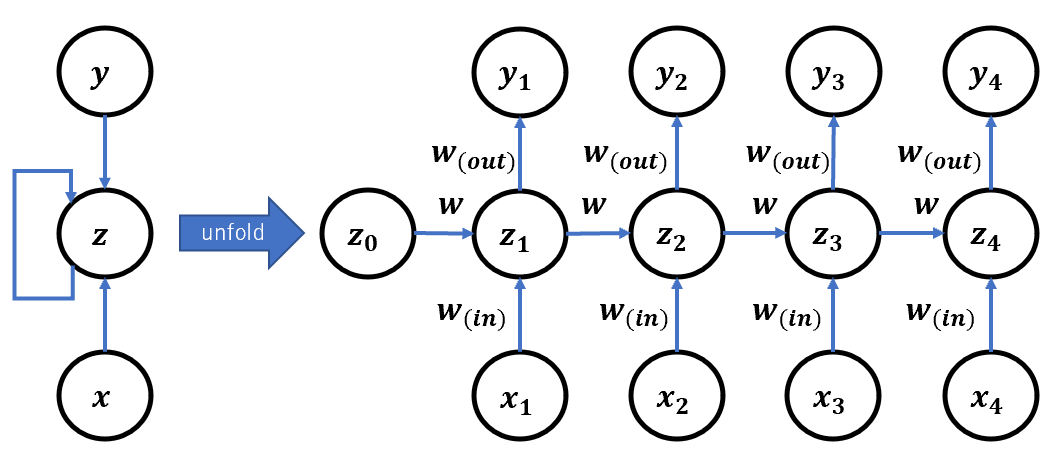
1. **再帰型ニューラルネットワークの概念**

**再帰的ニューラルネットワーク（Recurrent Neural Network: RNN）は、回帰型・循環型とも呼ばれ、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である。時系列データとは、ある時間の経過とともに値が変化していくようなデータを指し、店舗の日次売上データ、ホームページのアクセス数履歴、工場設備のセンサデータなど、多種多様なデータが時系列データとして表現される。また、音声データやテキストデータなども時系列データに含まれる。**

* **確認テスト**
* **サイズ5×5の入力画像を、サイズ3×3のフィルタで畳み込んだ時の出力画像のサイズを答えよ。なおストライドは2、パディングは1とする。**

**出力サイズの高さ：3、出力サイズの幅：3**

1. **RNN全体像**

****

* **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。**

**前時刻の中間層から現在の中間層を定義する際にかけられる重み**

1. **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である。これにより、RNNを、時間経過を含めた1つの大きなニューラルネットワークとみなすことで、誤差逆伝播法を適用することが可能となる。　この方法によって求めた勾配情報により、勾配降下法を使ってのチューニングを行う。誤差逆伝播法では下記数式を使用する。**

