Program Studi Teknik Informatika
Institut Teknologi Sumatera

Nama	:	**Fathan Andi Kartagama(122140055)**
Mata Kuliah	:	**Pembelajaran Mendalam (IF25-40401)**
Tugas	:	**Eksplorasi Vision Transformer**
Dosen Pengampu	:	1. **Imam Eko Wicaksono, S.Si., M.Si.**
		2. **Martin Clinton Tosima Manullang, S.T., M.T., Ph.D.**
Tanggal	:	November 21, 2025

LAPORAN TUGAS EKSPLORASI

Perbandingan Model Vision Transformer, Swin Transformer, dan DeiT Base

1 Pendahuluan

1.1 Latar Belakang

Dalam 20 tahun terakhir, *Convolutional Neural Networks* (CNN) sudah jadi raja di dunia *computer vision*. CNN berhasil menyelesaikan berbagai tugas seperti klasifikasi gambar, deteksi objek, dan segmentasi dengan performa yang sangat baik [1, 2]. Tapi ternyata, CNN punya kelemahan juga. Arsitekturnya yang bergantung pada konvolusi lokal membuat CNN kurang efektif dalam menangkap hubungan antar bagian gambar yang jaraknya jauh (*long-range dependencies*).

Nah, kesuksesan *Transformer* di bidang *Natural Language Processing* (NLP) [3] membuat para peneliti bertanya-tanya: "Kenapa nggak coba pakai *Transformer* buat gambar juga?". Di tahun 2020, Dosovitskiy dan timnya memperkenalkan *Vision Transformer* (ViT) [4]. Hasilnya cukup mengejutkan bahwa ternyata *transformer* murni tanpa konvolusi bisa mencapai hasil yang setara atau bahkan lebih baik dari CNN terbaik, terutama kalau dilatih dengan dataset besar seperti ImageNet-21k dan JFT-300M.

Cara kerja Vision Transformer sebenarnya cukup sederhana. Gambar dipotong-potong jadi *patches* berukuran kecil, lalu setiap *patch* diperlakukan seperti kata-kata dalam kalimat. Dengan mekanisme *self-attention*, model bisa "melihat" hubungan antar *patches* di seluruh gambar sekaligus. Ini berbeda dengan CNN yang hanya bisa melihat area lokal di sekitarnya. Pendekatan ini memberikan fleksibilitas lebih dalam memahami konteks global dari sebuah gambar.

Kesuksesan ViT memicu banyak inovasi baru. Liu dan timnya mengembangkan *Swin Transformer* [5] yang lebih efisien dengan menggunakan *shifted window attention* dan arsitektur hierarkis untuk menangkap informasi di berbagai skala. Touvron dkk. membuat *Data-efficient Image Transformer* (DeiT) [6] yang bisa dilatih dengan efisien pada dataset yang lebih kecil seperti ImageNet-1k dengan memanfaatkan teknik *knowledge distillation*. Sementara He dkk. mengusulkan *Masked Autoencoder* (MAE) [7] yang belajar dengan cara merekonstruksi bagian gambar yang disembunyikan.

Setiap varian Vision Transformer ini punya kelebihan dan kekurangannya masing-masing. Ada yang lebih akurat tapi berat komputasinya, ada yang lebih cepat tapi butuh data lebih banyak. Makanya penting banget buat saya memahami karakteristik masing-masing model supaya bisa pilih yang paling cocok untuk kebutuhan.

1.2 Motivasi Perbandingan Model

Walaupun berbagai model Vision Transformer sudah menunjukkan hasil yang bagus di paper-paper penelitian, ada beberapa hal yang perlu diperhatikan untuk aplikasi praktis. Pertama, kebanyakan model ini dievaluasi menggunakan dataset yang sangat besar seperti ImageNet-21k atau JFT-300M. Masalahnya, dataset sebesar itu nggak selalu tersedia atau praktis dipakai di dunia nyata. Kedua, jarang ada penelitian yang membandingkan secara lengkap antara akurasi, ukuran model, dan kecepatan inferensi dalam satu eksperimen yang terkontrol.

Di praktiknya, memilih model bukan cuma soal akurasi tertinggi. Ada banyak faktor lain yang perlu dipertimbangkan:

1. **Efisiensi Komputasi**: Berapa banyak parameter dan operasi yang dibutuhkan? Ini penting banget kalau mau deploy di perangkat dengan sumber daya terbatas seperti smartphone atau perangkat IoT.

