Gabriel Padilha Alves

Questionário - Deep Learning

Trabalho parcial da disciplina de Machine Learning
Professor: Eric Aislan Antonelo

Universidade Federal de Santa Catarina – UFSC

Centro Tecnológico

Programa de Pós-Graduação em Engenharia de Automação e Sistemas

Brasil

2023

Sumário

1 Questão 1

Considere uma rede neural de 1 camada oculta com 2 neurônios e 1 camada de saída com 2 neurônios, e função de ativação $f(.)$ do tipo sigmoide para todo neurônio da rede. A rede recebe como entrada dois valores numéricos (Figura 1). A função de custo E é o erro quadrático para um único exemplo de treinamento. Use obrigatoriamente a notação y_i para a ativação de um neurônio i, e z_i para a soma ponderada correspondente. Responda:

(a) Usando a regra da cadeia, calcule

$$\frac{\partial E}{\partial w_{kl}}$$

(referente à última camada de pesos) para um exemplo de treinamento $< x, t >$, em função das ativações neuronais e saída desejada t.

(b) Dado o exemplo $< x = (0, 1),\ t = (0.8, 1) >$:

(i) calcule a propagação (*forward*) de sinais da rede neural.

(ii) Para esse mesmo exemplo, calcule o valor numérico da derivada parcial $\frac{\partial E}{\partial w_{kl}}$, onde $k = l = 2$ (calcule o valor final com 4 casas decimais após a vírgula).

(iii) Use o método do gradiente para ajustar o peso w_{kl}, $k = l = 2$ com taxa $\alpha = 1$.

(iv) Com o novo peso w_{kl}, calcule mais uma vez a ativação do neurônio $l = 2$ da camada de saída. Explique o novo valor com relação ao valor passado.

(v) Adicione regularização $\lambda = 0.1$ no ajuste de w_{kl}. Explique o valor de w_{kl}.

(a)

Tem-se para um exemplo de treinamento:

$$E = \frac{1}{2}\sum (y^{(l)} - t)^2, \text{ com } y^{(l)} = f(z^{(l)}) \text{ e } z^{(l)} = w_{kl}\, y^{(l-1)}. \tag{1.1}$$

Então

$$\frac{\partial E}{\partial w_{kl}} = \frac{1}{2}\frac{\partial}{\partial w_{kl}}(y^{(3)} - t)^2, \tag{1.2}$$

onde

$$y^{(3)} = f(z^{(3)}), \qquad z^{(3)} = y^{(2)} w_{kl}. \tag{1.3}$$

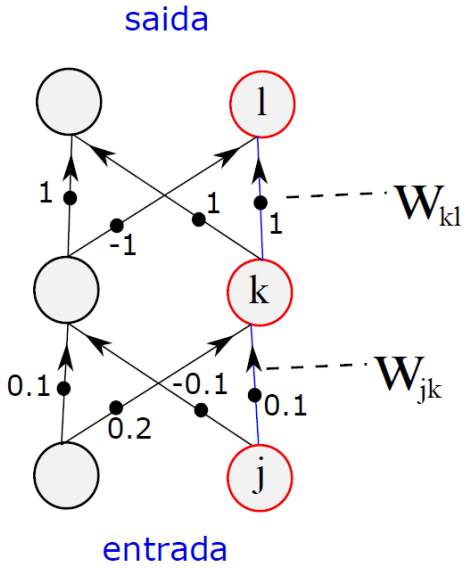

Figura 1 – Rede neural: Neurônios à direita em vermelho correspondem aos índices $j = 2$, $k = 2$, e $l = 2$ em suas respectivas camadas. Os pesos w_{kl} e w_{jk} conectam neurônios entre camadas adjacentes e tem valores inicializados conforme mostra a figura (ex.: $w_{jk} = 0.1$ para $j = 2$ e $k = 2$, aresta em azul).

Com isto,

$$\frac{\partial E}{\partial w_{kl}} = \frac{1}{2}\frac{\partial}{\partial w_{kl}} f(y^{(2)} w_{kl} - t)^2. \tag{1.4}$$

Pela regra da cadeia,

$$\frac{\partial E}{\partial w_{kl}} = \frac{1}{2}\left[\frac{\partial u^2}{du}\frac{du}{dw_{kl}}\right], \tag{1.5}$$

em que $u = f(y^{(2)} w_{kl} - t)$. Deste modo,

$$\frac{\partial E}{\partial w_{kl}} = u\frac{\partial u}{\partial w_{kl}} = \left[f(y^{(2)} w_{kl}) - t\right]\frac{\partial}{\partial w_{kl}}\left(f(y^{(2)} w_{kl}) - t\right). \tag{1.6}$$