**誤差関数（二乗誤差関数）：**

**出力層の活性化関数（恒等写像）：**

**総入力の計算：**

**微分の連鎖率の式：**

**各関数を微分すると下記となる。**

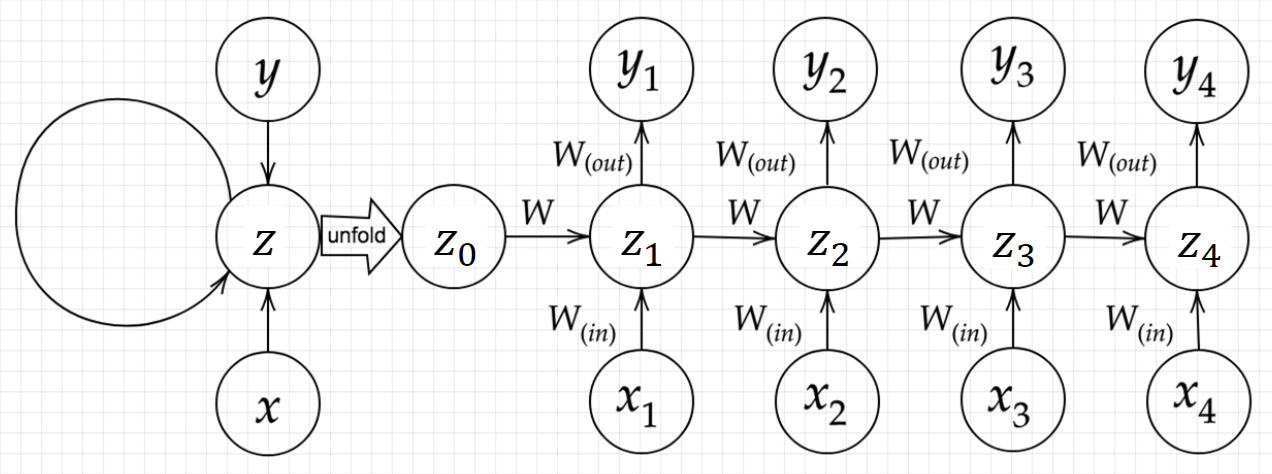
**BPTTを数式で表すと下記となる。**

**パラメータの更新式は下記となる。**

* **確認テスト**
* **連鎖律の原理を使い、を求めよ。**
* **確認テスト**
* **下図のを、を用いて数式で表せ。**

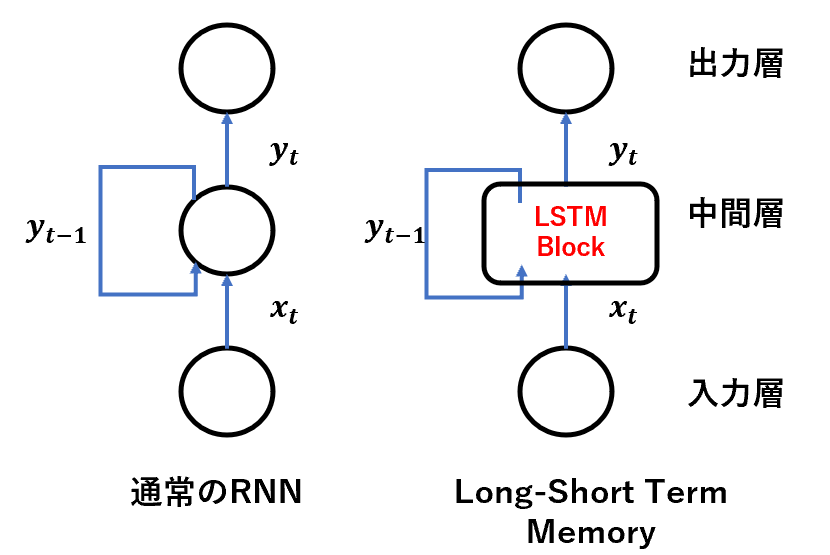
**※バイアスは任意の文字で定義せよ。**

**※また中間層の出力にシグモイド関数g(x)を作用させよ。**

****

1. **LSTM（Long short term memory）**

**長い時系列のデータの場合(Tが大きい場合)、計算上の問題で上層からの誤差が薄まったり逆に非常に大きくなったりする。値が大きくなる分には最大値の制限で何とかなるが、消えてしまうのはどうにもならないため、誤差が減衰しないよう伝播させるというのがLSTMの思想である。LSTMは、RNNの拡張として1995年に登場した、時系列データ(sequential data)に対するモデル、あるいは構造(architecture)の1種である。その名は、Long term memory(長期記憶)とShort term memory(短期記憶)という神経科学における用語から取られている。LSTMはRNNの中間層のユニットをLSTM blockと呼ばれるメモリと3つのゲートを持つブロックに置き換えることで実現されている（Qiitaより）。**

