Desain Eksperimen Komputasi

Metodologi Penelitian Kuantitatif

Hendri Karisma, M.T.
Dosen Teknik Informatika, STMIK Tazkia, Bogor, Indonesia
VP Engineering, Jejakin.com
hendri@stmik.tazkia.ac.id — hendri.karisma@jejakin.com

Program Studi Teknik Informatika
STMIK Tazkia, Bogor, Indonesia

2026

Agenda Pertemuan 7

1. Jenis Penelitian Kuantitatif di CS
2. Pembagian Dataset: *Train/Val/Test*
3. *K-Fold Cross Validation*
4. *Stratified K-Fold* & LOOCV
5. Isolasi *Environment* & *Reproducibility*
6. *Reproducibility Checklist*
7. *Stress Testing* & Skalabilitas
8. Instrumen Penelitian di CS
9. *Flowchart* Desain Eksperimen
10. Contoh Desain Eksperimen (3 Domain)

Tujuan Pembelajaran

Mahasiswa mampu merancang skenario eksperimen yang **valid** dan **reproducible**, serta mendokumentasikan instrumen dan desain secara lengkap.

4 Jenis Penelitian Kuantitatif di Ilmu Komputer

1. Eksperimental Murni

- Menguji algoritma/metode **baru**
- Kontrol variabel ketat
- Contoh: mengusulkan arsitektur CNN baru

2. Komparatif

- Membandingkan 2+ metode yang **sudah ada**
- *Fair comparison* (*benchmark*)
- Contoh: TensorFlow vs PyTorch vs JAX

3. Pengembangan Sistem + Evaluasi

- Membangun sistem utuh, lalu mengevaluasi
- Metrik: performa + *usability*
- Contoh: aplikasi deteksi penyakit tanaman

4. Simulasi

- Memodelkan fenomena nyata di komputer
- Analisis *what-if*
- Contoh: simulasi protokol di NS-3

Contoh Konkret Setiap Jenis Penelitian

Jenis	**Contoh Judul Penelitian**
Eksperimental	"FastDetectNet: Arsitektur Baru untuk Deteksi Objek Resolusi Rendah" "Algoritma Enkripsi Ringan untuk Perangkat IoT"
Komparatif	"Perbandingan MongoDB, Cassandra, dan Redis untuk *Mixed Workload*" "Evaluasi *Feature Selection*: Chi-Square vs *Information Gain*"
Pengembangan	"Sistem Rekomendasi Buku Perpustakaan Berbasis *Collaborative Filtering*" "*Dashboard Real-Time Monitoring* Jaringan Kampus"
Simulasi	"Simulasi Protokol AODV vs DSR pada *Wireless Sensor Network*" "Simulasi Penyebaran *Malware* dengan Model Epidemiologi SIR"

Penting

Jenis penelitian menentukan **metode evaluasi**, **desain eksperimen**, dan **cara pelaporan hasil**.

Pembagian Dataset: *Training*, *Validation*, *Testing*

Tiga Partisi Wajib

- **Training Set** — Melatih model
- **Validation Set** — *Tuning hyperparameter*, memilih model terbaik
- **Test Set** — Evaluasi akhir (**hanya sekali!**)

Aturan Emas

Test set **TIDAK BOLEH** digunakan selama pengembangan model. Jika dilanggar = *data leakage* = hasil bias!

Rasio	Cocok Untuk
70:15:15	Data sedang (1K–50K sampel)
80:10:10	Data besar (¿50K sampel)
60:20:20	Data kecil (¡1K sampel)

Default

Jika ragu, gunakan **70:15:15**.

Tips Penting Pembagian Dataset

1. **Selalu catat random seed** yang digunakan saat melakukan pembagian.
2. **Data time-series:** JANGAN gunakan *random split*. Gunakan *temporal split* — data lama untuk *training*, data baru untuk *testing*.
3. **Data berkelompok (grouped data):** Pastikan satu kelompok tidak tersebar ke *training* dan *testing* secara bersamaan (*group-aware split*).
4. **Data imbalanced:** Gunakan *stratified split* agar proporsi kelas sama di setiap partisi.

