Bab 09 — Neural Networks dari Nol

Cara membaca bab ini

Bab 9 mengikuti gaya Bab 7 dan Bab 8: berbasis subbab. Setiap subbab menggabungkan cerita, intuisi, persamaan, cara membaca persamaan, contoh hitung manual, cara membaca visual, dan tes cepat. Target bab ini bukan sekadar membuat pembaca tahu istilah “neural network”, tetapi memahami mengapa neural network bekerja, kapan ia cocok, bagaimana menghitung forward pass, bagaimana error mengalir balik, dan mengapa jaringan berlapis dapat menangani pola non-linear yang sulit untuk model linear sederhana.

Bab ini menjembatani Bab 7 tentang supervised learning, Bab 8 tentang representasi, dan Bab 10 tentang deep learning. Jika Bab 7 membahas banyak algoritma, Bab 9 memperdalam satu keluarga besar: neural network.

Subbab 1 — Apa itu neural network?

Inti subbab: neural network adalah fungsi berparameter yang menyusun banyak transformasi sederhana menjadi model yang fleksibel.

Secara praktis, neural network menerima input x, menghitung kombinasi bobot, melewati fungsi aktivasi, lalu menghasilkan output. Bentuk paling sederhana:

input → neuron → output

Satu neuron menghitung:

$$z = w \cdot x + b$$
$$a = \text{activation}(z)$$

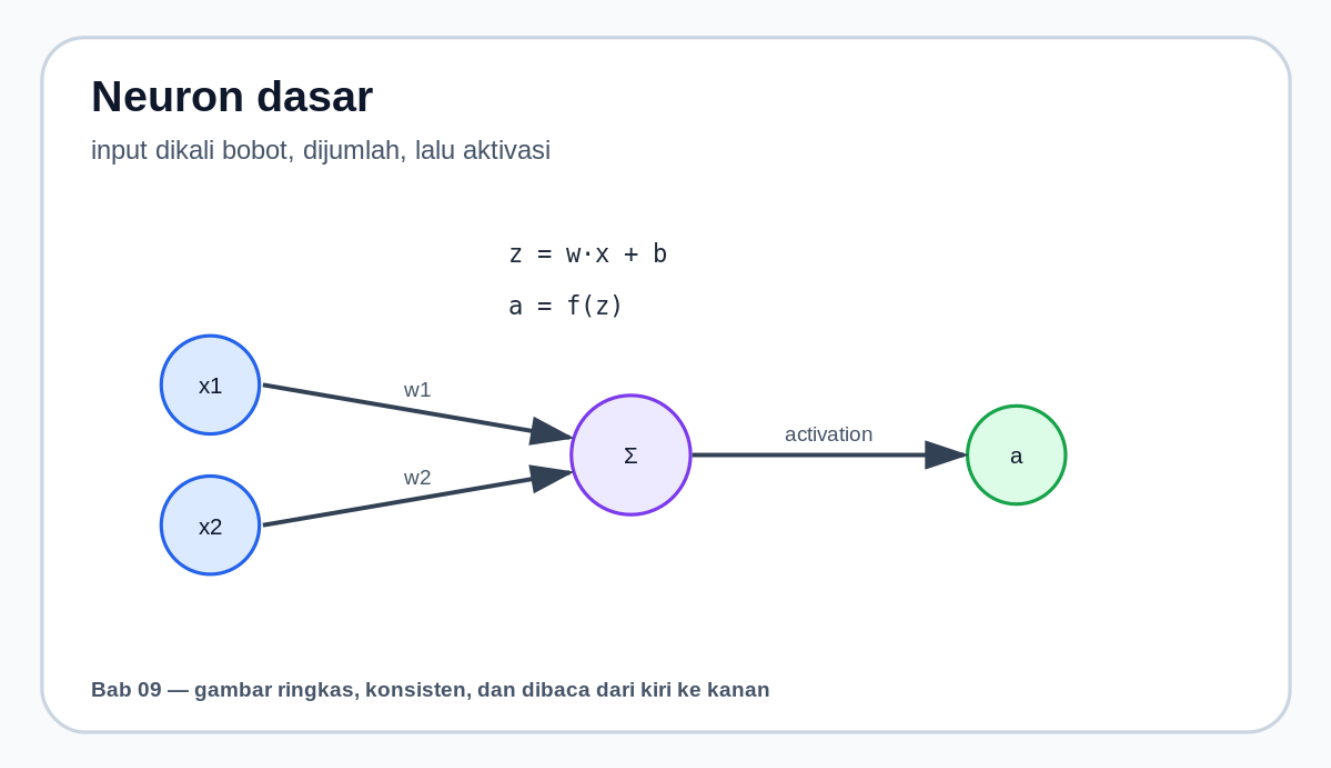
Cara membaca persamaan: $w \cdot x$ dibaca “dot product bobot dan input”. b adalah bias. z adalah skor mentah sebelum aktivasi. a adalah output setelah aktivasi.

Contoh: $x=[2,3]$, $w=[0,5,-1]$, $b=1$.

$$z = 0{,}5 \times 2 + (-1) \times 3 + 1 = 1 - 3 + 1 = -1$$

Jika aktivasi ReLU:

$$a = \max(0, -1) = 0$$



Neuron dasar

Tes cepat subbab 1

1. Apa perbedaan `z` dan `a`?

2. Hitung `z` untuk `x=[1,4]`, `w=[2,-1]`, `b=3`.

3. Mengapa bias `b` diperlukan?

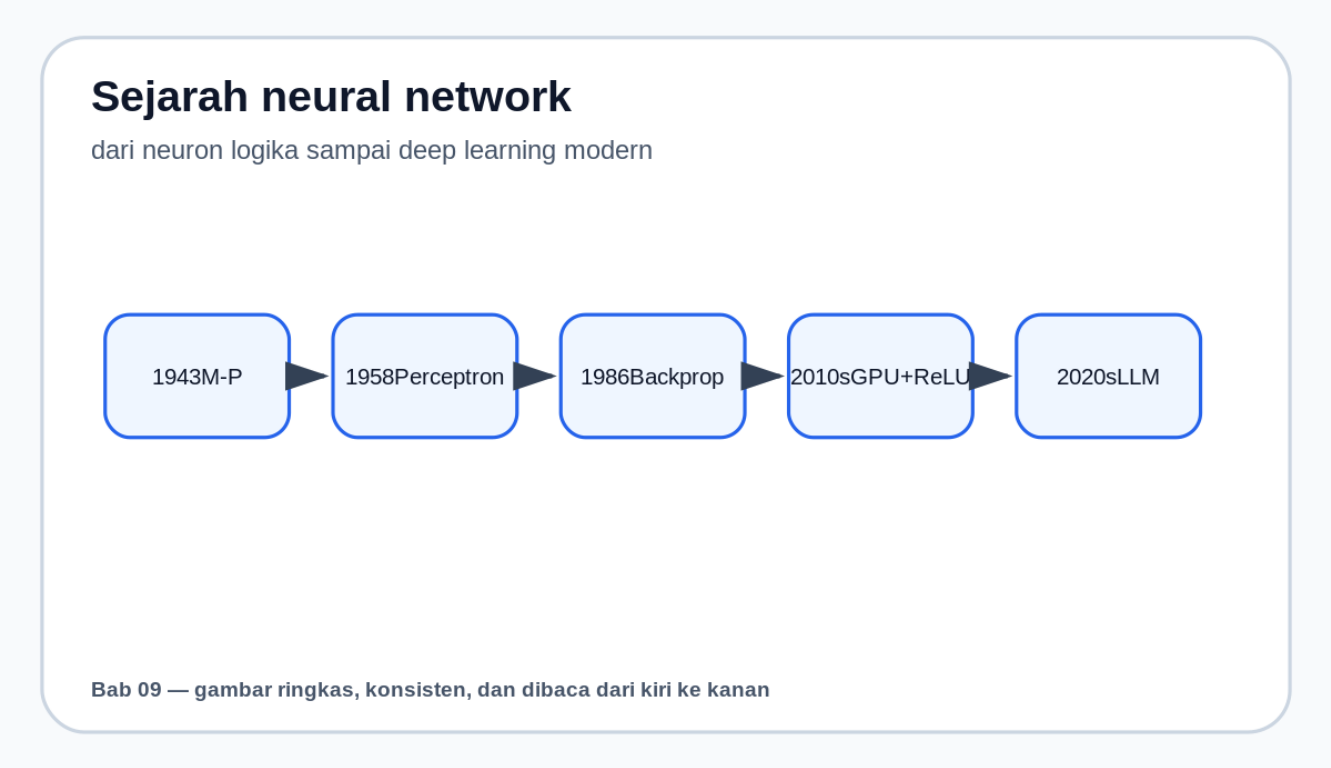
Subbab 2 — Sejarah singkat neural networks

Inti subbab: neural network modern lahir dari gabungan ide biologi, matematika, statistik, komputasi, dan data besar.

Timeline ringkas:

1943: McCulloch-Pitts neuron, model logika neuron sederhana
1958: Rosenblatt perceptron, model linear yang bisa belajar bobot
1969: Minsky & Papert menunjukkan keterbatasan perceptron, termasuk XOR
1980-an: backpropagation dipopulerkan untuk melatih multi-layer network
1990-an: neural network bersaing dengan SVM dan metode statistik lain
2010-an: GPU, data besar, ReLU, dropout, batch norm memicu deep learning modern
2020-an: neural network menjadi fondasi LLM, diffusion model, multimodal AI

Pelajaran sejarah penting: neural network pernah naik, turun, lalu naik lagi. Bukan karena idenya tiba-tiba berubah sepenuhnya, tetapi karena komputasi, data, teknik training, dan arsitektur membaik.



Sejarah neural network

Catatan kritis: neural network bukan “otak manusia digital”. Ia terinspirasi secara longgar dari neuron biologis, tetapi secara teknik lebih tepat dibaca sebagai sistem fungsi komposisi yang dilatih dengan optimisasi.

Tes cepat subbab 2

1. Mengapa perceptron penting dalam sejarah NN?

2. Apa keterbatasan perceptron klasik?

3. Mengapa era GPU penting untuk deep learning?

Subbab 3 — Neuron sebagai model linear kecil

Inti subbab: sebelum aktivasi, neuron adalah model linear.

Bagian linear neuron:

$$z = w_1x_1 + w_2x_2 + \dots + w_d x_d + b$$

Cara membaca: setiap fitur x_j dikalikan bobot w_j, semua dijumlahkan, lalu ditambah bias. Bobot positif menaikkan skor jika fitur naik. Bobot negatif menurunkan skor jika fitur naik.

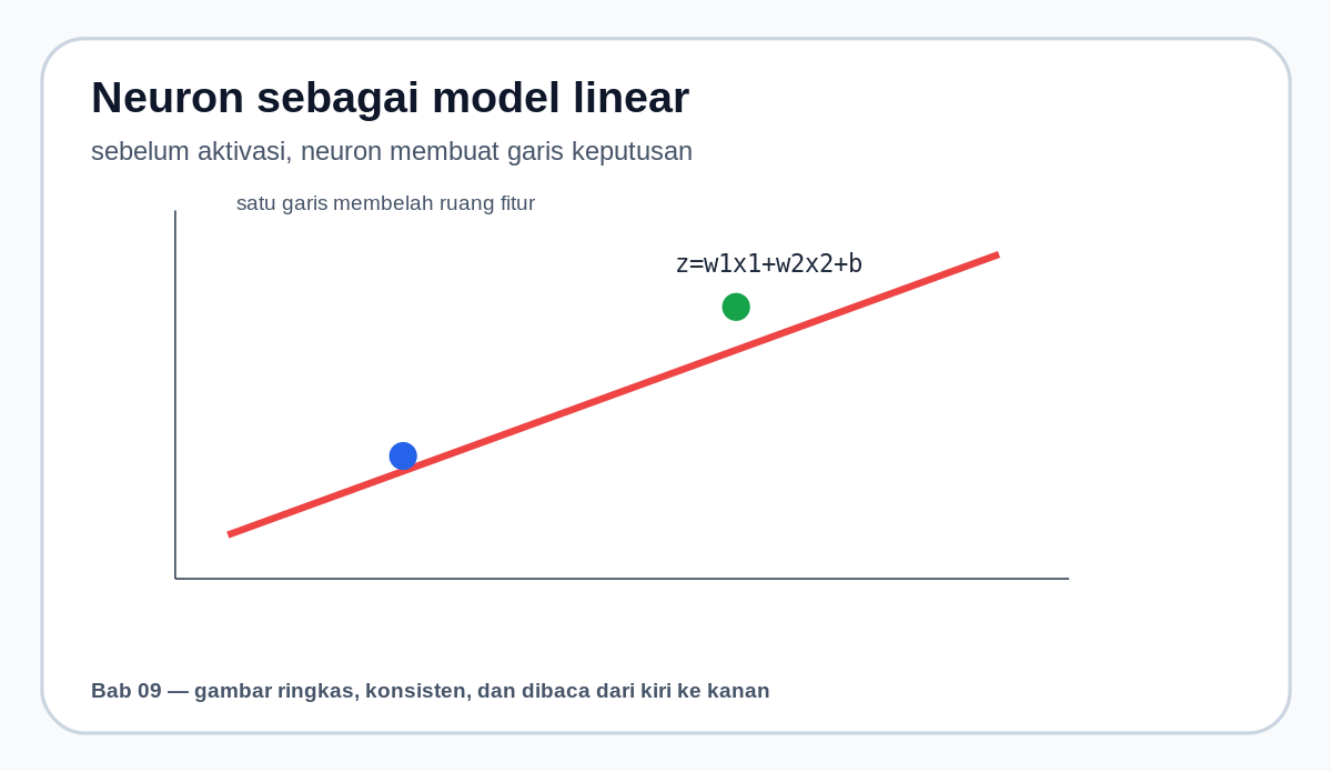
Contoh prediksi minat beli:

x_1 = jumlah kunjungan = 4
x_2 = diskon dilihat = 1
w_1 = 0,6
w_2 = 1,2
b = -2

Hitung:

$$z = 0{,}6\times4 + 1{,}2\times1 - 2$$
$$= 2{,}4 + 1{,}2 - 2$$
$$= 1{,}6$$

Jika z positif, neuron cenderung “aktif”. Jika z negatif, neuron cenderung “tidak aktif”, tergantung aktivasi.



Neuron linear

Tes cepat subbab 3

1. Jika w positif, apa efek fitur yang naik?

2. Hitung z untuk x■=3, x■=2, w■=1, w■=-0,5, b=0.

3. Mengapa bagian linear saja belum cukup untuk banyak pola kompleks?

Subbab 4 — Aktivasi: membuat jaringan tidak hanya garis lurus

Inti subbab: fungsi aktivasi memberi neural network kemampuan non-linear.

Tanpa aktivasi non-linear, tumpukan layer linear tetap setara dengan satu transformasi linear. Misalnya:

$$h = W_1x$$
$$\hat{y} = W_2h = W_2(W_1x) = (W_2W_1)x$$

Cara membaca: dua perkalian matriks linear dapat digabung menjadi satu matriks baru. Jadi menumpuk layer tanpa aktivasi tidak menambah kekuatan bentuk fungsi.

Aktivasi umum:

$$\text{sigmoid}(z) = 1 / (1 + e^{-z})$$
$$\tanh(z) = (e^{z} - e^{-z})/(e^{z} + e^{-z})$$
$$\text{ReLU}(z) = \max(0, z)$$

Cara membaca sigmoid: output selalu antara 0 dan 1, cocok untuk probabilitas biner. Jika z besar positif, sigmoid mendekati 1. Jika z besar negatif, mendekati 0.

Contoh:

$$\text{sigmoid}(0) = 1/(1+e^{0}) = 1/(1+1)=0,5$$
$$\text{ReLU}(-3)=0$$
$$\text{ReLU}(4)=4$$

Fungsi aktivasi

Tes cepat subbab 4

1. Mengapa aktivasi non-linear penting?

2. Hitung ReLU untuk `[-2,0,5]`.

3. Berapa nilai sigmoid(0)?

Subbab 5 — Perceptron: garis keputusan pertama

Inti subbab: perceptron adalah neuron linear dengan keputusan threshold.

