

# Dicas VIP: Aprendizado profundo

Afshine AMIDI e Shervine AMIDI

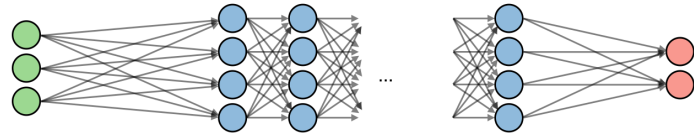
13 de Outubro de 2018

Traduzido por Gabriel Fonseca. Revisado por Leticia Portella.

## Redes neurais

Redes neurais são uma classe de modelos que são construídos com camadas. Os tipos de redes neurais comumente utilizadas incluem redes neurais convolucionais e recorrentes.

□ **Arquitetura** – O vocabulário em torno das arquiteturas de redes neurais é descrito na figura abaixo:



Camada de entrada    Camada escondida 1    ...    Camada escondida  $k$     Camada de saída

Dado que  $i$  é a  $i$ -ésima camada da rede e  $j$  a  $j$ -ésima unidade escondida da camada, nós temos:

$$z_j^{[i]} = w_j^{[i]T} x + b_j^{[i]}$$

onde é definido que  $w$ ,  $b$ ,  $z$ , o peso, o viés e a saída respectivamente.

□ **Função de ativação** – Funções de ativação são usadas no fim de uma unidade escondida para introduzir complexidades não lineares ao modelo. Aqui estão as mais comuns:

Sigmoide	Tanh	ReLU	Leaky ReLU
$g(z) = \frac{1}{1 + e^{-z}}$	$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	$g(z) = \max(0, z)$	$g(z) = \max(\epsilon z, z)$ with $\epsilon \ll 1$

□ **Perda de entropia cruzada** – No contexto de redes neurais, a perda de entropia cruzada  $L(z, y)$  é comumente utilizada e é definida como se segue:

$$L(z, y) = - \left[ y \log(z) + (1 - y) \log(1 - z) \right]$$

□ **Taxa de aprendizado** – A taxa de aprendizado, frequentemente indicada por  $\alpha$  ou às vezes  $\eta$ , indica a que ritmo os pesos são atualizados. Isso pode ser fixado ou alterado de modo adaptativo. O método atual mais popular é chamado Adam, o qual é um método que adapta a taxa de aprendizado.

□ **Retropropagação** – Retropropagação é um método para atualizar os pesos em uma rede neural levando em conta a saída atual e a saída desejada. A derivada relativa ao peso  $w$  é computada utilizando a regra da cadeia e é da seguinte forma:

$$\frac{\partial L(z, y)}{\partial w} = \frac{\partial L(z, y)}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial w}$$

Como resultado, o peso é atualizado como se segue:

$$w \leftarrow w - \eta \frac{\partial L(z, y)}{\partial w}$$

□ **Atualizando os pesos** – Em uma rede neural, os pesos são atualizados como a seguir:

- Passo 1 : Pegue um conjunto de dados de treinamento.
- Passo 2 : Realize propagação para frente a fim de obter a perda correspondente.
- Passo 3 : Retropropague a perda para obter os gradientes.
- Passo 4 : Use os gradientes para atualizar os pesos da rede.

□ **Abandono** – Abandono (*dropout*) é uma técnica que pretende prevenir o sobreajuste dos dados de treinamento abandonando unidades na rede neural. Na prática, neurônios são ou abandonados com a probabilidade  $p$  ou mantidos com a probabilidade  $1 - p$ .

