15-780: Graduate Artificial Intelligence
Homework 2

Due date: `2/10/2025 11:59pm`

Instructions

- Please provide detailed and well-reasoned responses to the questions below.
- Answers should be clear, concise, and well-supported with examples or explanations.
- Submit your responses as a PDF document on the course's gradescope page by the due date.

Questions

1 Linear language models and approximation error

Consider a language modeling task where the objective is to predict the next word x_i given the previous context $x_{1:i-1}$ where $x_{1:i-1} = x_1, x_2, \ldots x_{i-1}$ and every x_j is a word from the vocabulary $\mathcal{V}$ of size m.

Let $f^\star(x_i \mid x_{1:i-1})$ represent the "ground-truth" probability distribution. Our hypothesis class is linear classifiers over some feature representation of the context. We study the approximation error induced by different feature representations under different assumptions on the ground truth.

Let $\psi(x_{1:i-1})$ represent features of a context and the hypothesis class is parameterized by θ such that $h(x_{1:i-1}) = \theta^\top \psi(x_{1:i-1})$. The dimensionality of ψ, θ depends on how the features are constructed.

The one-hot encoding of a word $v \in \mathcal{V}$ is a vector $\delta(v) \in 0, 1^m$ where $\delta(v)$ has 1 in the position corresponding to v in the vocabulary and 0 elsewhere.

Part (a): Suppose the feature representation of $x_{1:i-1}$ is the one-hot encoding of the last word, i.e. $\psi(x_{1:i-1}) = \delta(x_{i-1})$.

Suppose $f^\star$ satisfies

$$f^\star(x_i \mid x_{1:i-1}) = f^\star(x_i \mid x_{i-1}), \tag{1}$$

Suppose we apply the cross-entropy loss function. Write the expression for $\theta^\star$ that minimizes the expected loss. Is the approximation error zero?

Part (b): We now make a different assumption on $f^\star$:

$$f^\star(x_i \mid x_{1:i-1}) = f^\star(x_i \mid x_{i-2}, x_{i-1}). \tag{2}$$

In other words, the probability of the next word depends on the two previous words.

Consider a natural different feature representations that is a concatenation of one-hot vectors of the previous two words, i.e. $\psi(x_{1:i-1}) = [\delta(x_{i-1}), \delta(x_{i-2})] \in R^{2m}$. Can a linear classifier with $\theta \in R^{2m\times m}$ always achieve an approximation error zero (under the cross-entropy loss)? If yes, write out the expression for $\theta^\star$ that achieves this for any $f^\star$ satisfying the condition above. If not, prove that there is an $f^\star$ that cannot be represented by a $\theta^\star$. You may use either a counting argument or given an explicit example of such $f^\star$.

Part (c): Repeat the exercise above for a new feature representation that is a flattened outer product of the one-hot vectors of the previous two words:

$$\psi(x_{1:i-1}) = \text{flatten}(\delta(x_{i-1}) \otimes \delta(x_{i-2})), \tag{3}$$

where $\psi(x_{1:i-1}) \in \mathbb{R}^{m^2}$.

2 Optimization methods

In this problem, you will explore the evolution of optimization algorithms, starting from Gradient Descent (GD) and progressively introducing improvements like Momentum, Adaptive Learning Rates, and Adam. Each part includes the update rule, and your task is to think critically about its purpose and effect while applying the algorithms to minimize the following convex function:

$$f(x, y) = 50x^2 + y^2,$$

where $x, y \in \mathbb{R}$. This function has "steep" curvature along the x-axis and "shallow" curvature along the y-axis, making it a good test case for optimization challenges.

2.1 Gradient Descent (GD)

Gradient Descent updates parameters by moving in the direction of steepest descent:

$$x_{k+1} = x_k - \eta \nabla f(x_k),$$

where $\eta > 0$ is the learning rate.

(a) Derive the gradient $\nabla f(x, y)$ for the given function.

(b) Why might a single fixed learning rate η be problematic for minimizing $f(x, y)$, given its steep curvature along the x-axis?

(c) Using $\eta = 0.1$ and the initial point $(x_0, y_0) = (1, 1)$, compute the first two iterations of Gradient Descent. Does this illustrate any challenges you described in part (b)?

2.2 Momentum

Momentum reduces oscillations by accumulating velocity from past gradients. The update rule is:

$$v_{k+1} = \beta v_k + \nabla f(x_k), \quad x_{k+1} = x_k - \eta v_{k+1},$$

where v_k is the accumulated velocity, $\beta \in [0, 1)$ is the momentum parameter, and $\eta > 0$ is the learning rate.

(a) Intuitively explain why adding a momentum term helps reduce oscillations along the steep x-axis.

(b) Compute the first two iterations of Momentum-Based Gradient Descent using $\beta = 0.9$, $\eta = 0.1$, and $(x_0, y_0) = (1, 1)$. Assume $v_0 = (0, 0)$. Compare the updates with those from Gradient Descent.

(c) Why might momentum alone still struggle to optimize $f(x, y)$?

2.3 Adaptive Learning Rates (AdaGrad)

AdaGrad adjusts the learning rate for each parameter based on the history of squared gradients:

$$\eta_i^{(k)} = \frac{\eta}{\sqrt{\sum_{j=1}^{k} (g_i^{(j)})^2 + \epsilon}}, \quad x_{i,k+1} = x_{i,k} - \eta_i^{(k)} g_i^{(k)},$$

where $g_i^{(j)}$ is the i-th component of the gradient at step j, and $\epsilon > 0$ prevents division by zero.

(a) Why is adjusting learning rates for each parameter helpful for a function like $f(x, y)$?

(b) Compute the effective learning rates $\eta_x^{(1)}$ and $\eta_y^{(1)}$ after the first iteration, using $\eta = 0.1$ and $\epsilon = 10^{-8}$.

(c) What happens to the learning rate over time?

2.4 RMSProp

RMSProp modifies AdaGrad by using an exponential moving average of squared gradients instead of the full history. The update rule is:

$$s_{k+1} = \gamma s_k + (1 - \gamma) g_k^2, \quad x_{k+1} = x_k - \frac{\eta}{\sqrt{s_{k+1} + \epsilon}} g_k,$$

where s_k is the EMA of squared gradients, and $\gamma \in [0, 1)$ is a decay factor.

(a) How does using an exponential moving average of squared gradients change the learning rates compared to Adagrad? Describe a clear advantage of RMSProp over Adagrad.

(b) Compare the role of γ in RMSProp with the role of β in momentum. What does each parameter control?

2.5 Adam

Adam combines momentum with adaptive learning rates. The update rule is:

$$m_{k+1} = \beta_1 m_k + (1-\beta_1) g_k, \quad v_{k+1} = \beta_2 v_k + (1-\beta_2) g_k^2,$$

$$\hat{m}_{k+1} = \frac{m_{k+1}}{1-\beta_1^k}, \quad \hat{v}_{k+1} = \frac{v_{k+1}}{1-\beta_2^k}, \quad x_{k+1} = x_k - \frac{\eta}{\sqrt{\hat{v}_{k+1}} + \epsilon} \hat{m}_{k+1}.$$

(a) Adam incorporates momentum and adaptive learning rate. Based on the equation above, identify which of m_k, v_k correspond to momentum and which to adaptive learning rate?