Contoh Kesalahan Umum

Mahasiswa menggunakan *random split* pada data harga saham — data masa depan "bocor" ke *training set*, menghasilkan akurasi yang tidak realistis (95%+). Seharusnya menggunakan *temporal split*.

K-Fold *Cross Validation*: Konsep dan Diagram

Cara Kerja (K=5):

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Iter 1	**TEST**	Train	Train	Train	Train
Iter 2	Train	**TEST**	Train	Train	Train
Iter 3	Train	Train	**TEST**	Train	Train
Iter 4	Train	Train	Train	**TEST**	Train
Iter 5	Train	Train	Train	Train	**TEST**

Hasil akhir = **rata-rata** $\pm$ **std** dari K iterasi.

Mengapa Lebih Baik dari *Single Split*?

- Setiap data pernah menjadi *test*
- Estimasi performa lebih stabil
- Mengurangi bias akibat pembagian data

Cara Melaporkan yang Benar

"Accuracy: **92.3%** $\pm$ **1.2%**"

Bukan hanya "Accuracy: 92.3%"

Memilih Nilai K: K=5 vs K=10

Aspek	**K=5**	**K=10**
Bias estimasi	Lebih tinggi	Lebih rendah
Variansi estimasi	Lebih rendah	Lebih tinggi
Waktu komputasi	Cepat ($5\times$)	Lambat ($10\times$)
Cocok untuk	Dataset besar, model lambat (*deep learning*)	Dataset kecil, model cepat (kNN, SVM)

Rekomendasi Praktis

- **K=5**: Pilihan *default* yang baik untuk kebanyakan kasus
- **K=10**: Jika dataset kecil dan model cepat dilatih
- **K=3**: Hanya jika sangat terbatas waktu komputasi

Catatan

Selalu laporkan nilai K yang digunakan beserta alasannya di Bab 3 skripsi.

Stratified K-Fold dan LOOCV

Stratified K-Fold:

- Memastikan **proporsi kelas sama** di setiap *fold*
- **Wajib** untuk data *imbalanced* (misal: 95% Normal, 5% Fraud)
- *Best practice* untuk semua tugas klasifikasi

LOOCV (Leave-One-Out):

- K = N (jumlah sampel)
- 1 sampel jadi *test*, sisanya *training*

Kelebihan	Kekurangan
Bias sangat rendah	Sangat mahal (N×)
Deterministik	Variansi tinggi
Semua data terpakai	Tidak praktis jika N ¿ 10.000

```
from sklearn.model_selection import StratifiedKFold

skf = StratifiedKFold(n_splits=5,
    shuffle=True, random_state=42
for train_idx, test_idx in skf.sp
    X_train = X[train_idx]
    X_test  = X[test_idx]
```

Kapan Pakai LOOCV?

Dataset sangat kecil (N ¡ 100) dan model cepat (kNN, regresi linear).

Isolasi *Environment*: Docker & *Virtual Machine*

Docker Container:

```
FROM python:3.10-slim
WORKDIR /experiment
COPY requirements.txt .
RUN pip install -r requirements.txt
COPY . .
ENV PYTHONHASHSEED=42
CMD ["python", "run_experiment.py"]
```

- Siapapun bisa replikasi: `docker build` + `docker run`
- Menghilangkan masalah "*works on my machine*"
- Versi *library* dan OS terkunci

Kapan Docker? Kapan VM?

Docker	VM
Eksperimen ML	Uji keamanan
Web service	Kernel OS tertentu
Ringan, cepat	Konfigurasi jaringan
Portabel	Isolasi penuh

Tools VM: VirtualBox, VMware, QEMU, AWS EC2, Google Compute Engine.

