Nama : Muhammad Husain Kaasyiful Ghitha

NIM : 1103220047

Kelas TK-45-G13

Deep Learning

Jawaban Soal Analisis UTS model MLP

1. Jika menggunakan model MLP dengan 3 hidden layer (256-128-64) menghasilkan underfitting pada dataset ini, modifikasi apa yang akan dilakukan pada arsitektur? Jelaskan alasan setiap perubahan dengan mempertimbangkan bias-variance tradeoff!

   Jika model MLP dengan arsitektur 3 hidden layer (256-128-64) mengalami underfitting, berikut modifikasi yang dapat dilakukan:
   Modifikasi yang Direkomendasikan:

   - Meningkatkan Kapasitas Model:
     - Menambah jumlah neuron per layer: (512-256-128) atau (1024-512-256)
     - Menambah jumlah hidden layer: menjadi 4-5 layer dengan penurunan neuron bertahap

     Alasan: Underfitting menunjukkan model tidak memiliki cukup kapasitas untuk menangkap kompleksitas data. Data memiliki 91 fitur (kolom 0-90) dan pola yang kompleks, sehingga membutuhkan representasi yang lebih kaya.

   - Mengubah Fungsi Aktivasi: Mengganti ReLU dengan Leaky ReLU atau ELU
     Alasan: Fungsi aktivasi non-linear yang lebih ekspresif dapat membantu model mempelajari hubungan non-linear yang kompleks dalam dataset. Leaky ReLU/ELU mencegah masalah "dying ReLU" yang bisa membatasi kemampuan belajar.

   - Mengurangi Regularisasi:
     - Menurunkan nilai regularisasi L1/L2 jika diterapkan
     - Mengurangi dropout rate

     Alasan: Regularisasi yang terlalu kuat dapat menjadi penyebab underfitting karena terlalu membatasi kapasitas model.

   - Menerapkan Residual Connections (Skip Connections): Menambahkan koneksi residual antara layer
     Alasan: Memungkinkan gradien mengalir lebih baik melalui network yang dalam, memudahkan optimasi dan meningkatkan kapasitas model tanpa menambah masalah vanishing gradient.

Perspektif Bias-Variance Tradeoff:
Dalam kasus underfitting, model memiliki bias tinggi dan varians rendah. Modifikasi di atas bertujuan menurunkan bias dengan meningkatkan kapasitas model untuk menangkap pola kompleks, sambil tetap menjaga varians pada level yang dapat diterima. Peningkatan kapasitas model berpotensi meningkatkan varians, tetapi dataset dengan 91 fitur kemungkinan memiliki kompleksitas tinggi yang membutuhkan model dengan kapasitas yang lebih besar.

2. Selain MSE, loss function apa yang mungkin cocok untuk dataset ini? Bandingkan kelebihan dan kekurangannya, serta situasi spesifik di mana alternatif tersebut lebih unggul daripada MSE!

   a. Huber Loss:
   Huber$(y, \hat{y}) = 0.5(y - \hat{y})^2$ jika $|y - \hat{y}| \leq \delta$, dan Huber$(y, \hat{y}) = \delta(|y - \hat{y}| - 0.5\delta)$ jika $|y - \hat{y}| > \delta$

      Kelebihan:
      - Kombinasi dari MSE dan MAE, lebih tahan terhadap outlier
      - Tetap differentiable di semua titik, berbeda dengan MAE

      Kekurangan:
      - Memerlukan hyperparameter tambahan (δ) yang perlu di-tuning
      - Komputasi lebih kompleks dibanding MSE

      Situasi unggul: Ketika dataset mengandung outlier atau nilai ekstrem (terlihat dari beberapa nilai yang sangat tinggi/rendah dalam data).

   b. Mean Absolute Error (MAE):
   $MAE = (1/n) \Sigma|y - \hat{y}|$

      Kelebihan:
      - Lebih robust terhadap outlier dibanding MSE
      - Memberikan bobot yang sama untuk semua error

      Kekurangan:
      - Gradien konstan tidak memberikan penalti lebih besar untuk error besar
      - Tidak differentiable pada titik nol (bisa menyulitkan optimasi)

      Situasi unggul: Ketika distribusi data mengandung outlier signifikan dan kita ingin mengurangi pengaruhnya.

   c. Log-Cosh Loss:
   Log-cosh $= (1/n) \Sigma \log(\cosh(y - \hat{y}))$

      Kelebihan:
      - Mirip MSE untuk error kecil, mirip MAE untuk error besar
      - Differentiable dua kali, memudahkan optimasi

- Lebih tahan terhadap outlier dibanding MSE

Kekurangan:
- Komputasi lebih berat dibanding MSE dan MAE

Situasi unggul: Ketika kita membutuhkan fungsi yang halus seperti MSE tapi dengan ketahanan terhadap outlier.

