Ringkasan Buku 1

Nama Buku: Tensorflow in Actions
Team: Wilhelmina Arlene, Luthfiah Maulidya, Fadhilah Dwi Istiani

Chapter 5 — *State-of-the-art in Deep Learning: Transformers*

🎯 **Tujuan Utama Chapter 5**
Chapter 5 memperkenalkan **Transformer**, model yang:

- merevolusi NLP,
- menggantikan dominasi RNN & LSTM,
- menjadi fondasi model modern seperti **BERT, GPT, T5**, dll.

Jika Chapter 4 mengajarkan *"model berdasarkan jenis data"*, maka Chapter 5 menjawab:
"Bagaimana model bisa memahami konteks panjang secara efisien?"

MASALAH BESAR SEBELUM TRANSFORMER
Keterbatasan RNN / LSTM
Walaupun RNN & LSTM punya memori, mereka punya masalah serius:

1. **Sequential processing**
    - kata diproses satu per satu
    - tidak bisa paralel
2. **Long-range dependency**
    - sulit mengingat konteks jauh
3. **Training lambat**
    - tidak efisien untuk data besar

➡️ Dunia butuh arsitektur baru.

5.1 Representing Text as Numbers
Masalah Fundamental NLP
Komputer **tidak memahami teks**, hanya angka.
Kalimat:
"I love machine learning"
harus diubah menjadi:

- token
- lalu angka

Tokenization

Langkah awal:

- memecah teks menjadi unit kecil (token)
- bisa berupa:
  - kata
  - subword
  - karakter

Embedding

Alih-alih one-hot encoding:

- Transformer menggunakan **embedding vector**
- setiap kata → vektor berdimensi tetap

Keunggulan embedding:

- lebih padat
- membawa makna semantik
- kata mirip → vektor mirip

5.2 Understanding the Transformer Model

Ide Revolusioner Transformer

"Attention is all you need"

Artinya:

- tidak perlu RNN
- tidak perlu CNN
- cukup **attention mechanism**

Arsitektur Besar Transformer

Transformer terdiri dari:

- **Encoder**
- **Decoder**

Digunakan untuk:

- translation
- summarization
- sequence-to-sequence tasks

Encoder–Decoder View

Encoder

- menerima seluruh input sequence sekaligus
- menghasilkan representasi kontekstual

Decoder

- menghasilkan output token satu per satu
- menggunakan:
  - encoder output

- token sebelumnya

5.2.1 Self-Attention: Jantung Transformer

Apa itu Self-Attention?

Self-attention memungkinkan model:

- melihat **seluruh kata dalam kalimat**
- sekaligus
- dan menentukan:
  - kata mana penting untuk kata lain

Contoh:

"The animal didn't cross the street because **it** was tired"

Self-attention membantu model memahami bahwa **"it" = animal**, bukan street.

Mekanisme Dasar Self-Attention

Setiap token diubah menjadi:

- **Query (Q)**
- **Key (K)**
- **Value (V)**

Proses:

1. hitung kesamaan Q dengan semua K
2. normalisasi (softmax)
3. gunakan bobot untuk menjumlahkan V

Hasil:

- representasi token yang **kontekstual**

Intuisi Penting

Self-attention =

"token bertanya ke semua token lain: siapa yang relevan untukku?"

5.2.2 Masked Self-Attention

Digunakan di **decoder**:

- mencegah model "melihat masa depan"
- menjaga sifat autoregressive

Tanpa masking:

- model bisa curang
- training tidak valid

5.2.3 Multi-Head Attention

Kenapa Multi-Head?

Satu attention saja:

- terlalu terbatas

Multi-head:

- banyak attention paralel
- tiap head fokus pada:
    - grammar
    - posisi
    - makna
    - hubungan kata

Hasil:

- representasi lebih kaya

5.2.4 Fully Connected Layer

Setelah attention:

- output dilewatkan ke **feed-forward network**
- sama untuk setiap token
- memperkaya transformasi non-linear

5.2.5 Positional Encoding

Masalah Attention

Attention **tidak tahu urutan**.

Solusi:

- **positional encoding**
- menambahkan informasi posisi token

Tanpa ini:

- "dog bites man"
- sama dengan "man bites dog"

Putting Everything Together

Blok Encoder

1. Self-attention
2. Add & Normalize
3. Feed-forward
4. Add & Normalize

Blok Decoder

1. Masked self-attention
2. Encoder–decoder attention
3. Feed-forward

KEUNGGULAN TRANSFORMER

1. **Parallel processing**
2. **Context panjang**

3. **Scalable**
4. **State-of-the-art performance**