Due Date: Nov 14th, 2023 at 11:00 pm

Instructions

- *For all questions, show your work!*
- *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.*
- *The use of AI tools like Chat-GPT to find answers or parts of answers for any question in this assignment is not allowed. However, you can use these tools to improve the quality of your writing, like fixing grammar or making it more understandable. If you do use these tools, you must clearly explain how you used them and which questions or parts of questions you applied them to. Failing to do so or using these tools to find answers or parts of answers may result in your work being completely rejected, which means you'll receive a score of 0 for the entire theory or practical section.*
- *Submit your answers electronically via Gradescope.*
- *TAs for this assignment are* ***Thomas Jiralerspong, Saba Ahmadi, and Shuo Zhang.***

Question 1 (3)**. (Normalization)**

This question is about normalization techniques.

Batch normalization, layer normalization and instance normalization all involve calculating the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma^2}$ with respect to different subsets of the tensor dimensions. Given the following 3D tensor, calculate the corresponding mean and variance tensors for each normalization technique: $\boldsymbol{\mu}_{batch}$, $\boldsymbol{\mu}_{layer}$, $\boldsymbol{\mu}_{instance}$, $\boldsymbol{\sigma}^2_{batch}$, $\boldsymbol{\sigma}^2_{layer}$, and $\boldsymbol{\sigma}^2_{instance}$.

$$\left[\begin{bmatrix}2,4,3\\2,1,2\end{bmatrix},\begin{bmatrix}1,2,3\\4,1,1\end{bmatrix},\begin{bmatrix}2,1,3\\3,2,4\end{bmatrix},\begin{bmatrix}1,4,2\\2,4,3\end{bmatrix}\right]$$

The size of this tensor is 4 x 2 x 3 which corresponds to the batch size, number of channels, and number of features respectively.

Answer 1. Batch normalization takes the mean over examples and features (one value per channel):

$$\boldsymbol{\mu}_{batch} = \begin{bmatrix}2.33\\2.42\end{bmatrix} \boldsymbol{\sigma}^2_{batch} = \begin{bmatrix}1.06\\1.24\end{bmatrix}$$

Layer normalization takes the mean and variance over channels and features (one value per example):

$$\boldsymbol{\mu}_{layer} = \left[[2.33],[2],[2.5],[2.67]\right] \boldsymbol{\sigma}^2_{layer} = \left[[0.89],[1.33],[0.92],[1.22]\right]$$

Instance normalization takes the mean and variance over features (one value for each channel-example pair):

$$\boldsymbol{\mu}_{instance} = \left[\begin{bmatrix}3\\1.67\end{bmatrix},\begin{bmatrix}2\\2\end{bmatrix},\begin{bmatrix}2\\3\end{bmatrix},\begin{bmatrix}2.33\\3\end{bmatrix}\right]$$

$$\boldsymbol{\sigma}^2_{instance} = \left[\begin{bmatrix}0.67\\0.22\end{bmatrix}, \begin{bmatrix}0.67\\2\end{bmatrix}, \begin{bmatrix}0.67\\0.67\end{bmatrix}, \begin{bmatrix}1.56\\0.67\end{bmatrix}\right]$$

Question 2 (4-6-6)**.** (**Regularization**)

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

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

2.1 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.

2.2 Consider a "decoupled" weight decay scheme for the Adam algorithm (see lecture slides) 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 $\boldsymbol{\Delta}\theta = -\left(\epsilon \frac{\hat{\mathbf{s}}}{\sqrt{\hat{\mathbf{r}}}+\delta} + \lambda\theta\right)$.

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.)

2.3 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).

Answer 2.

2.1 For the *L2 regularization* scheme, the update rule for standard SGD is

$$\theta_{i+1} = \theta_i - \eta \frac{\partial L_{\text{reg}}(f(\mathbf{x}^{(i)}, \theta), \mathbf{y}^{(i)})}{\partial \theta_i} = \theta_i - \eta \frac{\partial L(f(\mathbf{x}^{(i)}, \theta), \mathbf{y}^{(i)})}{\partial \theta_i} - 2\eta\gamma\theta_i$$

This is identical to the update rule under the weight decay scheme when $\lambda = 2\eta\gamma$.

2.2 (a) The larger weights θ_{large} are regularized less strongly than the smaller weights θ_{small}. This is because Adam-L2-reg adapts the gradients of the L2 regularization loss term (which includes the gradients of the usual loss term plus the gradients of the L2 penalty over the weights). This results in the weights with larger gradient magnitudes being adapted less than the weights with smaller gradient magnitudes.

(b) In this case, each weight is adapted by the same amount λ.

2.3 The objective of a regularization term (or of weight decay) is to prevent weights from deviating strongly from their current values. In the context of Adam, L2 regularization results in each weight having it's own update rate based on it's gradient magnitude, which can be undesirable (because weights with low gradient magnitudes are not as strongly regularized, and can deviate from their previous values). However, weight decay ensures that each parameter is regularized uniformly, and achieves better regularization.

