Due Date: March 23rd (11pm), 2022

Instructions

- *For all questions, show your work!*
- *Starred questions are **hard** questions, not **bonus** questions.*
- *Use LaTeX and the template we provide when writing your answers. You may reuse most of the notation shorthands, equations and/or tables. See the assignment policy on the course website for more details.*
- *Unless noted that questions are related, assume that notation and defintions for each question are self-contained and independent.*
- *Submit your answers electronically via Gradescope.*
- *TAs for this assignment are **Ankit Vani** and **Sai Aravind Sreeramadas**.*

Question 1 (6-9-6)**.** This question is about activation functions and vanishing/exploding gradients in recurrent neural networks (RNNs). Let $\sigma : \mathbb{R} \to \mathbb{R}$ be an activation function. When the argument is a vector, we apply σ element-wise. Consider the following recurrent unit:

$$\boldsymbol{h}_t = \boldsymbol{W}\sigma(\boldsymbol{h}_{t-1}) + \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{b}$$

1.1 Show that applying the activation function in this way results in an equivalent recurrence as the conventional way of applying the activation function: $\boldsymbol{g}_t = \sigma(\boldsymbol{W}\boldsymbol{g}_{t-1} + \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{b})$ (i.e. express $\boldsymbol{g}_t$ in terms of $\boldsymbol{h}_t$). More formally, you need to prove it using mathematical induction. You only need to prove the induction step in this question, assuming your expression holds for time step $t-1$.

*1.2 Let $||\boldsymbol{A}||$ denote the L_2 operator norm[1] of matrix $\boldsymbol{A}$ ($||\boldsymbol{A}|| := \max_{\boldsymbol{x}:||\boldsymbol{x}||=1} ||\boldsymbol{A}\boldsymbol{x}||$). Assume $\sigma(x)$ has bounded derivative, i.e. $|\sigma'(x)| \leq \gamma$ for some $\gamma > 0$ and for all x. We denote as $\lambda_1(\cdot)$ the largest eigenvalue of a symmetric matrix. Show that if the largest eigenvalue of the weights is bounded by $\frac{\delta^2}{\gamma^2}$ for some $0 \leq \delta < 1$, gradients of the hidden state will vanish over time, i.e.

$$\lambda_1(\boldsymbol{W}^\top \boldsymbol{W}) \leq \frac{\delta^2}{\gamma^2} \quad \implies \quad \left|\left|\frac{\partial \boldsymbol{h}_T}{\partial \boldsymbol{h}_0}\right|\right| \to 0 \text{ as } T \to \infty$$

Use the following properties of the L_2 operator norm

$$||\boldsymbol{A}\boldsymbol{B}|| \leq ||\boldsymbol{A}||\,||\boldsymbol{B}|| \qquad \text{and} \qquad ||\boldsymbol{A}|| = \sqrt{\lambda_1(\boldsymbol{A}^\top \boldsymbol{A})}$$

1.3 What do you think will happen to the gradients of the hidden state if the condition in the previous question is reversed, i.e. if the largest eigenvalue of the weights is larger than $\frac{\delta^2}{\gamma^2}$? Is this condition *necessary* and/or *sufficient* for the gradient to explode ? (Answer in 1-2 sentences).

Question 2 (8-8-8)**.** In this question you will demonstrate that an estimate of the first moment of the gradient using an (exponential) running average is equivalent to using momentum, and is biased by a scaling factor. The goal of this question is for you to consider the relationship between

[1] The L_2 operator norm of a matrix $\boldsymbol{A}$ is is an *induced norm* corresponding to the L_2 norm of vectors. You can try to prove the given properties as an exercise.

different optimization schemes, and to practice noting and quantifying the effect (particularly in terms of bias/variance) of *estimating* a quantity.

Let $\boldsymbol{g}_t$ be an unbiased sample of gradient at time step t and $\Delta\boldsymbol{\theta}_t$ be the update to be made. Initialize $\boldsymbol{v}_0$ to be a vector of zeros.

