Nama : Muhammad Husain Kaasyiful Ghitha
NIM : 1103220047
Kelas TK-45-G13

Deep Learning
Tugas Week 5 - Penjelasan Persamaan Matematika pada Arsitektur RNN, LSTM, dan GRU

1. Recurrent Neural Network (RNN)
   a. Persamaan Dasar RNN

      RNN adalah jaringan saraf yang memiliki loop internal, memungkinkannya untuk mempertahankan informasi. Persamaan sederhana untuk RNN adalah:

$$h_t = \tanh(W_{hx} \cdot x_t + W_{hh} \cdot h_{t-1} + b_h)$$
$$y_t = W_{hy} \cdot h_t + b_y$$

      Dimana:

      - h_t adalah hidden state pada waktu t
      - x_t adalah input pada waktu t
      - W_{hx} adalah bobot dari input ke hidden state
      - W_{hh} adalah bobot dari hidden state sebelumnya ke hidden state saat ini
      - W_{hy} adalah bobot dari hidden state ke output
      - b_h dan b_y adalah bias
      - tanh adalah fungsi aktivasi hyperbolic tangent yang memetakan nilai ke rentang [-1, 1]

   b. Masalah Vanishing/Exploding Gradient

      Problem utama dengan RNN sederhana adalah masalah vanishing atau exploding gradient selama backpropagation through time (BPTT). Ini terjadi ketika gradien dikalikan berulang kali dengan matriks bobot W_{hh}:

      Jika nilai eigen dari $W_{hh} < 1$, gradien akan menghilang (vanish) Jika nilai eigen dari $W_{hh} > 1$, gradien akan meledak (explode)

      Ini membuat RNN sederhana sulit untuk mempelajari dependensi jangka panjang.

2. Long Short-Term Memory (LSTM)

   LSTM dirancang untuk mengatasi masalah vanishing/exploding gradient dengan memperkenalkan mekanisme gating.

   a. Persamaan LSTM

      LSTM memiliki beberapa gerbang (gates) yang mengontrol aliran informasi:

      1. Forget Gate (f_t): Mengontrol informasi apa yang harus dilupakan dari cell state sebelumnya

$$f_t = \sigma\big(W_f \cdot [h_{t-1}, x_t] + b_f\big)$$

      2. Input Gate (i_t): Mengontrol informasi baru apa yang harus disimpan di cell state

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

      3. Cell State Candidate ($\tilde{C}_t$): Nilai baru yang berpotensi ditambahkan ke cell state

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + \ b_C)$$

      4. Cell State Update (C_t): Memperbarui cell state lama dengan informasi baru

$$C_t \ = \ f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

      5. Output Gate (o_t): Mengontrol bagian dari cell state yang akan menjadi output

$$o_t \ = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

      6. Hidden State (h_t): Output akhir dari unit LSTM

$$h_t \ = \ o_t \odot \tanh(C_t)$$

      Dimana:

      - σ adalah fungsi sigmoid, yang menghasilkan nilai antara 0 dan 1
      - $\odot$ adalah perkalian elemen-wise (Hadamard product)
      - $[h_{t-1}, x_t]$ adalah concatenation dari hidden state sebelumnya dan input saat ini

LSTM memungkinkan gradien mengalir melalui cell state tanpa banyak modifikasi, yang membantu mengurangi masalah vanishing gradient.

3. Gated Recurrent Unit (GRU)

   GRU adalah versi yang lebih sederhana dari LSTM dengan performa yang sebanding namun dengan parameter yang lebih sedikit.

   a. Persamaan GRU

      GRU hanya memiliki dua gerbang:

      1. Reset Gate (r_t): Menentukan seberapa banyak informasi sebelumnya yang akan dilupakan

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r)$$

      2. Update Gate (z_t): Menentukan seberapa banyak informasi sebelumnya yang akan dipertahankan

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z)$$

      3. Candidate Hidden State ($\tilde{h}_t$): Hidden state kandidat baru

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t] + b_h)$$

      4. Hidden State Update (h_t): Memperbarui hidden state

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

      Perhatikan bahwa GRU tidak memiliki cell state terpisah seperti LSTM. GRU menggabungkan forget gate dan input gate menjadi satu update gate.

4. Dropout

   Dropout adalah teknik regularisasi yang membantu mencegah overfitting dengan secara acak "mematikan" neuron selama training dengan probabilitas p:

$$\hat{h} = h \odot \text{mask}, \ \text{mask} \sim \text{Bernoulli}(1-p)$$

   Selama inferensi, semua neuron aktif tapi outputnya diskala dengan faktor $(1-p)$ untuk menjaga nilai yang diharapkan tetap sama:

$$\hat{h} = (1-p) \cdot h$$

Dalam implementasi modern, ini sering digantikan dengan "inverted dropout" di mana penskalaan dilakukan selama training, bukan inferensi.

5. Binary Cross-Entropy Loss

   Untuk klasifikasi sentimen biner (positif/negatif), kita menggunakan binary cross-entropy loss:

$$L = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

   Dimana:

   - N adalah jumlah sampel
   - y_i adalah label aktual (0 atau 1)
   - $\hat{y}_i$ adalah prediksi model (probabilitas antara 0 dan 1)

6. Metrik Evaluasi

   a. Akurasi

$$\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}}$$

   Akurasi mengukur proporsi prediksi yang benar dari total prediksi. Ini menunjukkan seberapa sering model memberikan prediksi yang tepat.

   b. Presisi

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}}$$

   Precision mengukur proporsi prediksi positif yang benar-benar positif. Ini penting ketika biaya false positive tinggi.

   c. Recall

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}}$$

Recall (juga dikenal sebagai sensitivitas) mengukur proporsi positif aktual yang diidentifikasi dengan benar. Ini penting ketika biaya false negative tinggi.

d. F1-Score

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

F1-Score adalah rata-rata harmonik dari precision dan recall, yang memberikan metrik tunggal yang menyeimbangkan keduanya. Ini berguna ketika distribusi kelas tidak seimbang.

e. Area Under the ROC Curve (AUC)

AUC mengukur area di bawah kurva ROC (Receiver Operating Characteristic), yang menggambarkan trade-off antara true positive rate (TPR) dan false positive rate (FPR) pada berbagai threshold.

$$\text{TPR} = \text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP}+\text{TN}}$$

AUC bernilai dari 0 hingga 1, di mana:

- AUC = 0.5 menunjukkan model tidak lebih baik dari random guessing
- AUC > 0.5 menunjukkan model memiliki kemampuan diskriminatif
- AUC = 1.0 menunjukkan model sempurna