Redes Neurais

Carlos Diego Rodrigues

16 de dezembro de 2021

Universidade Federal do Ceará

Redes Neurais

- Motivação: *Deep Learning*.
- Por que usar mais de uma camada?
- Descida de gradiente estocástica.
- Retropropagação do erro.
  - Rede de duas camadas
  - Generalização matricial

Deep Learning

- Redes neurais causaram grande impacto em anos recentes em aplicações como visão computacional e reconhecimento de linguagem natural.
- Um dos motivos para isto é a capacidade dos dispositivos lidarem com grande quantidade de dados.
- Estudos mostram que uma estrutura expandida, com diversas camadas de computação, também traz benefícios significativos na qualidade do algoritmo (em comparação com a aplicação de uma função diretamente: regressão).
- Os modelos *deep learning* são flexíveis e conseguem explorar mais combinações de informação enterradas nos conjuntos de instâncias com maior eficiência.
- Surgimento de software e hardware próprios para redes neurais: *tensors* e *GPUs*.

Deep Learning

1. Aprendizado de máquina clássico: fazer predições diretamente de um conjunto de atributos pré-especificados pelo usuário.
2. Aprendizado com técnicas de representação: transformar os atributos em uma representação antes de fazer uma predição.
3. Aprendizado profundo (*deep learning*): uma técnica de aprendizado com representação que pode utilizar múltiplos passos (camadas) de representação para criar características complexas.

MNIST - Reconhecimento de números escritos

Classifier	Test Error Rate (%)	References
Linear classifier (1-layer neural net)	12.0	LeCun et al. (1998)
K-nearest-neighbors, Euclidean (L2)	5.0	LeCun et al. (1998)
2-Layer neural net, 300 hidden units, mean square error	4.7	LeCun et al. (1998)
Support vector machine, Gaussian kernel	1.4	MNIST Website
Convolutional net, LeNet-5 (no distortions)	**0.95**	LeCun et al. (1998)
Methods using distortions		
Virtual support vector machine, deg-9 polynomial, (2-pixel jittered and deskewing)	0.56	DeCoste and Scholkopf (2002)
Convolutional neural net (elastic distortions)	0.4	Simard, Steinkraus, and Platt (2003)
6-Layer feedforward neural net (on GPU) (elastic distortions)	**0.35**	Ciresan, Meier, Gambardella, and Schmidhuber (2010)
Large/deep convolutional neural net (elastic distortions)	**0.35**	Ciresan, Meier, Masci, Maria Gambardella, and Schmidhuber (2011)
Committee of 35 convolutional networks (elastic distortions)	0.23	Ciresan, Meier, and Schmidhuber (2012)

Uma primeira rede neural: regressão

- Uma função de regressão (ou regressão logística) pode ser vista como uma primeira rede neural simples.
- Nesta rede temos um conjunto de entradas igual ao número de atributos da instância original e uma única saída que é calculada diretamente a partir das entradas:

$$h_w(x, b) = \sigma(w^T x + b)$$

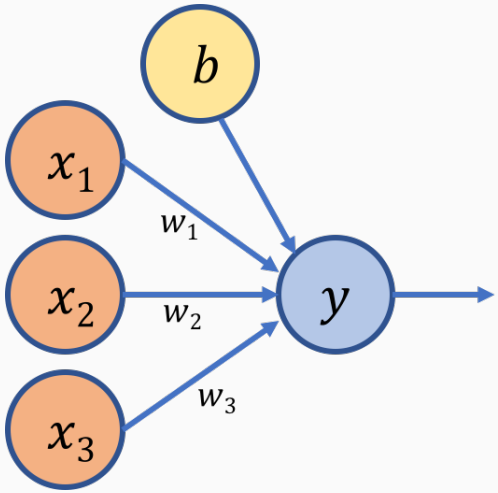
- Este modelo pode ser avaliado para cada exemplo, pelo erro quadrático proporcionado:

$$J^i(w) = \frac{1}{2}(h_w(x^i, b^i) - y^i)^2 \forall\ i = 1, \cdots, m$$

- Podemos adotar como critério para um modelo ideal a minimização do erro médio sobre todas as instâncias:

$$w^* = \arg\min_w \left\{ \frac{1}{m} \sum_{i=1}^{m} J^i(w) \right\}$$

Rede neural de regressão

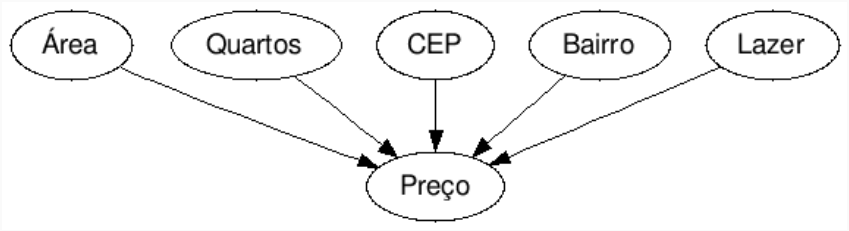


Mais camadas

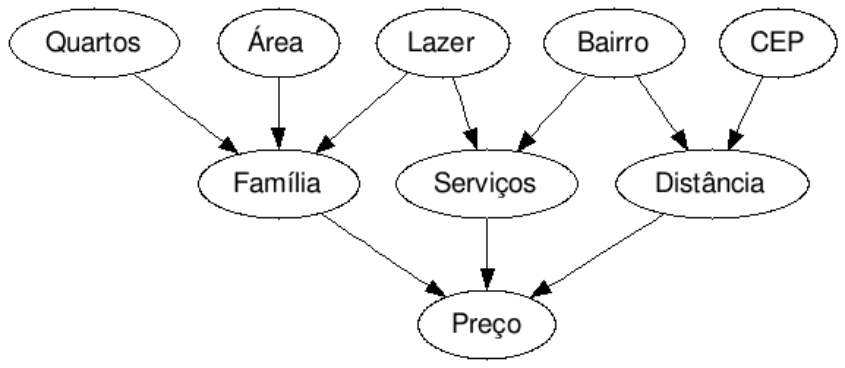
- Digamos que temos um problema de prever o preço de um imóvel.
- Alguns atributos interessantes podem ser:
  - Área do imóvel
  - Número de quartos
  - CEP
  - Bairro
  - Área de lazer
- Estas características estão relacionadas às características que fazem de fato com que as pessoas paguem mais por um imóvel:
  - Tamanho da família
  - Acesso a bons serviços
  - Proximidade com o centro da cidade
- Poderíamos então associar os atributos a essa camada escondida.