****

1. **CEC（Constant Error Carousel）**

**CECは誤差消滅問題や勾配爆発に対応するために導入された非常にシンプルな解決方法で、勾配が1であれば誤差は正しく伝播して解決できる。**

**※勾配爆発とは、勾配が層を逆伝播するごとに指数関数的に大きくなっていくことである。**

**CECの課題は、入力データについて、時間依存度に関係なく重みが一律であり、ニューラルネットワークの学習特性が無いということである。**

1. **入力ゲートと出力ゲート**

**入力・出力ゲートの役割は、入力・出力ゲートを追加することで、それぞれのゲートへの入力値の重みを、重み行列W,Uで可変可能とし、CECの課題を解決する。**

1. **入力重み衝突（input weight conflict）**

**一般のRNNを考えたとき、ユニットからの入力は、重みを与えられてユニットに入力される。誤差逆伝播法を用いる場合誤差信号がユニットを遡り、必要な場合重みを更新する。ところが、時系列データを学習する場合は次の矛盾する重み更新を同時に受ける場合がある。**

1. **ユニット を活性化されることによる入力信号の伝達**
2. **ユニット からの無関係な入力によってユニット の値が消去されることを防ぐ入力信号の保護**

**[i]はが入力されたことを示す情報を次々に未来のユニットに伝達するためにの値を大きくする場合を指す。一方で無関係な入力を受けてユニットの値が更新されても困るので、[ii]のようにの値を小さくしたい場合もある。従来型のRNNではこのような矛盾する重み更新が頻発し、学習を遅らせる主要な要因となっていた。そこで、LSTMでは入力ゲートを導入し、追加の重みパラメターを持たせることで、「前のユニット(1つ前の時間のユニット)の入力を受け取るか否か」を判断させるようにした。そうすることで必要に応じて誤差信号の伝播をゲート部で止め、必要な誤差信号だけが適切に伝播するようにゲートを開いたり閉じたりする。**

1. **出力重み衝突（output weight conflict）**

**出力ゲートも入力ゲートと同様に、以下の理由による重みの衝突を防ぐために導入されている。ユニット の情報が重みに従いユニット に出力を行うことを考えると、次の2つの衝突が起こる。**

1. **ユニット を活性化されることによる出力信号の受信**
2. **ユニット からの無関係な出力によってユニットの値が消去されることを防ぐ出力信号の保護**
3. **忘却ゲート**

**複数の時系列タスクにおいて目覚ましい成果を上げたLSTMだが、内部メモリセルの更新は線形で、その入力を貯め込む構造であった。入力系列のパターンががらりと変わったとき、セルの状態を一気に更新する術が無かった。そこで、忘却ゲート(Forget Gate)が導入された。忘却ゲートは、誤差信号を受け取ることで、一度メモリセルで記憶した内容を一気に「忘れる」ことを学習する。状態遷移が起こり、今までの記憶が不必要になった時点で素早くセルを初期化することを可能にした。**

1. **覗き穴結合（peephole connection）**

**忘却ゲートを導入して完成に見えたLSTMだが、ゲートの制御に関して次の致命的な問題を抱えていた。3つの制御ゲートの役割は、「メモリセルの内容を書き換えるか/忘れるか/出力するか」ということにあった。しかしこれまでのLSTMでは、そのゲートの制御はLSTMの外側と呼べる(1)入力層の出力(2)中間層の1ステップ前の出力をベースとして行われており、制御対象であるメモリセル自身の内部状態は制御に利用されていない状態だった。例えば、一見すると中間層出力であるがメモリセルの情報をすべて含んでいるように見えますが、出力ゲートが出力を遮断している場合メモリセルの真の状態は隠ぺいされる。そこで覗き穴結合（peephole connection）と呼ばれる接続をメモリセルから各ゲートに流し込むことで解決を図った。**

