Meeting 5: Varian Arsitektur, Pre-training & Scaling Laws

AI-40X: Generative AI & Large Language Models

Hendri Karisma, M.T.
Dosen Teknik Informatika STMIK Tazkia
VP Engineering at jejakin.com, 2026

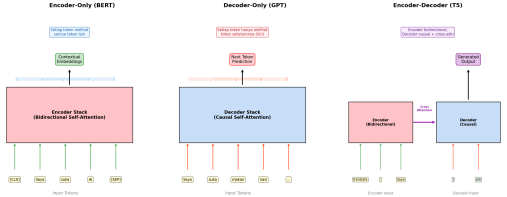
Semester 4 — 2025/2026

Outline

Tiga Paradigma Besar Transformer

Di semester 3, kalian sudah mengenal arsitektur Transformer secara umum. Sekarang kita dalami **tiga varian utama** yang mendominasi dunia NLP modern:

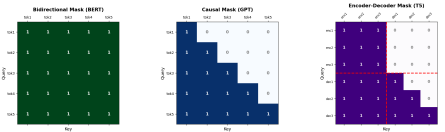
1. **Encoder-Only** — BERT, RoBERTa, ELECTRA
2. **Decoder-Only** — GPT, Llama, Mistral
3. **Encoder-Decoder** — T5, BART, mBART



Encoder-Only: BERT & Keluarganya

Ciri Khas: Bidirectional Context

- Setiap token bisa "melihat" **semua** token lain dalam kalimat — kiri *dan* kanan.
- **Pre-training:** Masked Language Modeling (MLM) — mask 15% token, model menebak token yang di-mask.
- *Contoh:* "Ibu pergi ke `[MASK]` untuk membeli sayur." → "pasar"
- **Kekuatan:** Pemahaman konteks mendalam — klasifikasi, NER, extractive QA.
- **Kelemahan:** Tidak bisa men-*generate* teks secara autoregressive.

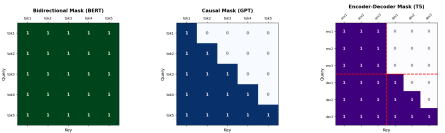


Attention mask penuh: semua token saling melihat.

Decoder-Only: GPT & Keluarganya

Ciri Khas: Causal Masking

- Token ke-t **hanya** bisa melihat token $w_1, w_2, \ldots, w_{t-1}$.
- Masa depan "gelap" — dicegah oleh **triangular mask** ($-\infty$ di bagian atas-kanan).
- **Pre-training:** Causal Language Modeling (CLM) — prediksi token berikutnya.
- **Kekuatan:** Generasi teks, coding, chat, reasoning.
- **Dominasi saat ini:** GPT-4, Llama 3, Mistral, Qwen — semua Decoder-Only.



Causal mask: segitiga bawah saja yang aktif.

Encoder-Decoder: T5 & BART

Ciri Khas: Sequence-to-Sequence

- **Encoder** membaca input secara bidirectional (seperti BERT).
- **Decoder** men-generate output secara autoregressive (seperti GPT), dengan tambahan **cross-attention** ke encoder.
- **T5 (Text-to-Text Transfer Transformer):** Semua task diubah jadi format teks-ke-teks.
  - Klasifikasi: "sentiment: I love this!" $\rightarrow$ "positive"
  - Terjemahan: "translate English to Indonesian: Hello" $\rightarrow$ "Halo"
- **Pre-training T5:** Span Corruption — ganti span acak dengan sentinel token `<extra_id_0>`, model harus me-reconstruct span tersebut.
- **BART:** Denoising autoencoder — mengorupsi input (mask, delete, shuffle), decoder merekonstruksi teks asli.

Tabel Perbandingan: Arsitektur vs Kesesuaian Task

Aspek	Encoder-Only	Decoder-Only	Enc-Dec
Contoh Model	BERT, RoBERTa	GPT, Llama	T5, BART
Attention	Bidirectional	Causal (unidirectional)	Bid. + Causal
Pre-training	MLM	CLM (next token)	Span Corruption
Klasifikasi	**Sangat Baik**	Baik	Baik
Generasi Teks	Tidak Cocok	**Sangat Baik**	**Sangat Baik**
Terjemahan	Kurang Cocok	Baik	**Sangat Baik**
Summarization	Kurang Cocok	Baik	**Sangat Baik**
Reasoning/Chat	Kurang Cocok	**Sangat Baik**	Baik
Dominasi 2024+	Menurun	**Dominan**	Niche

Tren: Decoder-Only semakin mendominasi karena sifatnya yang general-purpose dan kemampuan *in-context learning*.

Masked Language Modeling — BERT

Objective: Prediksi token yang di-mask berdasarkan konteks bidirectional.

Prosedur

1. Pilih 15% token secara acak dari input.
2. Dari 15% tersebut:
   - 80% diganti dengan `[MASK]`
   - 10% diganti dengan token acak
   - 10% dibiarkan asli (agar model tidak hanya bergantung pada `[MASK]`)
3. Model memprediksi token asli di posisi yang dipilih.