Rede neural com duas camadas

1 camada



2 camadas



Relações entre camadas

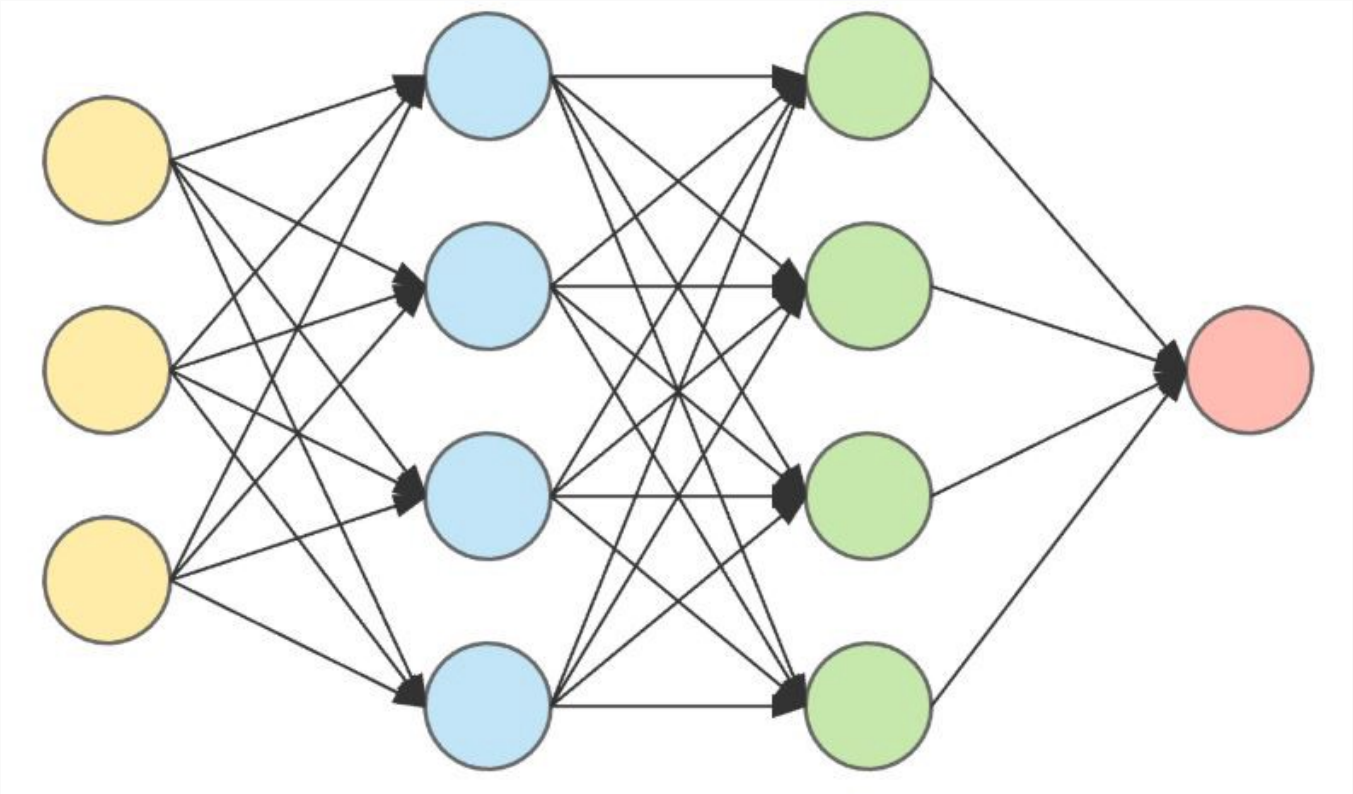
- Podemos observar que a saída agora nao depende mais diretamente da entrada através da relação: $h_w(x, b) = \sigma(w^T x + b)$.
- De maneira geral poderíamos dizer que cada nó além da camada de entrada pode ser escrito através de alguma relação dessa forma.
- O mais comum é considerarmos que todos os nós de uma camada estão conectados a todos os nós da camada seguinte e deixar que os coeficientes emerjam no processo de resolução.
- Portanto em cada camada k podemos considerar uma matriz $W^{[k]}$, onde cada coluna $W_j^{[k]}$ corresponde ao j-ésimo nó daquela camada e o valor $W_{ij}^{[k]}$ é o peso da sua ligação com o i-ésimo elemento da camada seguinte.

Relações entre camadas

- Dessa forma, para o caso de duas camadas, é possível obter um vetor intermediário $a = \sigma(W^{[1]}x + b^{[1]})$ e então calcular $h_W(x, b) = \sigma(W^{[2]}a + b^{[2]})$.
- Generalizando para r camadas temos:

$$
\begin{aligned}
a^{[1]} &= \sigma^{[1]}(W^{[1]}x + b^{[1]}) \\
a^{[2]} &= \sigma^{[2]}(W^{[2]}a^{[1]} + b^{[2]}) \\
&\dots \\
a^{[r-1]} &= \sigma^{[r-1]}(W^{[r-1]}a^{[r-2]} + b^{[r-1]}) \\
h_W(x) &= \sigma^{[r]}(W^{[r]}a^{[r-1]} + b^{[r]})
\end{aligned}
$$

Esquema de uma rede neural com 2 camadas internas



Funções de ativação

- A função de ativação pode mudar, a depender do interesse do desenvolvedor da rede.
- Observe que se a função de ativação for uma identidade, é possível reduzir aquela camada. Suponha que temos duas camadas, onde $a^{[1]} = W^{[1]}x + b^{[1]}$.

$$
\begin{aligned}
h_W(x) &= \sigma(W^{[2]}a^{[1]} + b^{[2]}) \\
h_W(x) &= \sigma(W^{[2]}(W^{[1]}x + b^{[1]}) + b^{[2]}) \\
h_W(x) &= \sigma(W^{[2]}W^{[1]}x + (W^{[2]}b^{[1]} + b^{[2]})) \\
h_W(x) &= \sigma(\tilde{W}x + \tilde{b})
\end{aligned}
$$

- Funções de ativação definem portanto o objetivo e necessidade da camada.

