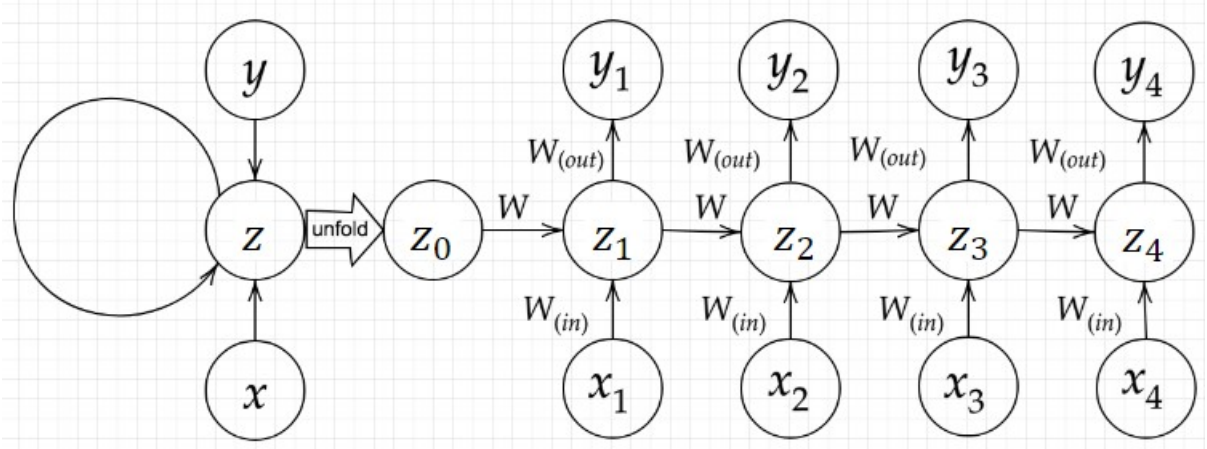
Study AI レポート

深層学習 Day3

Day3Section1:再帰型ニューラルネットワークの概念

RNN とはリカレントニューラルネットワークの事で、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 であり、会話や天気予報、株価など日常のあらゆる場面に存在している。再帰型とは一度使用したデータを保持、記憶しその情報に基づいて新しい事象を処理できる事に特徴がある。
関連する重要な学習アルゴリズムとして BPTT（Back Propagation Through Time ）がある。基本的な考えは全損失 L を時間 t ＝ 1 から t ＝ T までの全ての損失関数の総和であると考える点である。

確認テスト考察
・RNN は、入力→現在の中間層用、中間層→出力用、及び中間層→中間層への 3 つの重みがある



前の中間層から中間層にかかる重み W が一番重要な RNN の特徴を示している

・RNN の特徴として、時系列モデルを扱うには、初期の状態と過去の時間 t-1 の状態を保持し、そこから次の時間での t を再帰的に求める再帰構造が必要になる。その再帰構造を持っている事が重要な特徴である。

$$v^t = W_{(out)}z^t + c \qquad y^t = g\left(W_{(out)}z^t + c\right)$$

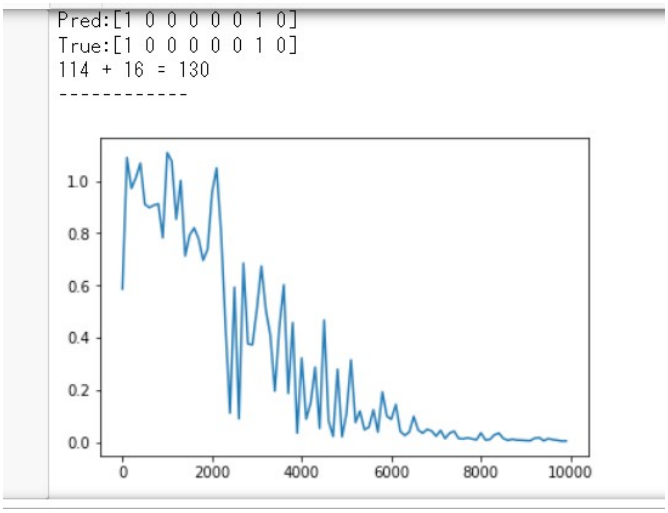
確認テスト 2　下図の y1 を x・s0・s1・win・w・wout を用いて数式で表せ。※バイアスは任意の文字で定義せよ。※また中間層の出力にシグモイド関数 g(x)を作用させよ

Y1= g(Wout*S1 + C)
S1 = f(Win*X1 + WS0 + b)

