Regularization method in Neural Network

Author : oilneck の人

1 tangent propagation

1.1 Tikhonov Regularization

入力ベクトル $\boldsymbol{x}_n$ の変換 $\boldsymbol{s}(\boldsymbol{x}_n, \xi)$ を考える．ただし，$\boldsymbol{s}(\boldsymbol{x}_n, 0) = \boldsymbol{x}_n$ であり，例えば乱数ノイズを付加する入力の変換に対しては $\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{\xi}) = \boldsymbol{x} + \boldsymbol{\xi}$ と変換される．正則化項係数 Ω を用いて誤差関数が $\tilde{E}(\boldsymbol{w}) = E(\boldsymbol{w}) + \lambda\Omega$(ただし λ は正則化の調節パラメータ) と表されるとき，次の正則化係数を考えたい．

$$\Omega = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}\left(\left.\frac{\partial y_{nk}}{\partial \xi}\right|_{\xi=0}\right)^2 \tag{1.1}$$

$y_{nk} = y_k|_{\boldsymbol{x}_n}$ であり，n 番目の訓練データに対してニューラルネットワークを順伝播した際，k 番目の出力ユニット値を意味する．$\Omega = \sum_n \Omega_n$ と記述するとき

$$\Omega_n = \frac{1}{2}\sum_{k=1}^{K}\left(\sum_{i=1}^{D}\frac{\partial y_{nk}}{\partial x_{ni}}\underbrace{\left.\frac{\partial x_{ni}}{\partial \xi}\right|_{\xi=0}}_{\tau_{ni}}\right)^2 = \frac{1}{2}\sum_{k}\left(\sum_{i} J_{nki}\tau_{ni}\right)^2 \tag{1.2}$$

のように正則化項 (Tikhonov 正則化項) は表現される．τ は入力の接線情報を意味する．また，$\sum_i$ は入力ユニットに関しての和の操作である．式 (1.2) 中に出てくる

$$J_{nki} = \left.\frac{\partial y_k}{\partial x_i}\right|_{\boldsymbol{x}_n} \tag{1.3}$$

はサンプル n の Jacobi 行列の (k, i) 成分を意味する．

1.2 Back propagation

以下に誤差逆伝播法のアルゴリズムをまとめる．$w^{(\ell)}$ は ℓ 番目の層における重み行列の成分を表す．

Error back-propagation

誤差関数が $E(\boldsymbol{w}) = \sum_n E_n(\boldsymbol{w})$ で与えられるとき，入力層 (第 0 層) と L 層ある出力ユニットから成るニューラルネットワークにおける誤差逆伝播法のアルゴリズムは以下で与えられる．

1. 活性化関数を $h(\cdot)$ と表すとき，次式を用いて**順伝播**を行う．

$$a_j^{(\ell)} = \sum_i w_{ji}^{(\ell)} z_i^{(\ell-1)} \qquad z_j^{(\ell)} \equiv h\left(a_j^{(\ell)}\right) = h\left(\sum_i w_{ji}^{(\ell)} z_i^{(\ell-1)}\right) \tag{1.4}$$

2. 出力ユニットの誤差 $\delta^{(L)}$ 及びネットワーク内全ての隠れユニットの誤差 δ_j を計算する．

$$\begin{cases} \delta_j^{(\ell)} = h'\left(a_j^{(\ell)}\right)\sum_k w_{kj}^{(\ell+1)}\delta_k^{(\ell+1)} & [1 \leqq \ell \leqq L-1] \\ \delta_j^{(L)} = y_j - t_j \end{cases} \tag{1.5}$$

3. ユニット誤差を利用して誤差関数の各重みパラメータに関する微分を評価する．

$$\frac{\partial E_n}{\partial w_{ji}^{(\ell)}} = \delta_j^{(\ell)} z_i^{(\ell-1)} \tag{1.6}$$