* **確認テスト**
* **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。**

**（1）0.15（2）0.25（3）0.35（4）0.45**

* **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\_\_\_\_。」**

* **忘却ゲート**

1. **GRU（Gated Recurrent Unit）**

**LSTMと同様にシンプルなRNNの勾配爆発や、勾配消失の欠点を補うために開発されたゲート付きRNNで、LSTMよりゲート数が少ない構成となっており、その分パラメータ数も少なく計算量が少なくて済む、といった特徴を持つ。**

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

**LSTMはパラメータの数が多く計算コストも高い。また、CECはパラメータ更新が行われない。**

* **確認テスト**
* **LSTMとGRUの違いを簡潔に述べよ。**

**LSTMは入力ゲート、出力ゲート、忘却ゲートを持っていて、パラメータの数が多い。**

**GRUはリセットゲートと更新ゲートを持っていて、パラメータの数が少ない。**

1. **双方向RNN（Bindirectional RNN）**

**通常のRNNは過去の情報を保持することで時系列データの学習を実現していた。双方向RNNの場合は、過去の情報に加えて未来の情報も加味させ、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。また、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。**

1. **Seq2Seq（sequence to sequence）**

**Seq2Seqは、EncoderとDecoderを備えたEncoder-Decoderモデルを使って、系列データを別の系列データに変換するモデルのことを指す。これは発話・応答のシーケンスのペアを学習させることで、発話から応答を生成するモデルである。**

1. **Encoder**

**InputData(画像、テキスト、音声、動画)を何かしらの(固定長)特徴ベクトルに変換する機構のこと。**

1. **Decoder**

**Encoderでエンコードされた特徴ベクトルをデコードして何か新しいデータを生む機構のこと**

1. **HRED（Hierarchical Recurrent Encoder-Decoder）**

**Seq2Seq は一問一答だが、これを過去の n-1 個の発話から次の n 個目の発話を推測するようにしたのが HRED である。Seq2Seq は Encoder RNN, Decoder RNN の2段構成だが、 HRED は Encoder RNN, Context RNN, Decoder RNN の3段構成である。**

1. **Encoder RNN: 一つ一つの文章（会話なら過去の一つ一つの発言）をそれを表すベクトルに変換する。**
2. **Context RNN: Encoder のまとめた各文章の系列をまとめて、これまでの会話コンテキスト全体を表すベクトルに変換する。**
3. **Decoder RNN: Context RNN の情報から応答を生成する。**

2 の Context RNN というレイヤーがあることによって、過去の発話の履歴を加味した返答をできるようになっているということである。

1. **VHRED**

**HRED と同じく過去の n-1 個の発話を与えられて、 n 個目の発話を生成する。しかし、 HRED は対話学習において下記の問題を持っていた。**

1. **HRED は確率的な多様性が字面にしかなく、会話の「流れ」のようなロングタームな多様性が無い。これは、Encoder RNN, Context RNN, Decoder RNN のうち確率的な処理が Decoder RNN の次ステップの単語を生成する部分にしか無いため、同じコンテキスト（発話リスト）を与えられても、答えの内容が毎回会話の流れとしては同じものしか出せない。**
2. **HRED は短く情報量に乏しい答えをしがちである。同じコンテキスト（発話リスト）を与えられても、それに続く発話は全く異なるものでありうるし、これらを決定論的に学習しようとすると、結果「無難な」答えつまり短いよくある答えを学ぶ傾向がある。**
3. **VAE**
4. **オートエンコーダ**

**教師なし学習の一つで、入力データと同じデータをNNの出力から得られるようにします。**

1. **VAE（Variational Auto encoder）**

**VAEはディープラーニングによる生成モデルの１つで、訓練データを元にその特徴を捉えて訓練データセットに似たデータを生成することができる。VAEはこの潜在変数に確率分布、通常を仮定したところが大きな違いである。通常のオートエンコーダーだと、何かしら潜在変数にデータを押し込めているものの、その構造がどうなっているかはよくわからない。VAEは、潜在変数を確率分布という構造に押し込めることを可能にする。**