2.1 For $t \geq 1$, consider the following update rules:

- SGD with momentum:

$$\boldsymbol{v}_t = \alpha \boldsymbol{v}_{t-1} + \epsilon \boldsymbol{g}_t \qquad \Delta\boldsymbol{\theta}_t = -\boldsymbol{v}_t$$

  where $\epsilon > 0$ and $\alpha \in (0,1)$.

- SGD with running average of $\boldsymbol{g}_t$:

$$\boldsymbol{v}_t = \beta \boldsymbol{v}_{t-1} + (1-\beta)\boldsymbol{g}_t \qquad \Delta\boldsymbol{\theta}_t = -\delta \boldsymbol{v}_t$$

  where $\beta \in (0,1)$ and $\delta > 0$.

Express the two update rules recursively ($\Delta\boldsymbol{\theta}_t$ as a function of $\Delta\boldsymbol{\theta}_{t-1}$). Show that these two update rules are equivalent ; i.e. express (α, ϵ) as a function of (β, δ).

2.2 Unroll the running average update rule, i.e. express $\boldsymbol{v}_t$ as a linear combination of $\boldsymbol{g}_i$'s $(1 \leq i \leq t)$.

2.3 Assume $\boldsymbol{g}_t$ has a stationary distribution independent of t. Show that the running average is biased, i.e. $\mathbb{E}[\boldsymbol{v}_t] \neq \mathbb{E}[\boldsymbol{g}_t]$. Propose a way to eliminate such a bias by rescaling $\boldsymbol{v}_t$.

Question 3 (8-8-6-9-3)**.** In this question, you will analyze the performance of dot-product attention and derive an efficient approximation of it. Consider that *multi-head* dot-product attention for a sequence of length n is defined as follows:

$$\begin{aligned}
\text{MultiHead}(\bar{\boldsymbol{Q}}, \bar{\boldsymbol{K}}, \bar{\boldsymbol{V}}) &= \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\boldsymbol{W}^O \\
\text{where} \quad \text{head}_i &= \text{Attention}_{\text{std}}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) \quad (\text{here, } \boldsymbol{Q} := \bar{\boldsymbol{Q}}\boldsymbol{W}_i^Q, \boldsymbol{K} := \bar{\boldsymbol{K}}\boldsymbol{W}_i^K, \boldsymbol{V} := \bar{\boldsymbol{V}}\boldsymbol{W}_i^V) \\
&= \text{softmax}_{\text{row}}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d_k}}\right)\boldsymbol{V}
\end{aligned}$$

where $\bar{\boldsymbol{Q}}, \bar{\boldsymbol{K}}, \bar{\boldsymbol{V}} \in \mathbb{R}^{n \times d_{\text{model}}}$ are the queries, keys, and values, and $\boldsymbol{W}_i^Q, \boldsymbol{W}_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $\boldsymbol{W}_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ $\forall i$, and $\boldsymbol{W}_O \in \mathbb{R}^{hd_v \times d_{model}}$ are the weights. The softmax subscript "row" indicates that the softmax is computed along the rows, and the Attention subscript "std" indicates that this is the standard variant (we will see other variants later in the question). For this question, you can assume that $d_k = d_v = d_{\text{model}}$ and call the value d.

For calculating the time and space complexities, you can also assume that matrix multiplications are performed naively. As an example, for $\boldsymbol{C} = \boldsymbol{A}\boldsymbol{B}$ where $\boldsymbol{A} \in \mathbb{R}^{p \times q}$, $\boldsymbol{B} \in \mathbb{R}^{q \times r}$, and $\boldsymbol{C} \in \mathbb{R}^{p \times r}$, the time complexity is $\Theta(pqr)$ due to the three nested loops, and the space complexity is $\Theta(pq + qr + pr)$ from storing the inputs and the result.

3.1 What is the time and space complexity of the attention operation carried out by a single head in Θ-notation in terms of n and d ? Use your answer to calculate the time and space complexity of multi-head dot-product attention in terms of n, d, and h, assuming that the heads are computed sequentially. For very long sequences, where does the bottleneck lie ?