複雑な式も分解していけば単純な形として理解しやすくなる。

コード確認　simpleRNN の実行

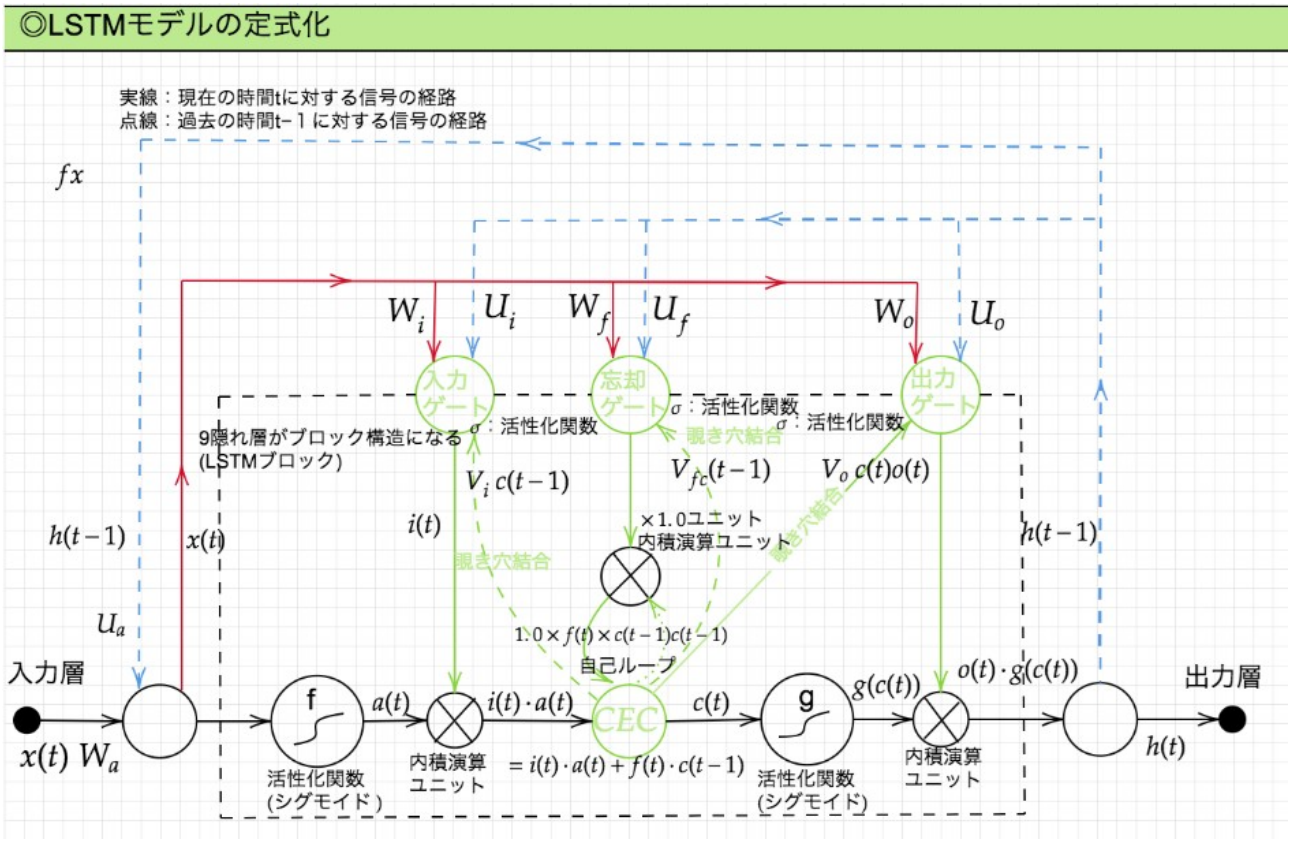
iter9900 の結果　学習が進んで良好な結果が取得できていることが右図の結果からわかる。



バイナリ加算というシンプルな例を用いて、RNN の機能を理解していくという方法が理解しやすかった。Xe や Xavier の初期化も試し、勾配爆発もどのように発生するのかグラフの実例をみながら実感する事ができた。

Day3Section2:LSTM

RNN の課題として時系列を遡れば遡るほど、勾配が消失していく。勾配消失を計算手法で対策する方法とは、別に構造自体を変えて解決したものが LSTM である。また学習を繰り返すごとに勾配が指数関数的に増大してしまう問題もある。Long　Short -Term Memory は 1997 年に提唱された。LSTM の構成要素はメモリセルである。メモリセルは基本的には隠れ層を表す。各メモリセルには勾配消失と勾配発散問題を克服するためのリカレントエッジが存在する。記憶機能を持つ CEC、忘却ゲート、入力ゲート、出力ゲートを持つことが特徴である。



確認テスト考察

シグモイド関数を微分した時、入力値が 0 の時に最大値をとる。その値として正しいものを選択肢から選べ。 →最大値 0.25　これは何度も学習し、この最大値が 1/4 にさせる 0.25 までしかない事が勾配消失につながる事が理解できた。

チャレンジ問題考察

RNN や深いモデルでは勾配の消失または爆発が起こる傾向がある。勾配爆発を防ぐために勾配のクリッピングを行うという手法がある。具体的には勾配のノルムがしきい値を超えたら、勾配のノルムをしきい値に正規化するというものである。以下は勾配のクリッピングを行う関数である。