* **確認テスト**
* **下記の選択肢から、seq2seqについて説明しているものを選べ。**

1. **時刻に関して順方向と逆方向のRNNを構成し、それら2つの中間層表現を特徴量として利用するものである。**
2. **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。**
3. **構文木などの木構造に対して、隣接単語から表現ベクトル（フレーズ）を作るという演算を再帰的に行い（重みは共通）、文全体の表現ベクトルを得るニューラルネットワークである。**
4. **RNNの一種であり、単純なRNNにおいて問題となる勾配消失問題をCECとゲートの概念を導入することで解決したものである。**

* **確認テスト**
* **seq2seqとHRED、HREDとVHREDの違いを簡潔に述べよ。**

**seq2seqは文脈を無視した応答になってしまうが、HREDは過去の会話を元に、文脈を考慮した応答になる。しかし、HREDは同じコンテキストを与えると同じ答えを出してしまい多様性がなくなり、短い答えを選ぶ傾向がある。VHREDはVAEの概念を利用し、多様な出力を行える。**

* **確認テスト**
* **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**

1. **Word2vec（word to vector）**

**Word2vecは、単語の埋め込みを生成するために使用される一連のモデル群である。 これらのモデルは、単語の言語コンテキストを再構築するように訓練された浅い2層ニューラルネットワークであり、大きなコーパスを受け取って一つのベクトル空間を生成する。このベクトル空間は典型的には数百次元からなり、コーパスの個々の単語はベクトル空間内の個々のベクトルに割り当てられる。コーパス内で同じコンテキストを共有する単語ベクトルは、ベクトル空間内の近くに配置される（wikiペディアより）。当時Googleに在籍していた研究者であるトマス・ミコロフ氏らにより提案され、自然言語処理に大きな技術的進展をもたらしたツールである。それまでの自然言語処理手法が抱えていたさまざまな課題を乗り越え、大規模データによる自然言語処理学習を可能にし、Googleのみならずさまざまな企業でも活用が始まっている（Ledgeより）。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（AINOWより）。**

1. **one-hot表現**

**文字や単語を、記号の世界から数値の世界に変換する手法で、ベクトルのすべての要素のうちひとつだけが1であり、残りはすべて0であるベクトルを意味します。One-hotベクトル表現の言語処理への応用では、世の中に存在するすべての単語を、（0,0……,0,1,0,……0,0）のベクトルで表現します。これにより、単語とベクトルが一対一の関係になり、プログラムで処理できるようになります。このように、各概念を一対一対応で表現する手法は「局所表現」と呼ばれる（Ledgeより）**

1. **分散表現**

**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である。**

1. **Attention Mechanism**

**seq2seq の問題は長い文章への対応が難しいことである。「意味のようなもの」を表すのが固定次元のベクトルであることに問題がある。つまり、3単語のとても短い文であっても、50単語あるとても長い文であっても、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。そこで、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ような何らかの仕組みが必要となる。Attention 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」を学習させることで対応する。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あげる。**

* **確認テスト**
* **RNNとword2vec、seq2seqとAttentionの違いを簡潔に述べよ。**

**RNNはシーケンスデータをベクトルに変換するニューラルネットワークでword2vecは単語をベクトルに変換するニューラルネットワーク。seq2seqはEncoderの最後の状態のみをDecoderに渡すが、Attentionは全ての状態をDecoderに渡す。**

1. **VQ-VAE（Vector Quantized Variational Auto Encoder）**

**「自然界の様々な事物は離散的に表現され得る」という発想から, 潜在変数を離散値にしたのがVector Quantised-VAE (VQ-VAE) である。VQ-VAEはVector Quantised(ベクトル量子化)という手法を使ったVAEである。従来のVAEでは潜在変数を正規分布(ガウス分布)のベクトルになるような学習を行うが、VQ-VAEでは潜在変数を離散化した数値になるような学習を行うVAEである。モデルは(Encoder)-(量子化部分)-(Decoder)から成るが、Encoder、Decoderについては畳み込みを行うVAEと大きく変わらない（Qiitaより）。**