Perceptron menghitung:

$$z = w \cdot x + b$$
$$\hat{y} = 1 \text{ jika } z \geq 0, \text{ selain itu } 0$$

Cara membaca: jika skor melewati ambang nol, prediksi positif. Jika tidak, prediksi negatif.

Contoh logika AND:

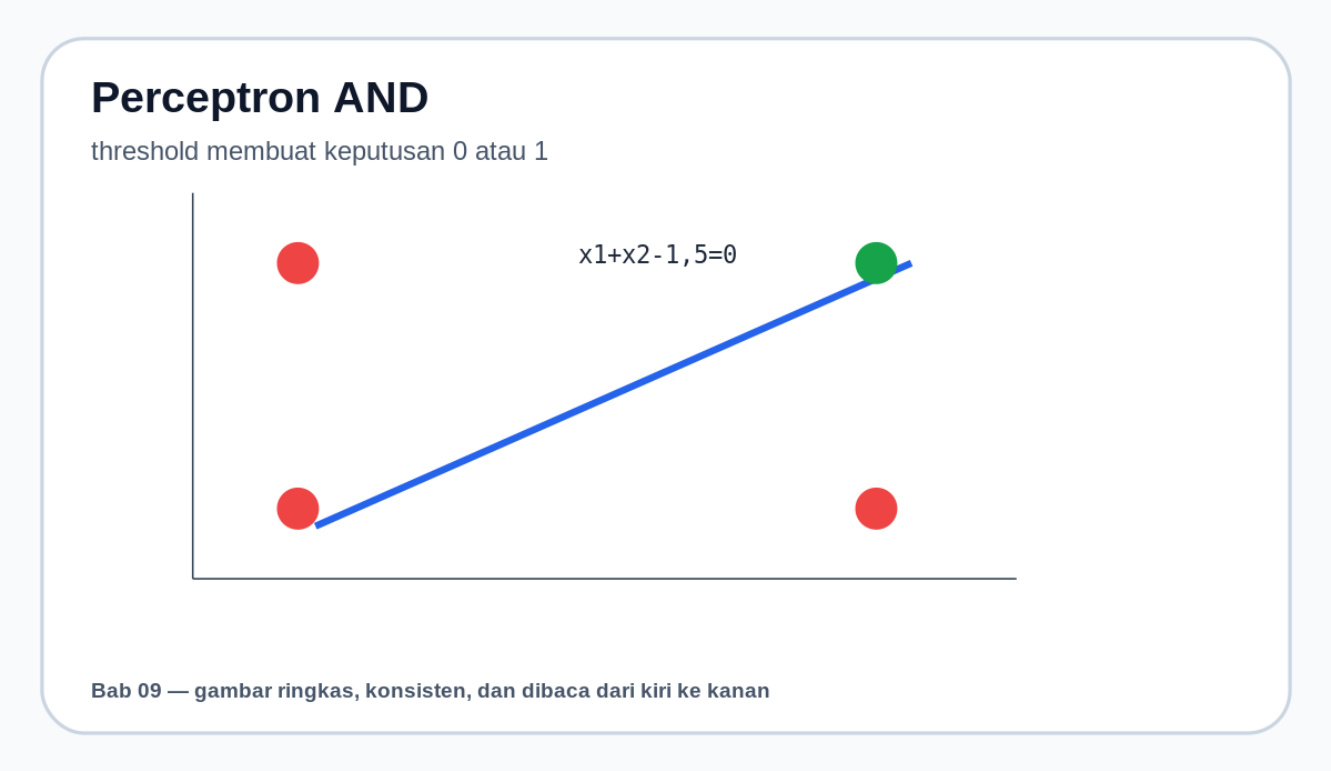
x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

Pilih `w=[1,1]`, `b=-1,5`.

Hitung:

[0,0]: z=0+0-1,5=-1,5 → 0
[0,1]: z=0+1-1,5=-0,5 → 0
[1,0]: z=1+0-1,5=-0,5 → 0
[1,1]: z=1+1-1,5=0,5 → 1

Perceptron berhasil untuk AND karena kelas bisa dipisahkan oleh satu garis.



Perceptron AND

Tes cepat subbab 5

1. Apa fungsi threshold pada perceptron?
2. Uji `w=[1,1]`, `b=-0,5` untuk logika OR.
3. Apa arti “dipisahkan oleh satu garis”?

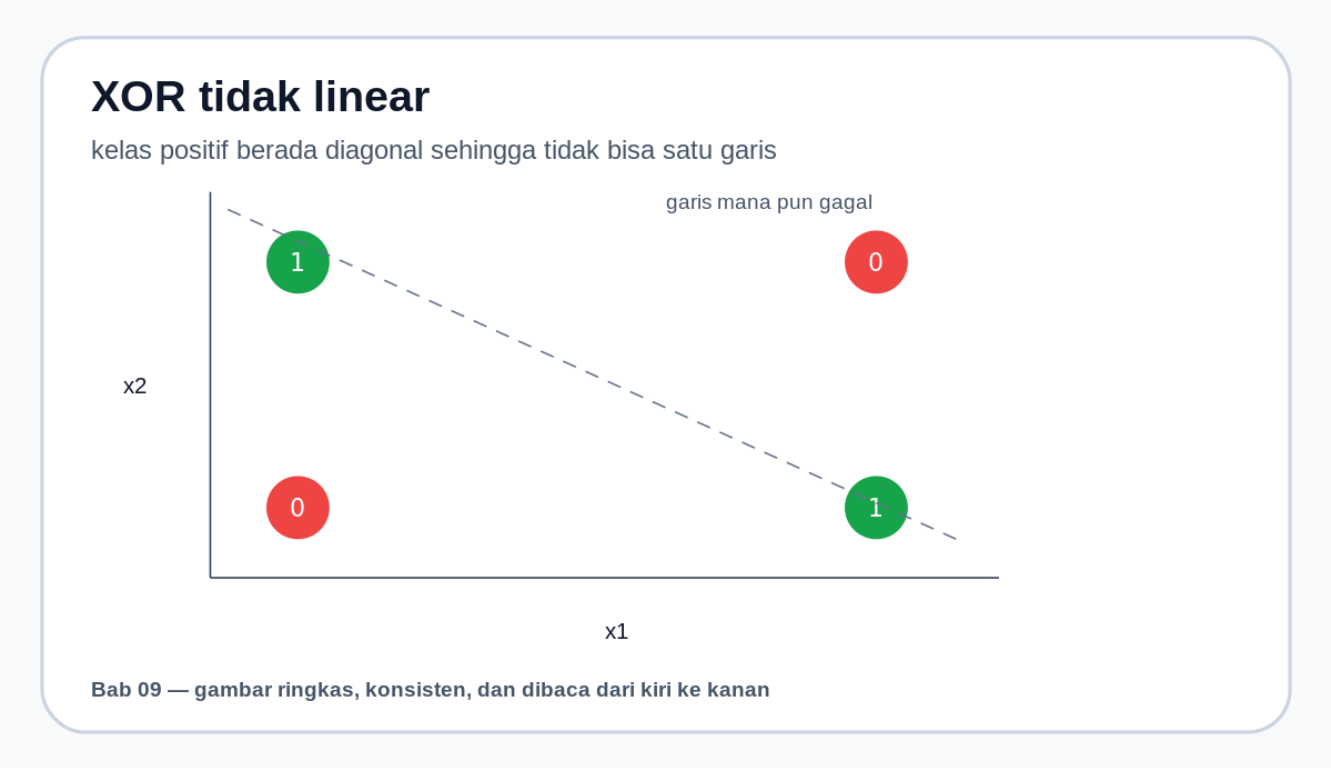
Subbab 6 — Mengapa model linear gagal pada XOR

Inti subbab: XOR adalah contoh pola non-linear yang tidak bisa dipisahkan oleh satu garis lurus.

XOR bernilai 1 jika input berbeda:

x_1	x_2	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Titik positif berada di `(0,1)` dan `(1,0)`. Titik negatif berada di `(0,0)` dan `(1,1)`. Tidak ada satu garis lurus yang bisa menaruh dua positif di satu sisi dan dua negatif di sisi lain.



XOR non-linear

Pembuktian intuisi dengan pertidaksamaan

Misalkan ada perceptron dengan:

$$z = w_1x_1 + w_2x_2 + b$$

Agar XOR benar:

(0,0) negatif: $b < 0$
(1,0) positif: $w_1 + b \geq 0$
(0,1) positif: $w_2 + b \geq 0$
(1,1) negatif: $w_1 + w_2 + b < 0$

Dari dua syarat positif:

$w_1 \geq -b$
$w_2 \geq -b$

Jumlahkan:

$w_1 + w_2 \geq -2b$

Karena `b < 0`, maka `-2b` positif. Untuk titik `(1,1)` negatif dibutuhkan:

$w_1 + w_2 + b < 0$
$w_1 + w_2 < -b$

Tetapi sebelumnya `w■+w■ ≥ -2b`. Jika `b<0`, maka `-2b > -b`. Kontradiksi. Jadi satu perceptron tidak bisa menyelesaikan XOR.

Tes cepat subbab 6

1. Mengapa XOR tidak linearly separable?

2. Titik mana yang positif pada XOR?

3. Mengapa bukti pertidaksamaan menunjukkan kontradiksi?

Subbab 7 — Hidden layer: membuat fitur baru dari data lama

Inti subbab: hidden layer dapat membuat representasi baru sehingga pola non-linear menjadi lebih mudah dipisahkan.

Ide utama neural network berlapis:

> input x → hidden representation h → output ŷ

Hidden neuron tidak hanya menerima fitur asli. Ia membuat fitur baru. Untuk XOR, dua hidden neuron bisa mewakili kondisi:

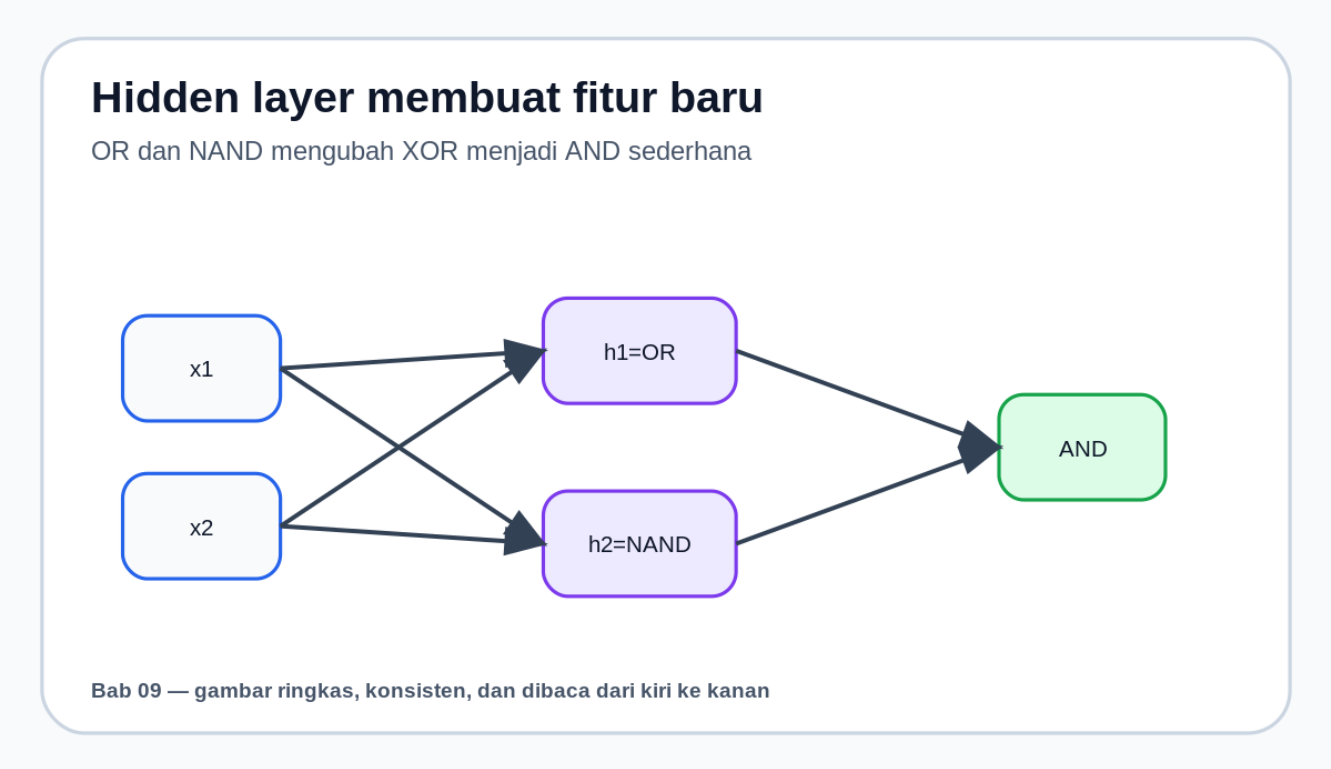
> $h_1 = OR(x_1, x_2)$
> $h_2 = NAND(x_1, x_2)$
> $output = AND(h_1, h_2)$

Mengapa ini menyelesaikan XOR?

> XOR = OR AND NAND

Tabel:

x_1	x_2	OR	NAND	OR AND NAND
0	0	0	1	0
0	1	1	1	1
1	0	1	1	1
1	1	1	0	0



Hidden layer XOR

Makna mendalam: hidden layer melakukan representation learning sederhana. Ia mengubah masalah yang tidak bisa dipisahkan di ruang input menjadi masalah yang bisa dipisahkan di ruang fitur baru.

Tes cepat subbab 7

1. Apa peran hidden layer?

2. Mengapa OR dan NAND bisa membantu XOR?

3. Apa hubungan hidden layer dengan representation learning Bab 8?

Subbab 8 — Contoh hitung lengkap neural network XOR

Inti subbab: neural network kecil dapat menghitung XOR secara manual.

Gunakan threshold activation:

$\text{step}(z)=1$ jika $z \geq 0$, selain itu 0

Hidden neuron 1 untuk OR:

$h_1 = \text{step}(1 \times x_1 + 1 \times x_2 - 0{,}5)$

Hidden neuron 2 untuk NAND:

$h_2 = \text{step}(-1 \times x_1 + -1 \times x_2 + 1{,}5)$

Output untuk AND dari h■ dan h■:

$\hat{y} = \text{step}(1 \times h_1 + 1 \times h_2 - 1{,}5)$

Hitung semua kasus

Kasus `(0,0)`:

$h_1 = \text{step}(0+0-0{,}5)=\text{step}(-0{,}5)=0$
$h_2 = \text{step}(0+0+1{,}5)=\text{step}(1{,}5)=1$
$\hat{y} = \text{step}(0+1-1{,}5)=\text{step}(-0{,}5)=0$

Kasus `(0,1)`:

$h_1 = \text{step}(0+1-0{,}5)=1$
$h_2 = \text{step}(0-1+1{,}5)=1$
$\hat{y} = \text{step}(1+1-1{,}5)=1$

Kasus `(1,0)`:

$h_1 = \text{step}(1+0-0{,}5)=1$
$h_2 = \text{step}(-1+0+1{,}5)=1$
$\hat{y} = \text{step}(1+1-1{,}5)=1$

Kasus `(1,1)`:

$h_1 = \text{step}(1+1-0{,}5)=1$
$h_2 = \text{step}(-1-1+1{,}5)=0$
$\hat{y} = \text{step}(1+0-1{,}5)=0$

Hasilnya `[0,1,1,0]`, tepat XOR.

Perhitungan XOR

Tes cepat subbab 8

1. Hitung h■, h■, dan output untuk input `(1,0)`.
2. Mengapa output memakai AND?
3. Apa yang berubah dari ruang input ke ruang hidden?

