Nama : Pandu Kaya Hakiki

NIM : 1103220016

Kelas : Deep Learning TK-45-G13

Analisis Matematis Pengembangan Model MLP untuk Dataset RegresiUTSTelkom

1. Modifikasi Arsitektur MLP untuk Mengatasi Underfitting

Ketika model MLP dengan arsitektur 3 hidden layer (256-128-64) mengalami underfitting, perlu dilakukan modifikasi sistematis dengan memperhatikan bias-variance tradeoff. Secara matematis, underfitting terjadi ketika model memiliki bias tinggi dan variance rendah, sehingga:

Formulasi Matematis Model MLP

Untuk model MLP dengan L layer, kita dapat mendefinisikan:

$$f(x;\theta) = h^{(L)} = \sigma^{(L)}(W^{(L)}h^{(L-1)} + b^{(L)})$$

dimana:

- $h^{(l)} = \sigma^{(l)}(W^{(l)}h^{(l-1)} + b^{(l)})$ untuk $l = 1, 2, \ldots, L$
- $h^{(0)} = x$ (input)
- $\sigma^{(l)}$ adalah fungsi aktivasi pada layer l
- $W^{(l)}$ adalah matriks bobot pada layer l
- $b^{(l)}$ adalah vektor bias pada layer l

Analisis Matematis Kapasitas Model

Kapasitas model dapat ditingkatkan melalui:

1. Penambahan kompleksitas layer:
   - $W^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$ dimana n_l adalah jumlah neuron
   - Peningkatan dari $(256, 128, 64)$ menjadi $(512, 256, 128)$ meningkatkan parameter dari $\sum_{l=1}^{L} n_l \times (n_{l-1} + 1)$ menjadi jumlah yang lebih besar
2. Optimalisasi fungsi aktivasi:
   - ReLU: $\sigma(z) = \max(0, z)$
   - Leaky ReLU: $\sigma(z) = \max(\alpha z, z)$ dimana α adalah parameter kecil (misal 0.01)
   - SELU: $\sigma(z) = \lambda \begin{cases} z & \text{jika } z > 0 \\ \alpha(e^z - 1) & \text{jika } z \leq 0 \end{cases}$
3. Pengurangan regularisasi:
   - L2 regularisasi: $\Omega(\theta) = \frac{\lambda}{2} \sum_{l=1}^{L} \|W^{(l)}\|_F^2$

- Dropout: $h^{(l)} = m^{(l)} \odot h^{(l)}$ dimana $m^{(l)} \sim \text{Bernoulli}(p)$
- Mengurangi nilai λ atau probabilitas dropout p

2. Analisis Matematis Fungsi Loss Alternatif

Selain Mean Squared Error (MSE), beberapa alternatif loss function dengan formulasi matematis sebagai berikut:

MSE (Mean Squared Error)

$$\mathcal{L}_{MSE}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

MAE (Mean Absolute Error)

$$\mathcal{L}_{MAE}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

Gradien terhadap prediksi:

$$\frac{\partial \mathcal{L}_{MAE}}{\partial \hat{y}_i} = \begin{cases} -1 & \text{jika } y_i > \hat{y}_i \\ 1 & \text{jika } y_i < \hat{y}_i \\ \text{tidak terdefinisi} & \text{jika } y_i = \hat{y}_i \end{cases}$$

Huber Loss

$$\mathcal{L}_{Huber}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} \begin{cases} \frac{1}{2}(y_i - \hat{y}_i)^2 & \text{jika } |y_i - \hat{y}_i| \leq \delta \\ \delta(|y_i - \hat{y}_i| - \frac{\delta}{2}) & \text{jika } |y_i - \hat{y}_i| > \delta \end{cases}$$

Gradien terhadap prediksi:

$$\frac{\partial \mathcal{L}_{Huber}}{\partial \hat{y}_i} = \begin{cases} -(y_i - \hat{y}_i) & \text{jika } |y_i - \hat{y}_i| \leq \delta \\ -\delta \cdot \text{sgn}(y_i - \hat{y}_i) & \text{jika } |y_i - \hat{y}_i| > \delta \end{cases}$$

