تمرین اول

سوال ۱: تحلیل انتشار رو به جلو، عقب و مرگ نورون‌ها (محاسباتی و مفهومی)

یک شبکه عصبی MLP با دو ورودی، یک لایه پنهان شامل دو نورون با تابع فعال‌سازی ReLU و یک لایه خروجی با یک نورون با تابع فعال‌سازی Sigmoid را در نظر بگیرید. فرض کنید برای یک نمونه آموزشی، بردار ورودی $x = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$ و برچسب واقعی $y = 1$ است. ماتریس‌های وزن و بایاس به صورت زیر مقداردهی اولیه شده‌اند:

$$W^{[1]} = \begin{bmatrix} 0.5 & 1.0 \\ 0.5 & -2.0 \end{bmatrix}, \quad b^{[1]} = \begin{bmatrix} -1.5 \\ 0.5 \end{bmatrix}$$

$$W^{[2]} = \begin{bmatrix} 1.0 & -1.0 \end{bmatrix}, \quad b^{[2]} = \begin{bmatrix} 0.0 \end{bmatrix}$$

همچنین تابع هزینه از نوع Binary Cross-Entropy (BCE) به همراه منظم‌سازی L2 با ضریب $\lambda = 0.1$ تعریف شده است.

الف) مقادیر پیش‌فعال‌سازی ($z^{[1]}$)، خروجی لایه پنهان ($a^{[1]}$) و پیش‌بینی نهایی ($\hat{y}$) را محاسبه کنید.

ب) گرادیان تابع هزینه کل (با احتساب ترم منظم‌سازی) نسبت به وزن $W^{[1]}_{1,1}$ (یعنی وزن متصل‌کننده ورودی اول به نورون اول لایه پنهان) را محاسبه کنید. *راهنمایی:* دقت کنید که وضعیت خروجی نورون اول لایه پنهان (Dying ReLU) چه تاثیری بر پس‌انتشار (Backpropagation) خطای پیش‌بینی دارد و آیا این موضوع به معنای صفر شدن **کامل** به‌روزرسانی این وزن در الگوریتم گرادیان کاهشی است؟ (به رابطه صفحه ۵۷ اسلایدها رجوع کنید).

سوال ۲: بحران محوشدگی گرادیان و اثرات ساختاری (مفهومی و اثباتی)

یک شبکه عمیق MLP با ۵۰ لایه پنهان را در نظر بگیرید که در تمامی لایه‌ها از تابع فعال‌سازی Sigmoid استفاده می‌کند.

الف) با استفاده از قاعده زنجیره‌ای (Chain Rule) و با استناد به مشتق تابع Sigmoid، اثبات کنید که چرا در طول آموزش، وزن‌های لایه‌های ابتدایی شبکه (نزدیک به ورودی) تقریباً هیچ تغییری نمی‌کنند (فرض کنید ماکزیمم مقدار مشتق Sigmoid برابر با ۰/۲۵ است و مقادیر وزن‌ها در حدود ۱ هستند).

ب) فرض کنید برای حل این مشکل، توابع فعال‌سازی را به ReLU تغییر می‌دهیم. یک دانشجوی بی‌دقت، به جای استفاده از Batch Normalization، تمام وزن‌ها را با مقادیر تصادفی بسیار بزرگ (مثلاً در بازه $[10, 20]$) مقداردهی اولیه می‌کند. توضیح دهید که چرا این کار با وجود تابع ReLU منجر به شکست فرآیند آموزش می‌شود؟ نقش Batch Normalization (طبق معادلات صفحه ۶۰: $\hat{z}_i = \frac{z_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$) در جلوگیری از این فاجعه چیست؟

سوال ۳: چالش کدنویسی بدون لپ‌تاپ (برداری‌سازی و پیاده‌سازی گام به گام)

یکی از مهم‌ترین مهارت‌ها در یادگیری عمیق، درک عملیات ماتریسی برای یک Mini-batch از داده‌ها بدون استفاده از حلقه‌های for است (همانطور که در اسلایدهای ۳۹ تا ۵۱ تاکید شده است).

