しおん研-ゼロつくゼミ

第６回

今日やること

- 様々な最適化手法
- パラメータの初期値の設定

学習の目的：損失が最小になるようなパラメータを決定する！（最適化）

そのためには、気をつけるべきことが沢山・・・

- モデルの表現力
- 汎化性能
- 局所最適解の問題
- 勾配消失
- 最適なパラメータへ辿り着くまでの速さ　　　etc…

それらを改善するために、様々な最適化手法が考えられている。

色々な最適化手法

- SGD（確率的勾配降下法）

- Momentum
- AdaGrad
- RMSprop
- Adam

今回は、これらを順に実装していく。

最適化手法をクラスで取り出してみる

Momentum

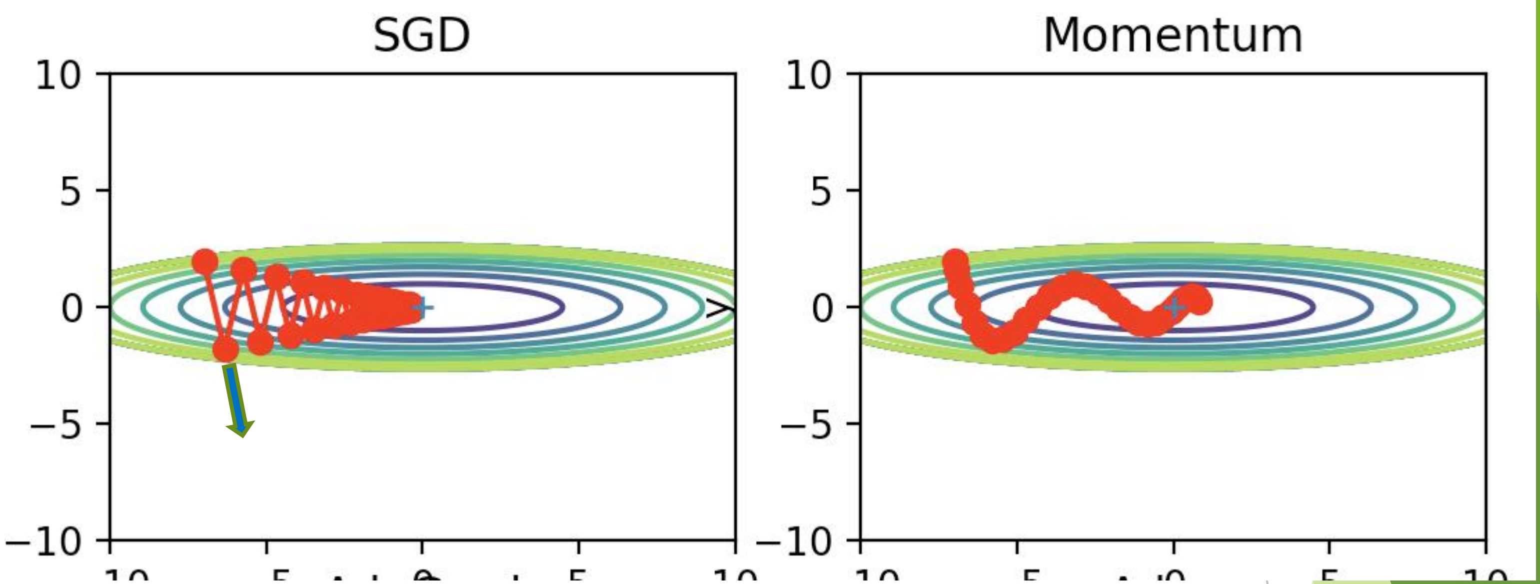
$$w_{t+1} = w_t + v_t$$

$$v_t = \mu v_{t-1} - \eta \frac{\partial C(w)}{\partial w}$$

目的：パラメータを調整する勾配を「勾配の移動平均」とすることで、最適解への到達を速くする

（出典：https://qiita.com/supersaiakujin/items/d9795c34cecd1438b711）

Momentum



式を書き換えると・・・

$$\nu_t = \beta \nu_{t-1} + (1-\beta)G$$
$$w_t = w_{t-1} - \alpha \nu_t$$

勾配の移動平均

：一定の力を受けた物体の速度

出典：
https://qiita.com/omiita/items/1735c1d048fe5f611f80

AdaGrad

$$G_t = G_{t-1} + dw^2$$

$$w_{new} = w_{old} - \frac{\alpha}{\sqrt{G_t + \epsilon}} \odot dw$$

目的：勾配の各成分が大きいとき、それに応じて各成分の学習率の値を小さくする
→「行きすぎること」を防いで、最適解までの到達を速くする！

問題点：学習が進むと学習率が0に近づいてしまう→勾配消失の可能性

RMSprop

$$\mathbf{h}_{i+1} = \rho\mathbf{h}_i + (1-\rho)(\nabla f)^2 \quad (1)$$

$$\mathbf{x}_{i+1} = \mathbf{x}_i - \eta\frac{1}{\sqrt{\mathbf{h}_{i+1}}}(\nabla f) \quad (2)$$