d. Quantile Loss:
$Q(y, \hat{y}, \tau) = \max(\tau(y - \hat{y}), (1-\tau)(\hat{y} - y))$

Kelebihan:
- Memungkinkan prediksi interval (tidak hanya titik)
- Berguna untuk kasus asimetris dimana underestimation dan overestimation memiliki dampak berbeda

Kekurangan:
- Memerlukan pemilihan kuantil τ
- Interpretasi lebih kompleks

Situasi unggul: Ketika konsekuensi kesalahan prediksi asimetris atau kita ingin mengkuantifikasi ketidakpastian melalui interval prediksi.

Berdasarkan sampel data yang terlihat memiliki variasi nilai yang besar, Huber Loss kemungkinan lebih cocok karena menawarkan keseimbangan yang baik antara sensitivitas MSE terhadap perubahan kecil dan ketahanan MAE terhadap outlier.

3. Jika salah satu fitur memiliki range nilai 0-1, sedangkan fitur lain 100-1000, bagaimana ini memengaruhi pelatihan MLP? Jelaskan mekanisme matematis (e.g., gradien, weight update) yang terdampak!

Ketika satu fitur memiliki range 0-1 dan fitur lain 100-1000, perbedaan skala ini secara signifikan mempengaruhi pelatihan MLP:

Mekanisme Matematis yang Terdampak:
- Gradien dan Weight Update:
  Update bobot dalam MLP mengikuti rumus:
  $w_{new} = w_{old} - learning\_rate * \partial Loss/\partial w$

  Ketika input x memiliki skala berbeda, gradien juga terpengaruh karena:
  $\partial Loss/\partial w = \partial Loss/\partial output * \partial output/\partial w = \partial Loss/\partial output * x$

  Fitur dengan skala 100-1000 akan menghasilkan gradien 100-1000 kali lebih besar dibanding fitur dengan skala 0-1 untuk kontribusi yang sebenarnya sama. Ini menyebabkan:

- Bobot untuk fitur berskala besar berubah jauh lebih cepat
- Bobot untuk fitur berskala kecil berubah sangat lambat

- Landscape Loss Function:
  Perbedaan skala membuat landscape loss function sangat memanjang di satu arah dan terlalu sempit di arah lain, menciptakan bentuk yang disebut "ravine" (jurang sempit).
  Ini membuat optimizer seperti gradient descent mengalami osilasi:
  - Langkah besar dalam dimensi fitur bernilai kecil
  - Langkah terlalu kecil dalam dimensi fitur bernilai besar

- Dampak pada Inisialisasi Bobot:
  Inisialisasi bobot yang umum (seperti Xavier/Glorot) mengasumsikan input terdistribusi dengan skala serupa. Ketika skala sangat berbeda:
  - Fitur bernilai besar dapat menyebabkan aktivasi saturasi (terutama dengan sigmoid/tanh)
  - Initial gradients tidak seimbang antar fitur

- Learning Rate yang Efektif:
  Learning rate efektif per fitur menjadi:
  effective_lr_feature = learning_rate * scale_feature^2

  Ini berarti learning rate terlalu besar untuk fitur berskala besar dan terlalu kecil untuk fitur berskala kecil.

Solusi Teknis:
Normalisasi fitur (seperti StandardScaler atau MinMaxScaler) sangat penting untuk mengatasi masalah ini, membuat semua fitur memiliki kontribusi potensial yang setara terhadap model.

4. Tanpa mengetahui nama fitur, bagaimana Anda mengukur kontribusi relatif setiap fitur terhadap prediksi model? Jelaskan metode teknikal (e.g., permutation importance, weight analysis) dan keterbatasannya!

   Tanpa mengetahui nama fitur, kita dapat menggunakan beberapa metode teknikal untuk mengukur kontribusi relatif. Berikut adalah metode yang dapat digunakan dan keterbatasannya:

   a. Permutation Feature Importance:
      Proses:
      - Train model dengan semua fitur

- Untuk setiap fitur:
  - Acak nilai fitur tersebut di dataset validasi (menghilangkan informasi)
  - Ukur penurunan performa model
  - Semakin besar penurunan, semakin penting fitur tersebut

Keterbatasan:

- Komputasi intensif untuk dataset besar atau model kompleks
- Tidak menangkap interaksi fitur dengan baik
- Sangat bergantung pada kualitas model yang digunakan
- Tidak bisa mendeteksi redundansi antar fitur

b. Weight Analysis (untuk layer pertama):
Proses:

- Hitung rata-rata magnitude (nilai absolut) bobot dari setiap fitur ke hidden layer pertama
- Normalisasi nilai untuk mendapatkan kepentingan relatif