For the remaining parts, let us focus on the attention operation carried out by a single head. Furthermore, you can omit the scaling factor $\sqrt{d}$ without loss of generality by considering that $\boldsymbol{Q}$ and $\boldsymbol{K}$ can be scaled as desired.

3.2 Let us consider an alternative form of attention, one that performs row-wise softmax on $\boldsymbol{Q}$ and column-wise softmax on $\boldsymbol{K}$ separately as follows:

$$\text{Attention}_{\text{separable}}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}_{\text{row}}(\boldsymbol{Q})\text{softmax}_{\text{col}}(\boldsymbol{K})^\top \boldsymbol{V}.$$

Prove that $\text{softmax}_{\text{row}}(\boldsymbol{Q})\text{softmax}_{\text{col}}(\boldsymbol{K})^\top$ produces valid categorical distributions in every row, like $\text{softmax}_{\text{row}}(\boldsymbol{Q}\boldsymbol{K}^\top)$. If $n \gg d$, show that $\text{Attention}_{\text{separable}}$ can be faster and requires less space than $\text{Attention}_{\text{std}}$. Is $\text{Attention}_{\text{separable}}$ as expressive as $\text{Attention}_{\text{std}}$?

(Hint: For a valid categorical distribution $\boldsymbol{p} \in \mathbb{R}^d$ over d categories, $p_i \geq 0 \, \forall i \in \{1, \dots, d\}$ and $\sum_{i=1}^{d} p_i = 1$.)

3.3 Verify that the standard attention can be written as

$$\text{Attention}_{\text{std}}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{V}$$

where $\boldsymbol{A} = \exp\left(\boldsymbol{Q}\boldsymbol{K}^\top\right)$ and $\boldsymbol{D} = \text{diag}(\boldsymbol{A}\mathbf{1})$, where exp is an element-wise operation, diag creates a diagonal matrix from a vector, and $\mathbf{1}$ is a vector of ones. Note that you can store diagonal matrices in linear space and compute matrix multiplications with them in linear time.

Let us now consider a variant $\text{Attention}_{\text{approx}}$ where the elements a_{ij} of $\boldsymbol{A}$ can be represented as $a_{ij} = f(\boldsymbol{q}_i)^\top f(\boldsymbol{k}_j)$ for some $f : \mathbb{R}^d \to \mathbb{R}_+^m$, where $\boldsymbol{q}_i$ and $\boldsymbol{k}_j$ are the ith row of $\boldsymbol{Q}$ and the jth row of $\boldsymbol{K}$ respectively.

If $n \gg m$ and $n \gg d$, how can you use this formulation to make attention efficient ? What is the time and space complexity of $\text{Attention}_{\text{approx}}$? You can assume that f takes $\Theta(md)$ time and space.

(Hint: Decompose the matrix $\boldsymbol{A}$.)

*3.4 Prove that in $\text{Attention}_{\text{std}}$,

$$a_{ij} = \exp\left(\frac{-\|\boldsymbol{q}_i\|^2}{2}\right) \cdot \mathbb{E}_{\boldsymbol{x} \in \mathcal{N}(\mathbf{0}, \mathbf{I})}\left[\exp(\boldsymbol{x}^\top \boldsymbol{q}_i)\exp(\boldsymbol{x}^\top \boldsymbol{k}_j)\right] \cdot \exp\left(\frac{-\|\boldsymbol{k}_j\|^2}{2}\right).$$

Use this result to devise the function $f : \mathbb{R}^d \to \mathbb{R}_+^m$ introduced in the previous part, such that $\text{Attention}_{\text{approx}}$ approximates the expectation in $\text{Attention}_{\text{std}}$ by sampling.