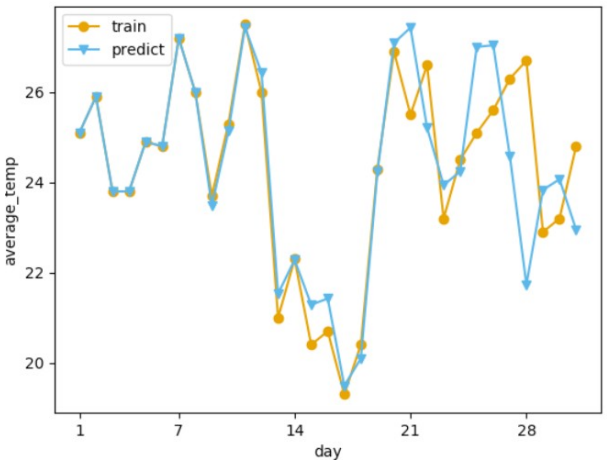
勾配のノルムがしきい値より大きいときは、勾配のノルムをしきい値に正規化するので、クリッピングした勾配は、勾配×(しきい値/勾配のノルム)と計算される。

考察）Python のプログラムの問題では、問題文から意味をとらえて数式に落とし込む能力が確認されることが良くわかった。

確認テスト考察
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
→忘れるためなので、忘却ゲートが作用する。

In [ ]:
```
def rnn_model_maker(n_samples, n_out):
    # 3層RNN(リカレントネットワーク)を定義
    model = Sequential()
    # 中間層(RNN)を定義
    model.add(LSTM(units=n_units, input_shape=(n_rnn, 1), dropout=r_dropout, return_sequences=False))
    # 出力層を定義(ニューロン数は1個)
    model.add(Dense(units=n_out, activation='linear'))
    # 回帰学習モデル作成
    model.compile(loss='mean_squared_error', optimizer='rmsprop')
    # モデルを返す
    return model
```

Keras での LSTM テスト



Keras での LSTM のコードテスト
RNN よりも LSTM のほうが予測の精度が上がっている事が確認できた。

Day3Section3:GRU

LSTM では、パラメータ数が多く計算負荷が高くなる問題がある。これはパラメータが多数存在していたため、計算負荷が大きかった事に起因している。GRU 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様にした構造 である。GRU は記憶用のＣＥＣがなく、更新ゲートとリセットゲートから構成されている。ＬＳＴＭに比較して計算量が少ないのが特徴である。

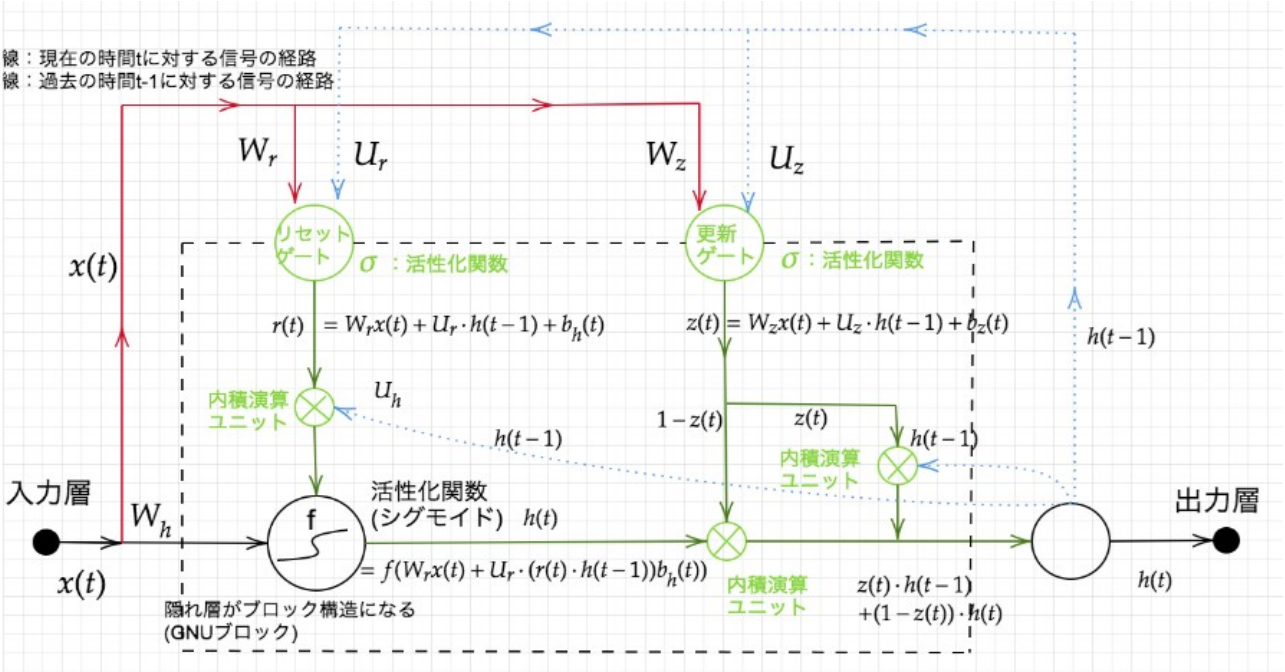
確認テスト考察
・LSTM と CEC が抱える課題について、それぞれ簡潔に述べよ。

LSTM は入力、出力、忘却ゲート、CEC と 4 個のコンポーネントを持ち、パラメータが非常に多い。CEC は勾配を 1 にしたため、学習能力がないのが課題である。総合的に複雑な構成でパラメータが非常に多い事がこれらの課題である。
→よくこんなに複雑な事仕組みを思いついたものだと感心した。忘却ゲートといった人間的なキーワードがでてきたが、また生物の忘却する仕組みとは異なったシステムになっていると直感的に感じた。

