Aprendizado Profundo

Prof. Lucas S. Kupssinskü

Formulário, Unidades 1 a 3

MLPs, CNNs e Redes Recorrentes

Este formulário reúne as principais expressões das unidades 1 a 3 e resultados matemáticos de apoio. Notação: $\odot$ é o produto elemento a elemento; $\sigma(\cdot)$ é a função sigmoide; $\varphi(\cdot)$ é uma função de ativação genérica; η é a taxa de aprendizado; $\nabla_{\boldsymbol{\theta}}\mathcal{L}$ é o gradiente da função de custo em relação aos parâmetros.

Matemática Básica

Logaritmos e exponenciais:

$$\ln(ab) = \ln a + \ln b, \qquad \ln\frac{a}{b} = \ln a - \ln b, \qquad \ln a^b = b\ln a, \qquad \log_2 x = \frac{\ln x}{\ln 2}, \qquad e^{a+b} = e^a e^b.$$

Funções de Ativação e Derivadas

$$\sigma(a) = \frac{1}{1+e^{-a}}, \qquad \sigma'(a) = \sigma(a)\big(1-\sigma(a)\big), \qquad \sigma'(0) = \tfrac{1}{4}.$$

$$\tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}, \qquad \tanh'(a) = 1 - \tanh^2(a), \qquad \tanh'(0) = 1.$$

$$\mathrm{ReLU}(a) = \max(0, a), \qquad \mathrm{ReLU}'(a) = \begin{cases} 1, & a > 0, \\ 0, & a < 0. \end{cases}$$

Softmax (converte *logits* $\mathbf{z} \in \mathbb{R}^K$ em probabilidades; com temperatura τ):

$$\mathrm{softmax}(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \qquad \mathrm{softmax}(\mathbf{z}/\tau)_i = \frac{e^{z_i/\tau}}{\sum_{j=1}^{K} e^{z_j/\tau}}.$$

Perceptron e MLP

Perceptron de Rosenblatt (saída binária em $\{-1, +1\}$; primeira componente de X igual a 1, truque do bias):

$$\hat{y} = \varphi\big(X\,\boldsymbol{\theta}^\top\big), \qquad \varphi(z) = \mathrm{sinal}(z) = \begin{cases} +1, & z \geq 0, \\ -1, & z < 0. \end{cases}$$

Algoritmo de aprendizado do Perceptron (atualização para uma instância mal classificada $X^{(i)}$):

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + y^{(i)}\,X^{(i)}.$$

Camada densa (pré-ativação e ativação, forma vetorizada; $A^{(0)} = X$):

$$Z^{(l)} = A^{(l-1)}\,{\theta^{(l)}}^\top + b^{(l)}, \qquad A^{(l)} = \varphi\big(Z^{(l)}\big).$$

Funções de Custo

Erro quadrático (L_2 *loss*, MSE quando dividida por N; por instância, $J = \frac{1}{2}(y - \hat{y})^2$):

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \big(y^{(i)} - \hat{y}^{(i)}\big)^2.$$

Entropia cruzada binária (classificação binária):

$$\mathcal{L}_{\text{BCE}}(\boldsymbol{\theta}) = -\sum_{i=1}^{N}\Big[\, y^{(i)} \log \hat{y}^{(i)} + \big(1 - y^{(i)}\big) \log\big(1 - \hat{y}^{(i)}\big)\Big].$$

Entropia cruzada categórica (K classes, saída *softmax*; $y^{(i)}$ é o índice da classe correta):

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{i=1}^{N} \log\, \text{softmax}\big(f(\mathbf{x}^{(i)}; \boldsymbol{\theta})\big)_{y^{(i)}}.$$

Backpropagation

Regra da cadeia por camada (rede de duas camadas com custo J):

$$\frac{\partial J}{\partial \theta^{(2)}} = \frac{\partial J}{\partial a^{(2)}} \frac{\partial a^{(2)}}{\partial z^{(2)}} \frac{\partial z^{(2)}}{\partial \theta^{(2)}}, \qquad \frac{\partial J}{\partial \theta^{(1)}} = \frac{\partial J}{\partial a^{(2)}} \frac{\partial a^{(2)}}{\partial z^{(2)}} \frac{\partial z^{(2)}}{\partial a^{(1)}} \frac{\partial a^{(1)}}{\partial z^{(1)}} \frac{\partial z^{(1)}}{\partial \theta^{(1)}}.$$

