Aprendizado Profundo

Prova, Unidades 2 e 3

Prof. Lucas S. Kupssinskü

Data: _____

Nome: _____

Quadro de respostas (questões objetivas):

Questão	a	b	c	d	e	f	g	h
1								
2								
3								
4								
5								

Questões Objetivas (6.0 pts)

Cada questão possui apenas uma alternativa correta. Marque a letra escolhida no quadro de respostas.

1) (1.2) Considere uma CNN com três camadas convolucionais em sequência, todas **sem** *padding*. A camada 1 usa *kernel* 3×3 com *stride* $S_1 = 1$; a camada 2 usa *kernel* 3×3 com *stride* $S_2 = 2$; a camada 3 usa *kernel* 3×3 com *stride* $S_3 = 1$. Qual o campo receptivo (lado, em *pixels*) de uma unidade na saída da camada 3?

   a) 5
   b) 6
   c) 7
   d) 9
   e) 11
   f) 13
   g) 15
   h) 17

2) (1.2) Em redes residuais, a soma $h = x + f(x)$ tende a aumentar a variância das ativações ao longo da profundidade. Qual o principal objetivo de inserir *batch normalization* dentro dos blocos residuais?

   a) Reduzir o número de parâmetros treináveis do bloco, já que os parâmetros γ e β substituem os pesos das convoluções internas do bloco.
   b) Substituir a conexão de atalho (*skip connection*), tornando a soma $h = x + f(x)$ desnecessária e reduzindo o número de caminhos de gradiente da rede.
   c) Normalizar a distribuição das ativações dentro do bloco, controlando o crescimento da variância introduzido pela soma e mantendo o treino estável em grande profundidade.
   d) Atuar como a única não-linearidade do bloco, dispensando funções de ativação como a ReLU entre as convoluções que compõem $f(x)$.
   e) Aumentar o campo receptivo das unidades do bloco sem acrescentar parâmetros, de forma análoga ao efeito de uma convolução dilatada.

f) Garantir matematicamente que cada bloco aprenda exatamente o mapeamento identidade $f(x) = 0$, independentemente dos dados de treino.

g) Eliminar a necessidade de inicialização cuidadosa dos pesos, pois fixa a variância dos pesos em $2/n_{\text{in}}$ como na inicialização de He.

h) Servir de regularizador que memoriza as estatísticas do conjunto de teste para acelerar a convergência durante a inferência.

3) (1.2) Você treina uma RNN *vanilla* (ativação tanh nas conexões recorrentes) para classificar sequências longas (cerca de 200 passos). O treino fica estagnado com erro alto; ao inspecionar, os gradientes da função de custo em relação aos pesos associados aos **primeiros** passos da sequência são praticamente nulos, enquanto os dos **últimos** passos têm magnitude normal. Qual a explicação mais consistente **e** a melhor mitigação?

a) O otimizador SGD é o único responsável; trocá-lo por Adam, sem nenhuma outra mudança, é suficiente para que os gradientes dos passos iniciais deixem de ser nulos.

b) O problema é de explosão de gradiente; aplicar *gradient clipping* aos gradientes dos primeiros passos restaura o sinal de aprendizado que chega a esses passos.

c) A taxa de aprendizado está baixa demais para os passos iniciais; usar uma taxa por passo, maior no início da sequência, compensa o desequilíbrio observado entre os passos.

d) A rede está em *overfitting* nos passos finais; reduzir o comprimento das sequências para poucos passos elimina o problema sem necessidade de trocar a arquitetura.

e) A ativação tanh é a causa direta; substituí-la por sigmoide nas conexões recorrentes aumenta as derivadas e resolve o desaparecimento do gradiente nos passos iniciais.

f) Os gradientes desaparecem: a retropropagação no tempo multiplica fatores $\theta_{hh}\,(1 - \tanh^2(z_k))$ ao longo de muitos passos e, como têm módulo ≤ 1, o sinal decai nos passos iniciais; a mitigação é usar arquitetura com portões (LSTM/GRU).

g) A camada de saída está mal escolhida; trocá-la por *softmax* com entropia cruzada propaga gradiente igualmente para todos os passos, eliminando o decaimento.

h) Há *dropout* insuficiente nos primeiros passos; adicioná-lo aumenta a magnitude do gradiente que chega a eles e compensa o decaimento multiplicativo da retropropagação.