Subbab 9 — Matriks dan bentuk data dalam neural network

Inti subbab: neural network modern menghitung banyak neuron sekaligus dengan matriks.

Untuk satu layer:

```
Z = XW + b
A = activation(Z)
```

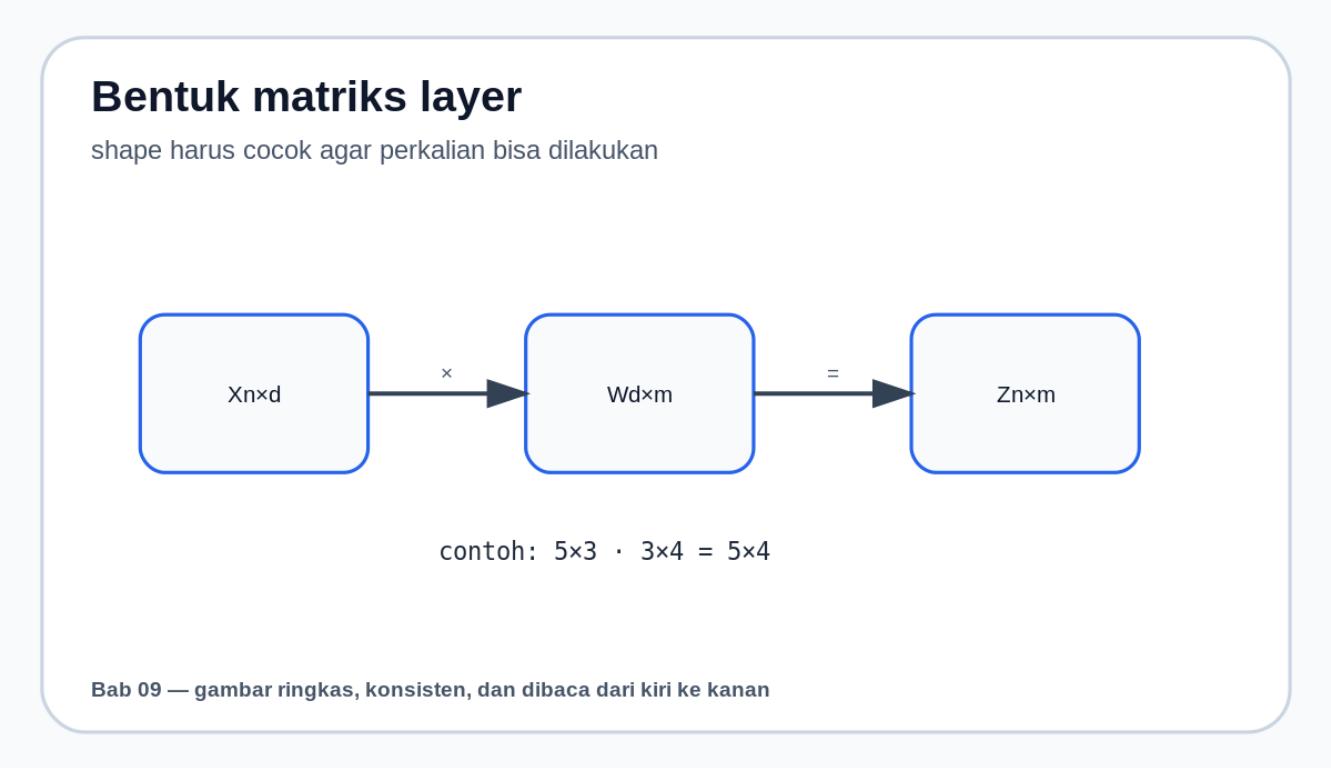
Cara membaca: `X` adalah matriks data. `W` adalah matriks bobot. `b` adalah bias yang ditambahkan ke setiap baris. `Z` skor mentah. `A` aktivasi.

Jika `X` berukuran `n×d` dan `W` berukuran `d×m`, maka:

```
Z berukuran n×m
```

Contoh: 5 data, 3 fitur, 4 neuron hidden.

```
X: 5×3
W: 3×4
Z: 5×4
```



Matriks NN

Kesalahan umum: shape mismatch. Jika jumlah kolom `X` tidak sama dengan jumlah baris `W`, perkalian matriks tidak bisa dilakukan.

Tes cepat subbab 9

1. Jika `X` 10×6 dan `W` 6×8, berapa ukuran `Z`?

2. Apa arti jumlah kolom `W`?

3. Mengapa shape penting dalam NN?

Subbab 10 — Forward pass satu layer dengan angka

Inti subbab: forward pass adalah proses menghitung output dari input ke depan.

Contoh input:

```
x = [2, 3]
```

Dua hidden neuron:

```
W = [[ 1, -1],
     [ 2,  1]]
b = [0, 1]
```

Hitung neuron 1:

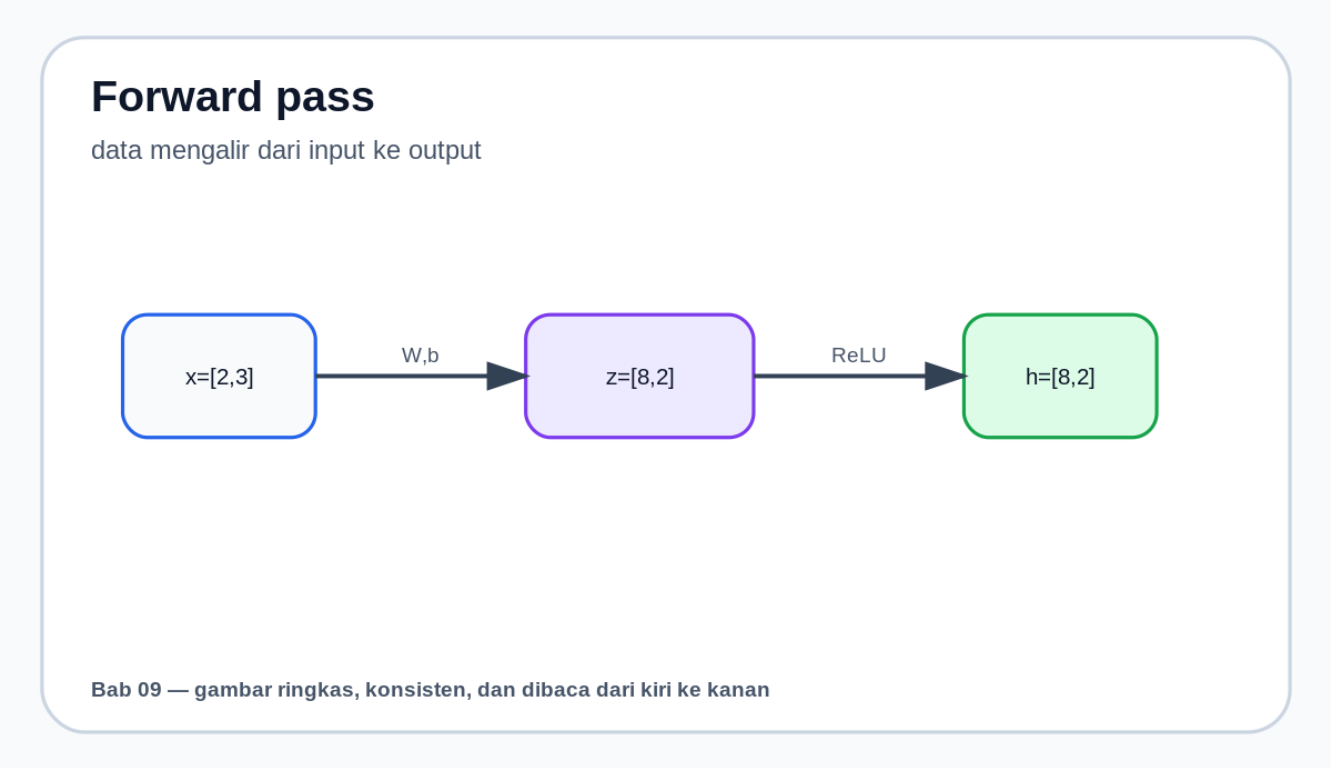
$z_1 = 2\times1 + 3\times2 + 0 = 8$

Hitung neuron 2:

$z_2 = 2\times(-1) + 3\times1 + 1 = 2$

Jika aktivasi ReLU:

$h = [\text{ReLU}(8), \text{ReLU}(2)] = [8, 2]$



Forward pass

Cara membaca visual: panah dari input ke neuron membawa bobot. Setiap neuron menjumlahkan kontribusi dari semua input.

Tes cepat subbab 10

1. Hitung z■ jika `x=[1,2]`, `w=[3,4]`, `b=-1`.

2. Apa itu forward pass?

3. Mengapa ReLU tidak mengubah nilai positif?

Subbab 11 — Loss function: mengukur seberapa salah model

Inti subbab: loss mengubah kesalahan model menjadi angka yang bisa diminimalkan.

Untuk regresi, loss umum:

$$MSE = (1/n)\ \Sigma(\hat{y}_i - y_i)^2$$

Cara membaca: setiap prediksi dikurangi target, dikuadratkan, dijumlahkan, lalu dirata-ratakan.

Contoh:

$$y = [3, 5]$$
$$\hat{y} = [2, 7]$$

Error:

$$[2-3, 7-5] = [-1, 2]$$

Kuadrat error:

$$[1, 4]$$

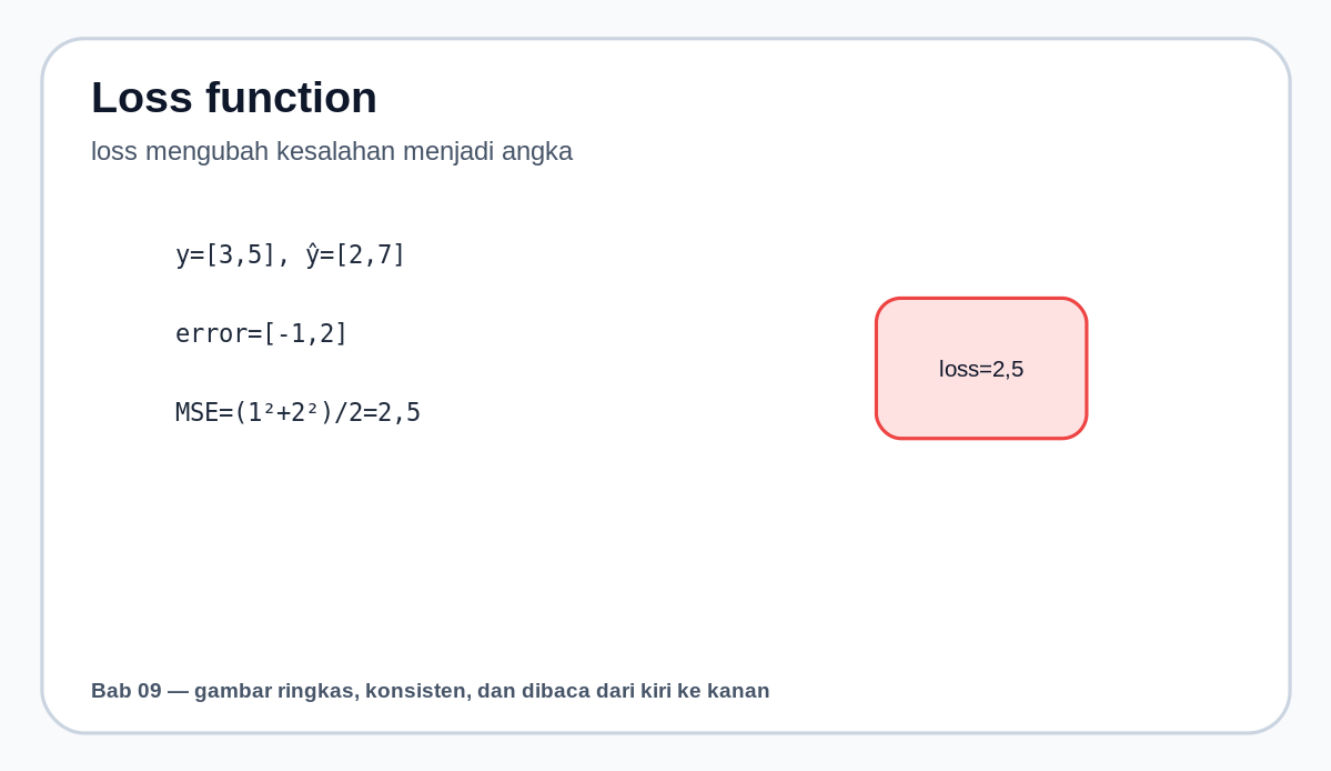
MSE:

$$(1+4)/2 = 2,5$$

Untuk klasifikasi biner, binary cross-entropy:

$$L = -[y \log(p) + (1-y) \log(1-p)]$$

Jika $y=1$, loss menjadi $-\log(p)$. Jika model memberi $p=0,9$, loss kecil. Jika $p=0,1$, loss besar.



Loss function

Tes cepat subbab 11

1. Hitung MSE untuk $y=[1,2]$, ■$=[2,2]$.
2. Mengapa error dikuadratkan dalam MSE?
3. Apa arti p pada binary cross-entropy?

Subbab 12 — Gradient descent: cara model memperbaiki bobot

Inti subbab: gradient descent mengubah bobot ke arah yang menurunkan loss.

Update umum:

$$\theta_{baru} = \theta_{lama} - \eta \, \partial L/\partial \theta$$

Cara membaca: parameter lama dikurangi learning rate η dikali turunan loss terhadap parameter. Jika turunan positif, parameter turun. Jika turunan negatif, parameter naik.

Contoh:

$w = 2$
$\eta = 0,1$
$\partial L/\partial w = 3$

Update:

$$w_{baru} = 2 - 0,1 \times 3 = 1,7$$

Jika gradient -4:

$$w_{baru} = 2 - 0,1 \times (-4) = 2,4$$

Gradient descent

Tes cepat subbab 12

1. Hitung update jika `w=5`, `η=0,01`, gradient `20`.

2. Apa efek learning rate terlalu besar?

3. Mengapa gradient negatif membuat bobot naik?

Subbab 13 — Turunan neuron linear untuk MSE

Inti subbab: turunan menunjukkan bagaimana setiap bobot memengaruhi loss.

Model satu neuron linear:

$$\hat{y} = wx + b$$
$$L = (\hat{y} - y)^2$$

Turunan terhadap `w`:

$$\partial L/\partial w = 2(\hat{y}-y)x$$

Turunan terhadap `b`:

$$\partial L/\partial b = 2(\hat{y}-y)$$

Cara membaca: error (■-y) dikalikan 2. Untuk bobot, dikalikan lagi dengan input x karena bobot hanya berpengaruh melalui wx.