## Redes neurais convolucionais

□ **Requisito de camada convolucional** – Dado que  $W$  é o tamanho do volume de entrada,  $F$  o tamanho dos neurônios da camada convolucional,  $P$  a quantidade de preenchimento de zeros, então o número de neurônios  $N$  que cabem em um dado volume é tal que:

$$N = \frac{W - F + 2P}{S} + 1$$

□ **Normalização em lote** – É uma etapa de hiperparâmetro  $\gamma, \beta$  que normaliza o lote  $\{x_i\}$ . Dado que  $\mu_B, \sigma_B^2$  são a média e a variância daquilo que queremos conectar ao lote, isso é feito como se segue:

$$x_i \leftarrow \gamma \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta$$

Isso é usualmente feito após de uma totalmente conectada/camada convolucional e antes de uma camada não linear e objetiva permitir maiores taxas de aprendizado e reduzir a forte dependência na inicialização.

## Redes neurais recorrentes

□ **Tipos de portas** – Aqui estão os diferentes tipos de portas (*gates*) que encontramos em uma rede neural recorrente típica:

Porta de entrada	Porta esquecida	Porta	Porta de saída
Escrever?	Apagar?	Quanto escrever?	Quanto revelar?

□ **LSTM** – Uma rede de memória de longo prazo (LSTM) é um tipo de modelo de rede neural recorrente (RNN) que evita o problema do desaparecimento da gradiente adicionando portas de 'esquecimento'.

## Aprendizado e controle reforçado

O objetivo do aprendizado reforçado é fazer um agente aprender como evoluir em um ambiente.

□ **Processos de decisão de Markov** – Um processo de decisão de Markov (MDP) é uma tupla de 5 elementos  $(S, \mathcal{A}, \{P_{sa}\}, \gamma, R)$  onde:

- $S$  é o conjunto de estados
- $\mathcal{A}$  é conjunto de ações
- $\{P_{sa}\}$  são as probabilidades de transição de estado para  $s \in S$  e  $a \in \mathcal{A}$
- $\gamma \in [0, 1[$  é o fator de desconto
- $R : S \times \mathcal{A} \rightarrow \mathbb{R}$  ou  $R : S \rightarrow \mathbb{R}$  é a função de recompensa que o algoritmo quer maximizar

□ **Diretriz** – Uma diretriz  $\pi$  é a função  $\pi : S \rightarrow \mathcal{A}$  que mapeia os estados a ações.

*Observação: dizemos que executamos uma dada diretriz  $\pi$  se dado um estado  $s$  nós tomamos a ação  $a = \pi(s)$ .*

□ **Função de valor** – Para uma dada diretriz  $\pi$  e um dado estado  $s$ , nós definimos a função de valor  $V^\pi$  como a seguir:

$$V^\pi(s) = E \left[ R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \dots | s_0 = s, \pi \right]$$

□ **Equação de Bellman** – As equações de Bellman ótimas caracterizam a função de valor  $V^{\pi^*}$  para a ótima diretriz  $\pi^*$ :

$$V^{\pi^*}(s) = R(s) + \max_{a \in \mathcal{A}} \gamma \sum_{s' \in S} P_{sa}(s') V^{\pi^*}(s')$$

*Observação: definimos que a ótima diretriz  $\pi^*$  para um dado estado  $s$  é tal que:*

$$\pi^*(s) = \operatorname{argmax}_{a \in \mathcal{A}} \sum_{s' \in S} P_{sa}(s') V^*(s')$$

□ **Algoritmo de iteração de valor** – O algoritmo de iteração de valor é realizado em duas etapas:

- Inicializamos o valor:

$$V_0(s) = 0$$

- Iteramos o valor baseado nos valores anteriores:

$$V_{i+1}(s) = R(s) + \max_{a \in \mathcal{A}} \left[ \sum_{s' \in S} \gamma P_{sa}(s') V_i(s') \right]$$

□ **Máxima probabilidade estimada** – A máxima probabilidade estimada para o estado de transição de probabilidades como se segue:

$$P_{sa}(s') = \frac{\text{\#vezes que a ação } a \text{ entrou no estado } s \text{ e obteve } s'}{\text{\#vezes que a ação } a \text{ entrou no estado } s}$$

□ **Aprendizado Q** – Aprendizado Q é um modelo livre de estimativa de Q, o qual é feito como se segue:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[ R(s, a, s') + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$$