・LSTM と GRU の違いを簡潔に述べよ

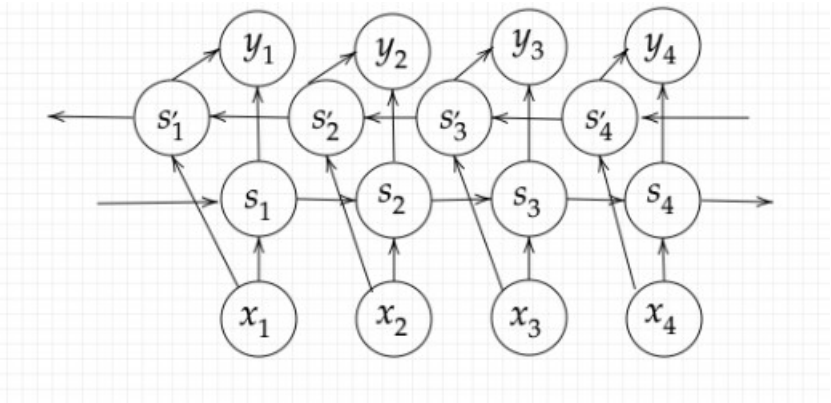
LSTM は入力、出力、忘却ゲート、CEC をもちパラメータが多い。GRU は CEC がなく、更新ゲートとリセットゲートを持つ。結果 GRU のほうがパラメータが少なく LSTM に比較して計算量が少なくなるのが特徴である。

→LSTM の仕組みは面白いと感じたが、計算量や複雑性を考えると確かにあまり実用的ではない雰囲気を感じた。ＧＲＵは構成をシンプルにして、精度を同等かそれ以上に出せる方法で、やはり物事をシンプルに解決できる方法はエレガントで主流になると思われる。