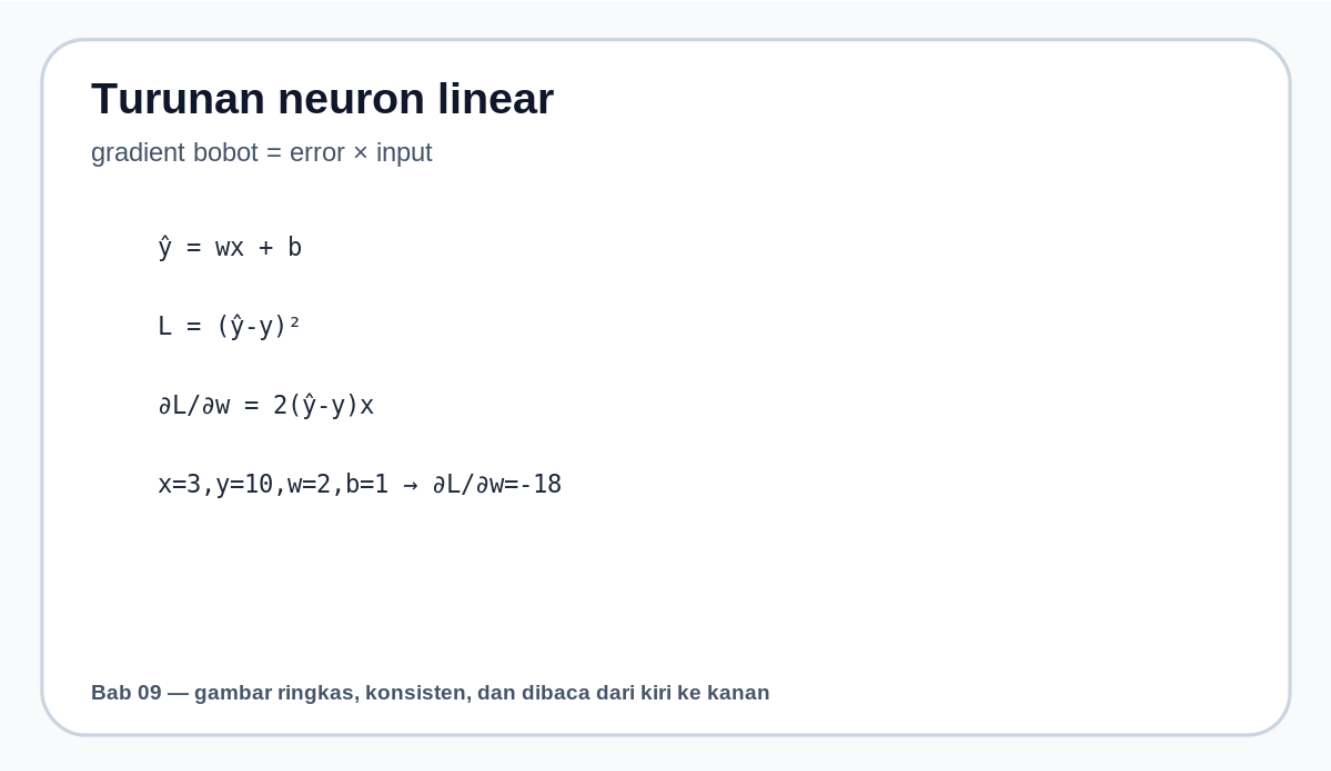
Contoh:

$x=3, y=10, w=2, b=1$
$\hat{y} = 2\times3 + 1 = 7$
error = $\hat{y}-y = -3$
$\partial L/\partial w = 2\times(-3)\times3 = -18$
$\partial L/\partial b = 2\times(-3) = -6$

Jika `η=0,01`:

w_baru = 2 - 0,01×(-18)=2,18
b_baru = 1 - 0,01×(-6)=1,06

Bobot naik karena prediksi terlalu kecil.



Turunan neuron

Tes cepat subbab 13

1. Hitung gradient untuk x=2, y=5, w=1, b=0.

2. Mengapa gradient dikalikan input x?

3. Jika prediksi terlalu kecil, arah update biasanya bagaimana?

Subbab 14 — Backpropagation: error mengalir balik dengan aturan rantai

Inti subbab: backpropagation adalah cara efisien menghitung gradient semua bobot dengan aturan rantai, bukan sekadar istilah “mundur”.

Forward sederhana satu neuron:

$$z = wx + b$$
$$a = \sigma(z)$$
$$L = (a-y)^2$$

Kita ingin tahu: jika w diubah sedikit, loss L berubah seberapa besar? Karena w tidak langsung menyentuh L, pengaruhnya melewati rantai:

$$w \to z \to a \to L$$

Maka:

$$\partial L/\partial w = \partial L/\partial a \times \partial a/\partial z \times \partial z/\partial w$$

Cara membaca persamaan: “turunan loss terhadap bobot sama dengan pengaruh loss terhadap aktivasi, dikali pengaruh aktivasi terhadap skor, dikali pengaruh skor terhadap bobot.” Ini seperti menelusuri jejak sebab-akibat dari output kembali ke bobot.

Turunan satu per satu

Loss:

$L = (a-y)^2$
$\partial L/\partial a = 2(a-y)$

Sigmoid:

$\sigma(z)=1/(1+e^{(-z)})$
$\sigma'(z)=\sigma(z)(1-\sigma(z)) = a(1-a)$

Skor linear:

$z = wx+b$
$\partial z/\partial w = x$
$\partial z/\partial b = 1$

Gabungkan:

$\partial L/\partial w = 2(a-y) \times a(1-a) \times x$
$\partial L/\partial b = 2(a-y) \times a(1-a)$

Contoh hitung lengkap

Diberikan:

$x = 2$
$y = 1$
$w = 0$
$b = 0$

Forward:

$z = 0\times2 + 0 = 0$
$a = \text{sigmoid}(0) = 0,5$
$L = (0,5 - 1)^2 = 0,25$

Backward:

$\partial L/\partial a = 2(0,5-1) = -1$
$\partial a/\partial z = 0,5(1-0,5) = 0,25$
$\partial z/\partial w = 2$
$\partial z/\partial b = 1$

Gradient:

$\partial L/\partial w = -1 \times 0,25 \times 2 = -0,5$
$\partial L/\partial b = -1 \times 0,25 \times 1 = -0,25$

Jika learning rate $\eta=0,1$:

$w\_baru = 0 - 0,1\times(-0,5) = 0,05$
$b\_baru = 0 - 0,1\times(-0,25) = 0,025$

Bobot dan bias naik karena prediksi $0,5$ masih terlalu kecil dibanding target 1.

Bentuk delta agar mudah diingat

Kita sering mendefinisikan:

$\delta = \partial L/\partial z$

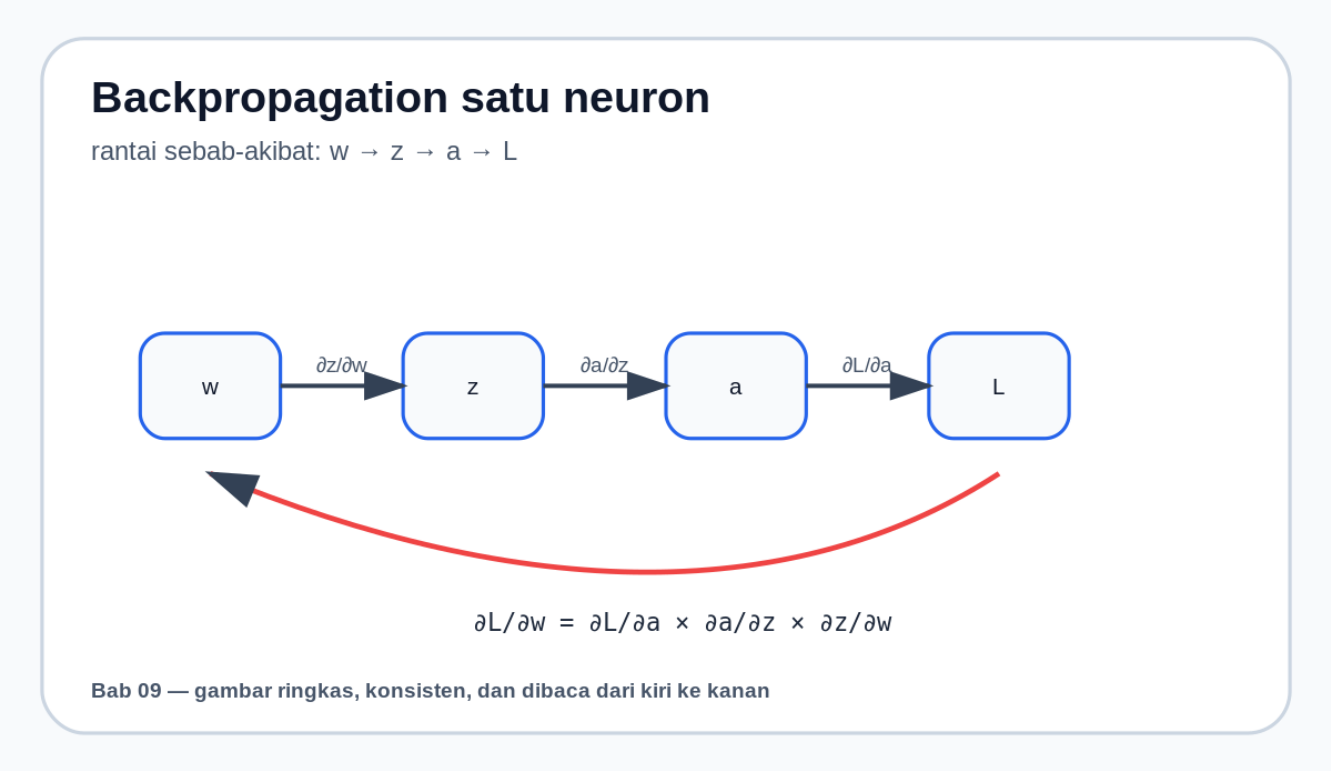
Untuk contoh sigmoid + MSE:

$\delta = 2(a-y)a(1-a)$

Lalu:

$\partial L/\partial w = \delta x$
$\partial L/\partial b = \delta$

Cara membaca: delta adalah “sinyal salah” pada neuron sebelum aktivasi. Gradient bobot adalah sinyal salah dikali input yang masuk ke bobot itu.



Backpropagation

Tes cepat subbab 14

1. Apa rantai sebab-akibat dari `w` ke `L`?

2. Hitung $\sigma'(0)$.

3. Jika $\delta=-0,2$ dan $x=5$, berapa $\partial L/\partial w$?

Subbab 15 — Backpropagation dua layer: dari output ke hidden layer

Inti subbab: pada jaringan berlapis, gradient output layer dihitung dulu, lalu dipakai untuk menghitung gradient hidden layer.

Gunakan jaringan kecil:

$$h = \mathrm{ReLU}(W_1 x + b_1)$$
$$\hat{y} = W_2 h + b_2$$
$$L = (\hat{y}-y)^2$$

Agar mudah, pakai 1 input, 2 hidden neuron, 1 output.

Parameter:

$x = 2,\ y = 5$
$w_{11} = 1,\ b_{11} = 0$ → neuron hidden 1
$w_{12} = -1,\ b_{12} = 3$ → neuron hidden 2
$w_{21} = 2,\ w_{22} = -1,\ b_2 = 0$

Forward pass

Hidden 1:

$z_1 = 1\times 2 + 0 = 2$
$h_1 = \mathrm{ReLU}(2) = 2$

Hidden 2:

$z_2 = -1\times2 + 3 = 1$
$h_2 = \text{ReLU}(1) = 1$

Output:

$\hat{y} = 2\times h_1 + (-1)\times h_2 + 0$
$= 2\times2 - 1\times1$
$= 3$

Loss:

$L = (\hat{y}-y)^2 = (3-5)^2 = 4$

Backward output layer

Turunan loss terhadap output:

$\partial L/\partial \hat{y} = 2(\hat{y}-y) = 2(3-5) = -4$

Gradient bobot output:

$\partial L/\partial w_{21} = \partial L/\partial \hat{y} \times \partial \hat{y}/\partial w_{21} = -4 \times h_1 = -4\times2 = -8$
$\partial L/\partial w_{22} = -4 \times h_2 = -4\times1 = -4$
$\partial L/\partial b_2 = -4$

Cara membaca: bobot output yang terhubung ke hidden aktif besar mendapat gradient besar karena kontribusinya besar terhadap prediksi.

Backward ke hidden layer

Sinyal error ke hidden 1:

$\partial L/\partial h_1 = \partial L/\partial \hat{y} \times \partial \hat{y}/\partial h_1 = -4 \times w_{21} = -4\times2 = -8$

Sinyal error ke hidden 2:

$\partial L/\partial h_2 = -4 \times w_{22} = -4\times(-1) = 4$

Karena hidden memakai ReLU:

$\text{ReLU}'(z)=1$ jika $z>0$, dan 0 jika $z\leq0$

Di sini z■=2 dan z■=1, jadi keduanya aktif, turunannya 1.

$\partial L/\partial z_1 = \partial L/\partial h_1 \times \text{ReLU}'(z_1) = -8\times1 = -8$
$\partial L/\partial z_2 = \partial L/\partial h_2 \times \text{ReLU}'(z_2) = 4\times1 = 4$

Gradient bobot hidden:

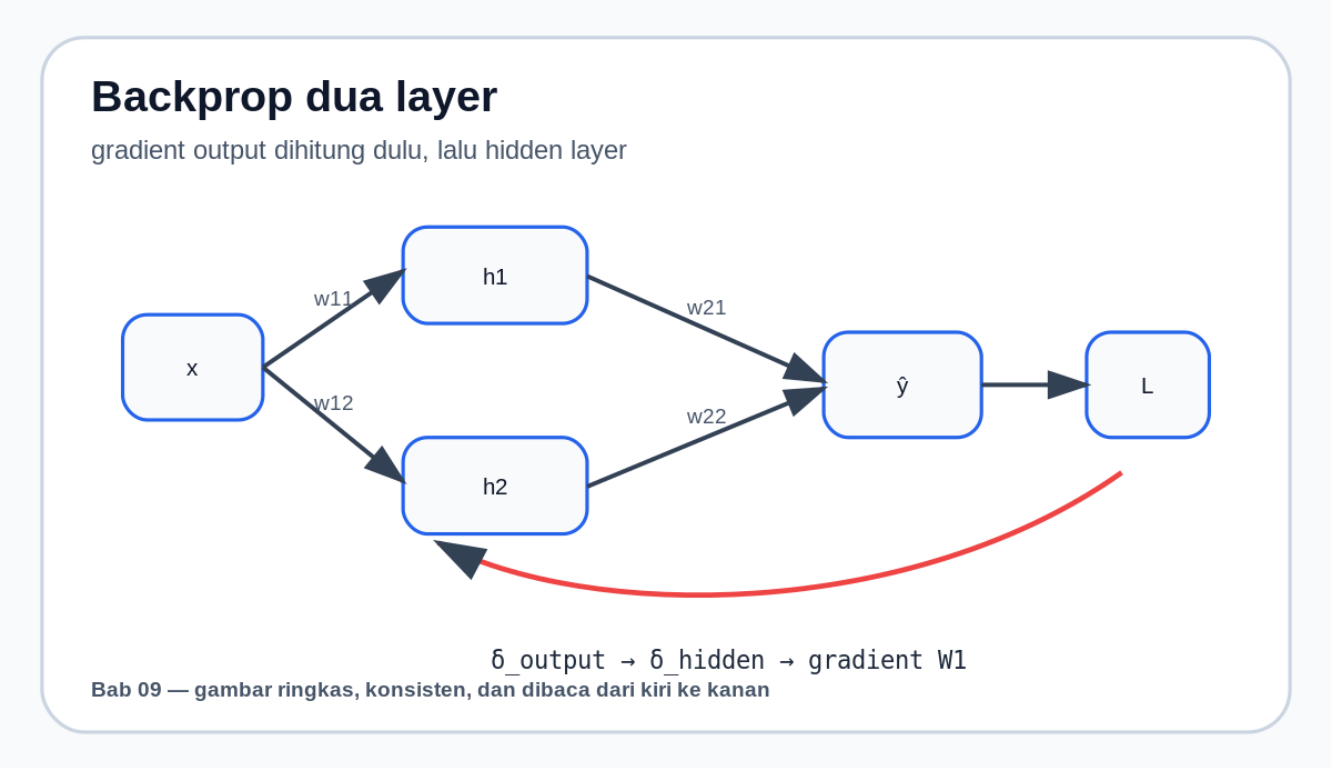
$\partial L/\partial w_{11} = \partial L/\partial z_1 \times x = -8\times2 = -16$
$\partial L/\partial b_{11} = -8$
$\partial L/\partial w_{12} = \partial L/\partial z_2 \times x = 4\times2 = 8$
$\partial L/\partial b_{12} = 4$

Update satu langkah

Jika $\eta=0,01$:

w_{21} baru = 2 - 0,01(-8) = 2,08
w_{22} baru = -1 - 0,01(-4) = -0,96
w_{11} baru = 1 - 0,01(-16) = 1,16
w_{12} baru = -1 - 0,01(8) = -1,08

Makna: backpropagation memberi tahu setiap bobot seberapa besar ia bertanggung jawab terhadap error. Bobot tidak diperbaiki secara acak; setiap update mengikuti jalur kontribusinya.



