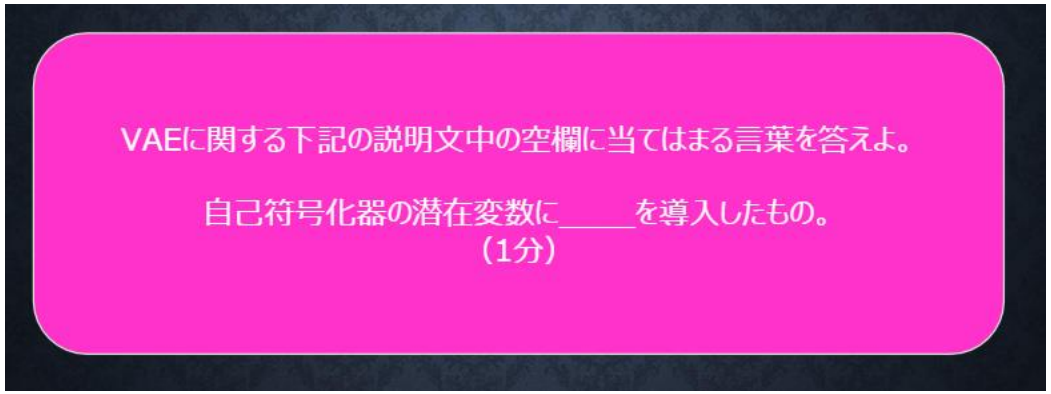
Seq2seqは、１文の一問一答に対して、処理ができるある時系列データから別の時系列データを作り出すニューラルネットワーク。

HREDとは文脈の意味ベクトルを解釈に加えられるようにしたニューラルネットワーク。

VHREDとはVAEの考え方を取り入れて改良したニューラルネットワーク。

実装演習レポート【深層学習_前編】 Section5 Seq2Seq

確認テスト



確率分布

実装演習レポート【深層学習_前編】 Section5 Seq2Seq

参考図書レポート

Seq2Seqの処理フロー

1. 入力シーケンス（英文）を状態ベクトルに変換する。
2. 学習済みLSTM層を利用したEncoderを使って入力データを与えて内部状態を取得する。
3. 状態ベクトル（内部状態）と１文字のターゲットシーケンスをDecoderに入力して、
   次の文字に対する予測を生成する。
4. 次の文字をサンプリングする。
5. サンプリングされた単語（日本語）をターゲットシーケンスに追加します。
6. Decoderで出力されたターゲットシーケンスと内部状態をDecoderの入力に与え、
   文字数の上限に達するかシーケンスの終了文字が生成されるまで処理を繰り返す。

実装演習レポート【深層学習_前編】 Section6 Word2vec

要点のまとめ

・ RNNでは、単語のような可変長の文字列をNNに与えることはできない。
固定長形式で単語に表す必要がある。
学習データからボキャブラリを作成。
One-hotベクトルでは単語毎に作成される為、データ量が膨大になる。
それに対して、word2vecでは大規模データの分散表現の学習が、現実的な計算速度とメモリで実現可能にした。

実装演習レポート【深層学習_前編】 Section6 Word2vec

参考図書レポート

- Word2vecとは、文章中の単語を数値ベクトルに変換してその意味を把握する
  自然言語処理の手法。初心者でも利用しやすい自然言語処理の手法です。
- Word2vecは、2013年にGoogleの研究者トマス・ミコロフ氏によって提案された手法。
  従来の自然言語処理手法に比べて精度が高く、特に文章の意味把握において
  以前より飛躍的に精度を向上させられるようになりました。
  Word2vecは、「意味ベクトル」という単語ベクトル表現を可能にする手法。
- 自然言語処理の手法のうち、単語をベクトルで表現する代表的な手法には、
  「one-hot表現」と「分散表現」の２つがある。
  Word2vecの意味ベクトルは「分散表現」に該当する。
  - 「one-hot表現」は単語を0or1のベクトル値で表現する。この表現の欠点は２つある。
    １つ目は、ベクトル間の演算で意味のある結果を得られないこと。
    ２つ目は、文章中の単語数が増えると次元が膨大になってしまうこと。
  - 「分散表現」はひとつの単語をおよそ数百次元のベクトルで表現する手法。
    分散表現では単語同士の演算が可能。単語数が膨大でもデータのサイズを抑えられる。