2. **Kecepatan Inferensi**: Berapa lama waktu yang dibutuhkan untuk memproses satu gambar? Kalau aplikasinya butuh real-time seperti autonomous driving atau video surveillance, kecepatan adalah hal yang sangat krusial.

3. **Efisiensi Data**: Seberapa banyak data yang dibutuhkan untuk training? Mengumpulkan dan memberi label pada data berskala besar itu mahal dan memakan banyak waktu

4. **Kemudahan Implementasi**: Seberapa mudah model di-*fine-tune* untuk domain spesifik? Beberapa arsitektur lebih mudah di-training ulang daripada yang lain.

1.3 Tujuan Eksperimen

Eksperimen ini bertujuan untuk membandingkan secara sistematis tiga arsitektur Vision Transformer yang berbeda. Secara spesifik, tujuannya adalah:

1. **Membandingkan Arsitektur Model**
   - Melihat perbedaan mendasar dalam desain arsitektur antara ViT, Swin Transformer, dan DeiT
   - Memahami bagaimana pilihan desain mempengaruhi karakteristik model
   - Mengidentifikasi kelebihan dan kekurangan masing-masing pendekatan
2. **Mengevaluasi Performa**
   - Mengukur dan membandingkan akurasi pada dataset test
   - Menghitung metrik evaluasi seperti precision, recall, dan F1-score
   - Menganalisis *confusion matrix* untuk melihat di mana model sering salah
   - Membandingkan kurva pembelajaran selama training
3. **Menganalisis Efisiensi**
   - Menghitung jumlah parameter dan ukuran model
   - Membandingkan kompleksitas komputasi dari masing-masing arsitektur
   - Melihat trade-off antara ukuran model dan akurasi yang dicapai
4. **Mengukur Kecepatan**
   - Mengukur waktu inferensi rata-rata per gambar
   - Menghitung berapa gambar yang bisa diproses per detik
   - Melihat apakah model cocok untuk aplikasi real-time atau tidak
5. **Analisis Trade-off**
   - Menganalisis hubungan antara akurasi, jumlah parameter, dan kecepatan
   - Mengidentifikasi model mana yang paling efisien
   - Mengevaluasi trade-off untuk berbagai skenario penggunaan
6. **Memberikan Rekomendasi**
   - Merekomendasikan model terbaik untuk akurasi maksimal
   - Merekomendasikan model terbaik untuk aplikasi dengan resource terbatas
   - Merekomendasikan model terbaik untuk aplikasi real-time
   - Memberikan panduan pemilihan model berdasarkan kebutuhan spesifik

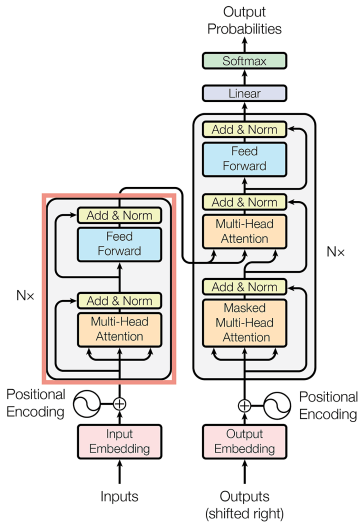
Dengan mencapai tujuan-tujuan di atas, diharapkan eksperimen ini bisa memberikan pemahaman yang lebih baik tentang karakteristik masing-masing arsitektur Vision Transformer, sekaligus memberikan panduan praktis untuk memilih model yang paling sesuai dengan kebutuhan

2 Landasan Teori

2.1 Transformer dan Mekanisme Self-Attention

2.1.1 Arsitektur Transformer

Transformer adalah arsitektur neural network yang diperkenalkan oleh Vaswani et al. [3] pada tahun 2017, yang pertama kali dirancang untuk tugas pemrosesan bahasa alami. Berbeda dengan arsitektur sekuensial seperti Recurrent Neural Networks (RNN) dan Long Short-Term Memory (LSTM), Transformer menghilangkan mekanisme rekurensi dan konvolusi, sepenuhnya bergantung pada mekanisme *attention* untuk menangkap dependensi global antara input dan output.