Question 3 (1-1-4-6-2)**.** (**Decoding**)

Suppose that we have a vocabulary containing N possible words, including a special token `<BOS>` to indicate the beginning of a sentence. Recall that in general, a language model with a full context can be written as

$$p(w_1, w_2, \ldots, w_T \mid w_0) = \prod_{t=1}^{T} p(w_t \mid w_0, \ldots, w_{t-1}).$$

We will use the notation $\boldsymbol{w}_{0:t-1}$ to denote the (partial) sequence $(w_0, \ldots, w_{t-1})$. Once we have a fully trained language model, we would like to generate realistic sequences of words from our language model, starting with our special token `<BOS>`. In particular, we might be interested in generating the most likely sequence $\boldsymbol{w}^{\star}_{1:T}$ under this model, defined as

$$\boldsymbol{w}^{\star}_{1:T} = \arg\max_{\boldsymbol{w}_{1:T}} p(\boldsymbol{w}_{1:T} \mid w_0 = \texttt{<BOS>}). \tag{1}$$

For clarity we will drop the explicit conditioning on w_0, assuming from now on that the sequences always start with the `<BOS>` token.

3.1 How many possible sequences of length $T + 1$ starting with the token `<BOS>` can be generated in total ? Give an exact expression, without the O notation. Note that the length $T + 1$ here includes the `<BOS>` token.

3.2 To determine the most likely sequence $\boldsymbol{w}^{\star}_{1:T}$, one could perform exhaustive enumeration of all possible sequences and select the one with the highest joint probability (as defined in equation 1). Comment on the feasibility of this approach. Is it scalable with vocabulary size ?

3.3 In light of the difficulties associated with exhaustive enumeration, it becomes essential to employ practical search strategies to obtain a reasonable approximation of the most likely sequence.

In order to generate B sequences having high likelihood, one can use a heuristic algorithm called *Beam search decoding*, whose pseudo-code is given in Algorithm 1 below

Algorithm 1: Beam search decoding

Input: A language model $p(\boldsymbol{w}_{1:T} \mid w_0)$, the beam width B

Output: B sequences $\boldsymbol{w}_{1:T}^{(b)}$ for $b \in \{1, \dots, B\}$

Initialization: $w_0^{(b)} \leftarrow$ `<BOS>` for all $b \in \{1, \dots, B\}$

Initial log-likelihoods: $l_0^{(b)} \leftarrow 0$ for all $b \in \{1, \dots, B\}$

for $t = 1$ **to** T **do**

  for $b = 1$ **to** B **do**

    for $j = 1$ **to** N **do**

      $s_b(j) \leftarrow l_{t-1}^{(b)} + \log p(w_t = j \mid \boldsymbol{w}_{0:t-1}^{(b)})$

  for $b = 1$ **to** B **do**

    Find (b', j) such that $s_{b'}(j)$ is the b-th largest score

    Save the partial sequence b': $\widetilde{\boldsymbol{w}}_{0:t-1}^{(b)} \leftarrow \boldsymbol{w}_{0:t-1}^{(b')}$

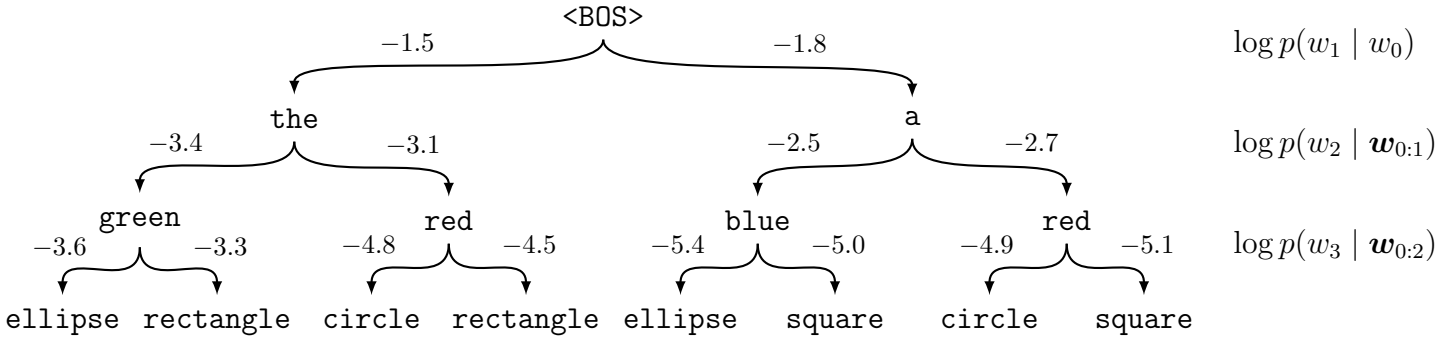
    Add the word j to the sequence b: $w_t^{(b)} \leftarrow j$