فرض کنید یک Mini-batch با اندازه m داریم. بردار ورودی X ابعاد (n_x, m) دارد. لایه پنهان اول دارای n_h نورون است. خروجی این لایه $A^{[1]}$ است که ابعاد (n_h, m) دارد. ما در مرحله پس‌انتشار (Backpropagation) هستیم و گرادیان لایه بعدی نسبت به خروجی لایه پنهان، یعنی $dZ^{[2]}$ (با ابعاد $n_y \times m$) را در اختیار داریم.

کد (یا شبه‌کد دقیق NumPy) توابع زیر را فقط با استفاده از ضرب ماتریسی (عملگر @ یا np.dot) و عملیات عنصر به عنصر بنویسید. استفاده از هرگونه حلقه for نمره صفر در پی خواهد داشت.

```python
def backward_and_adam_update(X, A1, W2, dZ2, W1, b1, m, t, m_t, v_t):
    """
    Inputs:
    X: Input data, shape (n_x, m)
    A1: Activation of hidden layer (ReLU), shape (n_h, m)
    W2: Weights of output layer, shape (n_y, n_h)
    dZ2: Gradient of loss w.r.t pre-activation of layer 2, shape (n_y, m)
    m: batch size
    t: current timestep for Adam
    m_t, v_t: Previous first and second moments for W1

    Task 1: Compute dW1 and db1 (Gradients for Layer 1)
    Task 2: Apply Adam Optimizer update rule for W1 ONLY
            (use beta1=0.9, beta2=0.999, lr=0.01, epsilon=1e-8)
    """

    # 1. Compute dZ1 (Gradient of ReLU pre-activation)
    # Hint: A1 > 0 creates a boolean mask that acts as the derivative of ReLU
    dZ1 = ???

    # 2. Compute dW1 and db1 explicitly
    # (Watch out for dimensions! Must result in (n_h, n_x) and (n_h, 1))
    dW1 = ???
    db1 = ???

    # 3. Adam Optimizer Updates for W1 (Refer to slide 55 equations)
    m_t_new = ???
    v_t_new = ???

    m_t_hat = ???
    v_t_hat = ???

    W1_updated = ???

    return W1_updated, m_t_new, v_t_new
```

سوال تحلیلی مرتبط با کد: در محاسبه db1 چرا استفاده از دستور np.sum نیازمند تنظیم پارامتر keepdims=True است؟ اگر این پارامتر تنظیم نشود، چه اتفاقی برای Broadcasting در پایتون طی تکرار بعدی می‌افتد؟

سوال ۴: ترکیب Dropout و استنتاج (Inference)

مطابق صفحه ۵۹ اسلایدها، Dropout در زمان آموزش اعمال شده و در زمان تست (استنتاج) غیرفعال می‌شود تا خروجی مورد انتظار ثابت بماند. فرض کنید در یک لایه پنهان، خروجی یک نورون خاص پیش از اعمال Dropout برابر با $a = 10$ است. نرخ Dropout برابر با $p = 0.5$ (احتمال خاموش شدن نورون) تنظیم شده است.

الف) مقدار مورد انتظار (امید ریاضی – Expected Value) خروجی این نورون در طول فاز آموزش چقدر است؟

ب) در زمان تست (Inference) که Dropout خاموش است، اگر خروجی نورون مستقیماً همان $a = 10$ به لایه بعد ارسال شود، چه تضادی با فاز آموزش پیدا می‌کند و برای رفع این مشکل دقیقاً چه عملیات ریاضیاتی باید روی وزن‌ها یا خروجی این لایه انجام دهیم؟

سوال ۵: کدنویسی برای کمربند حیات

فرض کنید شما یک محقق در آژانس فضایی هستید. داده‌های ارسالی از یک تلسکوپ فضایی، دو ویژگی از سیارات کشف شده را به شما می‌دهد: X_1 (فاصله از ستاره مادر) و X_2 (ضخامت اتمسفر). بررسی‌ها نشان می‌دهد سیاراتی که قابلیت حیات دارند (کلاس ۱)، در یک حلقه یا کمربند خاص (نه خیلی نزدیک و گرم، نه خیلی دور و سرد) قرار گرفته‌اند و سایر سیارات (کلاس ۰) خارج از این حلقه یا در مرکز آن هستند.