目的：ρ (0 <ρ < 1) をとることで、 AdaGradで発生していた勾配消失の可能性を排除

（出典；https://watlab-blog.com/2020/03/09/rmsprop/）

Adam

$$v_t = \beta_1 v_{t-1} + (1 - \beta_1)G$$

$$s_t = \beta_2 s_{t-1} + (1 - \beta_2)G^2$$

$$w_t = w_{t-1} - \alpha \frac{v_t}{\sqrt{s_t + \epsilon}}$$

Momentum と RMSprop の合体版

・これまでの勾配の影響を考慮
・勾配の大きさによって学習幅を調整

最適化手法の実装

初期値はどうやって決める？

- 初期値によって、学習の結果が大きく変化する！

- （例えば・・・）
  パラメータ行列の各要素が、ある一定の値の周辺に偏っている：表現力の低下
  重みが大きい：過学習

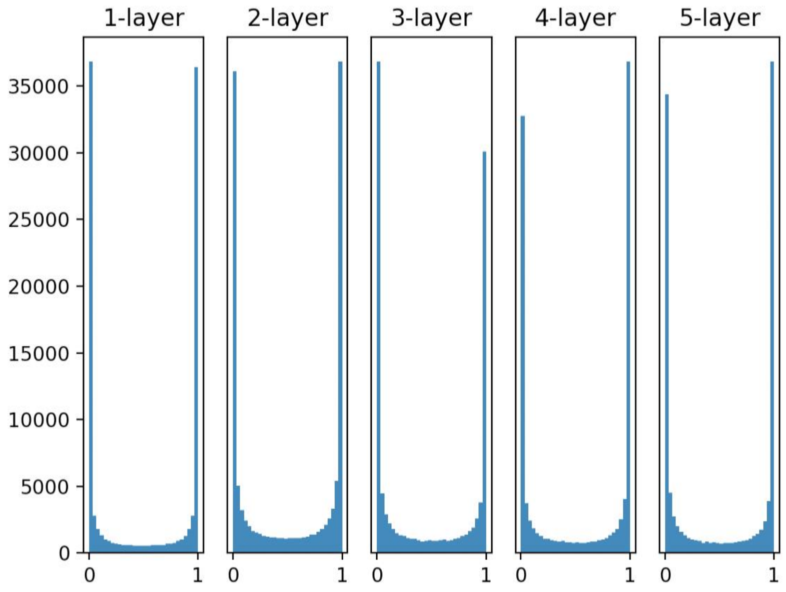
（ちなみに）「重みの初期値を全部0にすると、パラメータが重複した値を持つようになる」は、割と嘘です。ただし、学習がとても遅くなることがわかっています。

とにかく、適切な初期値を決めることがとても大切。

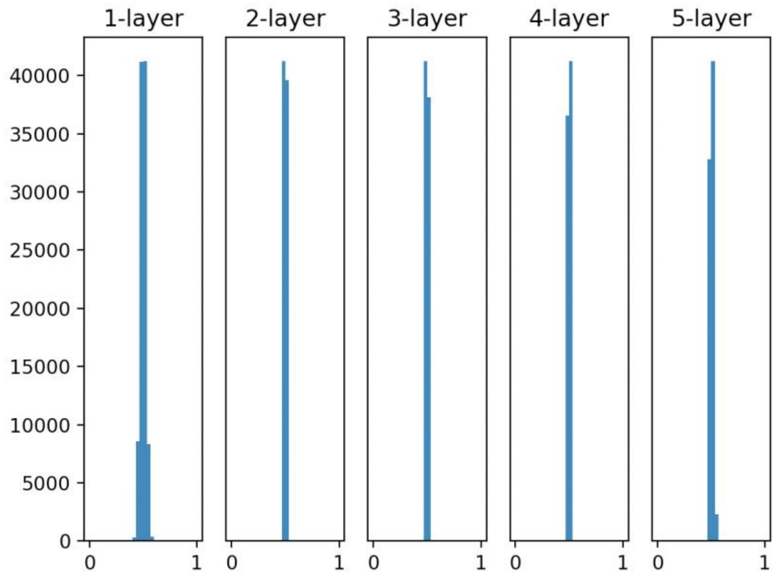
（出典：https://n3104.hatenablog.com/entry/2017/07/16/154042）

各層のアクティベーション分布

- 学習がうまく機能しているかどうかを調べるのに、各層のアクティベーション分布を調べることが多い。
- アクティベーション：活性化関数の後の出力データ
- 下は5層のニューラルネットワークに、100個のパラメータを持つデータを1000個突っ込んだ時のアクティベーション分布