Keterbatasan:

- Hanya valid untuk model linear atau layer pertama MLP
- Tidak memperhitungkan interaksi non-linear di layer berikutnya
- Sensitif terhadap inisialisasi dan regularisasi
- Bias terhadap fitur dengan skala besar (jika tidak dinormalisasi)

c. SHAP (SHapley Additive exPlanations):
Proses:

- Berbasis teori game, menghitung kontribusi setiap fitur terhadap prediksi
- Untuk setiap prediksi, hitung nilai Shapley dengan mempertimbangkan semua koalisi fitur

Keterbatasan:

- Komputasi sangat intensif
- Hasil bisa berbeda tergantung background distribution yang dipilih
- Untuk model kompleks, penjelasan lokal mungkin tidak konsisten secara global

d. Integrated Gradients:
Proses:

- Menghitung gradien dari output terhadap input, diintegrasikan sepanjang jalur dari baseline ke sampel aktual
- Mengukur sensitifitas model terhadap perubahan input

Keterbatasan:

- Memerlukan pemilihan baseline yang tepat
- Hasil bisa bervariasi tergantung jalur integrasi
- Komputasi intensif untuk model besar

e. Drop-Column Importance:
Proses:
- Train model dengan semua fitur
- Untuk setiap fitur, train model baru tanpa fitur tersebut
- Bandingkan performa dengan model lengkap

Keterbatasan:
- Sangat komputasi intensif (membutuhkan pelatihan model berulang)
- Tidak menangkap redundansi dan multikolinearitas
- Pelatihan ulang bisa menghasilkan model dengan sifat berbeda

Untuk dataset seperti yang diberikan dengan 91 fitur, kombinasi Permutation Importance untuk screening awal dan SHAP untuk analisis mendalam terhadap fitur-fitur penting akan memberikan pemahaman yang baik tentang kontribusi fitur.

5. Bagaimana Anda mendesain eksperimen untuk memilih learning rate dan batch size secara optimal? Sertakan analisis tradeoff antara komputasi dan stabilitas pelatihan!

Desain eksperimen untuk memilih learning rate dan batch size yang optimal:

- Grid Search dengan Pembagian Log-scale:
  - Learning Rate:
    - Rentang awal: 10^-5 hingga 10^-1 dengan 5 titik pada skala logaritmik
    - Rentang kedua: Fokus pada area yang menjanjikan dengan grid yang lebih halus
  - Batch Size:
    - Rentang: 2^5 (32) hingga 2^10 (1024) dengan peningkatan kelipatan 2
    - Pertimbangkan batasan memori dan ukuran dataset

- Learning Rate Range Test:
  Proses:
  - Mulai dengan learning rate sangat kecil dan tingkatkan secara eksponensial selama beberapa iterasi
  - Plot loss terhadap learning rate
  - Pilih learning rate pada titik dimana loss menurun paling cepat
  - Untuk adaptive optimizer (Adam), gunakan 1/10 dari nilai tersebut

- Batch Size Sweep dengan Fixed Learning Rate:
  Proses:
    - Gunakan learning rate terbaik dari langkah sebelumnya
    - Uji berbagai batch size, catat:
      - Waktu pelatihan per epoch
      - Konvergensi (jumlah epoch untuk mencapai target)
      - Performa validasi final

- Learning Rate Schedulers:
  Proses:
    - Uji scheduler berbeda (step, cosine, reduce on plateau)
    - Bandingkan dengan learning rate statis

Analisis Tradeoff:

- Batch Size:
    - Batch Besar (512-1024+):
      - Komputasi: Lebih efisien, lebih baik memanfaatkan paralelisme hardware
      - Stabilitas: Gradien lebih stabil, noise lebih rendah
      - Tradeoff: Konvergensi lambat, dapat terjebak di minima lokal, memori GPU besar
    - Batch Kecil (32-128):
      - Komputasi: Kurang efisien, lebih banyak update per epoch
      - Stabilitas: Gradien lebih noise, dapat melarikan diri dari minima lokal yang buruk
      - Tradeoff: Pelatihan lebih lama, tetapi seringkali generalisasi lebih baik

- Learning Rate:
    - Learning Rate Besar:
      - Komputasi: Konvergensi potensial lebih cepat
      - Stabilitas: Risiko divergensi, osilasi sekitar minimum
      - Tradeoff: Bisa menghemat waktu komputasi jika stabil
    - Learning Rate Kecil:
      - Komputasi: Konvergensi lambat, membutuhkan epoch lebih banyak
      - Stabilitas: Lebih stabil, langkah yang lebih konservatif
      - Tradeoff: Komputasi lebih lama, potensi terjebak di minima lokal