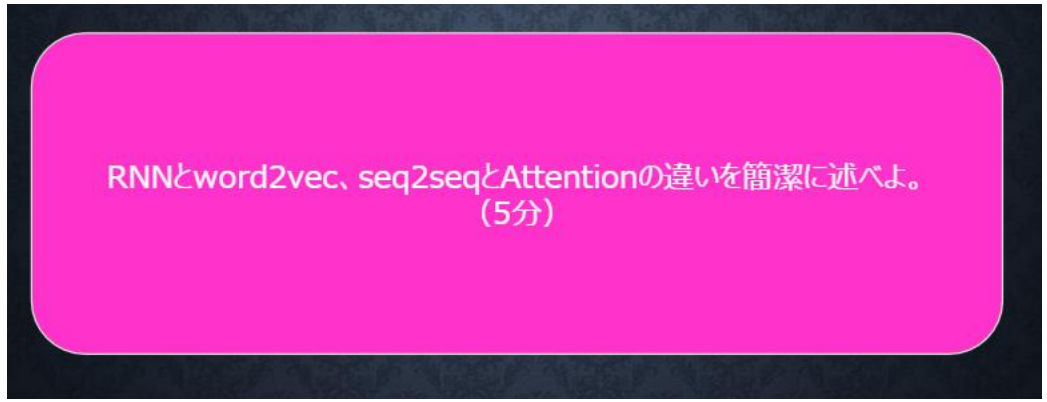
実装演習レポート【深層学習_前編】 Section7 Attention Mechanism

要点のまとめ

- seq2seqの問題は長い文章への対応が難しいこと。Seq2seqでは、2単語でも、100単語でも固定次元ベクトルの中に入力しなければならない。
  文章が長くなるほどそのシーケンスの内部表現の次元も大きくなっていく、仕組みが必要。
  そこで、AttentionMechanismを導入。「入力と出力のどの単語が関連しているのか」の関連度を学習する仕組み。

実装演習レポート【深層学習_前編】 Section7
Attention Mechanism

確認テスト



RNNは時系列データを扱うニューラルネットワーク。

Word2vecは単語の分散表現ベクトルを得る手法。

Seq2seqは一つの時系列データから別の時系列データを得る手法。

AttentionMechanismは時系列データの中身に対して、それぞれの関連性に重みを付ける手法。

実装演習レポート【深層学習_前編】 Section7 Attention Mechanism

参考図書レポート

・seq2seqの問題は長い文章への対応が難しいこと。
3単語の短い文でも、50単語あるような長い文でも、その意味をある固定次元ベクトルの中に押し込まなくてはならない。
文章が長くなるほど、シーケンスの内部表現の次元も大きくなっていくような仕組みが必要。
Attentino Mechanismではこの問題に対して、「入力と出力のどの単語が関連しているのか」を学習することで対応。
ネットワークは翻訳前後の単語の対応関係を学習し、単語列の出力時に対応する入力の単語を引っ張ってくることで長い文書でも翻訳の精度を上げる。

実装演習レポート【深層学習_前編】 Section1 強化学習

要点のまとめ

- 強化学習とは報酬を最大化できるように環境のなかで行動を選択できるエージェントを
  作ることを目標とする機械学習の一分野。
- 強化学習は通常の教師あり、なし学習と目標が異なる。
  教師あり、なし学習では、データに含まれるパターンを見つけ出すおよびそのデータから
  予測することが目標。 それに対して、強化学習では、優れた方策を見つけることが目標。
- 強化学習において、関数近似法と、Q学習を組み合わせる手法が登場
  関数近似法 ： 価値関数や方策関数を関数近似する手法のこと
  Q学習 ： 行動価値関数を、行動する毎に更新することにより学習を進める方法
- 価値関数は、状態価値関数と行動価値関数の 2 種類がある。
  ある状態の価値に注目する場合は、状態価値関数。
  状態と行動を組み合わせた価値に注目する場合は、行動価値関数。
- 方策関数とは方策ベースの強化学習手法において、ある状態でどのような行動を採るのか
  確率を与える関数のこと。
  方策勾配法を用いて、方策をモデル化して最適化を実施する。