重みの初期値：平均0,標準偏差1の正規分布



初期値：標準偏差0.01の正規分布

アクティベーション分布を実装で確認する回

各層のアクティベーション分布

- 標準偏差1→アクティベーション分布が0と1に偏る

  →（例えばシグモイド関数なら）勾配が0になってしまう・・・（勾配消失）

- 標準偏差0.01 → アクティベーション分布の偏りが大きすぎる

  →パラメータを増やす意味がなくなってしまう・・・（表現力の低下）

（復習）シグモイド関数の逆伝播

じゃあ、最適な初期値って何？

sigmoid関数：

$$sigmoid(x) = \frac{1}{1+e^{-x}}$$

sigmoid関数の微分：

$$sigmoid'(x) = \frac{1}{1+e^{-x}} * \left(1 - \frac{1}{1+e^{-x}}\right)$$

（出典：https://qiita.com/jun40vn/items/4469b947d4157cc98c39）

Sigmoid関数の初期値：Xavierの初期値

- 各層における入力時のノード数をnとすると、

- 重みを、

  平均0、標準偏差 1/√n の正規分布　　で初期化する！

・前層のパラメータの数が多いほど、対象ノードの初期値として設定する重みのスケールは小さくなる。

（イメージ：特徴量を増やせば増やすほど、その中の一つの特徴量に影響される度合いを少なくする）

初期値の設定とアクティベーション分布の変化の実装

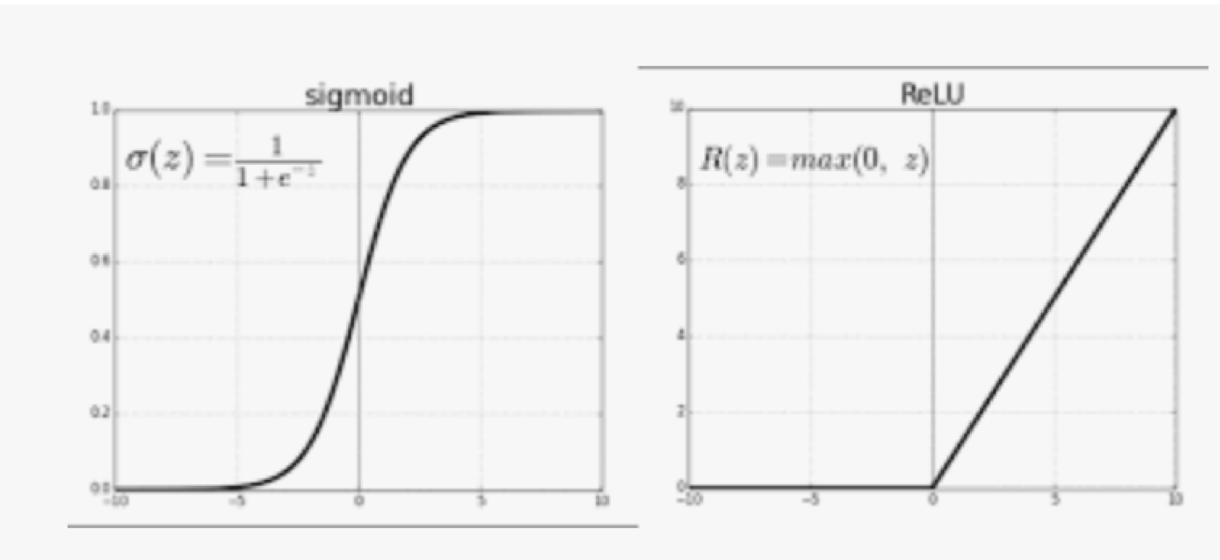
Relu関数の初期値：Heの初期値

- 各層における入力時のノード数をnとすると、

- 重みを、

  平均0、標準偏差 √(2/n) の正規分布　　で初期化する！

- Sigmoidより広がりを持たせた初期値の設定（アクティベーションがより均一になりやすいため）



Mnistでの実装

mirror object to mirror
operation == "MIRROR_X":
mirror_mod.use_x = True
mirror_mod.use_y = False
mirror_mod.use_z = False
operation == "MIRROR_Y":
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
operation == "MIRROR_Z":
mirror_mod.use_x = False
mirror_mod.use_y = False
mirror_mod.use_z = True
selection at the end -add
ob.select= 1
ier_ob.select=1
("Selected" + str(modifier
irror_ob.select = 0
bpy.context.selected_ob
ata.objects[one.name].se
int("please select exactly
OPERATOR CLASSES
types.Operator):
X mirror to the select
ject.mirror_mirror_x"
ror X"
context):
xt.active_object is not