1. **[フレームワーク演習]双方向RNN/勾配のクリッピング(Bidirectional RNN)**

**RNN では、ある状態の中間層の出力値を次の状態に順伝播するネットワークである。これに対して、Bidirectional RNN 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。例えば、DNA 塩基を k-mer ごとに区切れば、塩基配列解析に BRNN が使えるようになる。あるいは、1 文全体を入力して、文中にある誤字・脱字の検出などに応用されている。勾配クリッピングは、誤差逆伝播法を実行するときに時折発生する勾配爆発問題に対処するために使用される手法です。勾配の上限値を定義することで、実際にこの現象が抑制される。勾配クリッピングとは、誤差逆伝播法を実行するときに時折発生する勾配爆発問題に対処するために使用される手法である。勾配の上限値を定義することで、実際にこの現象が抑制することができる（シェルビン氏HPより）。**

1. **[フレームワーク演習]Seq2Seq**

**Seq2seqは翻訳技術が大きく向上させた技術であり、1つの入力(文章)を入れて1つの出力を出すモデルである。**

1. **[フレームワーク演習]data-augmentation（データ拡張）**

**学習用の画像データに対して「変換」を施すことでデータを水増しする手法である。反転、輝度変更、回転、平行移動、合成などを行い、全く異なるデータを多数作成することが可能である。しかし、data-augmentationには下記注意点がある。**

* **データセットに合わない変換**

**闇雲にデータ変換を行ってしまうと、常識ではあり得ない画像（例：数字を反対に書くこと、車は事故でもない限り逆さまにはならない）**

* **過学習**

**data-augmentationは過学習を防ぐために有効ではあるが、学習用データに似た特徴を与えてしまう可能性がある**

1. **[フレームワーク演習]activate\_functions**

**活性化関数（Activation function）とは、あるニューロンから次のニューロンへと出力する際に、あらゆる入力値を別の数値に変換して出力する関数である。機械学習における人工ニューラルネットワーク（ANN：Artificial Neural Network）は、人間における生体ニューラルネットワーク（BNN：Biological Neural Network）の基本的な挙動を模倣している。そのBNNでは、生体ニューロンが活性化（activation）することによって、電気信号がそのニューロンから次のニューロンへと伝播していくことになる。ANNで、この「活性化」を表現するのが、活性化関数である。複数のニューロンから、あるニューロンへの入力は、全結合（Fully connected）などの線形変換（Linear transformation、線形写像）処理によって1つの数値にまとめられる。活性化関数は、その数値を次のニューロンに「どのように出力するか」、つまり「どう活性化するか」が定義されたものだ（※よって活性化関数で重要なのは、「数式自体」よりも「どのような形状のグラフになっているか」である。形状については、例えばシグモイド関数のグラフ図を参照）。この活性化の変換は、非線形変換（Non-Linear transformation、非線形写像）である必要がある（※線形変換を何度重ねても線形にしか変化せず、意味がないので）。図1は、「複数のニューロンからの入力」→「線形変換」→「非線形変換」→「次のニューロンへの出力」の流れを示した図である。※ただし、出力層の活性化関数は次のニューロンに伝播するわけではないので、出力層の活性化関数には「恒等関数」と呼ばれる線形変換が用いられることもある。使われる活性化関数は時代とともに変化している。ニューラルネットワークの基礎となっている情報処理モデル「パーセプトロン」（後日解説）では「ステップ関数」という活性化関数が用いられていた。しかし、「バックプロパゲーション」が登場してからは「シグモイド関数」や「tanh関数」が使われるようになった。さらに、最近のディープニューラルネットワークでは「ReLU」がよく使われるようになっている。また、出力層においてよく使われる主な活性化関数としては下記関数がある。（@ITより）**

* **分類問題（二値）の場合は「シグモイド関数」**
* **分類問題（多クラス）の場合は「ソフトマックス（Softmax）関数」**
* **回帰問題の場合は「（活性化関数なし）」もしくは「恒等関数」**