Isolasi *Environment*: *Seed Random & Version Pinning*

Pengaturan Seed Random:

```
import random, numpy as np, torch

SEED = 42
random.seed(SEED)
np.random.seed(SEED)
torch.manual_seed(SEED)
torch.cuda.manual_seed_all(SEED)
torch.backends.cudnn.deterministic
torch.backends.cudnn.benchmark = F
```

Version Pinning:

```
# requirements.txt
numpy==1.24.3
pandas==2.0.1
scikit-learn==1.2.2
torch==2.0.1
matplotlib==3.7.1
```

Peringatan

Bahkan dengan *seed* sama, hasil bisa berbeda antar GPU atau versi *library*! Maka *seed* harus dikombinasikan dengan isolasi *environment*.

Best Practice

- Jalankan: `pip freeze > requirements.txt`
- Atau gunakan `environment.yml` (Conda)
- Simpan di repositori Git bersama kode

Reproducibility Checklist

Sebelum Eksperimen

- ☐ *Random seed* ditetapkan dan dicatat
- ☐ Versi semua *library* dikunci (`requirements.txt`)
- ☐ *Environment* diisolasi (Docker / venv)
- ☐ Spesifikasi *hardware* didokumentasikan
- ☐ Dataset disimpan dengan versi yang jelas

Selama & Sesudah Eksperimen

- ☐ Semua *hyperparameter* dicatat dalam tabel
- ☐ Kode sumber di-*version control* (Git)
- ☐ Hasil mentah (*raw results*) disimpan
- ☐ Skrip eksperimen otomatis (*runnable*)
- ☐ Instruksi replikasi tersedia di README

Rumus *Reproducibility*

Seed + Docker + Version Pinning + Git + Dokumentasi = Reproducible Research

Stress Testing dalam Konteks Riset

Definisi

Pengujian metode/sistem dalam **kondisi ekstrem** untuk menemukan titik lemah (*breaking point*).

Mengapa penting?

- *Reviewer* jurnal **pasti** bertanya: "Bagaimana jika datanya *noisy*?"
- Menunjukkan peneliti memahami **limitasi** metodenya
- Meningkatkan kredibilitas dan kualitas *paper* secara signifikan

Ukuran Dataset	**Tujuan**	**Yang Diukur**
100 sampel	Performa pada data sangat kecil	Akurasi, waktu, memori
1.000 sampel	Performa tipikal	Akurasi, waktu, memori
100.000 sampel	Performa pada data besar	Waktu: linear? kuadratik?
1.000.000 sampel	Batas skalabilitas	Apakah masih berjalan?

Scalability Analysis & Adversarial Testing

3 Dimensi Skalabilitas:

1. **Data Scalability** — Performa vs jumlah data
2. **Feature Scalability** — Performa vs jumlah fitur/dimensi
3. **Computational Scalability** — Performa vs jumlah *core*/*node*

Cara Melaporkan

Buat *log-log plot*: ukuran input (x) vs waktu (y). Tentukan kompleksitas: O(n)? O(n log n)? $O(n^2)$?

Adversarial Testing:

- Input yang **sengaja dirancang** untuk mengeksploitasi kelemahan metode

Domain	Contoh
Gambar	*FGSM/PGD attack*, tambah *noise*
NLP	Sinonim, *typo*, parafrase
Jaringan	*Malformed packets*
Optimasi	*Worst-case input*

Instrumen Penelitian: Spesifikasi *Hardware & Software*

WAJIB dilaporkan secara detail di Bab 3 agar eksperimen bisa direproduksi.

Spesifikasi Hardware:

Komponen	Spesifikasi
CPU	AMD Ryzen 9 5900X (12C/24T, 3.7 GHz)
GPU	NVIDIA RTX 3090 (24 GB VRAM)
RAM	64 GB DDR4 3600 MHz
Storage	2 TB NVMe SSD
Jaringan	1 Gbps Ethernet

Software Stack:

Komponen	Versi
OS	Ubuntu 22.04 LTS
Bahasa	Python 3.10.12
Framework	PyTorch 2.0.1 + CUDA 11.8
Library	scikit-learn 1.2.2
Container	Docker 24.0.5
VCS	Git 2.40.1

Jika Menggunakan *Cloud* (Google Colab, AWS)

Catat tipe *instance* (misal: `p3.2xlarge`), versi *runtime*, dan *region*.