Delta da camada (definição, erro na camada de saída L e propagação do erro; $a'^{(l)} = \varphi'(z^{(l)})$):

$$\delta^{(l)} = \frac{\partial J}{\partial z^{(l)}}, \qquad \delta^{(L)} = \frac{\partial J}{\partial a^{(L)}} \odot \frac{\partial a^{(L)}}{\partial z^{(L)}}, \qquad \delta^{(l)} = \big(\delta^{(l+1)}\, \theta^{(l+1)}\big) \odot a'^{(l)}.$$

Gradiente dos parâmetros (delta da camada vezes a ativação da camada anterior; $a^{(0)} = X$):

$$\frac{\partial J}{\partial \theta^{(l)}} = {\delta^{(l)}}^{\top} a^{(l-1)}.$$

Descida de Gradiente e Otimizadores

Descida de gradiente:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\, \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_t).$$

Momentum (média móvel exponencial dos gradientes, $v_0 = 0$):

$$v_{t+1} = \beta\, v_t + (1 - \beta)\, \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_t), \qquad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\, v_{t+1}.$$

Nesterov (gradiente avaliado no ponto deslocado pelo momentum):

$$v_{t+1} = \beta\, v_t + (1 - \beta)\, \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_t - \eta\, \beta\, v_t), \qquad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\, v_{t+1}.$$

AdaGrad (acumulador G cresce monotonicamente; operações elemento a elemento):

$$G_{t+1} = G_t + \big(\nabla_{\boldsymbol{\theta}} \mathcal{L}\big)^2, \qquad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \frac{\eta}{\sqrt{G_{t+1}} + \epsilon}\, \nabla_{\boldsymbol{\theta}} \mathcal{L}.$$

RMSProp (média móvel exponencial no lugar da soma):

$$\mathbb{E}[g^2]_{t+1} = \rho\, \mathbb{E}[g^2]_t + (1 - \rho)\, \big(\nabla_{\boldsymbol{\theta}} \mathcal{L}\big)^2, \qquad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \frac{\eta}{\sqrt{\mathbb{E}[g^2]_{t+1}} + \epsilon}\, \nabla_{\boldsymbol{\theta}} \mathcal{L}.$$

Adam (momentos m e v com correção de viés; $m_0 = v_0 = 0$):

$$m_{t+1} = \beta_1\, m_t + (1 - \beta_1)\, \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_t), \qquad v_{t+1} = \beta_2\, v_t + (1 - \beta_2)\, \big(\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_t)\big)^2,$$

$$\widetilde{m}_{t+1} = \frac{m_{t+1}}{1 - \beta_1^{\,t+1}}, \qquad \widetilde{v}_{t+1} = \frac{v_{t+1}}{1 - \beta_2^{\,t+1}}, \qquad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\, \frac{\widetilde{m}_{t+1}}{\sqrt{\widetilde{v}_{t+1}} + \epsilon}.$$

Regularização, Normalização e Inicialização

Regularização L1 e L2 (penalidade Ω somada à função de custo; L2 corresponde ao *weight decay*):

$$\tilde{\mathcal{L}}(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + \lambda\, \Omega(\boldsymbol{\theta}), \qquad \Omega_{L2} = \|\boldsymbol{\theta}\|_2^2, \qquad \Omega_{L1} = \|\boldsymbol{\theta}\|_1.$$

Batch normalization (estatísticas do *mini-batch* B; γ e β treináveis; em inferência usam-se as médias móveis μ_{mov} e σ_{mov}):

$$\mu_h = \frac{1}{|B|}\sum_{i \in B} h_i, \qquad \sigma_h = \sqrt{\frac{1}{|B|}\sum_{i \in B}(h_i - \mu_h)^2}, \qquad h_i \leftarrow \frac{h_i - \mu_h}{\sigma_h + \epsilon}, \qquad h_i \leftarrow \gamma\, h_i + \beta.$$

Inicialização de pesos (gaussianas de média zero com variâncias distintas; $n^{(l-1)}$ é o *fan-in* da camada; Xavier para tanh e sigmoide, He para ReLU):

$$\text{Xavier:} \quad \theta_{ij}^{(l)} \sim \mathcal{N}\left(0,\ \frac{1}{n^{(l-1)}}\right), \qquad \text{He:} \quad \theta_{ij}^{(l)} \sim \mathcal{N}\left(0,\ \frac{2}{n^{(l-1)}}\right).$$