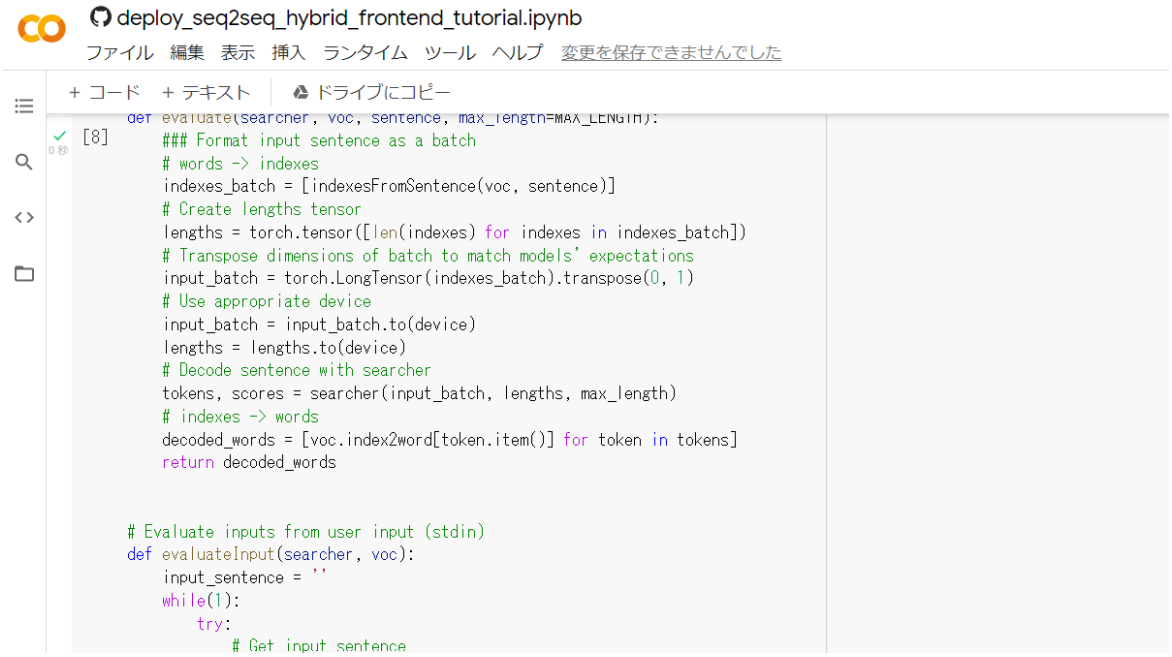
Day3Section4:双方向 RNN

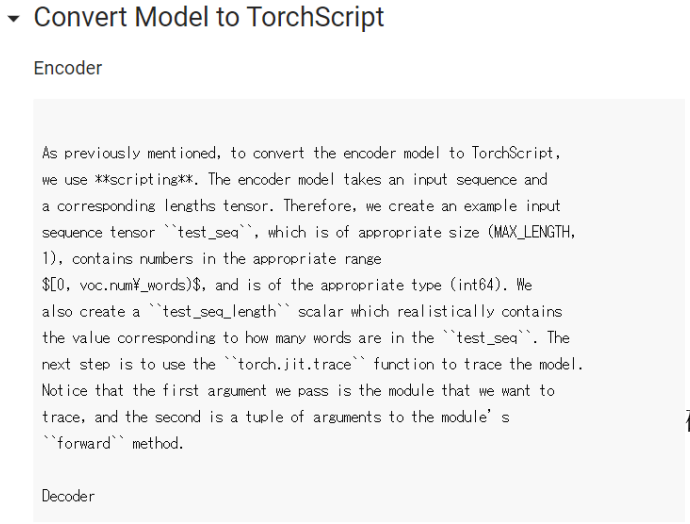
過去から未来の情報だけでなく、未来から過去への方向も考慮した再帰型ニューラルネットワークのこと。例えば文章内の文字の穴埋めタスクなど、過去から現在だけでなく未来から現在までの系列情報を用いる事が有効と考えられるタスクに用いられる。未来の情報を加味することで精度を向上させるためのモデルで実用例としては文章の推敲や、機械翻訳等がある。



Day3Section5:Seq2Seq

Seq2seq とは、Encoder-Decoder モデルの一種である。機械対話や機械翻訳などに使用されている。Encoder RNN はユーザーがインプットしたテキストデータを、単語等のトークンに区切って渡す構造をもつ。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 Taking、ID から、そのトークンを表す分散表現ベクトルに変換する Embedding などの機能がある。Encoder RNN:ベクトルを順番に RNN に入力していく・vec1 を RNN に入力し、hidden state を出力。この hidden state と次の入力 vec2 をまた RNN に入力してきた hidden state を出力という流れを繰り返す事で文章の。これを逆にしたのが **Decoder** になる



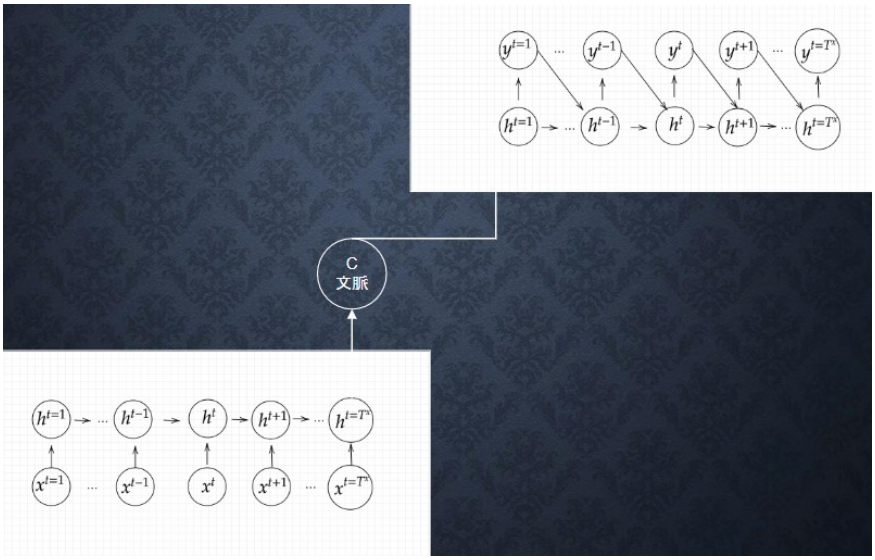


Pytorch のチュートリアルサンプルを動作させて確認した

確認テスト

RNN を用いた Encoder-Decoder モデルの一種であり、機械翻訳などのモデルに使われる。

VAE に関する下記の説明文中の空欄に当てはまる言葉を答えよ。
自己符号化器の潜在変数に____を導入したもの。　**確率分布を導入した物平均が Z で分散が 1 であることがポイント**



Day3Section6:Word2Vec

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。固定長形式で単語を表す必要がある。 学習データからボキャブラリを作成する I want to eat apples. I like apples. {apples,eat,I,like,to,want}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量で実現可能にしている。 Word2Vec には、埋め込み行列を獲得するための方法として Continuous Bags-of-Words と Skip-Gram の二つの NN が用意されている。CBOW は前後の単語からある単語を予測するための NN である。一方 Skip-Gram はある単語が与えられた時に、その周囲の単語を予測するための NN である。

Day3Section7:Attention Mechanism

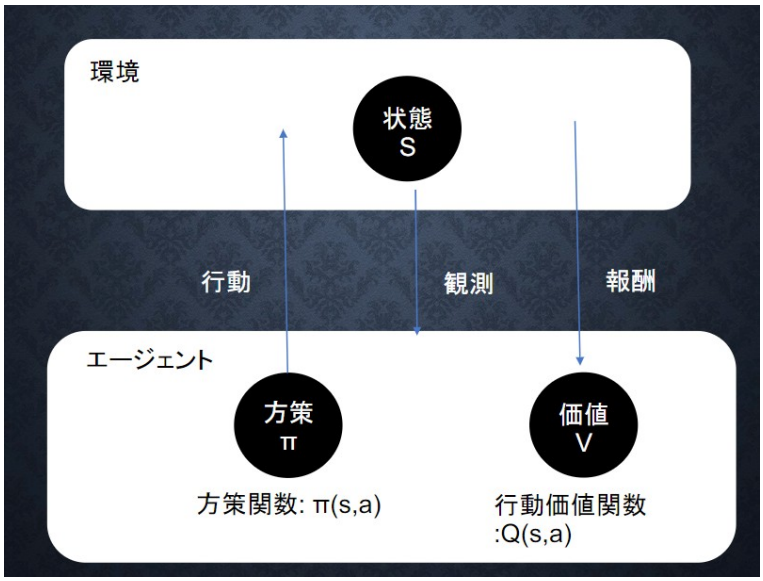
seq2seq の問題は長い文章への対応が困難である。seq2seq では、2 単語でも、100 単語でも、固定次元ベクトルの中に入力しなければならない。 文章が長くなるほどそのシーケンスの内部表現の次元も大きくなっていく仕組みが必要になる。Attention Mechanism は「入力と出力のどの単語が関連しているのか」の関連度を学習する仕組み。