Gambar 1: Arsitektur Transformer dengan encoder dan decoder. Sumber: Vaswani et al. [3]

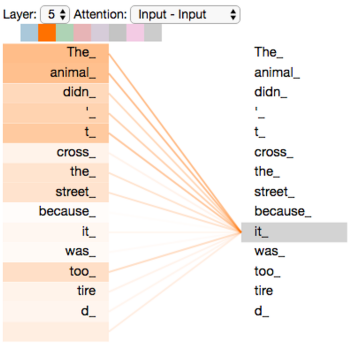
Arsitektur Transformer terdiri dari dua komponen utama: *encoder* dan *decoder*. Encoder memproses input sequence dan menghasilkan representasi abstrak, sementara decoder menggunakan representasi tersebut untuk menghasilkan output sequence. Namun, untuk aplikasi *computer vision* seperti klasifikasi gambar, umumnya hanya komponen encoder yang digunakan.

2.1.2 Mekanisme Self-Attention

Inti dari arsitektur Transformer adalah mekanisme *self-attention*, yang memungkinkan model untuk mempertimbangkan seluruh sequence input secara simultan dan menghitung representasi setiap elemen berdasarkan hubungannya dengan semua elemen lain dalam sequence. Secara matematis, mekanisme attention didefinisikan sebagai:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

di mana Q (Query), K (Key), dan V (Value) adalah matriks yang diproyeksikan dari input, dan d_k adalah dimensi dari key vector. Operasi ini menghitung *attention weights* yang menunjukkan seberapa besar setiap elemen dalam sequence harus memperhatikan elemen lainnya.



Gambar 2: Visualisasi mekanisme self-attention pada Transformer untuk tugas pemrosesan bahasa.

2.1.3 Multi-Head Attention

Untuk meningkatkan kemampuan model dalam menangkap berbagai jenis hubungan, Transformer menggunakan *multi-head attention*, yang menerapkan mekanisme attention secara paralel dengan beberapa set parameter yang berbeda (*heads*):

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \tag{2}$$

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{3}$$

di mana W_i^Q, W_i^K, W_i^V, dan W^O adalah parameter yang dapat dipelajari. Setiap *head* dapat mempelajari aspek yang berbeda dari hubungan antar elemen, seperti informasi posisi, semantik, atau sintaksis.

2.1.4 Position Encoding

Karena mekanisme self-attention tidak memiliki informasi tentang urutan atau posisi elemen dalam sequence, Transformer menggunakan *positional encoding* untuk menyediakan informasi posisi. Positional encoding ditambahkan ke input embeddings sebelum diproses oleh encoder. Vaswani et al. [3] mengusulkan penggunaan fungsi sinusoidal:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{4}$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{5}$$

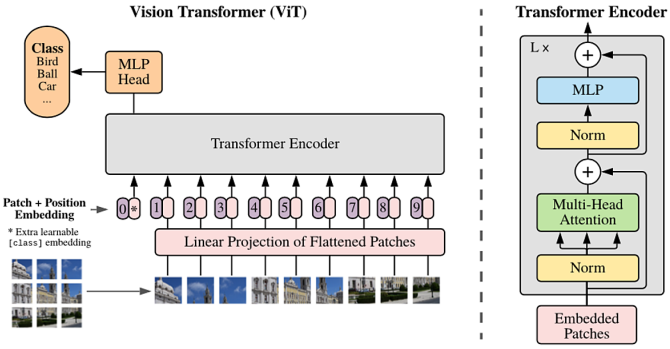
di mana *pos* adalah posisi dan i adalah dimensi. Alternatifnya, beberapa implementasi menggunakan *learned positional embeddings* yang dipelajari selama training.

2.2 Vision Transformer (ViT)

2.2.1 Arsitektur Umum

Vision Transformer (ViT), yang diperkenalkan oleh Dosovitskiy et al. [4], merupakan adaptasi langsung dari arsitektur Transformer untuk domain visi komputer. ViT mendemonstrasikan bahwa arsitektur

transformer murni, tanpa menggunakan konvolusi, dapat mencapai atau bahkan melampaui performa *state-of-the-art* CNN pada tugas klasifikasi gambar, terutama ketika dilatih pada dataset berskala besar.



Gambar 3: Arsitektur Vision Transformer (ViT).