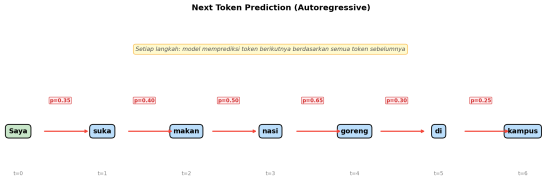
Loss Function:

$$\mathcal{L}_{\text{MLM}} = -\sum_{i \in \mathcal{M}} \log P(w_i \mid \mathbf{w}_{\setminus \mathcal{M}})$$

di mana $\mathcal{M}$ adalah himpunan posisi yang di-mask.

Causal Language Modeling — GPT

Objective: Prediksi token berikutnya berdasarkan konteks sebelumnya (autoregressive).



Loss Function:

$$\mathcal{L}_{\text{CLM}} = -\sum_{t=1}^{T} \log P(w_t \mid w_1, w_2, \ldots, w_{t-1})$$

Mengapa CLM Sangat Kuat?

"Compression is intelligence" (Ilya Sutskever) — untuk memprediksi kata berikutnya dengan baik, model *harus* memahami sintaksis, semantik, logika, dan pengetahuan dunia.

Span Corruption — T5

Objective: Mengganti beberapa *span* (rentang) token dengan sentinel token, lalu model merekonstruksi span tersebut.

Contoh

Input: "Mahasiswa <X> di kampus <Y> pagi."
Target: "<X> belajar AI <Y> setiap <Z>"

- Span rata-rata 3 token, total ≈15% token yang dikorupsi.
- Lebih efisien dari MLM: target sequence lebih pendek (hanya span yang di-mask, bukan seluruh kalimat).
- Model belajar **pemahaman** (encoder membaca konteks) **dan generasi** (decoder menghasilkan span).

Scaling Laws — Kaplan et al. (OpenAI, 2020)

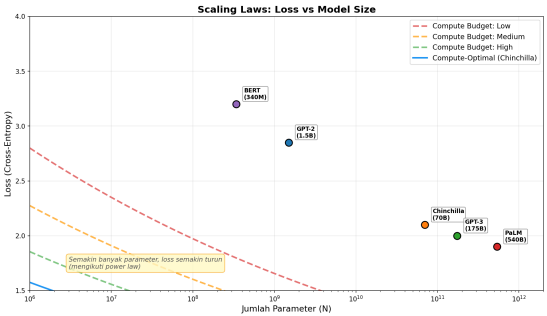
"Hukum Fisika"-nya LLM:

- Performa model (loss L) mengikuti **power law** terhadap tiga faktor:
  1. Jumlah Parameter (N)
  2. Ukuran Dataset (D)
  3. Budget Komputasi (C)
- Hubungan:

$$L(N) \approx \left(\frac{N_c}{N}\right)^{\alpha_N}$$

$$L(D) \approx \left(\frac{D_c}{D}\right)^{\alpha_D}$$

- Loss turun secara **predictable** seiring skala naik!



Grafik log-log: loss vs parameter/data/compute menghasilkan garis lurus (power law).

Chinchilla Optimal — Hoffmann et al. (DeepMind, 2022)

Koreksi penting terhadap Kaplan:

- Kaplan: "*Besarkan model, data secukupnya.*" $\rightarrow$ Hasilnya model besar tapi **undertrained**.
- Chinchilla: "*Seimbangkan parameter dan data!*"

Aturan Emas Chinchilla

Untuk budget komputasi C yang tetap, alokasikan secara seimbang:

- Jika C naik $10\times$, naikkan **parameter** dan **data** masing-masing $\approx 3.2\times$.
- **Rule of thumb:** Latih model N parameter dengan $\approx 20N$ token.

Contoh Praktis

- Llama-2 7B $\rightarrow$ Chinchilla optimal: $7 \times 10^9 \times 20 = 140$ miliar token.
- Llama-3 8B dilatih dengan 15 *triliun* token — jauh di atas Chinchilla optimal (**overtrained**) demi performa inferensi yang lebih kuat pada ukuran kecil.

Emergent Abilities — Kemampuan yang Muncul dari Skala

"More is Different" — Philip W. Anderson (Fisikawan Nobel)

- Pada model kecil (<10B parameter), banyak kemampuan yang **sama sekali tidak ada**.
- Ketika skala melewati ambang batas tertentu (*threshold*), kemampuan ini **muncul tiba-tiba** (*phase transition*).

Contoh Emergent Abilities:

1. **In-Context Learning:** Belajar dari contoh di prompt tanpa update parameter.
2. **Chain-of-Thought (CoT):** Reasoning step-by-step.
3. **Aritmatika multi-digit:** $3847 + 2916 = ?$
4. **Translasi bahasa langka** yang minim di data training.

Catatan Kritis

Wei et al. (2023) menunjukkan bahwa "emergence" mungkin merupakan artefak dari metrik evaluasi diskrit. Dengan metrik kontinu, peningkatan bisa terlihat gradual.