確認テスト
RNN と word2vec、seq2seq と Attention の違いを簡潔に述べよ seq2seq は 1 問一答しかできない。Word2vec は小さな単語表現に置換える。Attention は入力と出力の関連度を学習する仕組みである。

→自然言語処理は、時系列をもったデータを扱うため 1 枚の画像のみを対象とする方法と比較して複雑な仕組みが多くある事がわかった。一つポイントだと感じたのは、これらの方法は全て”意味”を理解してアウトプットを出すものではなく、あくまでディープラーニングのニューラルネットワークを基本技術として、経験則的に対応していく仕組みである所である。意味を説明できるディープラーニングなどが最近発表されたりしているが、この技術はまた別の工夫が必要になると考えられる。

Day4Section1:強化学習

強化学習とは長期的に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であり、行動の結果として与えられる利益(報酬)をもとに、行動を決定する原理を改善していく仕組み。マーケティングの行動改善などの例が分かりやすい。前提条件として、何をしたら最大利益が得られるかは分からない状態で、不完全な知識を元に行動しながら、データを収集し、最適な行動を見つけていく事である。



他の教師ありなし学習と区別される理由は、教師なし、あり学習では、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・強化学習では、優れた方策を見つけることが目標になるので目標が異なっている。

価値関数：価値を表す関数としては、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。状態と価値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：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の確率を与える関数のこと
方策勾配法を使う。
→関数があれば、ニューラルネットワークが適応できる発想が面白いと思った。そして損失関数ではなく収益最大化関数を用意してやれば、そのままＮＮが適用されることが分かった。実際の応用では、この方策関数、方策勾配法の定義が難しいと感じた。

Day4Section2:Alpha Go

2016 年に囲碁のチャンピオンに買った事で世間の耳目を集めたアルゴリズム。Alpha Go Lee と Alpha Go Zero の二つがある。Alpha Go Lee は PolicyNet と ValueNet を持っており、最初に

RollQutPolicy という精度は少し落ちるけど高速に学習できるネットワークを使って、人間の棋譜をもとに教師あり学習を行い、そこから発展させていった方法である。それと比較して、Alpha Go Zero は人間がこれが役に立つ要素だ、と判断するヒューリスティックな要素を排除し、初めから強化学習のみを使って行われている。Zero の特徴的なところは ResidualNet といって途中で学習をスキップするモデルが入っている事、WideResNet といって途中でネットワークを二つに分離する工夫が入っている。
→基本的なディープラーニングの要素は 1）畳み込み 2）プーリング 3）RNN 4）Attention であり、その他の物は仰々しい名前がついているが、基本この 4 要素の組み合わせである、という所がとても分かりやすく納得がいった。

Day4Secion3:軽量化・高速化技術

高速化技術としては、GPU を使う方法、並列処理を行う方法があげられる。GPU は CPU に比較して色々な種類の計算に対応は出来ないものの、簡単な行列計算ができるユニットが大量に入っているという特徴があるので、ニューラルネットワークの計算に向いている。
また並列処理では、データを複数に分割して並列処理する方法があり、各ワーカーの処理を同期させて処理する方法と非同期でやる方法があるが、精度面から現在は同期させて計算させる方法が主流である。それ以外にモデルを分割して並列処理する方法もある。分離の仕方は複数のニューラルネットワークをそれぞれ別のワーカーに割り振ってやる方法が現実的である。

量子化は、重みの計算を何ビットの計算で実施するか？という方法であり、16bit,32bit,64bit の選択肢があるが、現在 16bit で精度的にもほぼ十分であるという事が言われている。パラメータのデータ量が減れば直接計算負荷を減らすことにつながるので、高速化と軽量化が図れる。

その他の軽量化技術として、蒸留という方法がある。これは既に完成している精度の高い大きなモデルを使って軽いモデルを作る方法である。教師モデルと生徒モデルを作って生徒モデルを改善していくことで、軽量で高速なモデルが作成できる事が知られれている。

プルーニングは、結果に大きく寄与している重要なニューロンだけを残してモデルの圧縮を行う方法である。やり方は特定の閾値以下の重みを削減していって精度がどうなるかを確認していく。ある実例では、重みを 9 割も減らしても精度が 90%程度残っている事も確認されている。

