Dicas VIP: Aprendizado profundo

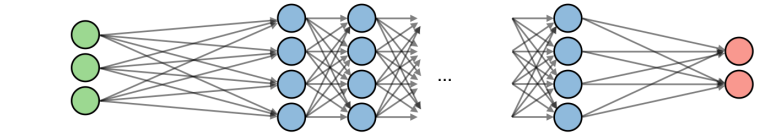
Afshine Amidi e Shervine Amidi

13 de Outubro de 2018

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:

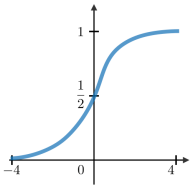
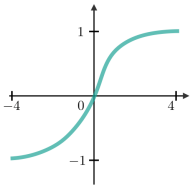
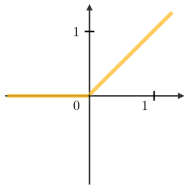
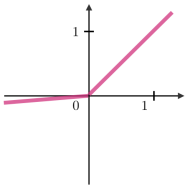


Dado que i é a i-ésima camada da rede e j a j-ésima unidade escondida da camada, nós temos:

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

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
$g(z) = \dfrac{1}{1+e^{-z}}$	$g(z) = \dfrac{e^z - e^{-z}}{e^z + e^{-z}}$	$g(z) = \max(0,z)$	$g(z) = \max(\epsilon z, z)$ 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:

$$\boxed{L(z,y) = -\Big[y\log(z) + (1-y)\log(1-z)\Big]}$$

❒ **Taxa de apredizado** – A taxa de aprendizado, frequentemente indicada por α ou às vezes η, 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:

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

Como resultado, o peso é atualizado como se segue:

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

❒ **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 treinamente abandonando unidades na rede neural. Na prática, neurônios são ou abandonados com a propabilidade p ou mantidos com a propabilidade $1-p$.

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:

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

❒ **Normalização em lote** – É uma etapa de hiperparâmetro γ, β que normaliza o lote $\{x_i\}$. Dado que μ_B, σ_B^2 são a média e a variância daquilo que queremos conectar ao lote, isso é feito como se segue:

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

Isso é usualmente feito após de uma totalmente conectada/camada concolucional 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 decição de Markov (MDP) é uma tupla de 5 elementos $(S,A,\{P_{sa}\},\gamma,R)$ onde:

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

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

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

❒ **Função de valor** – Para uma dada diretriz π e um dado estado s, nós definimos a função de valor V^{π} como a seguir:

$$\boxed{V^{\pi}(s) = E\Big[R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + ...|s_0 = s,\pi\Big]}$$

❒ **Equação de Bellman** – As equações de Bellman ótimas caracterizam a função de valor V^{π^*} para a ótima diretriz π^*:

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

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

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

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

- Inicializamos o valor:

$$\boxed{V_0(s) = 0}$$

- Iteramos o valor baseado nos valores anteriores:

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

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

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

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

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