実装演習レポート【深層学習_前編】 Section1 強化学習

参考図書レポート

- 強化学習では、コンピューターはある「環境」の中で、目的として設定された「報酬(スコア)」を最大化するための行動を学習する。

  代表例として、ロボットの歩行制御が挙げられる。この場合はロボットに「歩けた距離」を報酬として与えます。するとロボットは、歩行距離を最大化するために、自らさまざまな歩き方を試行錯誤する。そうすることで、歩行可能距離の長いアルゴリズムが構築される。

- 強化学習の流れ
  ① 「エージェント」がある「環境」の中に置かれ、その環境に対して「行動」を起こす。
  ② 環境がエージェントに、行動により更新された「状態」と「報酬」をフィードバックする。
  ③ 環境からのフィードバックを元に、「方策」を修正する。
  ④ これまでの一連の行動の結果として変化した環境の中で、再びエージェントが環境に対して行動を起こす。

実装演習レポート【深層学習_前編】 Section2 AlphaGo

要点のまとめ

- AlphaGo LeeはPoicyNetとValueNetを使用する。
  また,NNではなく線形の方策関数であるRollOutPolicyも使用される（高速に算出）。
- AlphaGo Leeの学習ステップ
  ① 教師あり学習によるRollOutPolicyとPolicyNetの学習
  ② 強化学習によるPolicyNetの学習
  ③ 強化学習によるValueNetの学習
- モンテカルロ木探索が価値関数を更新するときに使用される。
- AlphaGo LeeとAlphaGo Zeroの違い
  ① 教師あり学習を一切行わず、強化学習のみで作成
  ② 特徴入力からヒューリスティックな要素を排除し、石の配置のみにした。
  ③ PolicyNetとValueNetを１つのネットワークに統合した。
  ④ Residual Netを導入した
  　ネットワークにショートカット構造を追加し勾配の爆発、消失を抑える狙い
  ⑤ モンテカルロ木探索からRollOutシミュレーションをなくした

実装演習レポート【深層学習_前編】 Section2 AlphaGo

参考図書レポート

- AlphaGoは、①探索型、②盤面評価、③戦術予測という３つのAIが同時に働くことによって、最適な指し手を導きだしている。

  ①の探索型では「モンテカルロ木探索」と呼ばれる「統計的に勝つ確率の高い一手」を計算するアルゴリズムを使っている。
  ②の盤面評価では戦況の良し悪しを判断する。盤面と勝利（目的）の関係性について考える。その上で、「勝てそうな特徴の盤面を作れるか」を考える。
  ③の戦術予測では「盤面がこれからどのような展開になるのか」を予測。
  盤面と未来の状況についての関係性を考える。

実装演習レポート【深層学習_前編】 Section3 軽量化・高速化技術

要点のまとめ

- 深層学習は多くのデータを使用したり、パラメータ調整のために多くの時間を使用する。
  そこで、複数の計算資源（ワーカー）を使用し、並列的にニューラルネットワークを構成することで、効率の良い学習を行いたい。
  データ並列化、モデル並列化、GPUによる高速技術は不可欠である。

- データ並列化
  親モデルを各ワーカーに子モデルとしてコピー。データを分割し、各ワーカーごとに計算。
  同期型と非同期型がある。
  同期型は各ワーカーの計算が終わるのを待ち、全ての結果がでたところで、親モデルのパラメータを更新。それに対して、非同期型では、各ワーカーごとでパラメータ更新する。
  非同期型の方が計算スピードは速いが、同期型の方が精度が高い。
- モデル並列化
  親モデルを各ワーカーに分割し、それぞれで学習。終わった後で、一つのモデルに復元。
  モデルが大きい時はモデル並列化、データが大きい時はデータ並列化をすると良い。
- GPU　比較的低性能なコアが多数。簡単な並列処理が得意で、NN学習と相性が良い。