Backpropagation dua layer

Tes cepat subbab 15

1. Mengapa gradient output layer dihitung sebelum hidden layer?

2. Jika ReLU tidak aktif (z≤0), berapa gradient yang lewat?

3. Hitung ∂L/∂w■■ jika ∂L/∂■=-3 dan h■=4.

Subbab 16 — Multi-layer perceptron: jaringan feedforward

Inti subbab: MLP menyusun input, satu atau lebih hidden layer, dan output layer.

Bentuk dua layer:

h = activation($W_1x + b_1$)
ŷ = output_activation($W_2h + b_2$)

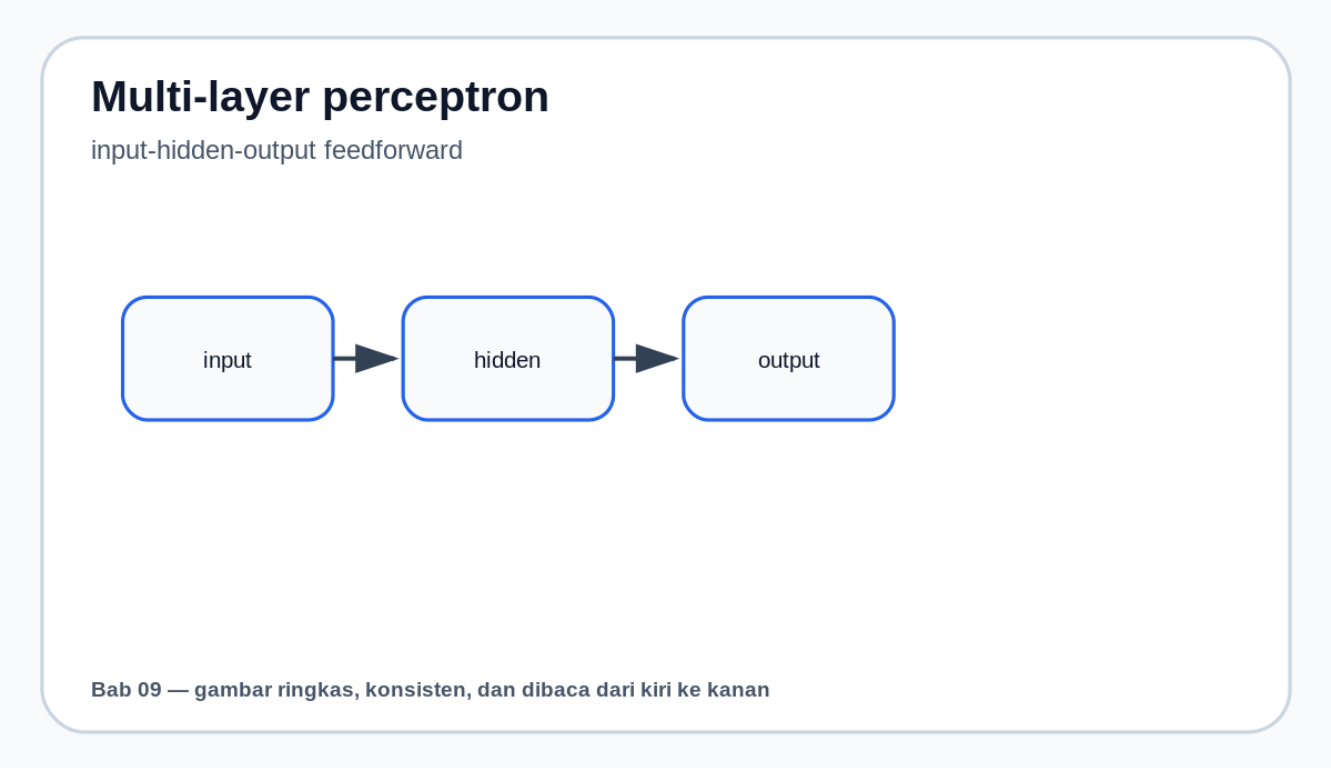
Cara membaca: layer pertama membuat representasi h. Layer kedua memakai representasi itu untuk prediksi. Jika ada banyak hidden layer, representasi makin bertingkat.

Contoh arsitektur:

2 input → 3 hidden ReLU → 1 output sigmoid

Jumlah parameter:

W_1: 2×3 = 6
b_1: 3
W_2: 3×1 = 3
b_2: 1
total = 13 parameter



MLP

Tes cepat subbab 16

1. Apa itu hidden layer?

2. Hitung parameter untuk `4 input` → `5 hidden` → `2 output`.

3. Mengapa output sigmoid cocok untuk klasifikasi biner?

Subbab 17 — Mengapa NN cocok untuk data non-linear?

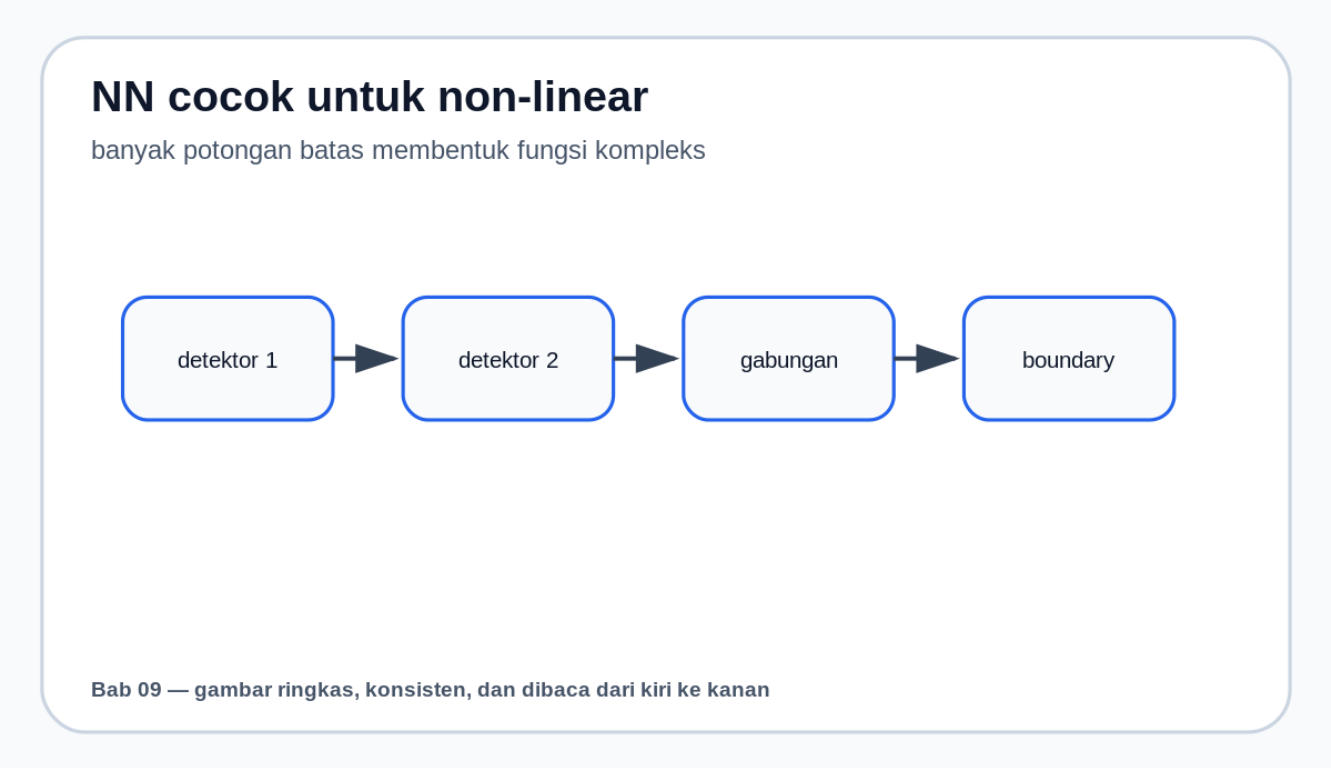
Inti subbab: neural network cocok untuk data non-linear karena menyusun banyak batas sederhana menjadi bentuk keputusan kompleks.

Model linear membuat satu garis, bidang, atau hyperplane. Jika pola data melingkar, XOR, gelombang, atau interaksi fitur kompleks, satu garis tidak cukup. Neural network berlapis dapat membuat fitur baru dan menggabungkan banyak potongan batas.

Contoh non-linear:

> kelas 1 jika x_1 dan x_2 berbeda (XOR)
> kelas 1 jika titik berada di dalam lingkaran
> harga naik karena kombinasi lokasi, luas, akses, musim, dan interaksi fitur

Hidden neuron bisa dipandang sebagai detektor kondisi. Beberapa neuron mendeteksi “wilayah kiri”, “wilayah atas”, “kombinasi fitur”, lalu output menggabungkannya.



NN non-linear

Contoh perhitungan lingkaran intuitif

Data kelas 1 jika:

$$x_1^2 + x_2^2 < 1$$

Ini bukan garis lurus. Model linear w■x■+w■x■+b hanya bisa membuat batas garis. NN dengan hidden layer dapat mendekati lingkaran memakai banyak potongan ReLU, seperti menggambar lingkaran dengan banyak garis kecil.

Tes cepat subbab 17

1. Mengapa XOR non-linear?
2. Mengapa satu garis tidak cukup untuk data melingkar?
3. Bagaimana hidden neuron membantu membentuk batas kompleks?

Subbab 18 — Contoh hitung non-linear lengkap: XOR dengan sigmoid halus

Inti subbab: bahkan dengan aktivasi halus, NN dapat mendekati XOR.

Gunakan bobot besar agar sigmoid mendekati step.

Hidden OR:

$$h_1 = \text{sigmoid}(10x_1 + 10x_2 - 5)$$

Hidden NAND:

$$h_2 = \text{sigmoid}(-10x_1 - 10x_2 + 15)$$

Output AND:

$$p = \text{sigmoid}(10h_1 + 10h_2 - 15)$$

Kasus `(0,1)`

$h_1 = \text{sigmoid}(0 + 10 - 5) = \text{sigmoid}(5) \approx 0{,}993$
$h_2 = \text{sigmoid}(0 - 10 + 15) = \text{sigmoid}(5) \approx 0{,}993$
$p = \text{sigmoid}(10 \times 0{,}993 + 10 \times 0{,}993 - 15)$
$= \text{sigmoid}(19{,}86 - 15)$
$= \text{sigmoid}(4{,}86)$
$\approx 0{,}992$

Prediksi mendekati 1.

Kasus `(1,1)`

$h_1 = \text{sigmoid}(10 + 10 - 5) = \text{sigmoid}(15) \approx 1$
$h_2 = \text{sigmoid}(-10 - 10 + 15) = \text{sigmoid}(-5) \approx 0{,}007$
$p = \text{sigmoid}(10 \times 1 + 10 \times 0{,}007 - 15)$
$= \text{sigmoid}(-4{,}93)$
$\approx 0{,}007$

Prediksi mendekati 0.

Sigmoid XOR

Tes cepat subbab 18

1. Mengapa bobot 10 membuat sigmoid mirip step?

2. Hitung kasar output untuk `(0,0)`.

3. Mengapa probabilitas 0,992 dibaca sebagai kelas 1?

Subbab 19 — Overfitting dalam neural network

Inti subbab: NN fleksibel, sehingga mudah menghafal data jika tidak dikendalikan.

Overfitting terjadi ketika loss training turun, tetapi performa validation buruk. Neural network dengan terlalu banyak parameter bisa menghafal noise.

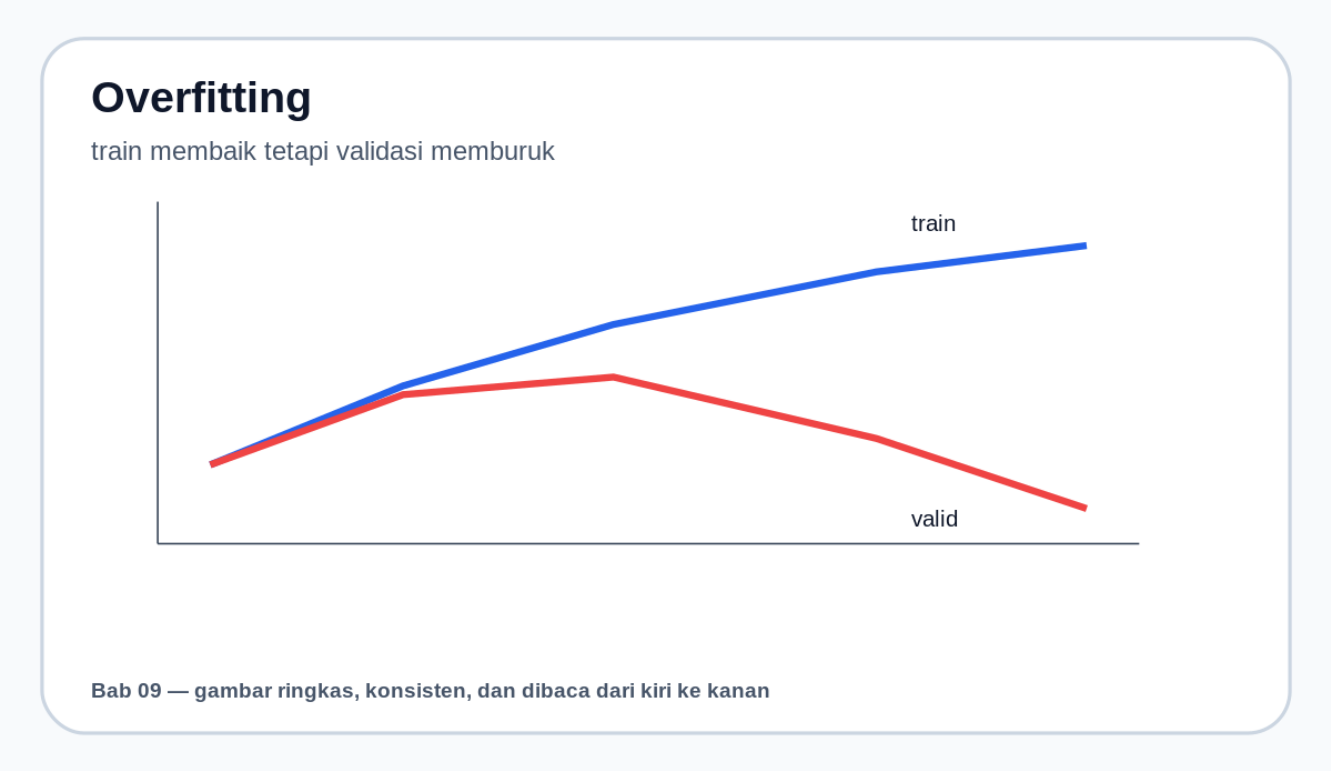
Tanda overfitting:

training loss turun terus
validation loss mulai naik

akurasi training tinggi, validation rendah
prediksi terlalu percaya diri pada data baru

Strategi mengurangi overfitting:

data lebih banyak
regularisasi L2
dropout
early stopping
augmentasi data
model lebih kecil
validasi yang benar



Overfitting NN

Tes cepat subbab 19

1. Apa tanda overfitting?

2. Mengapa model besar rawan menghafal?

3. Apa itu early stopping?

Subbab 20 — Regularisasi, dropout, dan early stopping

Inti subbab: regularisasi adalah rem agar model tidak terlalu liar.