Log-cosh Loss

$$\mathcal{L}_{LogCosh}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} \log(\cosh(y_i - \hat{y}_i))$$

Gradien terhadap prediksi:

$$\frac{\partial \mathcal{L}_{LogCosh}}{\partial \hat{y}_i} = -\tanh(y_i - \hat{y}_i)$$

3. Analisis Matematis Dampak Perbedaan Skala Fitur

Misalkan kita memiliki dua fitur $x_1 \in [0, 1]$ dan $x_2 \in [100, 1000]$ sebagai input untuk MLP.

Efek pada Forward Propagation

Untuk neuron pertama pada hidden layer pertama:

$$z_j^{(1)} = \sum_{i=1}^{d} w_{ji}^{(1)} x_i + b_j^{(1)} = w_{j1}^{(1)} x_1 + w_{j2}^{(1)} x_2 + b_j^{(1)}$$

Kontribusi relatif:

- $|w_{j1}^{(1)} x_1| \leq |w_{j1}^{(1)}|$ karena $x_1 \in [0, 1]$
- $|w_{j2}^{(1)} x_2| \in [100|w_{j2}^{(1)}|, 1000|w_{j2}^{(1)}|]$ karena $x_2 \in [100, 1000]$

Dengan inisialisasi bobot yang serupa (misalnya $w_{j1}^{(1)} \approx w_{j2}^{(1)}$), kontribusi x_2 akan mendominasi.

Dampak pada Backward Propagation

Gradien terhadap bobot:

$$\frac{\partial \mathcal{L}}{\partial w_{ji}^{(1)}} = \frac{\partial \mathcal{L}}{\partial z_j^{(1)}} \cdot \frac{\partial z_j^{(1)}}{\partial w_{ji}^{(1)}} = \frac{\partial \mathcal{L}}{\partial z_j^{(1)}} \cdot x_i$$

Untuk gradien bobot yang terkait dengan kedua fitur:

- $\frac{\partial \mathcal{L}}{\partial w_{j1}^{(1)}} = \frac{\partial \mathcal{L}}{\partial z_j^{(1)}} \cdot x_1 \in [0, \frac{\partial \mathcal{L}}{\partial z_j^{(1)}}]$
- $\frac{\partial \mathcal{L}}{\partial w_{j2}^{(1)}} = \frac{\partial \mathcal{L}}{\partial z_j^{(1)}} \cdot x_2 \in [100\frac{\partial \mathcal{L}}{\partial z_j^{(1)}}, 1000\frac{\partial \mathcal{L}}{\partial z_j^{(1)}}]$

Update bobot dengan Gradient Descent:

$$w_{ji}^{(1)} \leftarrow w_{ji}^{(1)} - \eta \frac{\partial \mathcal{L}}{\partial w_{ji}^{(1)}} = w_{ji}^{(1)} - \eta \frac{\partial \mathcal{L}}{\partial z_j^{(1)}} \cdot x_i$$

Perbandingan magnitude update:

$$\frac{|\Delta w_{j2}^{(1)}|}{|\Delta w_{j1}^{(1)}|} = \frac{|x_2|}{|x_1|} \in [100, 1000]$$

Solusi: Normalisasi Fitur

Standardisasi:

$$x_i' = \frac{x_i - \mu_i}{\sigma_i}$$

Min-Max Scaling:

$$x_i' = \frac{x_i - \min(x_i)}{\max(x_i) - \min(x_i)}$$

Setelah normalisasi, $x_1', x_2' \in [0, 1]$ atau $x_1', x_2' \sim \mathcal{N}(0, 1)$, sehingga $\frac{|\Delta w_{j2}^{(1)}|}{|\Delta w_{j1}^{(1)}|} \approx 1$.