実装演習レポート【深層学習_前編】 Section3 軽量化・高速化技術

要点のまとめ

- モデルの軽量化には３種類ある。
  モデルの軽量化とはモデルの精度を維持しつつパラメータや演算回数を低減する手法
  代表的な手法は３つ、量子化と蒸留とプルーリング。主には量子化がよく使われる。
- 量子化
  通常のパラメータの64bit浮動小数点を32bitなどの下位の精度に落とすことで、
  メモリと演算処理の削減を行う。しかし、精度は低下する。
- 蒸留
  精度の高いモデルはニューロンの規模が大きなモデルになっている。
  規模の大きなモデルの知識を使い軽量なモデル作成を行う。
  知識の継承により、軽量でありながら複雑なモデルに匹敵する精度のモデルを得る。
- プルーリング
  モデルの精度に寄与が少ないニューロンを削減することでモデルの軽量化、高速化が
  見込まれる。

実装演習レポート【深層学習_前編】 Section3 軽量化・高速化技術

参考図書レポート

- モデル圧縮方法
  ① Pruning（枝刈り）　② Quantize（量子化）　③ Distillation（蒸留）

① Pruning（枝刈り）
ディ―ニューラルネットワークのモデルは、各ノードが密に結合している。そのノード間の重みが小さい箇所の接続を削除する、または影響の小さいノードを削除することでパラメータ数を削減する手法。

③ Distillation（蒸留）

② Quantize（量子化）
重みなどのパラメータをより小さいビットで表現することで、モデルの軽量化を図る手法。使用するビットを制限することでネットワークの構造を変えずにメモリ使用量を削減。

③ Distillation（蒸留）
大きいモデルやアンサンブルモデルを教師モデルとして、その知識を小さいモデルの学習に利用する方法。これにより、大きいモデルに匹敵する精度を持つ小さいモデルを作ることが期待できます。

実装演習レポート【深層学習_前編】 Section4 応用モデル

要点のまとめ

- ディープラーニングモデルは精度は良いが、その分ネットワークが深くなり計算量が増える。
  ディープラーニングモデルの軽量化・高速化・高精度化を実現。（MobileNets)

- MobileNetsはDepthwise ConvolutionとPointwise Convolutionの組み合わせで
  軽量化を実現。
  Depthwise Convolutionはチャンネル毎に空間方向に畳み込む。すなわち、チャネル毎に
  $D_k \times D_k \times 1$のサイズのフィルターをそれぞれ用いて計算を行うため、その計算量は
  $H \times W \times C \times K \times K$ となる。
  次にDepthwise Convolutionの出力をPointwise Convolutionによってチャネル方向に
  畳み込む。すなわち、出力チャネル毎に$1 \times 1 \times M$ サイズのフィルターをそれぞれ用いて
  計算を行うため、その計算量は$H \times W \times C \times M$ となる。

- DenseNetとはDenseBlockと呼ばれるモジュールを用いたで前方の層から後方の層へ
  アイデンティティ接続を介して、学習が難しくなるという問題に対処した。

実装演習レポート【深層学習_前編】 Section4 応用モデル

要点のまとめ

- BatchNormではレイヤー間を流れるデータの分布を、ミニバッチ単位で平均が0・分散が1になるように正規化
  Batch Normalizationはニューラルネットワークにおいて学習時間の短縮や初期値への依存低減、過学習の抑制など効果がある。

- Batch Normの問題点
  Batch Sizeが小さい条件下では、学習が収束しないことがあり、代わりにLayer Normalizationなどの正規化手法が使われることが多い。

- Wavenet
  生の音声波形を生成する深層学習モデル
  Pixel CNN（高解像度の画像を精密に生成できる手法）を音声に応用したもの

実装演習レポート【深層学習_前編】 Section4 応用モデル

参考図書レポート

- WaveNetとはAI「AlphaGo」を作った、Google傘下のDeepMindが開発した。
  波形接続TTSではなくパラメトリックTTSを利用している。
- WaveNetは単純化や近似といった大きな劣化を生むプロセスがない為、従来型よりも
  自然な発音に大きく近づいた。
- WaveNetの誕生が可能になった土台には、インフラ的な側面とアルゴリズム的な側面が
  あるとされている。
  インフラ的な側面としては「データ量と計算能力」の圧倒的な向上があり、アルゴリズム的な
  側面として「音声を『点』の時系列として捉え。ディープラーニングに入力すること」が発展した
  ことにある。
- WaveNetはこの『点』を畳み込みニューラルネットワーク（CNN）で処理している。
  但し、音声信号特有の難しさとして、時系列の依存関係が非常に長い点がある。
  この長い依存関係を自己回帰モデルで扱うのは困難な課題であった。WaveNetでは
  dilated convolutionという仕組みを使って受容野が指数関数的に広くなるように、
  CNNを構築し、この問題を解決した。