L2 regularization menambahkan penalti bobot besar:

$$L_{total} = L_{data} + \lambda \Sigma w^2$$

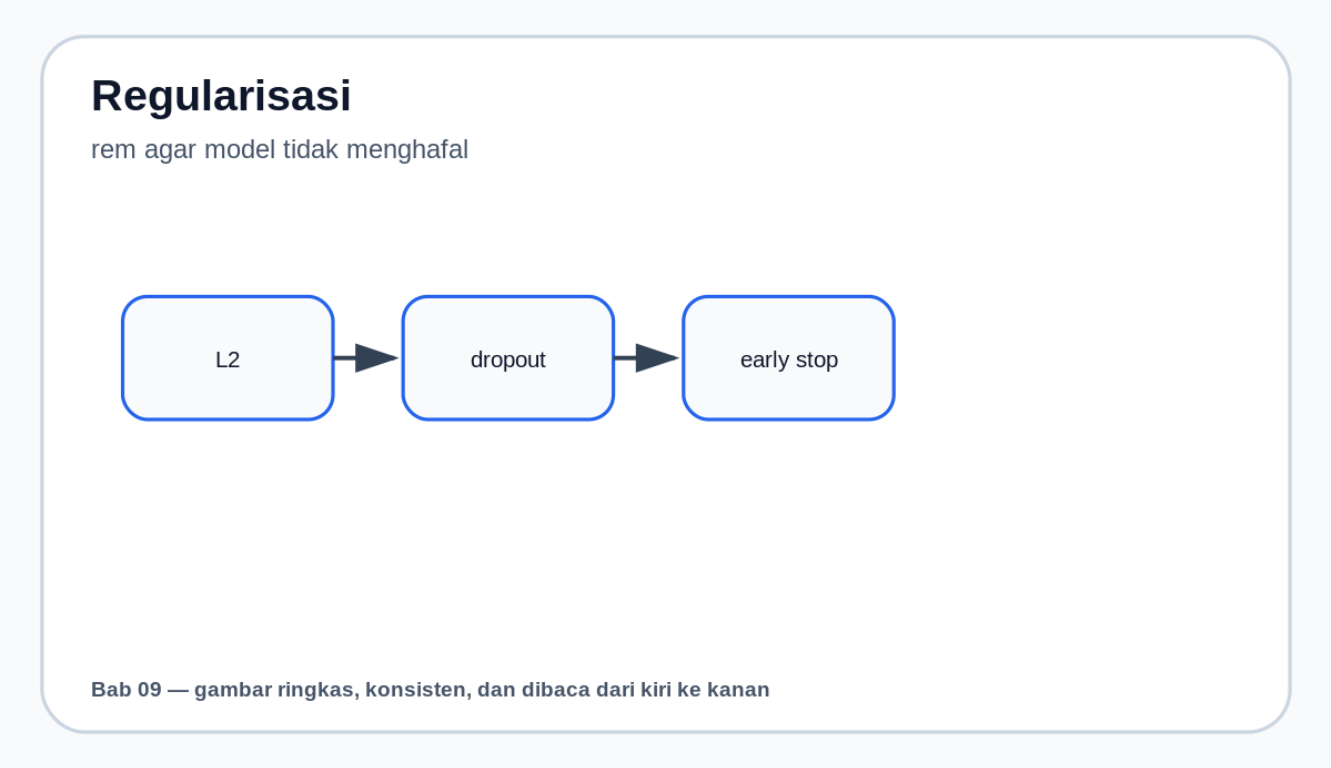
Cara membaca: loss total adalah loss data ditambah lambda kali jumlah kuadrat bobot. Jika bobot terlalu besar, penalti naik.

Dropout secara acak mematikan sebagian neuron saat training. Tujuannya agar jaringan tidak bergantung pada satu jalur saja.

Early stopping menghentikan training ketika validation loss tidak membaik.

Contoh L2:

L_data = 0,5
λ = 0,1
w = [2, -1]
Σw² = 4 + 1 = 5
L_total = 0,5 + 0,1×5 = 1,0



Regularisasi

Tes cepat subbab 20

1. Hitung L2 penalty untuk `w=[1,2,3]`, `λ=0,01`.

2. Apa fungsi dropout?

3. Mengapa validation loss dipakai untuk early stopping?

Subbab 21 — Learning rate, epoch, batch, dan optimizer

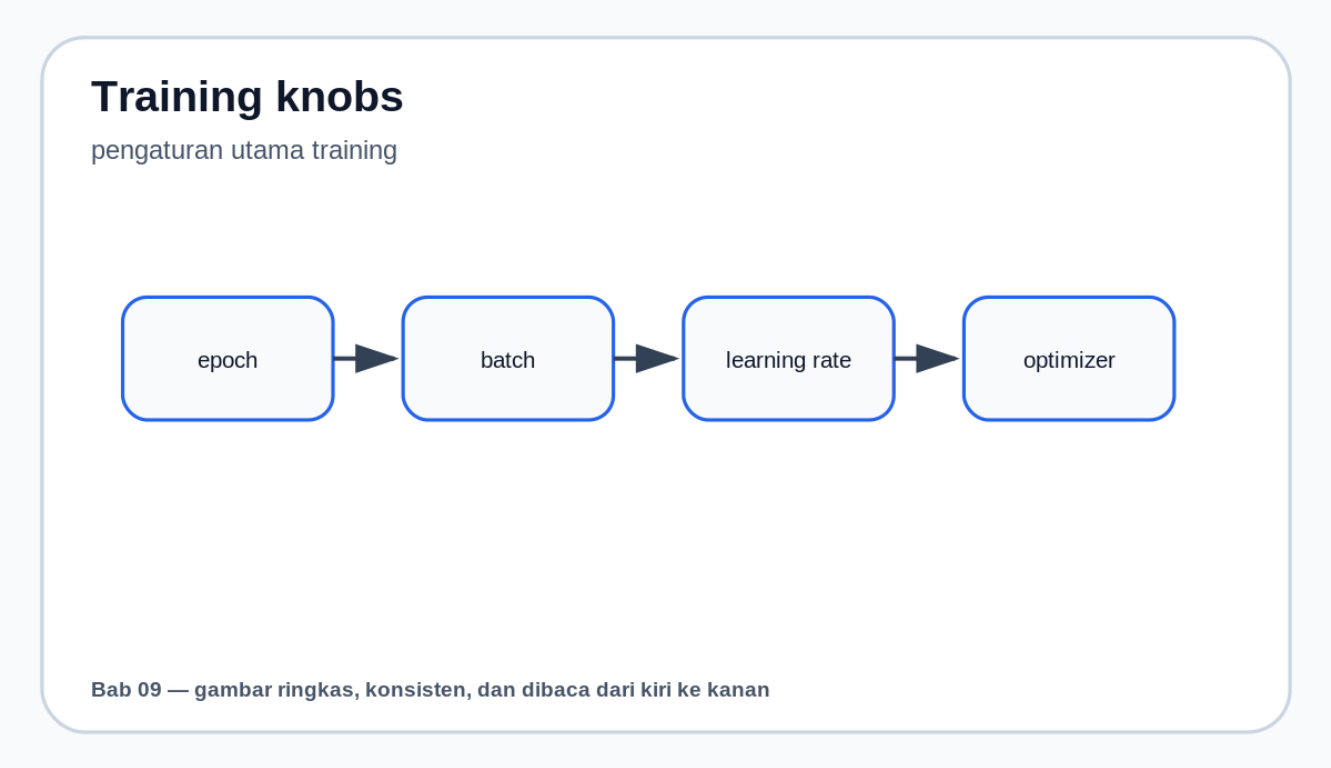
Inti subbab: training NN sangat dipengaruhi cara update dilakukan.

Istilah penting:

epoch = satu putaran melewati seluruh data
batch = subset data untuk satu update
learning rate = ukuran langkah update
optimizer = aturan update parameter

Batch gradient descent memakai semua data untuk satu update. Stochastic gradient descent memakai satu data. Mini-batch memakai sebagian data, paling umum di deep learning.

Jika learning rate terlalu kecil, training lambat. Jika terlalu besar, loss bisa berosilasi atau meledak.



Training knobs

Contoh: dataset 1.000 baris, batch size 100. Satu epoch berisi:

```
1000 / 100 = 10 update
```

Tes cepat subbab 21

1. Apa itu epoch?
2. Jika data 500 dan batch 50, berapa update per epoch?
3. Apa risiko learning rate terlalu besar?

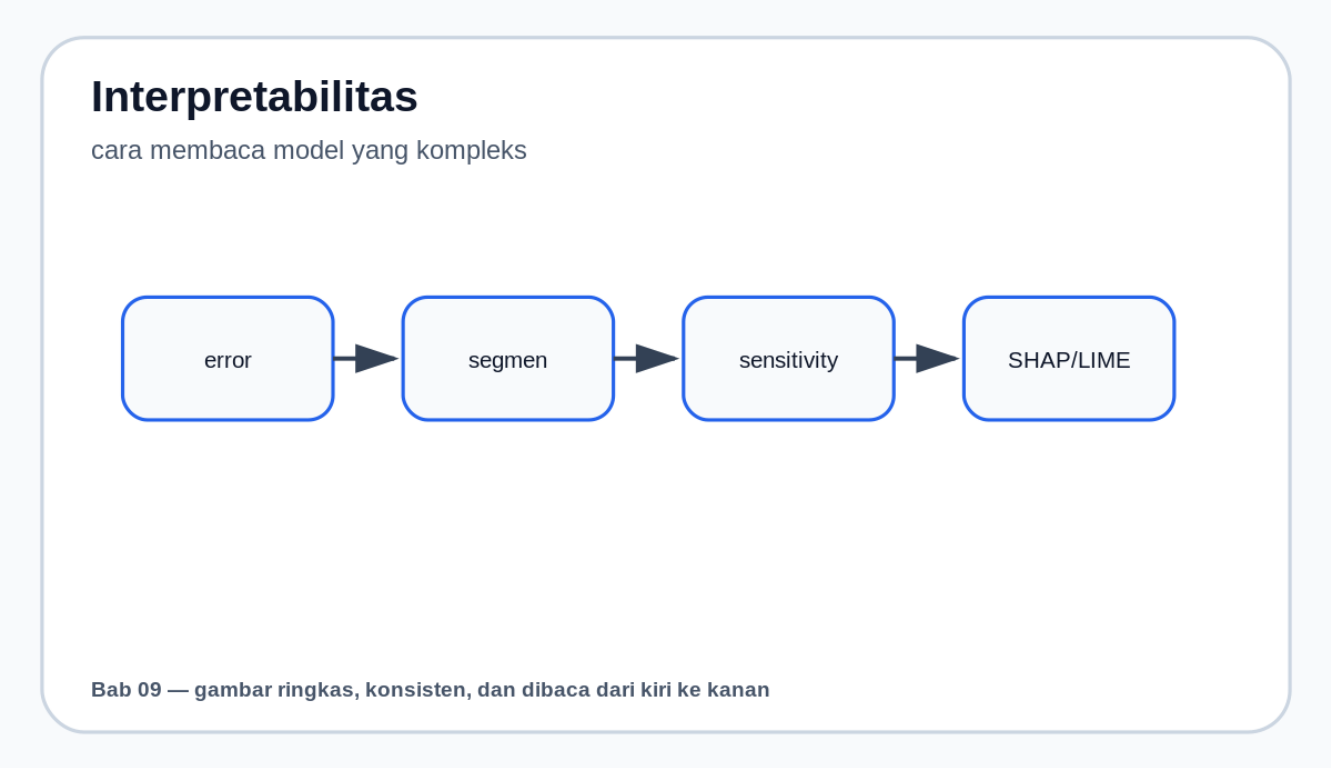
Subbab 22 — Interpretasi bobot dan keterbatasan penjelasan

Inti subbab: NN bisa kuat tetapi tidak selalu mudah dijelaskan.

Pada model linear, bobot mudah dibaca: fitur naik, skor naik/turun. Pada NN, bobot tersebar di banyak layer. Satu neuron hidden bisa menjadi detektor pola, tetapi maknanya tidak selalu jelas.

Cara memahami NN:

```
lihat performa per segmen
analisis error
uji sensitivity input
visualisasi aktivasi
pakai feature importance/permutation
pakai SHAP/LIME dengan hati-hati
```



Interpretasi NN

Catatan etika: untuk keputusan berdampak tinggi, model akurat saja tidak cukup. Harus ada evaluasi bias, dokumentasi data, dan mekanisme banding/manusia.

Tes cepat subbab 22

1. Mengapa NN lebih sulit dijelaskan daripada model linear?

2. Apa itu analisis error per segmen?

3. Mengapa interpretabilitas penting untuk keputusan sensitif?

Subbab 23 — Praktikum Bab 9: neural network kecil dari nol

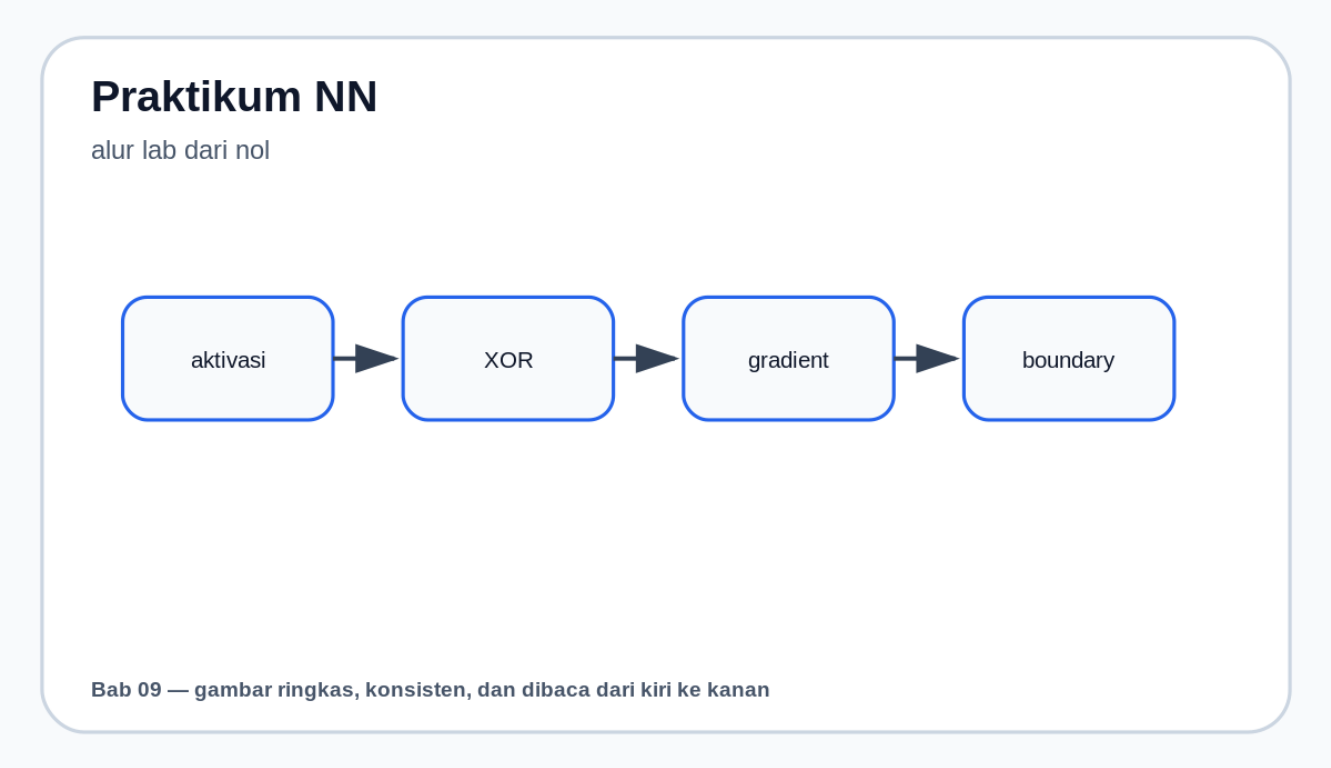
Inti subbab: pembaca membangun forward pass, XOR network, dan training linear neuron dengan Python standard library.