式 (1.4)〜(1.6) の公式は以下の導出によって得られる．まず，誤差関数が訓練集合の各データに対する誤差の和で表されると仮定する．

$$E(\boldsymbol{w}) = \sum_{n=1}^{N} E_n(\boldsymbol{w}) \tag{1.7}$$

一般のフィードフォワードネットワークでは，それぞれのユニットで

$$a_j = \sum_i w_{ji} z_i, \qquad z_j = h(a_j) \tag{1.8}$$

の形の出力 z_j が出現する．入力 z_i が出力ユニット a_j に影響を与え，その a_j が非線形活性化関数 $h(\cdot)$ を通して次の段の出力 z_j に信号を出す．ある特定のパターン E_n の重み w_{ji} に関する微分を考える．E_n は非線形活性化関数 $h(\cdot)$ の変数 a_j を介して w_{ji} に依存しているので

$$\frac{\partial E_n}{\partial w_{ji}} = \underbrace{\frac{\partial E_n}{\partial a_j}}_{\delta_j} \underbrace{\frac{\partial a_j}{\partial w_{ji}}}_{z_i} = \delta_j z_i \tag{1.9}$$

となる．ここで，式 (1.8) の線形和 a_j の結果より

$$\frac{\partial a_j}{\partial w_{ji}} = z_i$$

が成り立つことを用いた．また，誤差に関する便利な記法

$$\delta_j \equiv \frac{\partial E_n}{\partial a_j} \tag{1.10}$$

を導入した．次に式 (1.10) の誤差 δ_j に関して議論する．ロジスティックシグモイド活性化関数と交差エントロピー誤差関数の組み合わせや，多クラス交差エントロピー誤差関数とソフトマックス活性化関数の組み合わせのような，$h(\cdot)$ に正準連結関数を用いたスキームでは

$$\delta_j = \frac{\partial E_n}{\partial a_j} = y_j - t_j \tag{1.11}$$

の性質が成り立つ．この性質は正準連結関数を $h(\cdot)$ に選んだ場合の一般的な結果である．上式は単純に誤差関数に 2 乗和を選んだ際，出力ユニット値が活性の線形出力に対して $y_j = a_j$ が成り立つことから，

$$\frac{\partial E_n}{\partial a_j} = \frac{\partial}{\partial a_j}\left[\frac{1}{2}\sum_{k=1}^{K}(y_k - t_k)^2\right] = y_j - t_j$$

と導けばよい．次に隠れ層における δ_j の評価について考える．ユニット j につながっているユニット k を通して E_n 中にある a_j への影響があることを考えると，微分の連鎖則より

$$\frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k}\frac{\partial a_k}{\partial a_j} \tag{1.12}$$

と変形できる．ここで，式 (1.8) より a_j と a_k は z を経由して関係し

$$a_k = \sum_i w_{ki} h(a_i) \implies \frac{\partial a_k}{\partial a_j} = w_{kj} h'(a_j)$$

であることと，式 (1.10) の表記と合わせて，式 (1.12) は次の逆伝播の公式に変形される．

$$\delta_j = h'(a_j)\sum_k w_{kj}\delta_k \tag{1.13}$$

これまでに得られた結果である式 (1.4)∼(1.5) はベクトル形式で表すとより簡潔になる．

1. Forward propagation

$$\boldsymbol{a}^{(\ell)} = \hat{\boldsymbol{W}}^{(\ell)}\boldsymbol{z}^{(\ell-1)} \qquad \boldsymbol{z}^{(\ell)} \equiv h\left(\boldsymbol{a}^{(\ell)}\right) = h\left(\hat{\boldsymbol{W}}^{(\ell)}\boldsymbol{z}^{(\ell-1)}\right) \tag{1.14}$$

2. Error Back propagation

$$\begin{cases} \boldsymbol{\delta}^{(\ell)} = h'\left(\boldsymbol{a}^{(\ell)}\right) \circ \left[\hat{\boldsymbol{W}}^{(\ell+1)}\right]^{\mathrm{T}} \boldsymbol{\delta}^{(\ell+1)} & [\,1 \leqq \ell \leqq L-1\,] \\ \boldsymbol{\delta}^{(L)} = \boldsymbol{y} - \boldsymbol{t} \end{cases} \tag{1.15}$$

3. Evaluation of error gradient

$$\frac{\partial E_n}{\partial \hat{\boldsymbol{W}}^{(\ell)}} = \boldsymbol{\delta}^{(\ell)} \otimes \boldsymbol{z}^{(\ell-1)} \tag{1.16}$$