O termo t ao final da equação some, pois não depende de w_{kl}. Então, novamente, com a regra da cadeia:

$$\frac{\partial E}{\partial w_{kl}} = \left[f(y^{(2)} w_{kl}) - t\right]\frac{\partial f(v)}{\partial v}\frac{\partial v}{\partial w_{kl}}, \tag{1.7}$$

onde $v = y^{(2)} w_{kl}$. Logo, a derivada de v em relação a w_{kl} equivale a $y^{(2)}$, fazendo com que

$$\frac{\partial E}{\partial w_{kl}} = \left[f(y^{(2)} w_{kl}) - t\right] f'(v)\, y^{(2)}, \tag{1.8}$$

que é o mesmo que

$$\frac{\partial E}{\partial w_{kl}} = \left[f(y^{(2)} w_{kl}) - t\right] f'(y^{(2)} w_{kl})\, y^{(2)}, \tag{1.9}$$

ou

$$\frac{\partial E}{\partial w_{kl}} = (y^{(3)} - t)\ f'(y^{(2)} w_{kl})\ y^{(2)}. \tag{1.10}$$

(b)

A propagação *feedforward* da rede ocorre da seguinte maneira:

$$x = [0 \quad 1] \quad t = [0.8 \quad 1]$$

$$w_{jk} = \begin{bmatrix} 0.1 & 0.2 \\ -0.1 & 0.1 \end{bmatrix}$$

$$w_{kl} = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

$$y^{(2)} = f(x \cdot w_{jk}) = f([-0.1 \quad 0.1]) = [0.475 \quad 0.525]$$

$$y^{(3)} = f(y^{(2)} \cdot w_{kl}) = f([1 \quad 0.05]) = [0.7311 \quad 0.5125]$$

$$E = \frac{1}{2}\sum_{i}^{2}(y^{(3)} - t)^2 + \frac{\lambda}{2}\sum_{p=1}^{n} w_p^2 = 0.1212, \quad \text{para } \lambda = 0$$

$$E = 0.3247, \quad \text{para } \lambda = 0.1$$

onde n é o número de parâmetros da rede. A regularização da função custo é dada pela parcela $\frac{\lambda}{2}\sum_{p=1}^{n} w_p^2$, onde λ é o parâmetro da regularização e $\sum_{p=1}^{n} w_p$ representa a soma de todos os parâmetros da rede.

A derivada parcial $\frac{\partial E}{\partial w_{kl}}$ equivale a:

$$\frac{\partial E}{\partial w_{kl}} = (y^{(3)} - t)\ f'(z^{(3)})\ y^{(2)} + \lambda w_{kl} = \begin{bmatrix} -0.0064 & -0.0579 \\ -0.0071 & -0.0639 \end{bmatrix} \quad \text{para } \lambda = 0 \tag{1.11}$$

$$\frac{\partial E}{\partial w_{kl}} = (y^{(3)} - t)\ f'(z^{(3)})\ y^{(2)} + \lambda w_{kl} = \begin{bmatrix} 0.0936 & -0.1579 \\ 0.0929 & 0.0361 \end{bmatrix} \quad \text{para } \lambda = 0.1 \tag{1.12}$$

Portanto, para $k = l = 2$, o valor sem regularização e com regularização são de, respectivamente, -0.0639 e 0.0361.

Ajustando w_{kl} com $\alpha = 1$ no método do gradiente descendente, não utilizando regularização, w_{kl} com $k = l = 2$ equivale a 1.0639.

Agora recalculando a ativação do neurônio $l = 2$ – com apenas w_{kl} (onde $k = l = 2$) atualizado –, esta passa de 0.5125 para 0.5209. O custo cai de 0.1212 para 0.1172. Alterando w_{kl}

sem regularização reduziu o custo e a ativação do neurônio $l = 2$ sobe, aproximando-se do valor real, o qual equivale a 1.

Com a regularização $\lambda = 0.1$, w_{kl} $(k = l = 2)$ passa de 1 para 0.9639. A ativação do neurônio $l = 2$ desce de 0.5125 para 0.5078. O custo sobe de 0.1212 para 0.1235. Ou seja, a regularização teve um efeito contrário no parâmetro w_{kl}, reduzindo-o, mas aumentando o custo. Percebe-se que a ativação do neurônio $l = 2$ torna-se menor, distanciando-se do valor alvo (i.e., $t = 1$). Isso ocorre porque a regularização tenta reduzir o valor dos parâmetros da rede para evitar o sobreajuste (*overfitting*) do modelo, através da adição dos parâmetros w na função custo. Deste modo, a propagação do erro através do *backpropagation* acaba carregando essa parcela de erro devido aos parâmetros, como se pode observar nas equações desenvolvidas para este problema.