    Update the log-likelihood: $l_t^{(b)} \leftarrow s_{b'}(j)$

  Assign the partial sequences: $\boldsymbol{w}_{0:t-1}^{(b)} \leftarrow \widetilde{\boldsymbol{w}}_{0:t-1}^{(b)}$ for all $b \in \{1, \dots, B\}$

What is the time complexity of Algorithm 1 ? Its space complexity ? Write the algorithmic complexities using the O notation, as a function of T, B, and N. Is this a practical decoding algorithm when the size of the vocabulary is large ?

3.4 The different sequences that can be generated with a language model can be represented as a tree, where the nodes correspond to words and edges are labeled with the log-probability $\log p(w_t \mid \boldsymbol{w}_{0:t-1})$, depending on the path $\boldsymbol{w}_{0:t-1}$. In this question, consider the following language model (where the low probability paths have been removed for clarity)



3.4.a *Greedy decoding* is a simple algorithm where the next word $\overline{w}_t$ is selected by maximizing the conditional probability $p(w_t \mid \overline{\boldsymbol{w}}_{0:t-1})$ (with $\overline{w}_0 =$ `<BOS>`)

$$\overline{w}_t = \arg\max_{w_t} \log p(w_t \mid \overline{\boldsymbol{w}}_{0:t-1}).$$

Find $\overline{\boldsymbol{w}}_{1:3}$ using greedy decoding on this language model, and its log-likelihood $\log p(\overline{\boldsymbol{w}}_{1:3})$.

3.4.b Apply beam search decoding with a beam width $B = 2$ to this language model, and find $\boldsymbol{w}_{1:3}^{(1)}$ and $\boldsymbol{w}_{1:3}^{(2)}$, together with their respective log-likelihoods.

3.5 Please highlight the primary limitation that stands out to you for each of the discussed methods (greedy decoding and beam search).

Answer 3.

3.1 The number of possible sequences is N^T (we also accepted answer $(N-1)^T$ for students who applied the constraint that <BOS> token only appears at the beginning of the sentence).

3.2 The number of possible sequences scales exponentially to the length of the sequence. Therefore, it is not scalable, given the length of sentences and vocabulary size in practice.

3.3 The time complexity of the first inner-most loop of Beam search decoding is $O(BN)$ (iterations over B and N), and the time complexity of the second loop is $O(B^2N)$ (naively searching for the B highest scores among BN candidates) ; overall, the time complexity of Beam search decoding is therefore $O(B^2NT)$. The space complexity of this algorithm is $O(BN + BT)$ (the B partial sequences $\boldsymbol{w}_{0:t}^{(b)}$ need to be stored, as well as the BN next scores $s_b(j)$ and a vector of size B for the log-likelihoods $l_t^{(b)}$). This decoding algorithm is practical, even when the size of the vocabulary N is large, since it has a running time which is only linear in N.

3.4

3.4.a Applying greedy decoding on this language, we get

$$\overline{\boldsymbol{w}}_{1:3} = [\texttt{the}, \texttt{red}, \texttt{rectangle}] \qquad \text{and} \qquad \log p(\overline{\boldsymbol{w}}_{1:3}) = -9.1.$$

3.4.b We give the step-by-step details of beam-search decoding on this language model, when the beam width $B = 2$

	$\boldsymbol{w}_{1:t}^{(1)}$	$\boldsymbol{w}_{1:t}^{(2)}$	$l_t^{(1)}$	$l_t^{(2)}$	remaining scores	
$t=1$	[the]	[a]	-1.5	-1.8	–	–
$t=2$	[a, blue]	[a, red]	-4.3	-4.5	-4.6	-4.9
$t=3$	[a, blue, square]	[a, red, circle]	-9.3	-9.4	-9.6	-9.7

Therefore, we have

$$\boldsymbol{w}_{1:3}^{(1)} = [\texttt{a}, \texttt{blue}, \texttt{square}] \qquad \text{and} \qquad \log p(\boldsymbol{w}_{1:3}^{(1)}) = -9.3$$
$$\boldsymbol{w}_{1:3}^{(2)} = [\texttt{a}, \texttt{red}, \texttt{circle}] \qquad \text{and} \qquad \log p(\boldsymbol{w}_{1:3}^{(2)}) = -9.4.$$

3.5 Beam search, while demanding increased space and exhibiting a longer execution time compared to greedy decoding, tends to yield a more accurate answer by maintaining track of the B most probable sequences. It is crucial to emphasize, however, that neither approach guarantees the discovery of the most probable sequence.