2.2.2 Patch Embedding

Berbeda dengan pemrosesan teks yang secara natural berupa sequence of tokens, gambar perlu ditransformasi menjadi format yang sesuai untuk Transformer. ViT membagi gambar berukuran $H \times W \times C$ menjadi sequence of patches berukuran $P \times P$, menghasilkan $N = HW/P^2$ patches. Setiap patch kemudian di-flatten dan diproyeksikan ke dimensi D menggunakan linear projection:

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; ...; \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos} \tag{6}$$

di mana $\mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}$ adalah matriks embedding, $\mathbf{x}_p^i$ adalah patch ke-i, dan $\mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D}$ adalah positional embeddings. Token khusus $\mathbf{x}_{\text{class}}$ ditambahkan di awal sequence, yang representasinya pada output akan digunakan untuk klasifikasi.

2.2.3 Transformer Encoder

Setelah patch embedding, sequence diproses oleh L layer Transformer encoder. Setiap layer terdiri dari:

1. **Multi-Head Self-Attention (MSA)**: Memungkinkan setiap patch untuk berkomunikasi dengan semua patch lainnya

2. **Layer Normalization (LN)**: Normalisasi untuk stabilitas training
3. **Multi-Layer Perceptron (MLP)**: Two-layer feed-forward network dengan GELU activation
4. **Residual Connections**: Skip connections untuk aliran gradien yang lebih baik

Secara matematis, untuk layer ke-ℓ:

$$\mathbf{z}'_\ell = \text{MSA}(\text{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1} \tag{7}$$

$$\mathbf{z}_\ell = \text{MLP}(\text{LN}(\mathbf{z}'_\ell)) + \mathbf{z}'_\ell \tag{8}$$

2.2.4 Classification Head

Output dari class token pada layer terakhir $\mathbf{z}_L^0$ digunakan sebagai representasi gambar dan diproses oleh classification head (typically an MLP) untuk menghasilkan prediksi kelas:

$$\mathbf{y} = \text{LN}(\mathbf{z}_L^0)\mathbf{W}_{\text{head}} \tag{9}$$

di mana $\mathbf{W}_{\text{head}} \in \mathbb{R}^{D \times K}$ adalah weight matrix untuk K kelas.

2.2.5 Varian ViT

Dosovitskiy et al. [4] memperkenalkan tiga varian utama ViT berdasarkan ukuran model:

- **ViT-Base**: 12 layers, hidden size 768, 12 attention heads (∼86M parameters)
- **ViT-Large**: 24 layers, hidden size 1024, 16 attention heads (∼307M parameters)
- **ViT-Huge**: 32 layers, hidden size 1280, 16 attention heads (∼632M parameters)

2.3 Swin Transformer

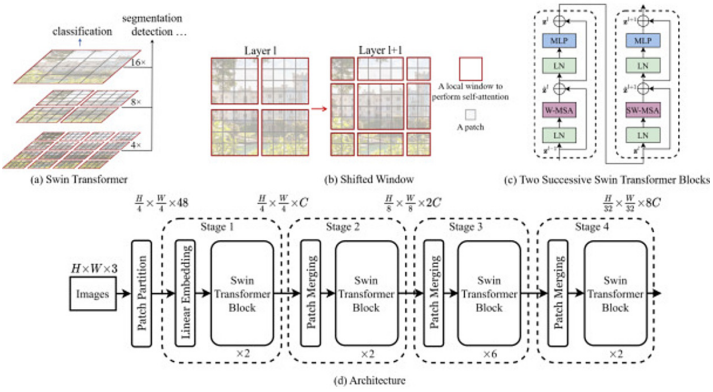
2.3.1 Motivasi dan Konsep Utama

Swin Transformer, yang diperkenalkan oleh Liu et al. [5], mengatasi beberapa keterbatasan ViT, terutama terkait efisiensi komputasi dan kemampuan untuk menangkap informasi multi-skala. Nama "Swin" merupakan singkatan dari *Shifted Windows*, yang merujuk pada mekanisme kunci dari arsitektur ini.

Dua inovasi utama Swin Transformer adalah:

1. **Hierarchical Feature Maps**: Seperti CNN, Swin Transformer membangun representasi hierarkis dengan resolusi yang menurun secara bertahap
2. **Shifted Window Attention**: Menghitung self-attention dalam window lokal yang bergeser antar layer untuk efisiensi dan komunikasi antar-window

2.3.2 Hierarchical Architecture



Gambar 4: Arsitektur hierarkis Swin Transformer. Sumber: Liu et al. [5]

Berbeda dengan ViT yang menggunakan patch size tetap dan menghasilkan feature maps dengan resolusi konstan, Swin Transformer mengadopsi arsitektur hierarkis dengan empat stage. Pada setiap stage, resolusi spasial berkurang setengah melalui *patch merging layer*, sementara jumlah channel meningkat dua kali lipat, mirip dengan arsitektur CNN seperti ResNet [2].