Mengapa Small Language Models Penting?

Tidak semua orang punya akses ke GPU cluster.

Masalah Model Besar:

- GPT-4: estimasi \$100M+ untuk training
- Llama-3 405B: butuh 8×A100 80GB untuk inferensi
- Biaya API mahal untuk skala besar
- Data sensitif tidak boleh dikirim ke cloud

Keunggulan Small Models:

- **Biaya:** Bisa jalan di laptop/HP
- **Latensi:** Respons cepat (edge deployment)
- **Privasi:** Data tetap lokal
- **Aksesibilitas:** Riset & startup kecil bisa berpartisipasi

Filosofi: "The best model is the smallest one that solves your problem."

Contoh Small Language Models

Model	Parameter	Pengembang	Highlight
SmolLM	135M – 1.7B	Hugging Face	Cocok untuk perangkat sangat kecil
TinyLlama	1.1B	Open-source	3T token, Chinchilla $\times$50 overtrained
Phi-2	2.7B	Microsoft	Mengalahkan Llama-2 7B di beberapa benchmark
Phi-3 Mini	3.8B	Microsoft	Competitive dengan Llama-3 8B
Gemma 2	2B	Google	Kualitas tinggi untuk ukurannya
Qwen2.5	0.5B – 3B	Alibaba	Kuat di bahasa Asia termasuk Indonesia

Rahasia keberhasilan: Bukan hanya soal arsitektur — **kualitas data** dan **lama pelatihan** (overtrain) sangat menentukan.

Knowledge Distillation: Mentransfer Kecerdasan

Ide: Gunakan model besar (*teacher*) untuk melatih model kecil (*student*).

Proses Distillation

1. **Teacher** menghasilkan "soft labels" — distribusi probabilitas penuh atas vocabulary (bukan hanya argmax).
2. **Student** dilatih untuk meniru distribusi teacher, bukan hanya ground truth.
3. Loss gabungan: $\mathcal{L} = \alpha \cdot \mathcal{L}_{\text{hard}} + (1 - \alpha) \cdot \mathcal{L}_{\text{soft}}$

- Soft labels mengandung **dark knowledge** — informasi relasi antar kelas yang hilang di hard labels.
- *Contoh:* Teacher bilang "kucing" 70%, "harimau" 20%, "anjing" 5% — student belajar bahwa kucing mirip harimau.
- **Contoh sukses:** DistilBERT (40% lebih kecil, 97% performa BERT).

Relevansi untuk Indonesia

Mengapa small models sangat relevan untuk konteks Indonesia?

1. **Infrastruktur terbatas:** Tidak semua institusi punya akses GPU mahal. Google Colab free tier cukup untuk model $<$3B.
2. **Bahasa Indonesia kurang terwakili** di training data model besar — fine-tune model kecil pada data Indonesia bisa lebih efektif.
3. **Deployment edge:** Aplikasi di daerah dengan koneksi internet terbatas butuh model yang bisa jalan offline.
4. **Startup & UMKM:** Biaya inferensi model kecil jauh lebih terjangkau.
5. **Riset akademik:** Mahasiswa dan dosen bisa melakukan eksperimen penuh tanpa budget cloud besar.

Peluang

Melatih small LM yang fokus pada Bahasa Indonesia (Pantun, hukum Islam, pertanian, dll.) — niche yang belum banyak digarap!

Kesimpulan

1. **Tiga paradigma Transformer:** Encoder-Only (pemahaman), Decoder-Only (generasi, dominan saat ini), Encoder-Decoder (seq2seq).
2. **Pre-training objectives** menentukan kemampuan model: MLM → pemahaman, CLM → generasi, Span Corruption → keduanya.
3. **Scaling Laws:** Performa model mengikuti power law — predictable, tapi mahal.
4. **Chinchilla:** Seimbangkan ukuran model dan data. Overtrain model kecil → alternatif efisien.
5. **Small Language Models:** AI yang terjangkau, relevan untuk Indonesia.

Minggu Depan (Meeting 6)

NanoGPT — Membangun LLM dari Nol. Kita akan membedah kode dan melatih GPT kecil sendiri!

Referensi

- Devlin, J., et al. (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." *NAACL*.
- Radford, A., et al. (2018). "Improving Language Understanding by Generative Pre-Training." (GPT-1).
- Raffel, C., et al. (2020). "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer." (T5). *JMLR*.
- Kaplan, J., et al. (2020). "Scaling Laws for Neural Language Models." *arXiv:2001.08361*.
- Hoffmann, J., et al. (2022). "Training Compute-Optimal Large Language Models." (Chinchilla). *NeurIPS*.
- Wei, J., et al. (2022). "Emergent Abilities of Large Language Models." *TMLR*.
- Hinton, G., et al. (2015). "Distilling the Knowledge in a Neural Network." *NeurIPS Workshop*.
- Zhang, P., et al. (2024). "TinyLlama: An Open-Source Small Language Model." *arXiv:2401.02385*.