Bài tập Llama2

Trần Hồng Đăng MSSV 22022646 | Hồ Tú Minh MSSV 22022674 | Bùi Thế Long MSSV 22022647 | Ngô Văn Kiệt MSSV 22022643

I. KIẾN TRÚC CỦA LLAMA2

LLaMA2 là một mô hình **Transformer Decoder-only**, gồm nhiều lớp Transformer xếp chồng lên nhau. Mỗi khối Transformer bao gồm:

RMSNorm (Root Mean Square Layer Normalization)

RMSNorm thay thế LayerNorm truyền thống, giúp ổn định huấn luyện và giảm độ phức tạp tính toán. Công thức chuẩn hóa RMSNorm:

$$\|\mathbf{x}\|_{RMS} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2 + \epsilon}$$

trong đó:

- x là tensor đầu vào.
- n là số phần tử của tensor x.
- x_i là thành phần của tensor x.

Grouped Query Attention (GQA)

GQA giúp giảm yêu cầu bộ nhớ bằng cách nhóm Query, sử dụng chung Key-Value. Công thức tính Attention Score:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

Rotary Positional Embedding (RoPE)

Thay vì Positional Encoding cố định, RoPE mã hóa vị trí bằng cách xoay ma trận embedding, cụ thể: Query và Key được reshaped để phân tách thành phần chẵn/lẻ, tính toán tần số quay cho mỗi chiều của head:

$$\theta_i = \theta^{-\frac{2i}{d}}$$

Áp dụng phép quay (RoPE Transform)

$$\begin{bmatrix} q_1' \\ q_2' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \tag{2}$$

trong đó θ phụ thuộc vào vị trí token.

II. HÀM TỐI ƯU

AdamW Optimizer

AdamW tối ưu hóa việc cập nhật trọng số, giúp huấn luyện nhanh và giảm overfitting, cụ thể là: Lấy **state** hiện tại của p (các tham số **weight, bias**), nếu **state** chưa được khởi tạo thì khởi tạo **state** với:

- Moment bậc 1: m
- Moment bậc 2: v
- Số bước: $step = 0$

Nếu state đã tồn tại, lấy $m, v, step$ của state hiện tại. Cập nhật gradient bậc 1, bậc 2 theo công thức:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{3}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{4}$$

$$\tag{5}$$

Bias correction để hiệu chỉnh độ lệch nhằm khắc phục sự thiên lệch trong các giá trị moment ban đầu

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

Cập nhật tham số theo công thức

$$\theta_t = \theta_{t-1} - \frac{\alpha}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t \tag{6}$$

Thêm **weight decay**:

$$\theta_t = \theta_t - \alpha \cdot \lambda \cdot \theta_t$$

Cross Entropy Loss

Hàm mất mát sử dụng trong huấn luyện mô hình:

$$L = -\sum y_i \log(\hat{y}_i) \tag{7}$$

III. TEXT GENERATION

Mô hình sinh văn bản bằng cách lấy mẫu theo nhiệt độ:

1) Tính xác suất softmax sau khi chia logits cho T:

$$P(y_i) = \frac{e^{z_i/T}}{\sum_j e^{z_j/T}} \tag{8}$$

2) Nếu $T = 0.0$, chọn token có xác suất cao nhất.
3) Nếu $T > 0.0$, lấy mẫu từ phân phối xác suất.

IV. KẾT QUẢ VÁ TỐI ÚU THAM SỐ

A. Text Continuation

- **temperature = 0**:
  "I have wanted to see this thriller for a while, and it didn't disappoint. Keanu Reeves, playing the hero John Wick, is this day. He was playing with his toy car, driving it around the living room. Suddenly, he heard a loud crash. He had broken the car and was very sad. John was angry and he shouted at his little brother. He was only three years old and he was only three. He was only three years old. He was very ups." (Câu văn đúng ngữ pháp tiếng Anh, từ vựng hẹp)
- **temperature = 1**:
  "I have wanted to see this thriller for a while, and it didn't

disappoint. Keanu Reeves, playing the hero John Wick, is it!" As John toddled up to the sweet aroma, he held his mum's hand tight, hoping he would finally reach the top step. But as he stepped, he felt so heavy, like an apple. He had made a mistake! His mum apologized, then hurriedly left, so he started to cry. His m." (Câu văn gần đúng ngữ pháp, có lỗi chính tả, từ vựng sáng tạo hơn)

B. Zero-shot, Prompt-based Sentiment Analysis

Dataset	Epoch	Batch Size	Learning Rate	Dropout
SST	5	80	$2e^{-5}$	0.3
CFIMDB	5	10	$2e^{-5}$	0.3

TABLE I
CÁC THAM SỐ SỬ DỤNG TRONG PHÂN TÍCH CẢM XÚC

C. Task-specific Fine Tuning

Dataset	Dev Acc	Test Acc
SST	0.411	0.410
CFIMDB	0.857	0.469

TABLE II
KẾT QUẢ FINE-TUNING VỚI SIÊU THAM SỐ MẶC ĐỊNH

1) Với siêu tham số mặc định:

2) Tối ưu hóa siêu tham số (5 Epochs):

- **SST**: Batch Size = 64, Learning Rate = $2e^{-5}$

Dev Acc	Test Acc
0.428	0.426

TABLE III
KẾT QUẢ TỐI ƯU HÓA SST

- **CFIMDB**: Batch Size = 10, Learning Rate = $2e^{-6}$

Dev Acc	Test Acc
0.918	0.674

TABLE IV
KẾT QUẢ TỐI ƯU HÓA CFIMDB

3) Thử nghiệm với các tỉ lệ dropout khác nhau:

Dataset	Dropout = 0	Dropout = 0.3	Dropout = 0.5
SST Dev Acc	0.421	**0.428**	0.427
SST Test Acc	0.406	**0.426**	0.414
CFIMDB Dev Acc	0.690	**0.918**	0.600
CFIMDB Test Acc	0.545	**0.674**	0.193

TABLE V
KẾT QUẢ THỬ NGHIỆM VỚI CÁC TỈ LỆ DROPOUT KHÁC NHAU

D. Nhận xét và Kết luận

- **Tham số tối ưu cho SST**:
  - Learning rate = $2e^{-5}$
  - Batch size = 64
  - Dropout = 0.3
- **Tham số tối ưu cho CFIMDB**:
  - Learning rate = $2e^{-6}$
  - Batch size = 10
  - Dropout = 0.3
- **Hướng cải thiện**:
  - Train trên nhiều epoch hơn