実装演習レポート【深層学習_前編】 Section5 Transformer

要点のまとめ

- TransformerはRNNやCNNを使用せず、Attentionのみを用いるSeq2Seqモデル
  並列計算が可能なためRNNに比べて計算が高速な上、Self-Attentionと呼ばれる機構を用いることにより、局所的な位置しか参照できないCNNと異なり、系列内の任意の位置の情報を参照することを可能にしている。
- ニューラルネットワーク機械翻訳の問題点は長さに弱い。
  翻訳元の文の内容をひとつのベクトルで表現しているので、
  文長が長くなると表現力が足りなくなる。
  これに対して、Attention（注意機構）を使用することで、文長が長くなっても
  翻訳精度が落ちなくなる。
- Attentionとは辞書オブジェクト。
  query（検束クエリ）に一致するkeyを牽引し、対応するvalueを取り出す操作であると見做すことができる。

実装演習レポート【深層学習_前編】 Section5 Transformer

参考図書レポート

- 自然言語処理の文脈から誕生したTransformerはそのモデルのシンプルさにも関わらず、大きな成果をあげることに成功しました。そのため、その後 自然言語処理にブレイクスルーをもたらしたBERTやGPT-2などのモデルはTransformerをもとに作られています。
- Transformerはそれまで自然言語処理の世界で主流であったRNN（及びCNN）を利用するモデル構造をやめ、Attentionのみをもちいてエンコーダ-デコーダ型のモデルを設計しました。
- Transformerの特徴
  ① RNNやCNNを使わずAttention層のみで構築。
  ⇒ 計算の高速化
  ② PositionEncoding層の採用
  ⇒ 文脈情報の保持
  ③ Attention層におけるQuery-Key-Valueモデルの採用
  ⇒ より正確な変換（翻訳）

実装演習レポート【深層学習_前編】 Section6 物体検知　SS

要点のまとめ

- 物体認識には４種類ある。
  ①　分類：画像に対し単一または複数のクラスラベル
  ②　物体検知：Bounding Box
  ③　意味領域分割：各ピクセルに対し単一のクラスラベル
  ④　個体領域分割：各ピクセルに対し単一のクラスラベル（各個体まで分割する）
- 代表的なデータセットとして、VOC12，ILSVRC17，MS COCO18，OICOD18の４種類がある。
  クラス、Train＋Valのデータ量、Box/画像の3種類でそれぞれ異なる。
  特にBox/画像が小さいものはアイコン的な映りであり、日常感とはかけ離れやすい。
  逆に、Box/画像が大きいものは部分的な重なり等も見られて、日常生活のコンテキストに近い。
- 評価指標としては、ConfusionMatrixを使用する。PrecisionとRecall。
  Threshold（閾値）を変えて物体検出の精度を確認する。
  物体検出においてはクラスラベルだけでなく、物体位置の予測精度も評価したい。
  その時に用いられる評価指標が、IOU（Intersection over Union)である。

実装演習レポート【深層学習_前編】 Section6 物体検知　SS

参考図書レポート

- Semantic Segmentationとは、各ピクセルをその意味（周辺のピクセルの情報）に基づいて、カテゴリ分類する手法。
  セグメンテーションは、機械モデルに人間と同様の認識能力を学習させるため非常に重要な技術の一つ。実際、人間は物体を認識する際、自然と各物体の境界を把握し、各物体を識別しています。
  物体の縁を囲う枠（境界線）の幅を小さくすれば小さくするほど、より精密な認識が可能になると言えます。
  セグメンテーションを精密・細分化することで画像認識の精度全体を底上げすることができ、誤作動を減らすことに貢献することでしょう。