(Hint 1: If $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{I})$, $p(\boldsymbol{x}) = (2\pi)^{-d/2}\exp\left(-\frac{1}{2}\|\boldsymbol{x} - \boldsymbol{\mu}\|^2\right)$ and $\int_{\boldsymbol{x}} p(\boldsymbol{x})d\boldsymbol{x} = 1$.)

(Hint 2: $\boldsymbol{x}^\top \boldsymbol{y} = -\frac{1}{2}(\boldsymbol{x}^\top \boldsymbol{x} - (\boldsymbol{x} + \boldsymbol{y})^\top(\boldsymbol{x} + \boldsymbol{y}) + \boldsymbol{y}^\top \boldsymbol{y})$. This can be useful when starting the proof in the reverse direction.)

3.5 Discuss the implications of the choice of m for $\text{Attention}_{\text{approx}}$. What are the trade-offs to think about ?

Question 4 (4-5-6-6)**.** In this question, you will reconcile the relationship between L2 regularization and weight decay for the Stochastic Gradient Descent (SGD) and Adam optimizers. Imagine you are training a neural network (with learnable weights θ) with a loss function $L(f(\mathbf{x}^{(i)}, \theta), \mathbf{y}^{(i)})$, under two different schemes. The *weight decay* scheme uses a modified SGD update rule: the weights θ decay exponentially by a factor of λ. That is, the weights at iteration $i + 1$ are computed as

$$\theta_{i+1} = \theta_i - \eta \frac{\partial L(f(\mathbf{x}^{(i)}, \theta_i), \mathbf{y}^{(i)})}{\partial \theta_i} - \lambda\theta_i$$

where η is the learning rate of the SGD optimizer. The *L2 regularization* scheme instead modifies the loss function (while maintaining the typical SGD or Adam update rules). The modified loss function is

$$L_{\text{reg}}(f(\mathbf{x}^{(i)}, \theta), \mathbf{y}^{(i)}) = L(f(\mathbf{x}^{(i)}, \theta), \mathbf{y}^{(i)}) + \gamma \|\theta\|_2^2$$

4.1 Prove that the *weight decay* scheme that employs the modified SGD update is identical to an *L2 regularization* scheme that employs a standard SGD update rule.

4.2 This question refers to the Adam algorithm as described in the lecture slide (also identical to Algorithm 8.7 of the deep learning book). It turns out that a one-line change to this algorithms gives us Adam with an L2 regularization scheme. Identify the line of the algorithm that needs to change, and provide this one-line modification.

4.3 Consider a "decoupled" weight decay scheme for the original Adam algorithm (see lecture slides, or equivalently, Algorithm 8.7 of the deep learning book) with the following two update rules.

- The **Adam-L2-reg** scheme computes the update by employing an L2 regularization scheme (same as the question above).
- The **Adam-weight-decay** scheme computes the update as $\mathbf{\Delta}\theta = -\left(\epsilon \frac{\hat{\mathbf{s}}}{\sqrt{\hat{\mathbf{r}}+\delta}} + \lambda\theta\right)$.

Now, assume that the neural network weights can be partitioned into two disjoint sets based on their gradient magnitude: $\theta = \{\theta_{\text{small}}, \theta_{\text{large}}\}$, where each weight $\theta_s \in \theta_{\text{small}}$ has a much smaller gradient magnitude than each weight $\theta_l \in \theta_{\text{large}}$. Using this information provided, answer the following questions. In each case, provide a brief explanation as to why your answer holds.

(a) Under the **Adam-L2-reg** scheme, which set of weights among θ_{small} and θ_{large} would you expect to be regularized (i.e., driven closer to zero) more strongly than the other ? Why ?

(b) Would your answer change for the **Adam-weight-decay** scheme ? Why/why not ?

(Note: for the two sub-parts above, we are interested in the rate at which the weights are regularized, *relative* to their initial magnitudes.)

4.4 In the context of all of the discussion above, argue that weight decay is a better scheme to employ as opposed to L2 regularization ; particularly in the context of adaptive gradient based optimizers. (Hint: think about how each of these schemes regularize each parameter, and also about what the overarching objective of regularization is).