Instrumen Penelitian: Dataset & Parameter Konfigurasi

Deskripsi Dataset (checklist):

1. **Nama & sumber:** Kaggle, UCI, *self-collected*
2. **Ukuran:** jumlah sampel, fitur, ukuran file
3. **Distribusi kelas:** seimbang / *imbalanced*?
4. **Tipe data:** numerik, teks, gambar, *time-series*?
5. **Preprocessing:** *missing values*? normalisasi?
6. **Lisensi:** CC-BY, MIT, dsb.

Tabel Parameter Konfigurasi:

Parameter	Nilai
Learning rate	0.001
Batch size	32
Epoch	100
Optimizer	Adam
Loss function	CrossEntropyLoss
Dropout rate	0.5
Random seed	42
Early stopping	10 epoch
LR *scheduler*	CosineAnnealing

Flowchart Desain Eksperimen

Komponen Wajib dalam Flowchart:

1. **Input:** Dataset mentah, parameter awal
2. **Preprocessing:** Pembersihan, normalisasi, *feature engineering*
3. **Pembagian Data:** *Split / cross-validation*
4. **Pelatihan:** Model *baseline* + *proposed method*
5. **Evaluasi:** Metrik, perbandingan antar metode
6. **Output:** Hasil akhir, model terbaik, kesimpulan

Simbol Standar:

2. Oval = Mulai / Selesai

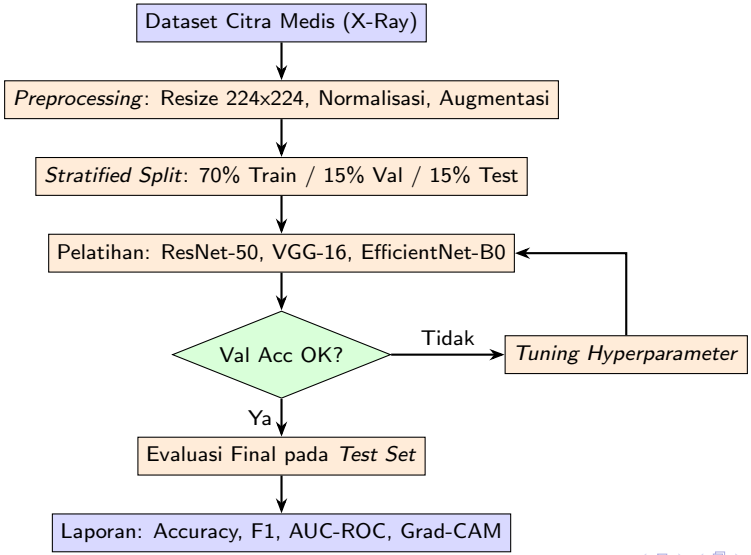
Tools Rekomendasi:

Tool	Kelebihan
draw.io	Gratis, *offline*, ekspor PDF
Lucidchart	Kolaborasi *real-time*
PlantUML	*Code-based*, integrasi LaTeX
Mermaid	Ringan, *embed* di GitHub
Visio	Standar industri, fitur lengkap

Saran untuk Skripsi

Gunakan **draw.io** (gratis). Gunakan **PlantUML** jika ingin *version control* pada

Contoh *Flowchart*: Eksperimen Klasifikasi ML



Contoh Desain Lengkap 1: Klasifikasi *Spam* Email

Judul: Perbandingan *Ensemble Learning* untuk Deteksi *Spam* Email Berbahasa Indonesia

Variabel:

- *Independen:* Metode (RF, XGBoost, LightGBM) $\times$ *Feature* (TF-IDF, Word2Vec)
- *Dependen:* F1-score, Accuracy, waktu *training*
- Kontrol: *Seed* = 42, 5-fold *Stratified CV*

Dataset: 10.000 email (6K *ham*, 4K *spam*)

Skenario Eksperimen:

1. *Preprocessing*: *lowercase*, *stopword removal*, *stemming* (Sastrawi)
2. *Feature extraction*: TF-IDF (5000) + Word2Vec (dim=100)
3. 3 model $\times$ 2 fitur = **6 kombinasi**
4. 5-fold *Stratified Cross Validation*
5. Uji statistik: *Friedman* + *Nemenyi post-hoc*

Stack: Python 3.10, scikit-learn, XGBoost, LightGBM

Contoh Desain Lengkap 2: Performa Jaringan IoT

Judul: Analisis Performa MQTT vs CoAP pada Jaringan *Bandwidth* Terbatas

Variabel:

- *Independen:* Protokol (MQTT QoS 0/1/2, CoAP CON/NON)
- *Dependen: Latency* (ms), *throughput* (msg/s), *packet loss* (%)
- Kontrol: *Payload* 256 bytes, 1 msg/detik

Hardware: Raspberry Pi 4, Laptop, Switch Gigabit

Skenario Eksperimen:

1. *Bandwidth*: 1M, 512K, 256K, 128K, 64K bps
2. *Packet loss* simulasi: 0%, 1%, 5%, 10%, 20%
3. Durasi: 10 menit/skenario $\times$ 5 repetisi
4. Total: $5 \times 5 \times 5 \times 5 =$ **625 run**

Uji Statistik: ANOVA + *Post-hoc* Tukey HSD
Software: Mosquitto 2.0, libcoap 4.3, Wireshark 4.0

Contoh Desain Lengkap 3: Optimasi Penjadwalan

Judul: *Hybrid* GA-SA untuk Penjadwalan Mata Kuliah dengan *Constraint* Kompleks

Variabel:

- *Independen:* Algoritma (GA, SA, GA-SA), populasi (50, 100, 200), *cooling* (linear, exp, log)
- *Dependen: Fitness* (*constraint violation*), waktu konvergensi
- Kontrol: 30 matkul, 10 ruang, 5 hari, 8 slot, maks 10.000 iterasi

Skenario Eksperimen:

1. Implementasi 3 algoritma, *interface* sama
2. **30 independent runs** per konfigurasi
3. Rekam: *best*, *mean*, *std fitness*
4. Uji: *Wilcoxon signed-rank* + *Kruskal-Wallis*
5. Analisis: *convergence plot*, *box plot*, *sensitivity analysis*

Stack: Java 17 (OpenJDK), JMetal 5.11

Catatan Penting

Untuk algoritma stokastik, **WAJIB** menjalankan minimal 30 kali dan melaporkan rata-rata $\pm$ standar deviasi, bukan hanya hasil terbaik saja.

Ringkasan Pertemuan 7

Aspek	**Poin Kunci**
Jenis Penelitian	Eksperimental, Komparatif, Pengembangan Sistem, Simulasi
Pembagian Dataset	70:15:15 (*default*), 80:10:10 (data besar), 60:20:20 (data kecil)
Cross Validation	K-Fold (K=5 atau K=10), *Stratified* K-Fold, LOOCV
Reproducibility	Docker + *seed random* + *version pinning* + Git + dokumentasi
Stress Testing	Berbagai ukuran data, *adversarial testing*, *scalability analysis*
Instrumen	*Hardware*, *software stack*, dataset, parameter konfigurasi
Flowchart	Wajib ada di Bab 3 skripsi; gunakan draw.io atau PlantUML

Tugas Pertemuan 7

Tugas Individu

Buatlah **desain eksperimen lengkap** untuk topik skripsi/penelitian Anda yang mencakup:

1. Jenis penelitian yang dipilih beserta **alasannya**
2. Variabel penelitian (independen, dependen, kontrol)
3. Skenario pembagian dataset dan metode validasi (*cross-validation*)
4. Spesifikasi instrumen (*hardware*, *software*, dataset)
5. Tabel parameter konfigurasi
6. *Flowchart* desain eksperimen (menggunakan draw.io atau tool lain)
7. Rencana *stress testing* dan analisis skalabilitas

Ketentuan

Format: PDF, maksimal 5 halaman.
Deadline: Sebelum Pertemuan 8.