File:

```
chapters/09-neural-networks/code/neural_network_playground.py
chapters/09-neural-networks/code/neural_network_playground.ipynb
```

Praktikum berisi:

1. fungsi sigmoid, ReLU, step
2. perceptron AND/OR
3. bukti XOR gagal dengan perceptron tunggal
4. XOR network dua hidden neuron
5. forward pass MLP kecil
6. gradient neuron linear untuk MSE
7. training loop sederhana
8. visualisasi decision boundary SVG



Praktikum NN

Tes cepat subbab 23

1. Mengapa praktikum dimulai dari standard library?

2. Apa output yang harus cocok untuk XOR?

3. Apa manfaat visualisasi decision boundary?

Subbab 24 — Output layer: sigmoid, softmax, regresi, dan pilihan loss

Inti subbab: bentuk output layer harus mengikuti tipe masalah.

Neural network tidak selalu berakhir dengan aktivasi yang sama. Untuk regresi, output sering linear:

$$\hat{y} = z$$

Untuk klasifikasi biner, output sigmoid:

$$p = \text{sigmoid}(z)$$

Untuk klasifikasi multi-kelas, output softmax:

$$\text{softmax}(z_i) = e^{z_i} / \Sigma_j e^{z_j}$$

Cara membaca softmax: eksponensial skor kelas ke-i dibagi jumlah eksponensial semua skor. Hasilnya menjadi distribusi probabilitas yang jumlahnya 1.

Contoh skor tiga kelas:

$$z = [1, 2, 0]$$
$$e^z \approx [2{,}718, 7{,}389, 1]$$
$$\text{jumlah} = 11{,}107$$
$$\text{softmax} \approx [0{,}245, 0{,}665, 0{,}090]$$

Model paling percaya kelas kedua.

Turunan penting: sigmoid + binary cross-entropy

Untuk klasifikasi biner, kombinasi sigmoid dan binary cross-entropy sering dipakai:

$$p = \text{sigmoid}(z)$$
$$L = -[y \log(p) + (1-y)\log(1-p)]$$

Turunan akhirnya sangat sederhana:

$$\partial L/\partial z = p - y$$

Cara membaca: sinyal error sebelum output adalah probabilitas prediksi dikurangi label. Jika $p=0,8$ dan $y=1$, maka:

$$\partial L/\partial z = 0,8 - 1 = -0,2$$

Gradient negatif membuat skor z naik agar probabilitas mendekati 1.

Jika $p=0,8$ tetapi $y=0$:

$$\partial L/\partial z = 0,8 - 0 = 0,8$$

Gradient positif membuat skor turun agar probabilitas mengecil.

Turunan penting: softmax + cross-entropy

Untuk multi-kelas satu label:

$$p = \text{softmax}(z)$$
$$L = -\Sigma\, y_i \log(p_i)$$

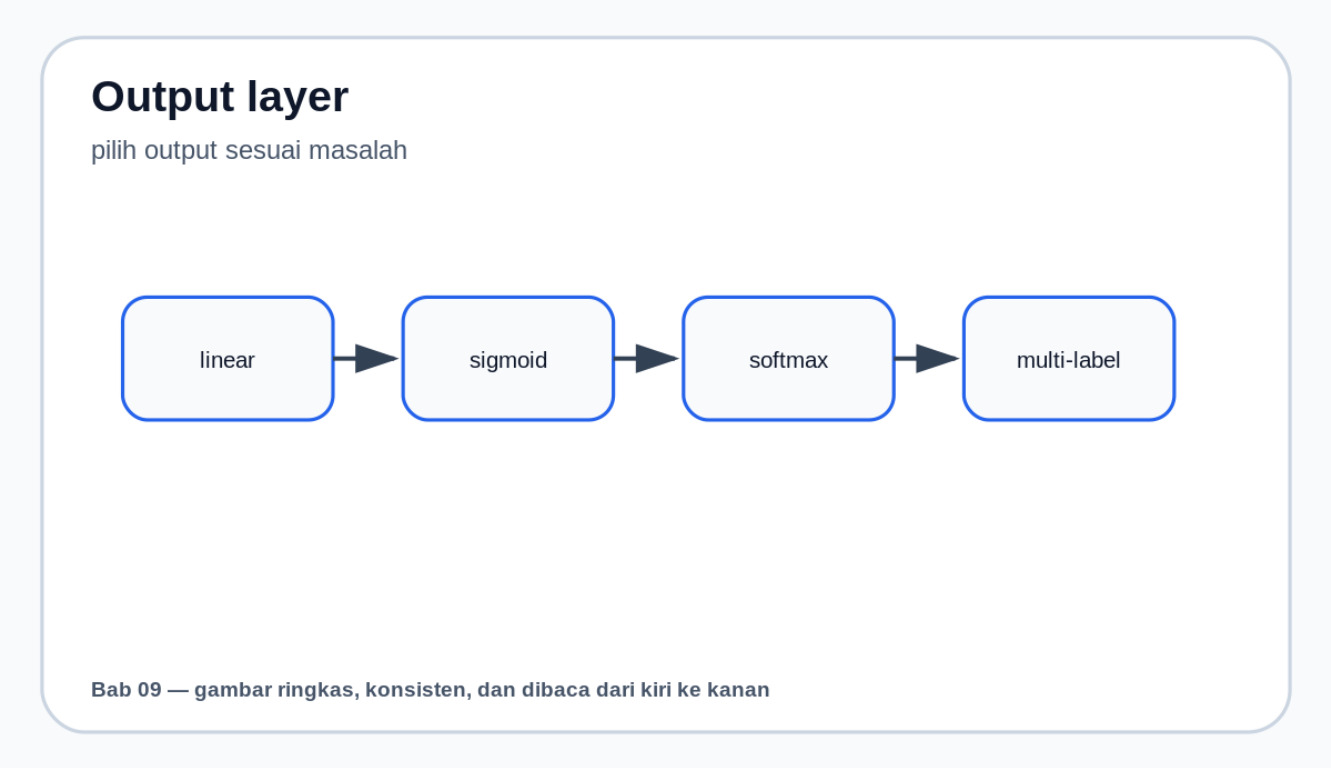
Turunan terhadap skor kelas ke-i:

$$\partial L/\partial z_i = p_i - y_i$$

Cara membaca: untuk setiap kelas, sinyal error adalah probabilitas prediksi dikurangi target one-hot. Misalnya target kelas kedua:

$$p = [0,245,\ 0,665,\ 0,090]$$
$$y = [0,\ \ 1,\ \ 0]$$
$$\partial L/\partial z = [0,245,\ -0,335,\ 0,090]$$

Skor kelas benar naik karena gradientnya negatif; skor kelas salah turun karena gradientnya positif.



Output layer

Peta pilihan:

Masalah	Output	Loss umum
Regresi angka	linear	MSE/MAE
Biner	sigmoid	binary cross-entropy
Multi-kelas satu label	softmax	categorical cross-entropy
Multi-label	banyak sigmoid	binary cross-entropy per label

Tes cepat subbab 24

1. Mengapa sigmoid cocok untuk klasifikasi biner?

2. Apa sifat utama softmax?

3. Untuk prediksi harga rumah, output apa yang masuk akal?

Subbab 25 — Inisialisasi bobot dan masalah gradient hilang/meledak

Inti subbab: training NN tidak hanya soal rumus, tetapi juga kondisi awal dan stabilitas gradient.

Jika semua bobot diinisialisasi sama, neuron dalam layer bisa belajar hal yang sama. Karena itu bobot biasanya diinisialisasi acak kecil. Namun acak juga tidak boleh sembarangan. Bobot terlalu besar bisa membuat aktivasi saturasi atau gradient meledak. Bobot terlalu kecil bisa membuat sinyal hilang.

Masalah gradient hilang (*vanishing gradient*) terjadi ketika gradient menjadi sangat kecil saat mengalir ke layer awal. Pada sigmoid, turunan maksimum hanya 0,25:

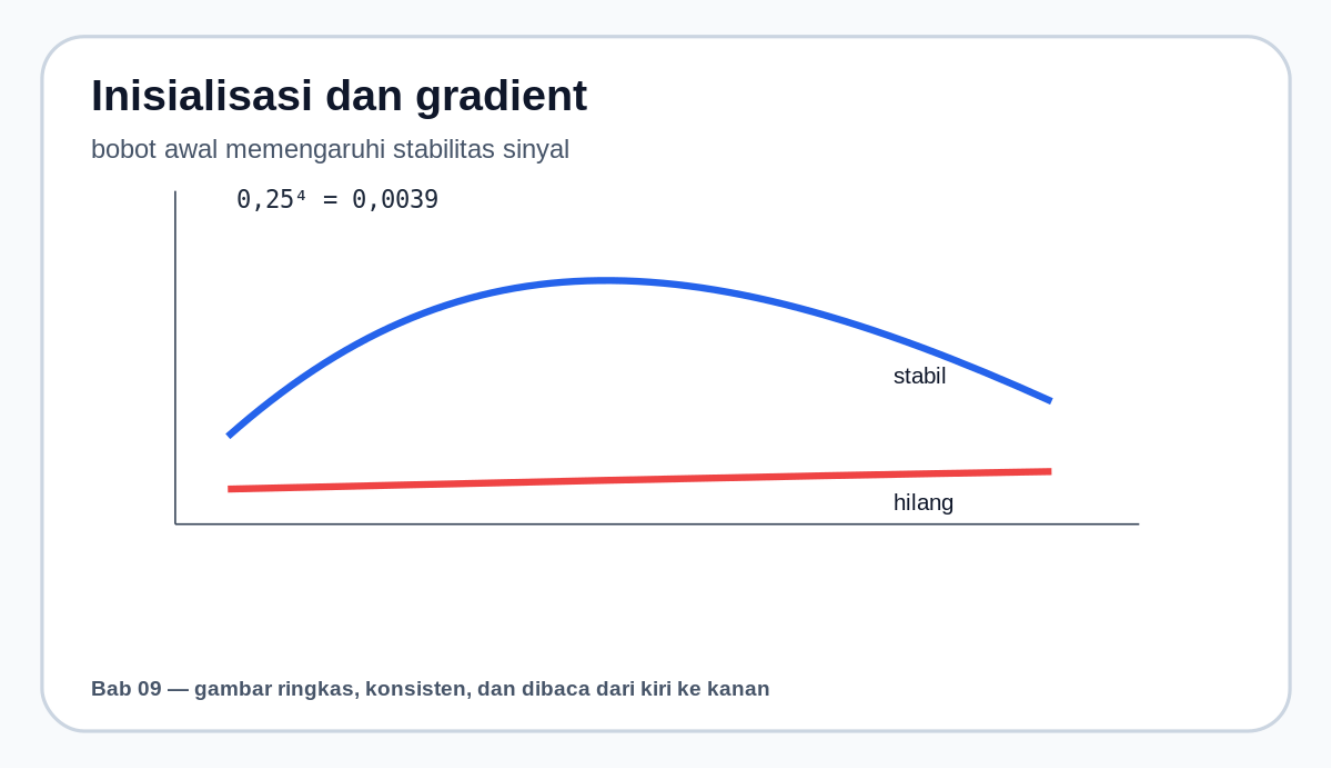
$$\sigma'(z)=\sigma(z)(1-\sigma(z)) \le 0{,}25$$

Cara membaca: setiap kali gradient melewati sigmoid, ia bisa dikalikan angka kecil. Jika banyak layer, hasil perkalian bisa sangat kecil.

Contoh sederhana:

$$0{,}25 \times 0{,}25 \times 0{,}25 \times 0{,}25 = 0{,}0039$$

Gradient hampir hilang. ReLU membantu karena turunannya 1 untuk bagian positif, tetapi ReLU juga bisa mati jika neuron terus berada di sisi negatif.



Inisialisasi dan gradient

Solusi praktis: Xavier/He initialization, ReLU/variant, batch normalization, residual connection, learning rate yang tepat, dan monitoring gradient.

Tes cepat subbab 25

1. Mengapa semua bobot tidak boleh diinisialisasi sama?

2. Hitung `0,25^3`.

3. Apa itu vanishing gradient?

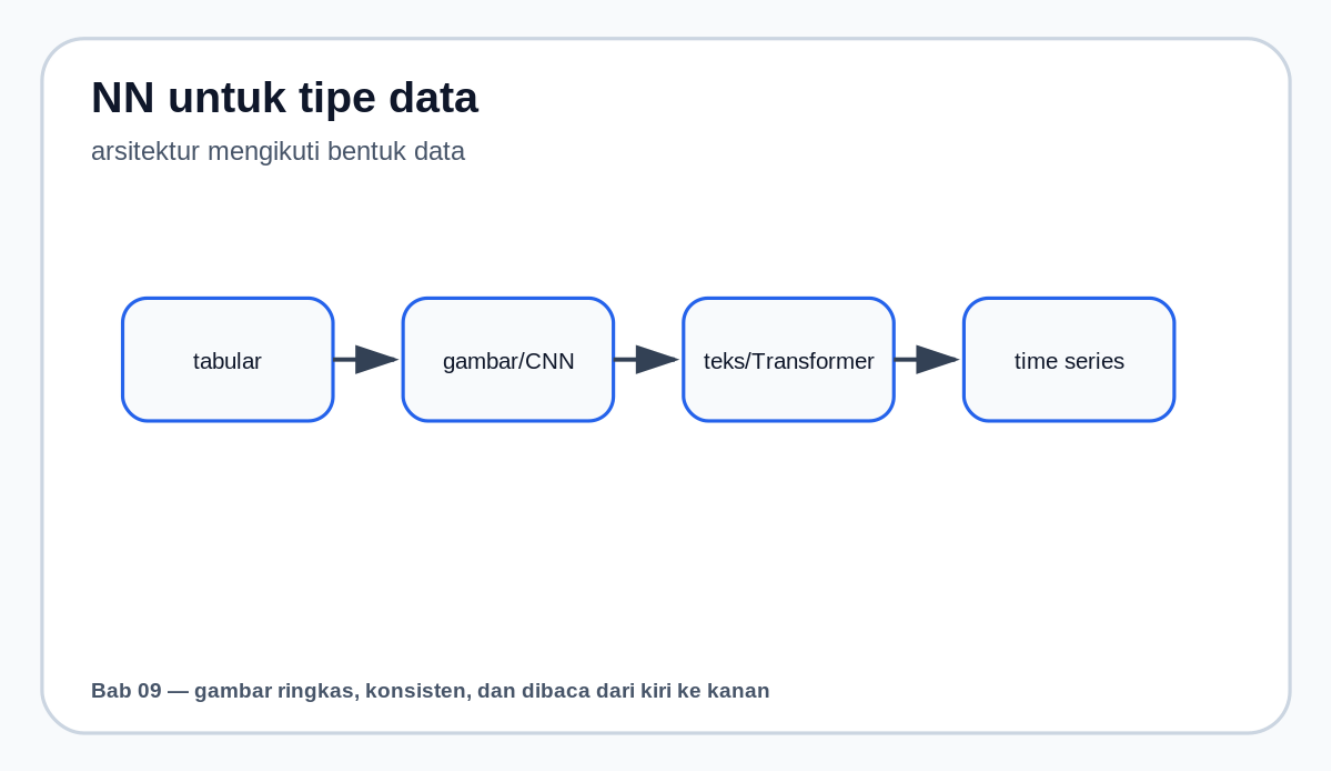
Subbab 26 — Neural network untuk tipe data berbeda

Inti subbab: arsitektur NN mengikuti bentuk data.

Data tabular, gambar, teks, audio, dan time series punya struktur berbeda. Memakai MLP untuk semua hal bisa menjadi baseline, tetapi tidak selalu optimal.

Tipe data	Struktur	Arsitektur NN umum
Tabular	kolom fitur	MLP, embedding kategori, TabNet/FT-Transformer
Gambar	grid piksel 2D	CNN, Vision Transformer
Teks	urutan token	RNN lama, Transformer modern
Audio	sinyal waktu/spectrogram	CNN, Transformer audio
Time series	urutan waktu	RNN/LSTM/GRU, Temporal CNN, Transformer

Untuk tabular kecil, model tree/boosting sering lebih kuat daripada NN. Untuk gambar dan teks besar, NN modern sangat dominan karena dapat belajar representasi dari data mentah.