2 Questão 2

Como as redes neurais convolucionais conseguem ser mais eficientes do que redes multicamadas totalmente conectadas (fully connected multilayer networks) para vários problemas, em especial processamento de imagens e visão computacional, mas também outros problemas envolvendo dados em grade (grid), como séries temporais?

A arquitetura das redes neurais convolucionais consiste em camadas convolucionais, camadas de *pooling* (agrupamento) – geralmente são usadas imediatamente após camadas convolucionais – e, finalmente, camadas totalmente conectadas, como uma *softmax* para realizar classificação.

As camadas convolucionais aplicam filtros a pequenas regiões dos dados de entrada, chamadas de campos receptivos locais. Estes campos receptivos locais são conectados às camadas ocultas. Por exemplo, cada conexão de uma região de tamanho 4x4 (16 pixels) conectada a um neurônio de uma camada oculta aprende um peso, como se o neurônio estivesse aprendendo a analisar seu campo receptivo local. Este campo vai deslizando por toda a imagem de entrada, conectando-se sempre a um neurônio diferente na camada oculta. O tamanho do "deslizamento" ou do passo desse campo é chamado de *stride length*. A convolução consiste no campo receptivo percorrer toda a imagem. Os pesos do campo receptivo formam um filtro, o qual acende com mais intensidade neurônios que ressaltam regiões onde existem peculiaridades que se assemelham ao filtro. Estes filtros são ajustados durante o treinamento e esse "mapa" da imagem de entrada para a camada oculta denomina-se *feature map*. Uma camada convolucional é formada por conjunto de distintos *feature maps*.

Já as camadas *pooling* reduzem a dimensionalidade dos dados, diminuindo o número de parâmetros necessários em camadas posteriores, e extraem características relevantes das camadas convolucionais, mesclando características semanticamente semelhantes em uma só. Uma vez encontrada uma característica – como, por exemplo, uma orelha de gato na tentativa de tentar classificar imagens que contêm um gato ou não –, sua localização relativa passa a ser mais importante que a localização exata (o que permite encontrar a orelha de gato em qualquer lugar da imagem, mas sempre perto do nariz).

Por estes motivos, uma rede convolucional consegue extrair características semelhantes nas imagens e é também capaz de utilizar a localização relativa destas características para o aprendizado. Por isto, redes convolucionais conseguem lidar melhor com dados em grade do que redes com multicamadas totalmente conectadas.

3 Questão 3

Um programador obteve de um amigo o código do algoritmo de retropropagação de erros para redes neurais de múltiplas camadas. Ao realizar o treinamento de uma rede neural, observou que os pesos da camada $n-1$ são atualizados conjuntamente com os pesos da camada n. E possível que o algoritmo esteja errado,devido a essa observação? Justifique.

No algoritmo de retropropagação (*backpropagation*) é calculado o erro entre a saída e o valor real dos dados de treinamento. Este erro é retropropagado para cada camada. O gradiente do erro em relação aos pesos é calculado usando a regra da cadeia, o qual é utilizado para atualizar os pesos das conexões entre os neurônios, todos de uma vez, através de $\theta := \theta - \alpha \cdot \frac{\partial J}{\partial \theta}$. Portanto, considerando apenas a observação realizada, não se pode constatar que há erro no código.

4 Questão 4

No treinamento de uma hipótese, a saída desejada é 0.7 e a saída da hipótese é 0.6 para um certo exemplo de treinamento. Após executar o código que ajusta os parâmetros da hipótese aplicando o descenso do gradiente para esse exemplo, a saída da hipótese muda para 0.6001 ao realizar uma nova propagação de sinais para o mesmo exemplo. Isso é esperado ou ocorreu um erro de implementação? Explique.

Isso pode ser esperado quando a taxa de aprendizado é muito baixa, tornando a descida do gradiente muito lenta. Deste modo, o processo de aprendizado pode se tornar custoso. No caso, a saída da hipótese está se aproximando da saída desejada, mostrando que a rede está aprendendo e que, baseando-se nas informações dadas, o algoritmo está correto.

5 Questão 5

Um estudante observou que ao retirar um peso de um neurônio, o desempenho da rede neural melhora para novos sinais de dados. Qual é a provável explicação disso? Discorra.

Um sobreajuste (*overfitting*) do modelo pode estar ocorrendo por conta de uma arquitetura de rede muito complexa. O peso de um neurônio que está contribuindo para o sobreajuste pode ajudar a rede a melhorar seu desempenho para novos dados.