4) (1.2) Sobre as asserções abaixo a respeito de células LSTM:

I. Na LSTM, a atualização da memória $c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$ combina a memória anterior (modulada pelo *forget gate*) com o novo candidato (modulado pelo *input gate*); o caminho de c_{t-1} para c_t é predominantemente aditivo, o que ajuda a preservar o gradiente ao longo do tempo.

II. Na LSTM, o *forget gate* f_t e o *input gate* i_t são sempre acoplados pela restrição $i_t = 1 - f_t$, de modo que a fração esquecida da memória anterior é exatamente a fração escrita pelo novo candidato.

III. O estado oculto da LSTM é obtido como $h_t = o_t \odot \tanh(c_t)$; portanto o *output gate* o_t controla qual parte da memória é exposta como saída no passo t.

a) Apenas I está correta.

b) Apenas II está correta.

c) Apenas III está correta.

d) Apenas I e II estão corretas.

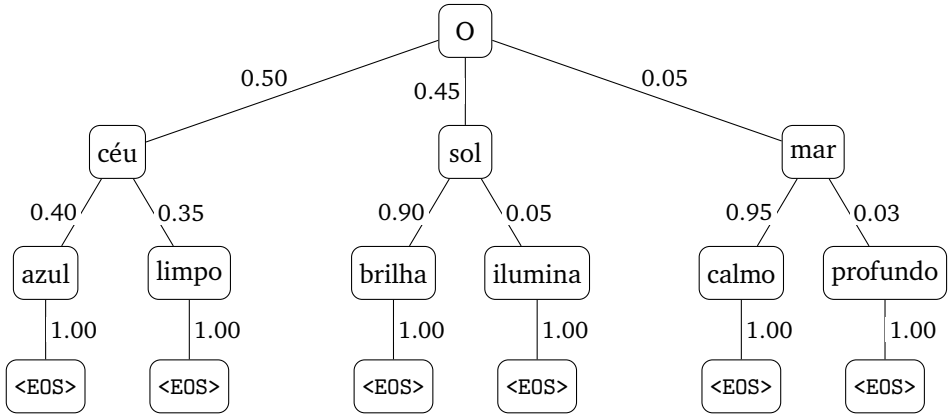
e) Apenas I e III estão corretas.

f) Apenas II e III estão corretas.

g) Todas estão corretas.

h) Nenhuma está correta.

5) (1.2) A árvore abaixo mostra a expansão de hipóteses de um modelo de linguagem ao completar a frase iniciada pelo *token* “O”: cada aresta indica a probabilidade condicional do próximo *token* e os nós do segundo nível representam as duas continuações mais prováveis de cada hipótese. Considere decodificação com *beam search* de largura $b = 2$ em contraste com *greedy search*. Com base na figura, qual a análise correta?



a) O *greedy* escolhe “O sol brilha” (0.405) e o *beam search* escolhe “O céu azul” (0.20); as duas estratégias trocam de papel neste exemplo específico.

b) Tanto o *greedy* quanto o *beam search* selecionam “O céu azul”, pois “céu” tem a maior probabilidade no primeiro passo (0.50) e nenhuma das duas chega a expandir “sol”.

c) “O sol brilha” é selecionada porque “sol” mantém a maior probabilidade em cada passo isolado; nesse exemplo o *greedy* e o *beam search* coincidem na escolha.

d) “O céu azul” vence no *beam search*, pois $0.50 \times 0.40 = 0.20$ é maior que 0.45×0.90; o *greedy* e o *beam* concordam nessa escolha.