Dimulai dengan patch size 4×4 (lebih kecil dari ViT's 16×16), input gambar berukuran $H \times W \times 3$ diubah menjadi feature maps berukuran $\frac{H}{4} \times \frac{W}{4} \times C$ pada stage pertama. Pada stage berikutnya, patch merging mengurangi resolusi menjadi $\frac{H}{8} \times \frac{W}{8}$, $\frac{H}{16} \times \frac{W}{16}$, dan $\frac{H}{32} \times \frac{W}{32}$, dengan channel yang meningkat menjadi $2C$, $4C$, dan $8C$ respectively.

2.3.3 Window-based Multi-Head Self-Attention (W-MSA)

Untuk mengatasi kompleksitas kuadratik dari global self-attention terhadap ukuran gambar, Swin Transformer membatasi komputasi self-attention dalam window lokal non-overlapping. Dengan membagi feature map menjadi windows berukuran $M \times M$, kompleksitas komputasi untuk gambar dengan $h \times w$ patches menjadi:

$$\Omega(\text{MSA}) = 4hwC^2 + 2(hw)^2C \tag{10}$$

$$\Omega(\text{W-MSA}) = 4hwC^2 + 2M^2hwC \tag{11}$$

di mana C adalah dimensi channel. Untuk $M \ll h, w$ (typically $M = 7$), kompleksitas menjadi linear terhadap jumlah patches, bukan kuadratik.

2.3.4 Shifted Window Multi-Head Self-Attention (SW-MSA)

Meskipun W-MSA efisien, ia membatasi komunikasi antar-window. Untuk mengatasi ini, Swin Transformer menggunakan strategi *shifted window* yang menggeser window partitioning antara consecutive layers:

$$\hat{\mathbf{z}}^{\ell} = \text{W-MSA}(\text{LN}(\mathbf{z}^{\ell-1})) + \mathbf{z}^{\ell-1} \tag{12}$$

$$\mathbf{z}^{\ell} = \text{MLP}(\text{LN}(\hat{\mathbf{z}}^{\ell})) + \hat{\mathbf{z}}^{\ell} \tag{13}$$

$$\hat{\mathbf{z}}^{\ell+1} = \text{SW-MSA}(\text{LN}(\mathbf{z}^{\ell})) + \mathbf{z}^{\ell} \tag{14}$$

$$\mathbf{z}^{\ell+1} = \text{MLP}(\text{LN}(\hat{\mathbf{z}}^{\ell+1})) + \hat{\mathbf{z}}^{\ell+1} \tag{15}$$

Window shifting dilakukan dengan menggeser window sebesar $(\lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor)$ pixels, yang memungkinkan koneksi antar-window pada layer sebelumnya.

2.3.5 Relative Position Bias

Berbeda dengan ViT yang menggunakan absolute positional embeddings, Swin Transformer menggunakan *relative position bias* $B \in \mathbb{R}^{M^2 \times M^2}$ yang ditambahkan ke attention computation:

$$\text{Attention}(Q, K, V) = \text{SoftMax}\left(\frac{QK^T}{\sqrt{d}} + B\right)V \tag{16}$$

Relative position bias terbukti lebih efektif untuk transfer learning dan dapat beradaptasi dengan resolusi gambar yang berbeda [5].

2.3.6 Varian Swin Transformer

Liu et al. [5] memperkenalkan empat varian dengan kompleksitas yang berbeda:

- **Swin-T (Tiny)**: $C = 96$, layer numbers $= \{2, 2, 6, 2\}$ (∼29M parameters)
- **Swin-S (Small)**: $C = 96$, layer numbers $= \{2, 2, 18, 2\}$ (∼50M parameters)
- **Swin-B (Base)**: $C = 128$, layer numbers $= \{2, 2, 18, 2\}$ (∼86M parameters)
- **Swin-L (Large)**: $C = 192$, layer numbers $= \{2, 2, 18, 2\}$ (∼197M parameters)

2.4 Data-efficient Image Transformer (DeiT)

2.4.1 Motivasi dan Kontribusi

Data-efficient Image Transformer (DeiT), yang dikembangkan oleh Touvron et al. [6], mengatasi salah satu keterbatasan utama ViT: kebutuhan akan data pre-training yang sangat besar. Sementara ViT original memerlukan pre-training pada dataset berskala ratusan juta gambar (JFT-300M), DeiT menunjukkan bahwa Vision Transformer dapat dilatih secara efektif hanya menggunakan ImageNet-1k (∼1.3 juta gambar) melalui strategi training yang carefully designed.