Day4Secion4:応用モデル

MobileNet・・ 小さな計算量でいい精度をだすための工夫がなされたネットワークの事。ポイントは DepthWise Convolution と PointWise Convolution の二つを定義し、その二つを通常の Convolution の変わりに使用することで計算量を大きく減らしながら精度を保つ事を実現している。DepthWise Convolution の特徴はフィルターが 1 個に限定されていて、通常の Convolution の計算量が HxWxKxKxCxM になるのに対し HxWxCxKxK で済む。また PointWise Convolution ではカーネル

サイズ 1 で畳みこみ、フィルターを M 個で計算するので、計算量は HxWxCxM になる。この工夫によりメモリの使用量を 1/40 程度にまで削減できる例も知られている

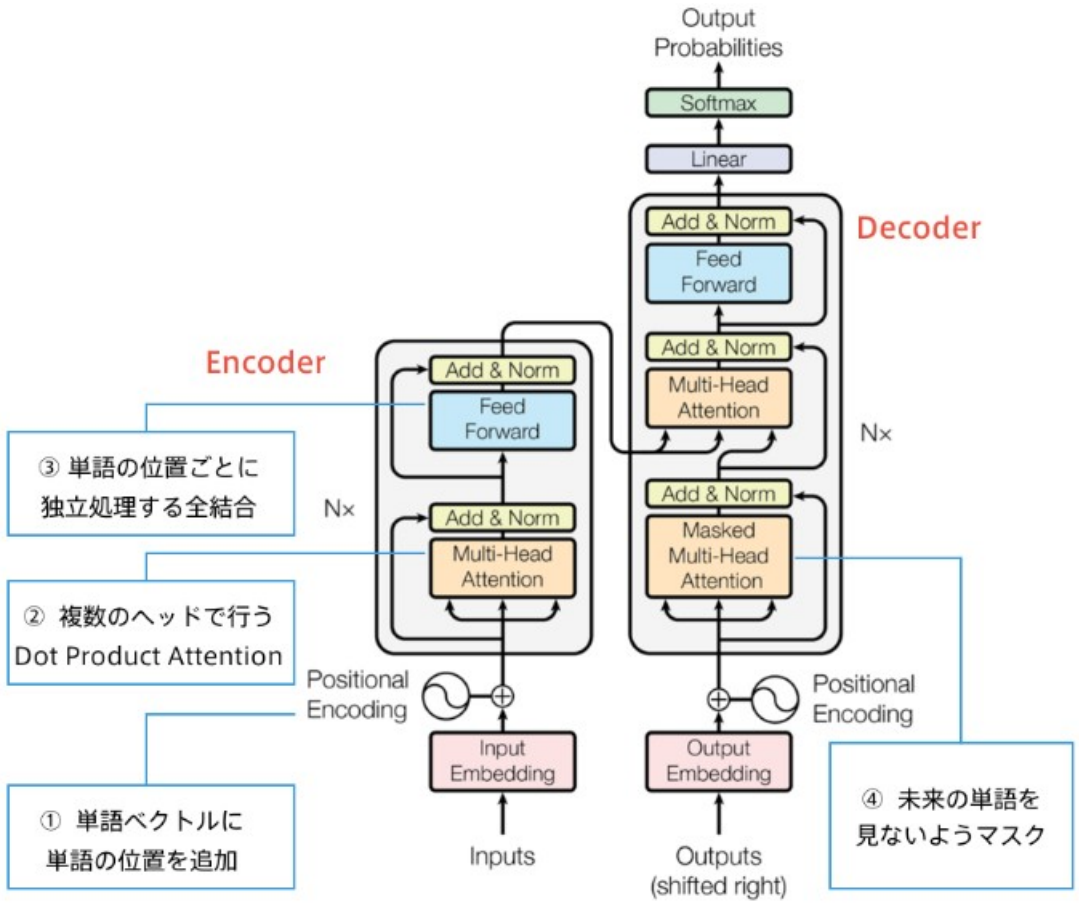
DenseNet・・Dense Block という BatchNorm,Relu,3x3Convolultion からなるブロックをネットワークの間にもつ特徴を持つ。この Dense Block の中を通るごとに人がハイパーパラメータとして決めた growth Rate K 個づつチャンネルが増えていく。そしてこの増えたチャンネルを Transition Layer でもとに戻す構造をもつ。ResNet との違いは、ResNet は過去 1 層の情報を使用するのに対して DenseBlock は過去の全ての層の情報を使用するところが異なっている。

BatchNorm・・決められたある塊バッチごとに、平均がゼロ分散が 1 になるように正規化を行う方法である。実際の使用時の問題点は、CPU や GPU などで保有しているメモリ数が異なるためにバッチサイズをそろえる事が困難である点である。そのため LayerNorm や InstanceNorm といった BatchNorm のように n 真っ当な正規化でなく、ある単位で切り取った正規化を行われる方法がある。深層学習でよくある話であるが、理由ははっきりしないけど、やってみたら効果があったのでデフォルト的になっているのがこの LayerNorm や InstanceNorm である。

WaveNet・・音声生成モデルに使用されているネットワーク。特徴は音声と言った時系列データに足しいてＲＮＮではなくコンボリューションを適用している事と、パラメータに対する需要野が広い、つまりは長い時間幅に対してのデータを、スキップをかませ間を間引いたりしながらネットワークを作成することで実現している。Dilated casual convolution と定義されている。