e) O *beam search* maximiza a soma das probabilidades ($0.45 + 0.90 = 1.35$) e por isso escolhe “O sol brilha”; o produto das probabilidades da sequência não é levado em conta.

f) “O mar calmo” é selecionada, pois o *beam search* prioriza *tokens* de baixa probabilidade no primeiro passo para aumentar a diversidade da geração.

g) “O sol brilha” é selecionada: sua probabilidade conjunta ($0.45 \times 0.90 = 0.405$) supera as demais; o *greedy* compromete-se com “céu” por ser localmente mais provável e perde a melhor sequência.

h) Nenhuma sequência pode ser escolhida sem antes normalizar pela penalidade de comprimento, pois sem normalização as probabilidades conjuntas das sequências não são comparáveis.

Questões Dissertativas (4.0 pts)

6) (2.0) Marque **Verdadeiro** ou **Falso** nas asserções abaixo. Em caso **Falso**, proponha uma modificação *não-trivial* que torna a asserção verdadeira (a simples negação **não** é uma modificação válida).

a) ( ) V ( ) F Uma camada convolucional com *kernel* 5×5, *stride* 1 e *padding* 1 preserva as dimensões espaciais $H \times W$ da entrada.

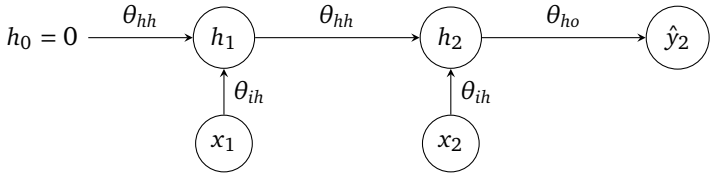
b) ( ) V ( ) F Na derivada da saída de um bloco residual em relação à sua entrada surge um termo identidade I; esse termo cria um caminho direto para o gradiente, o que ajuda a mitigar o desaparecimento do gradiente em redes profundas.

c) ( ) V ( ) F Durante a inferência, a *batch normalization* normaliza cada ativação usando a média e o desvio padrão calculados sobre o próprio *mini-batch* de teste em que a amostra se encontra.

d) ( ) V ( ) F Um modelo de linguagem que atribui probabilidade *uniforme* sobre um vocabulário de tamanho $|V|$ a cada passo tem perplexidade igual a 1.

e) ( ) V ( ) F Na LSTM, quando o *forget gate* $f_t \to 1$ e o *input gate* $i_t \to 0$, a atualização $c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$ reduz-se a c_{t-1}, ou seja, a memória anterior é mantida praticamente inalterada.

7) (2.0) Considere a RNN *vanilla* **escalar** desdobrada abaixo, com ativação tanh no estado oculto e saída linear. Os pesos (todos escalares) são $\theta_{ih} = 0.5$, $\theta_{hh} = 0.2$ e $\theta_{ho} = 1.0$, com vieses $b_h = 0$ e $b_o = 0$. A sequência de entrada é $x_1 = 1$, $x_2 = 1$, o estado inicial é $h_0 = 0$ e a saída esperada (no passo 2) é $y = 1$. A recorrência é $h_t = \tanh(x_t\,\theta_{ih} + h_{t-1}\,\theta_{hh})$, a saída é $\hat{y}_2 = h_2\,\theta_{ho}$ e a função de custo é $J = \frac{1}{2}(y - \hat{y}_2)^2$. Pode arredondar para 4 casas decimais.



a) (0.5) Compute o *forward pass* e indique os valores de h_1, h_2 e $\hat{y}_2$.

b) (0.5) Compute o valor da função de custo J.

c) (0.5) Compute os gradientes $\dfrac{\partial J}{\partial \theta_{ho}}$, $\dfrac{\partial J}{\partial \theta_{ih}}$ e $\dfrac{\partial J}{\partial \theta_{hh}}$.

d) (0.5) Realize uma atualização dos três parâmetros usando descida de gradiente estocástica $\theta_{t+1} = \theta_t - \eta\,\nabla_\theta J$ com taxa de aprendizado $\eta = 0.1$.