Kontribusi utama DeiT meliputi:

1. Strategi data augmentation yang agresif
2. Regularization techniques yang efektif
3. Knowledge distillation melalui distillation token
4. Training procedure yang dioptimalkan

2.4.2 Arsitektur DeiT

Dari segi arsitektur, DeiT menggunakan struktur yang sama dengan ViT dengan beberapa modifikasi minor. Namun, DeiT memperkenalkan konsep *distillation token*, sebuah token tambahan selain class token yang khusus didesain untuk knowledge distillation:

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{x}_{\text{dist}}; \mathbf{x}_p^1\mathbf{E}; ...; \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos} \tag{17}$$

di mana $\mathbf{x}_{\text{dist}}$ adalah distillation token yang berinteraksi dengan patch embeddings dan class token melalui self-attention layers.

2.4.3 Knowledge Distillation Strategy

DeiT menggunakan *distillation* untuk mentransfer pengetahuan dari teacher model (typically a convolutional network seperti RegNetY) ke student model (Vision Transformer). Berbeda dengan traditional distillation yang hanya menggunakan soft labels dari teacher, DeiT menggunakan *hard-label distillation* yang terbukti lebih efektif untuk Vision Transformers.

Fungsi loss untuk training adalah kombinasi dari tiga komponen:

$$\mathcal{L}_{\text{global}} = (1-\lambda)\mathcal{L}_{CE}(\psi(Z_s), y) + \lambda\mathcal{L}_{CE}(\psi(Z_s), y_t) \tag{18}$$

di mana $\mathcal{L}_{CE}$ adalah cross-entropy loss, ψ adalah softmax function, Z_s adalah logits dari student, y adalah ground-truth label, y_t adalah prediction dari teacher, dan λ adalah balancing parameter.

Untuk distillation token, loss tambahan ditambahkan:

$$\mathcal{L}_{\text{distill}} = \mathcal{L}_{CE}(\psi(Z_{\text{dist}}), y_t) \tag{19}$$

Inference dapat menggunakan rata-rata dari kedua classifiers atau hanya salah satu, tergantung pada preferensi accuracy-speed trade-off.

2.4.4 Training Strategy

Keberhasilan DeiT sangat bergantung pada training strategy yang carefully tuned [6]:

1. **Data Augmentation**:
   - Auto-Augment atau RandAugment

- Random Erasing
- CutMix dan Mixup dengan probabilitas tinggi

2. **Regularization**:
   - Stochastic Depth (drop path)
   - Label smoothing
   - Weight decay
3. **Optimization**:
   - AdamW optimizer
   - Cosine learning rate schedule dengan warmup
   - Gradient clipping

2.4.5 Varian DeiT

Touvron et al. [6] memperkenalkan beberapa varian dengan ukuran berbeda:

- **DeiT-Tiny**: 12 layers, hidden size 192, 3 heads (∼5M parameters)
- **DeiT-Small**: 12 layers, hidden size 384, 6 heads (∼22M parameters)
- **DeiT-Base**: 12 layers, hidden size 768, 12 heads (∼86M parameters)

Setiap varian juga memiliki versi distilled (dengan suffix ↑) yang menggunakan distillation token.

2.5 Perbandingan Arsitektur

Tabel 1 merangkum perbedaan kunci antara ketiga arsitektur Vision Transformer yang dibandingkan dalam eksperimen ini.

Tabel 1: Perbandingan Arsitektur Vision Transformer

Aspek	ViT	Swin Transformer	DeiT
Patch Size	16×16	4×4 (initial)	16×16
Attention Scope	Global	Local (windowed)	Global
Architecture Type	Flat (single scale)	Hierarchical (multi-scale)	Flat (single scale)
Complexity	$O(n^2)$	$O(n)$	$O(n^2)$
Position Encoding	Absolute learnable	Relative bias	Absolute learnable
Inductive Bias	Minimal	Moderate (locality)	Minimal
Pre-training Data	JFT-300M / ImageNet-21k	ImageNet-1k/21k	ImageNet-1k
Training Strategy	Standard	Standard	Distillation + augmentation
Special Tokens	[CLS] token	None	[CLS] + [DIST] tokens
Window Size	N/A	7×7 typical	N/A

2.6 Kelebihan dan Kekurangan

Tabel 2 merangkum kelebihan dan kekurangan teoritis dari masing-masing arsitektur berdasarkan desain dan karakteristik mereka.