برای اجرای این تمرین، نیازی به دانلود دیتاست ندارید. کد پایتون زیر دیتاست شما را در کسری از ثانیه تولید و رسم می‌کند:

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_circles
from sklearn.model_selection import train_test_split

# Generate the "Habitable Zone" dataset
X, y = make_circles(n_samples=600, noise=0.05, factor=0.5, random_state=42)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Plotting the dataset
plt.scatter(X[y==0, 0], X[y==0, 1], color='red', label='Uninhabitable (Class 0)')
plt.scatter(X[y==1, 0], X[y==1, 1], color='blue', label='Habitable (Class 1)')
plt.xlabel('Distance from Star (X1)')
plt.ylabel('Atmospheric Thickness (X2)')
plt.legend()
plt.title('Exoplanet Habitable Zone')
plt.show()
```

شرح ماموریت: شما مجاز هستید از هر کتابخانه‌ای مانند scikit-learn (کلاس MLPClassifier)، PyTorch یا TensorFlow/Keras استفاده کنید. آموزش این دیتاست روی ساده‌ترین CPU نیز کمتر از ۵ ثانیه زمان می‌برد.

فاز اول: اثبات بن‌بست مدل‌های خطی

یک مدل طبقه‌بند خطی ساده (مانند Logistic Regression یا Perceptron بدون لایه پنهان) روی این داده‌ها آموزش دهید.

- **سوال:** دقت (Accuracy) این مدل روی داده‌های تست چقدر است؟ با رسم مرز تصمیم‌گیری (Decision Boundary) روی نمودار، دلیل ریاضی و هندسی این شکست را بر اساس مفهوم قابلیت تفکیک‌پذیری خطی (Linear Separability) که در کلاس تدریس شد، در دو خط توضیح دهید.

فاز دوم: چالش معماری کمینه (The Minimalist Architect)

همان‌طور که در قضیه تقریب عمومی (Universal Approximation Theorem) خواندیم، یک MLP می‌تواند هر تابعی را تخمین بزند. اکنون یک MLP **فقط با یک لایه پنهان** بسازید. تابع فعال‌سازی لایه پنهان را ReLU و لایه خروجی را Sigmoid قرار دهید.

- **سوال:** هدف رسیدن به دقت بالای ٪۹۵ روی داده‌های تست است. **کمترین** تعداد نورونی که می‌توانید در این یک لایه پنهان قرار دهید تا شبکه بتواند این مجموعه داده (حلقه‌های تودرتو) را با موفقیت تفکیک کند، چند تاست؟
- **تحلیل هندسی:** شبکه را با ۱، ۲، ۳ و ۴ نورون در لایه پنهان آموزش دهید و مرز تصمیم را برای هر کدام رسم کنید. چرا با ۱ یا ۲ نورون، شبکه **هرگز** نمی‌تواند این مسئله را حل کند، اما با عدد بزرگتر حل می‌شود؟ (راهنمایی: هر نورون ReLU در لایه پنهان، معادل کشیدن چه شکل هندسی در فضای ورودی است تا یک چندضلعی بسته دور سیارات کلاس ۱ ایجاد شود؟)

فاز سوم: دام توابع فعال‌سازی (Activation Trap)

حال معماری شبکه را به ۳ لایه پنهان، هر کدام با ۱۰ نورون تغییر دهید (یک شبکه نسبتاً عمیق برای این مسئله). نرخ یادگیری (Learning Rate) را روی 0.01 تنظیم کنید. شبکه را دو بار آموزش دهید:

۱. دفعه اول: تابع فعال‌سازی تمام لایه‌های پنهان Sigmoid باشد.
۲. دفعه دوم: تابع فعال‌سازی تمام لایه‌های پنهان ReLU باشد.

- **سوال:** منحنی افت خطای (Loss Curve) هر دو شبکه را در طول Epochها روی یک نمودار رسم کنید. چه تفاوت فاحشی در سرعت همگرایی مشاهده می‌کنید؟ این پدیده دقیقاً به کدام مشکل معروف در شبکه‌های عصبی (که در اسلایدها بررسی شد) اشاره دارد و مشتقات توابع چگونه آن را توجیه می‌کنند؟