4. Metode Matematis Pengukuran Kontribusi Fitur

Permutation Importance

Untuk fitur j, permutation importance didefinisikan sebagai:

$$I(x_j) = \mathbb{E}_{X,y}[L(f(X), y)] - \mathbb{E}_{X,y}[L(f(X^j), y)]$$

dimana:

- X^j adalah dataset dengan nilai fitur j yang diacak
- L adalah fungsi loss
- f adalah model yang dilatih

Implementasi empiris:

$$I(x_j) \approx \frac{1}{n} \sum_{i=1}^{n} L(f(x_i), y_i) - \frac{1}{n} \sum_{i=1}^{n} L(f(x_i^j), y_i)$$

Integrated Gradients

Untuk fitur i, integrated gradients didefinisikan sebagai:

$$IG_i(x) = (x_i - x_i') \times \int_{\alpha=0}^{1} \frac{\partial f(x' + \alpha(x - x'))}{\partial x_i} d\alpha$$

dimana:

- x' adalah baseline input (biasanya vektor nol)
- α adalah parameter interpolasi
- $\frac{\partial f}{\partial x_i}$ adalah gradien output model terhadap fitur input i

Aproksimasi diskrit:

$$IG_i(x) \approx (x_i - x_i') \times \sum_{k=1}^{m} \frac{\partial f(x' + \frac{k}{m}(x - x'))}{\partial x_i} \times \frac{1}{m}$$

SHAP (SHapley Additive exPlanations)

Nilai Shapley untuk fitur i:

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} [f(S \cup \{i\}) - f(S)]$$

dimana:

- N adalah himpunan semua fitur

- S adalah subset fitur tanpa fitur i
- $f(S)$ adalah nilai prediksi model dengan hanya menggunakan fitur dalam S

5. Desain Eksperimen Matematis untuk Optimalisasi Parameter

Learning Rate Range Test

Prosedur:

1. Definisikan learning rate η yang meningkat secara eksponensial:

$$\eta_t = \eta_{min} \cdot \left(\frac{\eta_{max}}{\eta_{min}}\right)^{\frac{t}{T}}$$

   dimana t adalah iterasi dan T adalah total iterasi

2. Evaluasi loss pada setiap iterasi dan identifikasi η^* yang memberikan penurunan loss terbesar:

$$\eta^* = \arg\min_{\eta_t} \frac{d\mathcal{L}(\eta_t)}{d\eta_t}$$

Analisis Batch Size

Ketika meningkatkan batch size dari B menjadi kB (untuk $k > 1$):

1. Noise dalam estimasi gradien menurun dengan faktor $\sqrt{k}$:

$$\text{Var}(\nabla\mathcal{L}_B) = \frac{\sigma^2}{B}$$

$$\text{Var}(\nabla\mathcal{L}_{kB}) = \frac{\sigma^2}{kB} = \frac{\text{Var}(\nabla\mathcal{L}_B)}{k}$$

2. Learning rate dapat ditingkatkan secara proporsional:

$$\eta_{kB} \approx k \cdot \eta_B$$

Optimalisasi Terkait Komputasi

Waktu pelatihan per epoch dapat dimodelkan sebagai:

$$T(B) = c_1 + \frac{c_2}{B}$$

dimana:

- c_1 adalah overhead konstan
- c_2 adalah biaya komputasi yang bergantung pada jumlah batch

Tradeoff antara kecepatan konvergensi dan waktu komputasi:

$$\mathcal{E}(B, \eta) = \frac{\text{Konvergensi}(\eta)}{T(B)}$$

dimana $\text{Konvergensi}(\eta)$ mengukur seberapa cepat model mencapai error minimal dengan learning rate η.

Dengan pendekatan matematis ini, kita dapat menentukan kombinasi optimal (B^*, η^*) yang memaksimalkan efisiensi pelatihan:

$$B^*, \eta^* = \arg\max_{B,\eta} \mathcal{E}(B, \eta)$$

Analisis matematika ini memberikan kerangka yang sistematik dan terukur untuk mengoptimalkan arsitektur dan proses pelatihan model MLP pada dataset RegresiUTSTelkom.