Day4Secion5:Transformer

Transformer（Encoder-Decoder x Attention）は、2017 年に提案されたモデルで、RNN を使わずにアテンションのみで系列データを扱う点が最大の特徴。再帰結合がないため計算効率が高く、翻訳タスクにおいて最高性能を達成し重要なモデルに位置づけられている。アテンションは m 個の訓練データがある場合に、任意の説明変数 x に対して予測値 y を求める考え方を示している。アテンションでは Query,Key,Value に分けて、クエリに近いキーを探し、そこに記述されている本文バリューを取り出すような辞書に近い構成になっている。



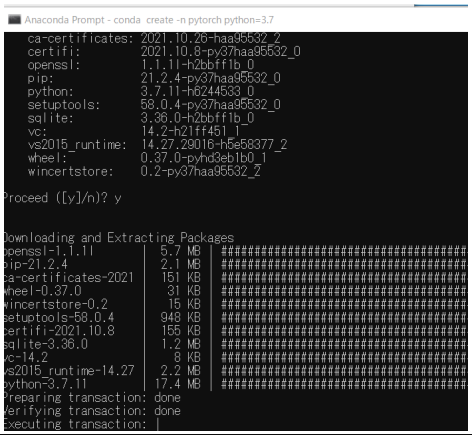
サンプルコードチェック

Day4Secion6:物体検知・セグメンテーション

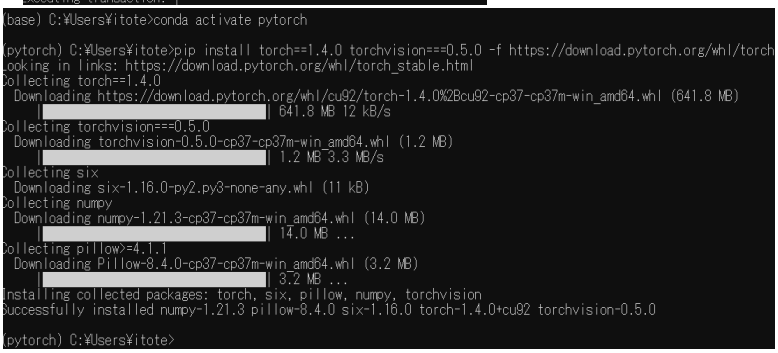
物体認識は、１分類、２物体検知、３意味領域分割、４個体領域分割の４種類に分かれる。物体検知は画像の中に人、車、犬など何がうつっているかを検知する方法である。意味領域分割は、さらに画像のどこに風船が映っているかなどを判別する技術で、個体領域分割は、更に赤い風船、青い風船など個別の個体まで認識してバウンディングボックス（BB）で、その領域を教示する手法である。

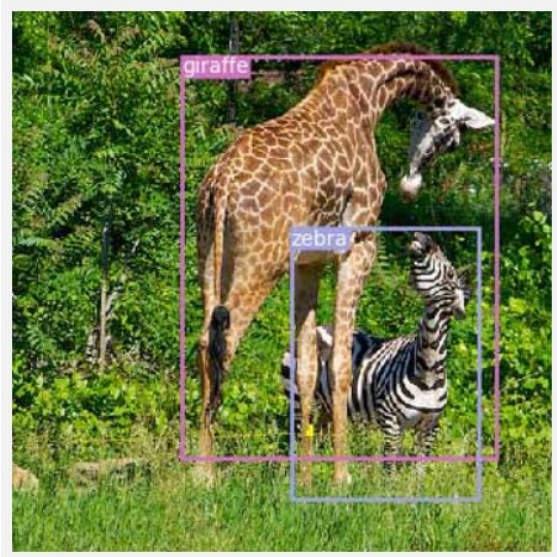
画面に対して、この領域が標識である、なと人が正しいラベルをつけた部分を GrandTruth と定義し、推論によって、この領域が何%の自信をもって標識である、とする Conf という値や、Area of Overlay /Area of Union といった IOU（インターセクションオブユニオン）といった評価指標がある。
各ピクセル毎に意味をもって分割できる方法をセマンテックセグメンテーションという。例えば犬と猫が映っている画像があった場合に、このピクセルが犬、このピクセルは猫と領域分けをする。
有名なアルゴリズムとしては SingleShotDetection （SSD）は１枚の画像から、画像のどこに何が映っているか？を出力する方法である。このネットワークは VGG １６から発展した手法である。物体認識には、検出とクラス分類を同時にやる１段階検出と、検出とクラス分類を別々に行う二段階検出がある。精度と計算負荷のトレードオフになるので、ネットワーク特性の理解が必要である。DSSD という手法は LesNet がベースになっており、使用技術の原型ベースを理解しておくことは重要である。

PyTorch-YOLOv3 でのコードテスト



conda で仮想環境を構築してテスト
トーチビジョンをインストールした





【感想】
一昔前なら夢の技術だったのが、ここまで無料でモジュールが後悔されていることに驚きを覚えた。
そして今回の学習を通じて、基本をしっかり押さえていること、随時変化しているネット上のコードやモジュールを使いこなせるだけの知識とスキルを得るためには、実際にコードを動かして行くことが必須であることが理解できた。