Ativações típicas

- Função logística: $f(x) = \frac{1}{1+e^{-x}}$
- Tangente hiperbólica: $f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$
- *Softmax*: $f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$
- *Softsign*: $f(x) = \frac{x}{|x|+1}$
- *Rectified Linear Unit*: $f(x) = max\{0, x\} = [max\{0, x_i\}]$
- *Softplus*: $f(x) = log(1 + e^{x})$
- Outras ...

Como resolver com mais camadas

- O processo de regressão nos elucida um caminho de resolução para uma rede neural onde entradas e saídas estão diretamente conectadas.
- A descida de gradiente consiste na atualização iterativa dos valores (w, b) de forma que:

$$w_j \leftarrow w_j + \alpha(y^i - h_w(x^i))x_j$$

- De forma mais geral poderíamos pensar no seguinte esquema:

$$w_j \leftarrow w_j + \alpha \frac{d}{dw_j} h_w(x)$$

- Dois desafios surgem no caso das redes neurais:
  1. Uma grande quantidade de entradas, neurons, exemplos ...
  2. Como atualizar os valores para mais de uma camada.

Descida de gradiente estocástica

- Consiste em aplicar exatamente a mesma metodologia do método de descida de gradiente.
- Contudo nem todos os exemplos são considerados para fazer uma atualização dos valores.
- Divide-se o conjunto de instâncias em amostras chamadas "*mini-batches*".
- Esta amostra aleatória não conduz ao valor ideal de descida de gradiente, mas permite uma atualização no sentido da otimização. Após uma nova avaliação, um outro "*mini-batch*" é usado e o método segue.

Algoritmo Descida de gradiente estocástica

Algorithm 1 Descida do gradiente estocástica

Entrada: X, Y: conjunto de atributos e classes

Saída: (w, b): coeficientes aproximados

1: Encontre um valor inicial para (w, b)
2: **for** $k \leftarrow 1 \cdots N_{iter}$ **do**
3: Escolha uma amostra (X', Y') de (X, Y)
4: **for** $(x', y') \in (X', Y')$ **do**
5: **for** $j \leftarrow 1 \cdots n$ **do**
6: $w_j \leftarrow w_j + \alpha \frac{d}{dw_j} h_w(x)$
7: $b \leftarrow b + \alpha \frac{d}{db} h_w(x)$
8: Retorna (w, b)

Retropropagação do erro

- A técnica para lidar com as várias camadas da rede neural é chamada de retropropagação do erro.
- Similarmente à teoria vista para regressão e regressão logística ela consiste em utilizar a margem de erro obtida por determinados candidatos (W, B) da rede para melhorá-los iterativamente com o auxílio da descida de gradiente (estocástica).
- **Teorema :** *Suponha que uma rede de tamnho N calcula uma função real $f : \mathbb{R}^I \to \mathbb{R}$. Então o gradiente ∇f pode ser calculado por um circuito de tamanho $O(N)$ em tempo $O(N)$.*
- Auto-diferenciação.