Question 4 (3-4-2-2)**.** (**RNN**)

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:

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

4.1 Show that applying the activation function in the following way: $\boldsymbol{g}_t = \boldsymbol{W}\sigma(\boldsymbol{g}_{t-1})+\boldsymbol{U}\boldsymbol{x}_t+\boldsymbol{b}$ results in an equivalent recurrence as that defined above (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 $t-1$.

4.2 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 (here, use g_t as the definition of the hidden state), i.e.

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

Use the following properties of the L_2 operator norm

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

4.3 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? If this condition is not sufficient, in what scenario is it not exploding given the condition is met?(Answer in 1-2 sentences)

4.4 Assume we have the reverse problem of exploding gradient, what measures can we take to address it? Propose 2 strategies where one takes into account the direction of the gradient and the other does not. Which is preferred according to you and why?

Answer 4.

4.1 The hypothesis to prove is $\boldsymbol{h}_t = \sigma(\boldsymbol{g}_t)$. Assume that $\boldsymbol{h}_{t-1} = \sigma(\boldsymbol{g}_{t-1})$. Then, given the definititions of $\boldsymbol{h}_t$ and $\boldsymbol{g}_t$,

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

4.2 For consecutive units, the Jacobian is

$$\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_{t-1}} = \boldsymbol{W}\frac{\partial \sigma(\boldsymbol{h}_{t-1})}{\partial \boldsymbol{h}_{t-1}}$$

Recall the following properties of 2-norm:

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

from which we have

$$\left|\left|\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_{t-1}}\right|\right| \leq ||\boldsymbol{W}||\left|\left|\frac{\partial \sigma(\boldsymbol{h}_{t-1})}{\partial \boldsymbol{h}_{t-1}}\right|\right| \leq \frac{\delta}{\gamma}\gamma = \delta$$

which means the 2-norm is bounded by some $\delta \in [0, 1)$. Applying the sub-multiplicativity T times gives

$$\left|\left|\frac{\partial \boldsymbol{h}_T}{\partial \boldsymbol{h}_0}\right|\right| \leq \prod_{t=1}^{T}\left|\left|\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_{t-1}}\right|\right| \leq \delta^T \to 0 \text{ as } T \to \infty$$

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.

4.3 By contraposition, gradients of the hidden state would not become arbitrarily large if the largest eigenvalue of weights is not larger than $\frac{\delta^2}{\gamma^2}$. Thus it is a necessary condition for gradient explosion. It is not sufficient: the product of the norms can be greater than the norm of the product. This can happen if the hidden state is orthogonal to the largest eigenvector of $\boldsymbol{W}$.

4.4 The following strategies are all applicable:

(a) direction sensitive

i. gradient clipping by norm

ii. Backward Gradient Normalization

(b) direction agnostic

i. gradient clipping by value

ii. regularizations (eg. L2)

iii. initializations (eg. orthogonal initialization)

iv. special architectures(eg. LSTM)

Question 5 (5-5-5-5)**.** (**Paper Review: Show, Attend and Tell**) In this question, you are going to write a **one page review** of the Show, Attend and Tell paper. Your review should have the following four sections: Summary, Strengths, Weaknesses, and Reflections. For each of these sections, below we provide a set of questions you should ask about the paper as you read it. Then, discuss your thoughts about these questions in your review. In this question, you are going to write a **one page review** of the Show, Attend and Tell paper. Your review should have the following four sections: Summary, Strengths, Weaknesses, and Reflections. For each of these sections, below we provide a set of questions you should ask about the paper as you read it. Then, discuss your thoughts about these questions in your review.

(5.1) **Summary:**

(a) What is this paper about ?

(b) What is the main contribution ?

(c) Describe the main approach and results. Just facts, no opinions yet.

(5.2) **Strengths:**

(a) Is there a new theoretical insight ?

(b) Or a significant empirical advance ? Did they solve a standing open problem ?

(c) Or a good formulation for a new problem ?

(d) Any good practical outcome (code, algorithm, etc) ?

(e) Are the experiments well executed ?

(f) Useful for the community in general ?

(5.3) **Weaknesses:**

(a) What can be done better ?

(b) Any missing baselines ? Missing datasets ?

(c) Any odd design choices in the algorithm not explained well ? Quality of writing ?

(d) Is there sufficient novelty in what they propose ? Minor variation of previous work ?

(e) Why should anyone care ? Is the problem interesting and significant ?

(5.4) **Reflections:**

(a) How does this relate to other concepts you have seen in the class ?

(b) What are the next research directions in this line of work ?

(c) What (directly or indirectly related) new ideas did this paper give you ? What would you be curious to try ?

This question is subjective and so we will accept a variety of answers. You are expected to analyze the paper and offer your own perspective and ideas, beyond what the paper itself discusses.

Answer 5. We have accepted a diverse range of responses that effectively analyzed the paper and provided perspectives beyond what covered in the paper itself.