ただし，以上の表記で $\circ$ はベクトルの対応する成分同士の積を意味するアダマール積であるとした．また，$\otimes$ は直積の記号を意味し，$\boldsymbol{a}\otimes\boldsymbol{b}=\boldsymbol{a}\boldsymbol{b}^{\mathrm{T}}$ の略記である．直積の結果は行列となることに注意されたい．これは，物理でいうところの内積 $\langle a|b\rangle$ がスカラー出力を与えるのに対し，$|a\rangle\langle b|$ は演算子として作用するのに似ている．その意味で直積の演算は外積 (内積の対義語) とも呼ばれる．python 言語を用いてプログラミングする際はこの直積を使った表式のほうが実装に便利である．演算は例えば numpy パッケージの outer 関数を用いればよい．式 (1.16) のようにひとたび誤差関数の重み勾配が求まると，後は以下に示す確率勾配降下法によって重みの更新を行い，この操作を適宜繰り返すことになる．

$$\hat{\boldsymbol{W}}^{(\ell)}_{(\mathrm{new})}=\hat{\boldsymbol{W}}^{(\ell)}_{(\mathrm{old})}-\eta\left.\frac{\partial E_n}{\partial\hat{\boldsymbol{W}}^{(\ell)}}\right|_{\hat{\boldsymbol{W}}_{(\mathrm{old})}}$$

2 Regularization of NN

ここでは，逆伝播アルゴリズムと正則化について考える．

$$\hat{\mathcal{G}}\equiv\sum_{m=1}^{D}\tau_{nm}\frac{\partial}{\partial x_{nm}} \tag{2.1}$$

というオペレータを導入すると，正則化項の式 (1.1) は

$$\Omega_n=\frac{1}{2}\sum_{k=1}^{K}\left(\hat{\mathcal{G}}y_{nk}\right)^2 \tag{2.2}$$

と簡潔に表記される．x_{ni} はサンプル n における入力ユニット値 $x_i=z_i^{(0)}$ である．次に順伝播の更新式を導出する．以下のように $\hat{\mathcal{G}}$ オペレータを順伝播公式 (1.4) に作用させる．

$$\alpha_j^{(\ell)}\equiv\hat{\mathcal{G}}z_j^{(\ell)}\qquad\beta_j^{(\ell)}\equiv\hat{\mathcal{G}}a_j^{(\ell)} \tag{2.3}$$

上の定義から，順伝播公式は

$$\alpha_j^{(\ell)}=h'\left(a_j^{(\ell)}\right)\hat{\mathcal{G}}a_j^{(\ell)}=h'\left(a_j^{(\ell)}\right)\beta_j^{(\ell)} \tag{2.4}$$

$$\beta_j^{(\ell)}=\hat{\mathcal{G}}\left(\sum_i w_{ji}^{(\ell)}z_i^{(\ell-1)}\right)=\sum_i w_{ji}^{(\ell)}\underbrace{\hat{\mathcal{G}}z_i^{(\ell-1)}}_{\alpha_i^{(\ell-1)}} \tag{2.5}$$

と変形され，特に入力層で τ の伝播は

$$\alpha_j^{(0)}=\hat{\mathcal{G}}x_{nj}=\sum_m\tau_{nm}\frac{\partial x_{nj}}{\partial x_{nm}}=\tau_{nj}$$

が得られる．以上をまとめると正則化順伝播の公式は次の表式となる．

$$\alpha_j^{(\ell)}=h'\left(a_j^{(\ell)}\right)\sum_i w_{ji}^{(\ell)}\alpha_i^{(\ell-1)}\equiv h'\left(a_j^{(\ell)}\right)\beta_j^{(\ell)}\qquad\left[\alpha_i^{(0)}=\tau_{ni}\right] \tag{2.6}$$

2.1 Error gradient

式 (2.3) を用いると $y_k=z_k^{(L)}$ であることから

$$\Omega_n=\frac{1}{2}\sum_{k=1}^{K}\left[\hat{\mathcal{G}}y_{nk}\right]^2=\frac{1}{2}\sum_{k=1}^{K}\left[\alpha_{nk}^{(L)}\right]^2 \tag{2.7}$$

