Rani Nirmala Prakoso (22/493982/TK/54153)

LAPORAN TUGAS INDIVIDU

Implementasi Arsitektur Transformer dari Nol dengan NumPy

A. Desain Arsitektur

Model yang dibangun adalah decoder-only Transformer: urutan token → embedding + positional encoding → L × (Decoder Block) → proyeksi ke logits → softmax token terakhir.

- Decoder Block (pre-norm): tiap blok berisi dua sub-lapisan:
- LayerNorm → Multi-Head Self-Attention (+ residual)
- LayerNorm → Feed-Forward (FFN) (+ residual)
- Alasan pre-norm : LayerNorm diletakkan sebelum sub-lapisan untuk stabilitas numerik saat menumpuk banyak layer; residual menjaga aliran gradien dan mempertahankan dimensi tetap d sehingga blok bisa ditumpuk.
- Multi-Head Self-Attention: memproyeksikan masukan menjadi Q, K, V menghitung perhatian per-head (scaled dot-product), menggabungkan (concat), lalu proyeksi W0. Ini memungkinkan model menangkap berbagai pola relasi secara paralel.
- FFN (position-wise): dua linear + aktivasi (ReLU/GELU), diterapkan per posisi untuk memperkaya transformasi fitur.

B. Alasan Pemilihan Positional Encoding

Positional encoding yang digunakan adalah versi sinusoidal seperti pada paper asli Transformer. Di paper ini, pada **Bagian 3.5 (Positional Encoding)** dijelaskan bahwa karena Transformer tidak memiliki struktur berurutan seperti RNN/CNN, posisi token ditambahkan menggunakan **sinusoidal positional encoding**.

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \qquad PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

- pos → posisi dalam urutan
- i → dimensi embedding
- d_model → dimensi model

Alasan pemilihan metode ini adalah karena tidak menambah parameter baru dan dapat menggeneralisasi ke panjang sekuens yang lebih besar dari data pelatihan. Representasi sinusoidal juga memberikan pola periodik sehingga model bisa membedakan urutan posisi token.

C. Penjelasan Causal Mask

Self-attention secara default dapat "melihat" semua posisi, termasuk masa depan. Agar autoregresif, kita menambahkan causal mask (segitiga atas) bernilai $-\infty$ke skor $QK^T/\sqrt{d_k}$ sehingga setelah softmax bobot perhatian ke masa depan menjadi 0. Artinya, token di posisi ttt hanya boleh menggunakan informasi dari posisi $\leq t$.

D. Bukti Uji Sederhana

Pengujian dilakukan dengan input acak berupa token integer. Hasil uji menunjukkan bahwa dimensi keluaran sesuai spesifikasi:

1. Logits : (batch, seq_len, vocab_size)
2. Probabilitas (softmax) : distribusi valid pada posisi terakhir token

3. Attention map : (batch, heads, seq_len, seq_len)

Dengan hasil,

bentuk input:

[[82 54 170 80 191 138 19 71 153 152 72 125 93 171 148 95 93 83 106 121]
[0 198 15 128 39 110 108 47 188 176 40 159 5 48 179 94 199 136 30 178]]

logits shape: (2, 20, 200)

probs shape: (2, 20, 200)

attn layers: 4

attn layer shape example: (2, 8, 20, 20)

E. Kesimpulan

Implementasi decoder-only Transformer dengan NumPy berhasil dibuat sesuai instruksi. Semua komponen utama berfungsi, hasil uji sesuai ekspektasi, dan fitur bonus telah ditambahkan.

F. Lampiran

Komponen	Fungsi / Peran	Input → Output	Keterangan
Token Embedding	Mengubah indeks token menjadi vektor representasi berdimensi d_model.	Token ID (batch, seq_len) → Vektor (batch, seq_len, d_model)	Matriks embedding diinisialisasi acak.
Positional Encoding	Memberikan informasi posisi ke token embedding.	(seq_len, d_model)	Versi sinusoidal, tidak menambah parameter.
Scaled Dot-Product Attention	Menghitung skor perhatian antar token dengan Q, K, V.	Q,K,V (batch, heads, seq, d_head) → Output + Weights	Menggunakan causal mask untuk mencegah akses masa depan.
Multi-Head Attention	Menjalankan attention pada beberapa head lalu menggabungkannya.	Input (batch, seq, d_model) → Output (batch, seq, d_model)	Menangkap pola relasi berbeda per head.
Feed Forward Network (FFN)	Transformasi non-linear untuk tiap posisi secara independen.	(batch, seq, d_model) → (batch, seq, d_model)	Dua layer linear dengan aktivasi ReLU.
Residual Connection + LayerNorm	Menstabilkan training, menjaga informasi awal.	Input + Output layer → Normalized Output	Pre-norm digunakan di setiap sub-layer.
Decoder Block	Satu unit Transformer decoder: MHA + FFN + residual + layer norm.	(batch, seq, d_model) → (batch, seq, d_model)	Bisa ditumpuk berkali-kali.
Causal Mask	Membatasi token hanya bisa melihat ke kiri (masa lalu).	(seq, seq) lower-triangular mask	Penting untuk model autoregresif (GPT-style).
Output Projection + Softmax	Proyeksi ke ukuran vocab lalu softmax untuk distribusi probabilitas.	(batch, seq, vocab_size)	Prediksi token berikutnya diambil dari posisi terakhir.

G. Referensi

1. DorsaRoh. *transformer-from-scratch* (GitHub repository). Tersedia secara daring di: https://github.com/DorsaRoh/transformer-from-scratch
2. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). *Attention Is All You Need*. arXiv preprint arXiv:1706.03762.