Retropropagação em uma rede de duas camadas

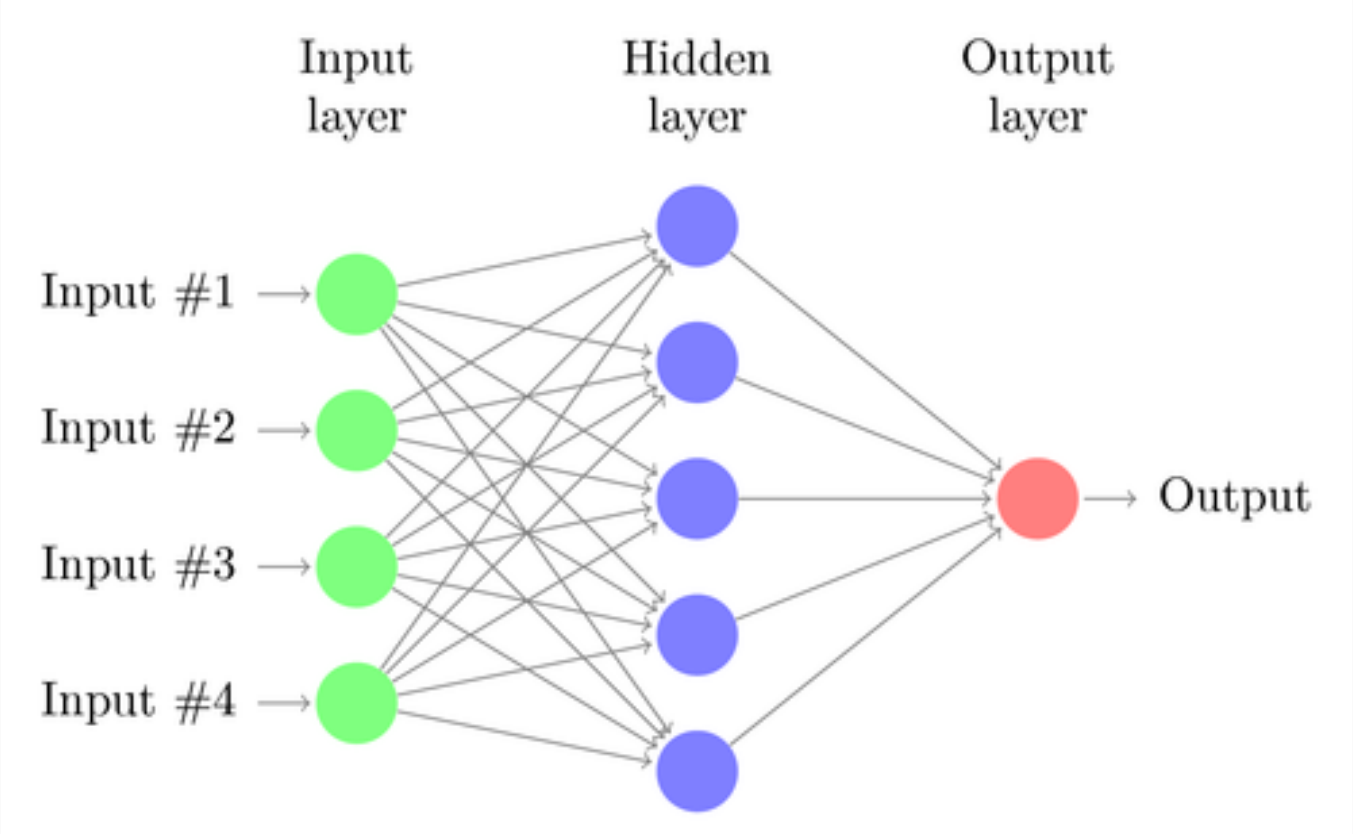
- Vamos voltar ao caso da rede de duas camadas explicitando cada passo do cálculo:

$$z = W^{[1]}x + b^{[1]}$$
$$a = \sigma(z)$$
$$o = W^{[2]}a + b^{[2]}$$
$$J = \frac{1}{2}(y - o)^2$$

- Vamos utilizar ao longo desta seção a regra da cadeia:

$$g_j = g_j(x_1, x_2, \cdots, x_n)$$
$$J = J(g1, g2, \cdots, g_k)$$
$$\frac{dJ}{dx_i} = \sum_{i=1}^{k} \frac{dJ}{dg_j}\frac{dg_j}{dx_i}$$

MNIST - Reconhecimento de números escritos



Retropropagação em uma rede de duas camadas

- Nosso objetivo é calcular as derivadas de cada camada para propor atualizações para $W^{[1]}, W^{[2]}, b^{[1]}$ e $b^{[2]}$.
- Calculando $\frac{dJ}{dW^{[2]}}$.

$$\begin{aligned}\frac{dJ}{dW_i^{[2]}} &= \frac{dJ}{do} \cdot \frac{do}{dW_i^{[2]}} \\ &= (o - y) \cdot \frac{do}{dW_i^{[2]}} \\ &= (o - y) \cdot a_i\end{aligned}$$