Variante fan_avg (compromisso entre *forward* e *backward*, com $n^{(l)}$ o *fan-out*):

$$\text{Xavier:} \quad \sigma_\theta^2 = \frac{2}{n^{(l-1)} + n^{(l)}}, \qquad \text{He:} \quad \sigma_\theta^2 = \frac{4}{n^{(l-1)} + n^{(l)}}.$$

Camadas Convolucionais

Dimensão espacial de saída (entrada $W \times W$, *kernel* K, *stride* S, *padding* P; vale também para *pooling*):

$$O = \left\lfloor \frac{W - K + 2P}{S} \right\rfloor + 1.$$

Kernel efetivo com dilatação D (substitui K na fórmula acima):

$$K_{\text{ef}} = D\,(K - 1) + 1.$$

Conexão residual (bloco com atalho de identidade):

$$y = F(x) + x.$$

IoU (*intersection over union* entre duas *bounding boxes* A e B):

$$\text{IoU}(A, B) = \frac{|A \cap B|}{|A \cup B|}.$$

Redes Recorrentes

RNN *vanilla* (um passo de tempo):

$$z_t = x_t\, \theta_{ih}^T + h_{t-1}\, \theta_{hh}^T + b_h, \qquad h_t = \tanh(z_t).$$

Gradiente retropropagado no tempo (fator multiplicado a cada passo; origem do desaparecimento e da explosão do gradiente):

$$\frac{\partial h_t}{\partial h_{t-1}} \propto \theta_{hh}\left(1 - \tanh^2(z_t)\right).$$

Gradient clipping por norma (limita o passo preservando a direção):

$$g \leftarrow c\,\frac{g}{\|g\|} \quad \text{se } \|g\| > c.$$

LSTM (portões f_t, i_t, o_t e candidato $\tilde{c}_t$; $[h_{t-1}, x_t]$ é a concatenação):

$$f_t = \sigma\big(\theta_f\,[h_{t-1}, x_t] + b_f\big), \qquad i_t = \sigma\big(\theta_i\,[h_{t-1}, x_t] + b_i\big), \qquad \tilde{c}_t = \tanh\big(\theta_c\,[h_{t-1}, x_t] + b_c\big),$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \qquad o_t = \sigma\big(\theta_o\,[h_{t-1}, x_t] + b_o\big), \qquad h_t = o_t \odot \tanh(c_t).$$

Caminho da memória (derivada que mitiga a dissipação do gradiente):

$$\frac{\partial c_t}{\partial c_{t-1}} = f_t.$$

GRU (portões de *reset* r_t e de *update* z_t):

$$r_t = \sigma\big(\theta_r\,[h_{t-1}, x_t] + b_r\big), \qquad z_t = \sigma\big(\theta_z\,[h_{t-1}, x_t] + b_z\big),$$

$$\tilde{h}_t = \tanh\big(\theta_h\,[r_t \odot h_{t-1},\, x_t] + b_h\big), \qquad h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t.$$

Modelagem de Linguagem e Atenção

Probabilidade de uma sequência (fatoração autorregressiva de m *tokens*):

$$p(y_1, \ldots, y_m) = \prod_{t=1}^{m} p\big(y_t \mid y_{<t}\big).$$

Perplexidade (a partir da *log-likelihood* em bits, com $\log_2$):

$$L(y_{1:m}) = \sum_{t=1}^{m} \log_2\big(p(y_t \mid y_{<t})\big), \qquad \text{Perplexidade}(y_{1:m}) = 2^{-\frac{1}{m}L(y_{1:m})}, \qquad \text{Perplexidade} \in [1, |V|].$$

Atenção em *seq2seq* (estado do decodificador h_t, estados do codificador $s_1, \ldots, s_m$; *scores* de produto escalar, pesos via *softmax* e vetor de contexto):

$$\text{score}(h_t, s_k) = h_t^{\top} s_k, \qquad a_k^{(t)} = \frac{\exp\big(\text{score}(h_t, s_k)\big)}{\sum_{i=1}^{m} \exp\big(\text{score}(h_t, s_i)\big)}, \qquad c^{(t)} = \sum_{k=1}^{m} a_k^{(t)}\, s_k.$$