と正則化項が別表式で表現できる．このとき，正則化項の重み勾配を調べる．

$$\begin{aligned}\frac{\partial \Omega_n}{\partial w_{rs}^{(\ell)}} &= \sum_k \left(\hat{\mathcal{G}} y_{nk}\right) \frac{\partial}{\partial w_{rs}^{(\ell)}} \left(\hat{\mathcal{G}} y_{nk}\right) \\ &= \sum_k \alpha_{nk}^{(L)} \hat{\mathcal{G}} \frac{\partial y_{nk}}{\partial w_{rs}^{(\ell)}} \\ &= \sum_k \alpha_{nk}^{(L)} \hat{\mathcal{G}} \left(\delta_{nkr}^{(\ell)} z_{ns}^{(\ell-1)}\right) \end{aligned} \tag{2.8}$$

上の式の 2 番目の等号では式 (1.9) と同様のロジックで

$$\frac{\partial y_{nk}}{\partial w_{rs}^{(\ell)}} = \frac{\partial y_{nk}}{\partial a_{nr}^{(\ell)}} \frac{\partial a_{nr}^{(\ell)}}{\partial w_{rs}^{(\ell)}} \equiv \delta_{nkr}^{(\ell)} z_{ns}^{(\ell-1)}$$

が成り立つことを用いた．誤差 δ_{nkr} の定義については後に述べる．式 (2.8) で $\hat{\mathcal{G}}$ は積の微分則によって δ と z のそれぞれに作用することと，式 (2.3) 及び，便利な表記

$$\hat{\mathcal{G}} \delta_{nkr}^{(\ell)} = \phi_{nkr}^{(\ell)} \tag{2.9}$$

を用いると，次の正則化項の重み勾配が与えられる．

$$\frac{\partial \Omega_n}{\partial w_{rs}^{(\ell)}} = \sum_k \alpha_{nk}^{(L)} \left\{ \phi_{nkr}^{(\ell)} z_{ns}^{(\ell-1)} + \delta_{nkr}^{(\ell)} \alpha_{ns}^{(\ell-1)} \right\} \tag{2.10}$$

2.2 Back propagation

ここで，誤差 δ の更新式についても考慮する．前述したように δ は出力ユニットの活性微分で定義される．式 (1.12) を真似ると

$$\begin{aligned}\delta_{nkr}^{(\ell)} \equiv \frac{\partial y_{nk}}{\partial a_{nr}^{(\ell)}} &= \sum_u \frac{\partial y_{nk}}{\partial a_{nu}^{(\ell+1)}} \frac{\partial a_{nu}^{(\ell+1)}}{\partial a_{nr}^{(\ell)}} \\ &= \sum_u \delta_{nku}^{(\ell+1)} \frac{\partial}{\partial a_{nr}^{(\ell)}} \left\{ \sum_s w_{us}^{(\ell+1)} h\left(a_{ns}^{(\ell)}\right) \right\}\end{aligned}$$

となるので，誤差 δ の更新式は以下のようになる．

$$\delta_{nkr}^{(\ell)} = h'\left(a_{nr}^{(\ell)}\right) \sum_u w_{ur}^{(\ell+1)} \delta_{nku}^{(\ell+1)} \tag{2.11}$$

また，式 (2.9) から ϕ に関する更新式は

$$\phi_{nkr}^{(\ell)} \equiv \hat{\mathcal{G}} \delta_{nkr}^{(\ell)} = \hat{\mathcal{G}} \left[h'\left(a_{nr}^{(\ell)}\right) \sum_u w_{ur}^{(\ell+1)} \delta_{nku}^{(\ell+1)} \right]$$

となるが，式 (2.3) から $\hat{\mathcal{G}} a_{nr}^{(\ell)} = \beta_{nr}^{(\ell)}$ が成り立つことと，式 (2.9) を用いて

$$\phi_{nkr}^{(\ell)} = h''\left(a_{nr}^{(\ell)}\right) \beta_{nr}^{(\ell)} \sum_u w_{ur}^{(\ell+1)} \delta_{nku}^{(\ell+1)} + h'\left(a_{nr}^{(\ell)}\right) \sum_u w_{ur}^{(\ell+1)} \phi_{nku}^{(\ell+1)} \tag{2.12}$$

が与えられる．実際の正則化項の更新は次のステップを実行すればよい．

1. 式 (2.6) を用いて接線情報 τ の順伝播
2. 式 (2.11),(2.12) により δ, ϕ の更新
3. 式 (2.10) から誤差勾配を計算

References

[1] Pattern Recognition and Machine Learning (C.M.Bishop)