- Calculando $\frac{dJ}{db^{[2]}}$

$$\begin{aligned}\frac{dJ}{db^{[2]}} &= \frac{dJ}{do} \cdot \frac{do}{db^{[2]}} \\ &= (o - y) \cdot \frac{do}{db} \\ &= (o - y)\end{aligned}$$

Retropropagação em uma rede de duas camadas

- Calculando $\frac{dJ}{dW^{[1]}}$.

$$\frac{dJ}{dW_{ij}^{[1]}} = \frac{dJ}{dz_i} \cdot \frac{dz_i}{dW_{ij}^{[1]}}$$
$$= \frac{dJ}{dz_i} \cdot x_j$$

- Calculando $\frac{dJ}{dz}$

$$\frac{dJ}{dz_i} = \frac{dJ}{da_i} \cdot \frac{da_i}{dz_i}$$
$$= \frac{dJ}{da_i} \cdot \sigma'(z_i)$$

- Calculando $\frac{dJ}{da}$

$$\frac{dJ}{da_i} = \frac{dJ}{do} \cdot \frac{do}{da_i}$$
$$= (o - y) \cdot W_i^{[2]}$$

Retropropagação em uma rede de duas camadas

- Portanto, podemos concluir o processo para a primeira camada:

$$\frac{dJ}{dW_{ij}^{[1]}} = \frac{dJ}{do} \cdot \frac{do}{da_i} \cdot \frac{da_i}{dz_i} \cdot \frac{dz_i}{dW_{ij}^{[1]}}$$
$$= (o - y) \cdot W_i^{[2]} \cdot \sigma'(z_i) \cdot x_j$$

- Em argumento similar podemo concluir que:

$$\frac{dJ}{db_i^{[1]}} = \frac{dJ}{do} \cdot \frac{do}{da_i} \cdot \frac{da_i}{dz_i} \cdot \frac{dz_i}{db_i^{[1]}}$$
$$= (o - y) \cdot W_i^{[2]} \cdot \sigma'(z_i)$$

Retropropagação em uma rede de várias camadas

- Adotando a sistemática (pensando $x = a^{[0]}$):

$$
\begin{aligned}
a^{[1]} &= \sigma^{[1]}(W^{[1]}a^{[0]} + b^{[1]}) \\
a^{[2]} &= \sigma^{[2]}(W^{[2]}a^{[1]} + b^{[2]}) \\
&\dots \\
a^{[r-1]} &= \sigma^{[r-1]}(W^{[r-1]}a^{[r-2]} + b^{[r-1]}) \\
a^{[r]} &= \sigma^{[r]}(W^{[r]}a^{[r-1]} + b^{[r]}) \\
J &= \frac{1}{2}(a^{[r]} - y)^2
\end{aligned}
$$

- Podemos adotar desenvolvimento similar àquele apresentado para a primeira camada da rede de duas camadas.

Algoritmo de Retropropagação em uma rede neural

Algorithm 2 Retropropagação

Entrada: X, Y: conjunto de atributos e classes

Saída: (W, b): coeficientes da rede

1: **repeat**
2: Escolher uma amostra (X', Y')
3: $a^{[0]} \leftarrow X'$
4: **for** $k = 1 \cdots r$ **do**
5: Calcular $a^{[k]} = \sigma^{[k]}(W^{[k]} a^{[k-1]} + b^{[k]})$
6: **for** $k \leftarrow r \cdots 1$ **do**
7: **if** $k = r$ **then**
8: Calcular $\delta^{[r]} = \frac{dJ}{dz^{[r]}} = \left(W^{[r]} \cdot (a^{[r]} - Y')\right) \cdot \sigma'^{[r]}(z^{[r]})$
9: **else**
10: Calcular $\delta^{[k]} = \frac{dJ}{dz^{[k]}} = \left(W^{[k+1]} \cdot \delta^{[k+1]}\right) \cdot \sigma'^{[k]}(z^{[k]})$
11: Calcular $\frac{dJ}{dW^{[k]}} = \delta^{[k]} {a^{[k]}}^T$
12: Calcular $\frac{dJ}{db^{[k]}} = \delta^{[k]}$
13: Atualizar (W, b) de acordo com a descida de gradiente.
14: **until** convergência
15: Retorna (W, b)