NN untuk tipe data

Contoh: gambar kucing tidak cocok direpresentasikan hanya dengan rata-rata warna. CNN belajar pola tepi, tekstur, bentuk, lalu objek. Teks tidak cukup dihitung panjangnya; embedding/transformer menangkap konteks kata.

Tes cepat subbab 26

1. Mengapa CNN cocok untuk gambar?
2. Mengapa MLP tidak selalu terbaik untuk tabular?
3. Apa arsitektur modern yang banyak dipakai untuk teks?

Subbab 27 — Evaluasi, error analysis, dan debugging neural network

Inti subbab: NN harus dievaluasi seperti sistem ilmiah, bukan dipercaya karena loss turun.

Checklist debugging:

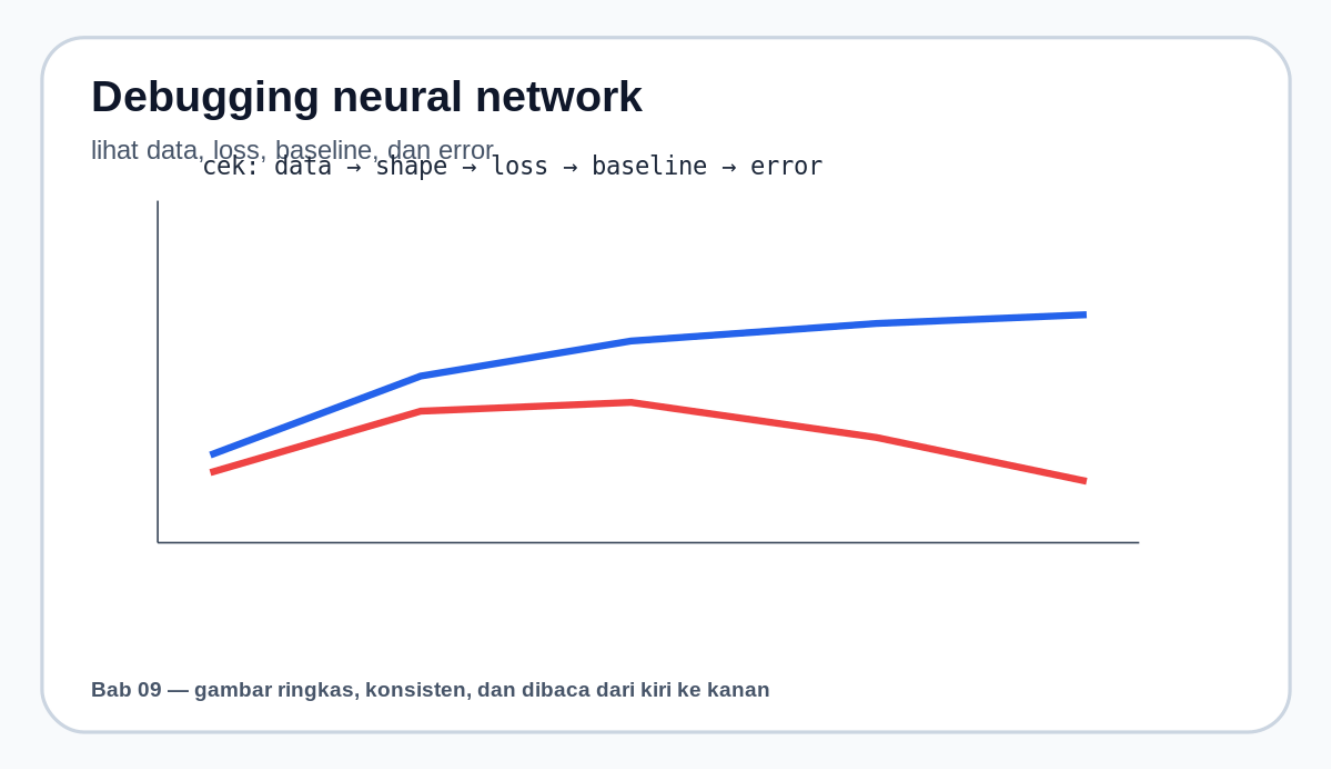
Apakah data dan label benar?
Apakah split bebas leakage?
Apakah loss training turun?
Apakah validation ikut membaik?
Apakah metrik sesuai masalah?
Apakah ada kelas/segmen yang gagal?
Apakah prediksi terlalu percaya diri?
Apakah baseline sederhana sudah dikalahkan?

Langkah debugging praktis:

1. Latih model kecil pada dataset sangat kecil sampai overfit. Jika tidak bisa, ada bug.
2. Cek satu batch manual: shape, nilai input, target, loss.
3. Bandingkan dengan baseline linear/tree.

4. Plot training vs validation loss.

5. Lihat contoh error, bukan hanya angka rata-rata.



Debugging NN

Contoh error analysis: model klasifikasi pelanggan terlihat akurat 90%, tetapi gagal pada pelanggan daerah tertentu. Ini bisa terjadi karena data training tidak representatif. Solusinya bukan hanya menambah epoch; perlu audit data, fairness, dan mungkin fitur tambahan.

Tes cepat subbab 27

1. Mengapa baseline tetap penting untuk NN?

2. Apa arti validation loss naik saat training loss turun?

3. Mengapa analisis error per segmen penting?

Pembahasan latihan hitung terstruktur Bab 9

Latihan 1 — Perceptron OR

Gunakan `w=[1,1]`, `b=-0,5`.

[0,0]: z=-0,5 → 0
[0,1]: z=0,5 → 1
[1,0]: z=0,5 → 1
[1,1]: z=1,5 → 1

Ini cocok dengan OR.

Latihan 2 — Update gradient neuron linear

Diberikan:

x=2, y=5, w=1, b=0
$\hat{y} = wx+b = 2$
error = $\hat{y}-y = -3$

$\partial L/\partial w = 2(\hat{y}-y)x = 2\times(-3)\times 2 = -12$
$\partial L/\partial b = 2(\hat{y}-y) = -6$

Jika $\eta = 0,1$:

$w_{baru} = 1 - 0,1\times(-12) = 2,2$
$b_{baru} = 0 - 0,1\times(-6) = 0,6$

Prediksi naik karena sebelumnya terlalu kecil.

Latihan 3 — Jumlah parameter

Arsitektur $4 \text{ input} \rightarrow 5 \text{ hidden} \rightarrow 2 \text{ output}$:

$W_1 = 4\times 5 = 20$
$b_1 = 5$
$W_2 = 5\times 2 = 10$
$b_2 = 2$
total = 37 parameter

Latihan 4 — Mengapa NN cocok untuk non-linear

Data XOR tidak bisa dipisahkan satu garis. Hidden layer membuat fitur baru `OR` dan `NAND`. Di ruang hidden, output cukup melakukan AND. Ini contoh sederhana bahwa NN bisa mengubah representasi agar masalah non-linear menjadi lebih mudah.

Pendalaman teknis — dari model linear ke neural network yang benar-benar belajar

Bagian ini memperkuat ide utama bab: neural network bukan mantra, melainkan kumpulan operasi matematika yang dapat dihitung satu per satu. Jika pembaca memahami alur berikut, maka deep learning di Bab 10 akan terasa jauh lebih masuk akal.

1. Model linear melihat dunia sebagai satu kemiringan besar

Model linear untuk dua fitur:

$z = w_1x_1 + w_2x_2 + b$

Batas keputusan $z=0$ adalah garis. Semua titik di satu sisi garis diprediksi kelas 1, semua titik di sisi lain kelas 0. Ini sangat kuat untuk pola sederhana: harga naik seiring luas rumah, risiko naik seiring jumlah keterlambatan, atau probabilitas beli naik seiring frekuensi kunjungan.

Masalahnya, banyak data lapangan tidak mengikuti satu kemiringan global. Contoh pelanggan:

pelanggan sering datang + belanja sedang → loyal
pelanggan jarang datang + belanja sangat tinggi → VIP sesekali
pelanggan sering datang + belanja rendah → pemburu promo

Satu garis sulit menangkap kombinasi seperti itu. Neural network membuat beberapa detektor kondisi, lalu menggabungkannya.

2. Hidden neuron sebagai pembuat “aturan lunak”

Satu hidden neuron dapat dibaca sebagai aturan lunak:

$h = \text{ReLU}(w\cdot x+b)$

Jika $w\cdot x+b$ negatif, neuron diam. Jika positif, neuron aktif. Dengan banyak neuron, jaringan membuat banyak potongan wilayah. Gabungan potongan ini dapat membentuk batas zig-zag, melengkung, atau bertingkat.

Contoh dua neuron:

h_1 aktif jika $x_1 + x_2$ cukup besar
h_2 aktif jika $x_1 - x_2$ cukup besar

Output kemudian membaca kombinasi h■ dan h■. Ini mirip membuat fitur baru:

fitur asli → fitur kondisi → prediksi

Inilah mengapa Bab 8 tentang representation learning penting. Neural network belajar representasi internal yang membuat tugas prediksi lebih mudah.

3. Contoh mini training loop satu data

Misalkan model linear kecil:

$\hat{y} = wx + b$
$L = (\hat{y}-y)^2$

Data:

$x=2, y=6$
$w=1, b=0, \eta=0{,}1$

Forward pass:

$\hat{y} = 1\times2 + 0 = 2$
$L = (2-6)^2 = 16$

Gradient:

$\partial L/\partial w = 2(\hat{y}-y)x = 2(2-6)2 = -16$
$\partial L/\partial b = 2(\hat{y}-y) = -8$

Update:

$w_{baru} = 1 - 0{,}1(-16) = 2{,}6$
$b_{baru} = 0 - 0{,}1(-8) = 0{,}8$

Forward ulang:

$\hat{y} = 2{,}6\times2 + 0{,}8 = 6{,}0$
$L = (6-6)^2 = 0$

Untuk satu data sederhana, satu update bisa langsung cocok. Pada data nyata, update untuk satu titik bisa memperbaiki titik itu tetapi mengganggu titik lain. Karena itu kita memakai banyak data, batch, epoch, dan validation set.

4. Mengapa backpropagation efisien

Tanpa backpropagation, menghitung pengaruh setiap bobot secara manual akan sangat mahal. Backpropagation menyimpan hasil forward, lalu memakai chain rule dari output ke input. Ia tidak mencoba semua perubahan bobot satu per satu; ia menghitung gradient secara sistematis.

Bayangkan jaringan punya 10.000 bobot. Finite difference akan mencoba mengubah bobot satu per satu. Backpropagation menghitung semua gradient dengan biaya yang sebanding dengan beberapa forward pass. Inilah alasan neural network besar bisa dilatih.

5. Kapan neural network bukan pilihan terbaik

Walaupun bab ini memperlihatkan kekuatan NN, pembaca harus tetap kritis. Neural network tidak selalu pilihan pertama. Untuk data tabular kecil, baseline linear, decision tree, random forest, atau gradient boosting sering lebih cepat, lebih mudah dijelaskan, dan lebih kuat. Neural network bersinar ketika:

data besar
pola non-linear kompleks
data tidak terstruktur seperti gambar/teks/audio

representasi perlu dipelajari otomatis
komputasi cukup
validasi ketat tersedia

Jika data kecil, noisy, dan butuh interpretasi tinggi, model sederhana bisa lebih baik. Prinsip buku ini tetap: mulai dari baseline, naik kompleksitas jika ada bukti.

6. Checklist memahami NN sebelum lanjut ke deep learning

Pembaca siap lanjut jika dapat menjawab:

Apa itu neuron?
Apa fungsi aktivasi?
Mengapa XOR gagal untuk perceptron tunggal?
Bagaimana hidden layer menyelesaikan XOR?
Apa itu forward pass?
Apa itu loss?
Bagaimana gradient descent mengubah bobot?
Apa itu backpropagation?
Mengapa NN cocok untuk pola non-linear?
Kapan NN tidak perlu dipakai?

Jika jawaban masih kabur, ulangi contoh XOR dan gradient manual. Dua contoh itu adalah kunci pintu menuju deep learning.

7. Cara membaca grafik decision boundary NN

Grafik decision boundary biasanya menunjukkan bidang input dua dimensi. Warna latar menunjukkan prediksi model untuk setiap wilayah. Titik data menunjukkan contoh asli. Jika warna berubah mengikuti pola titik, model belajar struktur. Jika warna terlalu bergerigi di sekitar setiap titik, model mungkin overfitting. Jika warna terlalu lurus padahal data melengkung, model underfitting.

Untuk data non-linear, boundary NN sering tampak seperti gabungan beberapa garis kecil. ReLU membuat potongan linear; banyak potongan membentuk kurva kasar. Inilah intuisi penting: NN tidak selalu menggambar kurva halus secara ajaib, tetapi menyusun banyak transformasi sederhana menjadi bentuk kompleks.

Saat membaca boundary, tanyakan:

Apakah boundary mengikuti pola umum atau hanya menghafal titik?
Apakah area dekat boundary penuh ketidakpastian?
Apakah ada kelas minoritas yang tertelan kelas mayoritas?
Apakah boundary masuk akal menurut domain?

Pertanyaan ini menjaga pembaca agar tidak terpesona visual tanpa analisis.

Kamus cara membaca persamaan Bab 9

Persamaan	Cara membaca	Makna
`z=w·x+b`	skor mentah adalah dot product bobot-input plus bias	bagian linear neuron
`a=activation(z)`	aktivasi mengubah skor mentah menjadi output neuron	memberi non-linearitas
`ReLU(z)=max(0,z)`	ambil z jika positif, 0 jika negatif	aktivasi populer
`sigmoid(z)=1/(1+e^-z)`	mengubah skor menjadi angka 0 sampai 1	probabilitas biner
`Z=XW+b`	banyak data dikalikan bobot layer lalu ditambah bias	forward layer
`MSE=(1/n)Σ(■-y)²`	rata-rata kuadrat error prediksi	loss regresi

Persamaan	Cara membaca	Makna
$\theta \leftarrow \theta - \eta \partial L/\partial \theta$	parameter dikurangi learning rate kali gradient	gradient descent
$\partial L/\partial w = 2(\blacksquare - y)x$	gradient bobot linear MSE	arah update bobot
$\partial L/\partial w = \partial L/\partial a \times \partial a/\partial z \times \partial z/\partial w$	aturan rantai backprop	error mengalir balik
$L_{total} = L_{data} + \lambda \Sigma w^2$	loss data ditambah penalti bobot	regularisasi L2

Ringkasan Bab 9

• Neural network menyusun neuron menjadi fungsi fleksibel.

• Satu neuron sebelum aktivasi adalah model linear.

• Aktivasi non-linear membuat layer bertumpuk menjadi lebih kuat.

• Perceptron bisa menyelesaikan AND/OR tetapi gagal pada XOR.

• XOR menunjukkan mengapa hidden layer penting.

• Hidden layer membuat representasi baru.

• NN cocok untuk data non-linear karena dapat menyusun banyak batas sederhana menjadi fungsi kompleks.

• Forward pass menghitung prediksi dari input ke output.

• Loss function mengukur kesalahan.

• Gradient descent memperbaiki parameter.

• Backpropagation memakai chain rule untuk menghitung gradient banyak layer.

• NN fleksibel tetapi rawan overfitting.

• Regularisasi, dropout, early stopping, dan validasi penting.

• Interpretabilitas dan audit tetap penting untuk keputusan berdampak tinggi.

Referensi utama bab

• McCulloch, W. S., & Pitts, W. (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity.

• Rosenblatt, F. (1958). The Perceptron.

• Minsky, M., & Papert, S. (1969). Perceptrons.

• Rumelhart, Hinton, Williams (1986). Learning representations by back-propagating errors.

• Goodfellow, Bengio, Courville. Deep Learning.

• Bishop. Pattern Recognition and Machine Learning.

• Nielsen. Neural Networks and Deep Learning.

Catatan validasi internal v0.1

• Bab memakai format subbab seperti Bab 7 dan Bab 8.

• Memuat sejarah, persamaan, cara membaca persamaan, contoh hitung manual, XOR non-linear, hidden layer, backpropagation, regularisasi, dan praktikum.