Tabel 2: Kelebihan dan Kekurangan Masing-masing Arsitektur

Model	Kelebihan	Kekurangan
ViT	• Arsitektur sederhana dan elegan • Global receptive field dari layer pertama • Excellent scalability dengan data • Strong performance pada dataset besar • Mudah diimplementasikan dan dipahami	• Memerlukan data pre-training sangat besar • Kompleksitas kuadratik terhadap jumlah patches • Kurang efisien untuk high-resolution images • Tidak capture hierarchical features • Sulit dilatih dari scratch pada dataset kecil
Swin Transformer	• Kompleksitas linear terhadap image size • Hierarchical features untuk multi-scale tasks • Efisien untuk high-resolution images • Dapat digunakan sebagai backbone umum • Balance antara locality dan global modeling • Lebih baik untuk dense prediction tasks	• Arsitektur lebih kompleks • Window shifting menambah overhead • Implementasi lebih challenging • Relative position bias perlu careful tuning • Sedikit lebih lambat dalam inference per layer
DeiT	• Data-efficient (dapat dilatih pada ImageNet-1k) • Tidak memerlukan pre-training ekstensif • Knowledge distillation meningkatkan performa • Training procedure yang well-documented • Arsitektur sama dengan ViT (familiar) • Cocok untuk practitioners dengan data terbatas	• Memerlukan teacher model yang kuat • Training lebih kompleks (distillation) • Augmentation strategy sangat aggressive • Hyperparameter tuning lebih sensitif • Inference dapat lebih lambat (dua classifiers) • Masih memiliki kompleksitas kuadratik

3 Metodologi

3.1 Dataset

3.2 Preprocessing dan Augmentasi Data

3.3 Konfigurasi Training

Tabel 3: Konfigurasi Training

Parameter	Nilai
Optimizer	Adam
Learning Rate	0.001
Batch Size	32
Epochs	100
Loss Function	Cross Entropy
Learning Rate Scheduler	CosineAnnealingLR
Weight Decay	0.0001

3.4 Library dan Framework

3.5 Spesifikasi Hardware

3.6 Metrik Evaluasi

4 Hasil dan Analisis

4.1 Perbandingan Jumlah Parameter

Tabel 4: Perbandingan Jumlah Parameter

Model	Parameters (M)	FLOPs (G)
Model 1	-	-
Model 2	-	-
Model 3	-	-

4.2 Perbandingan Metrik Performa

Tabel 5: Perbandingan Metrik Performa

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Model 1	-	-	-	-
Model 2	-	-	-	-
Model 3	-	-	-	-

Tabel 6: Perbandingan Waktu Inferensi

Model	Inference Time (ms)	Throughput (images/s)
Model 1	-	-
Model 2	-	-
Model 3	-	-

4.3 Perbandingan Waktu Inferensi

4.4 Visualisasi

4.4.1 Kurva Learning

4.4.2 Confusion Matrix

4.4.3 Contoh Prediksi

4.5 Analisis Mendalam

4.5.1 Analisis Performa Model

4.5.2 Trade-off Akurasi, Parameter, dan Kecepatan

4.5.3 Kesesuaian Model dengan Dataset

5 Kesimpulan dan Saran

5.1 Kesimpulan

5.2 Rekomendasi Model

5.2.1 Untuk Akurasi Maksimal

5.2.2 Untuk Efisiensi Komputasi

5.2.3 Untuk Aplikasi Real-time

5.3 Saran untuk Pengembangan Lebih Lanjut

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 25, 2012.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[4] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations (ICLR)*, 2021. [Online]. Available: https://arxiv.org/abs/2010.11929

[5] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 10 012–10 022. [Online]. Available: https://arxiv.org/abs/2103.14030

[6] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International Conference on Machine Learning (ICML)*. PMLR, 2021, pp. 10 347–10 357. [Online]. Available: https://arxiv.org/abs/2012.12877

[7] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 16 000–16 009. [Online]. Available: https://arxiv.org/abs/2111.06377

Lampiran

A. Source Code

B. Output Training Log

C. Screenshot Hasil Eksperimen