Lab 1 - Eksperimen Chunking (2/3)

2_Machine_Learning_dan_Deep_Learning.pdf | Python (pdfplumber + fungsi potong)

Langkah 1. Membaca PDF dan memeriksa teks

File yang dibaca: 2_Machine_Learning_dan_Deep_Learning.pdf. Berikut 500 karakter pertama hasil baca_pdf():

```
Machine Learning dan Deep
Learning
Cara mesin belajar dari data
1. Machine Learning
Machine Learning (ML) adalah pendekatan AI di mana komputer belajar pola dari data untuk
membuat prediksi atau keputusan, tanpa aturan yang ditulis secara manual. Semakin
berkualitas dan banyak data, umumnya semakin baik performa model.
Alur kerja umum ML: (1) mengumpulkan data, (2) membersihkan dan menyiapkan data, (3)
memilih model, (4) melatih model, (5) mengevaluasi, dan (6) menerapkan serta memantau
model.
2
```

Pemeriksaan	Hasil
File	2_Machine_Learning_dan_Deep_Learning.pdf
Total karakter	2732
Karakter rusak (U+FFFD)	0
Sisa kode glyph (cid:)	11
Persentase huruf	80%

Temuan kualitas teks: 11 kode glyph '(cid:N)' (simbol gagal diterjemahkan); 2 baris nomor halaman 'Halaman N' yang menyelip di tengah isi (sudah dibuang sebelum pemotongan).

Langkah 2. Jumlah chunk tiap pengaturan

Fungsi potong(teks, ukuran, overlap) memakai jendela geser per karakter; langkah = ukuran x (1 - overlap).

Ukuran (karakter)	overlap 10%	overlap 15%	overlap 20%
500	6	7	7
1000	3	4	4
1500	2	2	3

Tabel 1. Jumlah chunk untuk 3 ukuran x 3 overlap. Makin besar ukuran, makin sedikit chunk; makin besar overlap, makin banyak chunk.

Langkah 3. Tiga chunk acak per ukuran (overlap 15%, seed 42)

Ukuran	Jumlah chunk	Awal di tengah kalimat	Akhir di tengah kalimat	Memotong tabel
500	7	6/7 (86%)	6/7 (86%)	3/7 (43%)
1000	4	3/4 (75%)	3/4 (75%)	2/4 (50%)
1500	2	1/2 (50%)	1/2 (50%)	0/2 (0%)

Tabel 2. Statistik seluruh chunk pada tiap ukuran (bukan hanya yang dicuplik).

Ukuran 500 karakter

Chunk #1 (karakter 0-500) | awal di tengah kalimat: tidak | akhir di tengah kalimat: ya | memotong tabel: tidak

```
Machine Learning dan Deep
Learning
Cara mesin belajar dari data
1. Machine Learning
Machine Learning (ML) adalah pendekatan AI di mana komputer belajar pola dari data untuk
membuat prediksi atau keputusan, tanpa aturan yang ditulis secara manual. Semakin
berkualitas dan banyak data, umumnya semakin baik performa model.
Alur kerja umum ML: (1) mengumpulkan data, (2) membersihkan dan menyiapkan data, (3)
memilih model, (4) melatih model, (5) mengevaluasi, dan (6) menerapkan serta memantau
model.
2
```

Chunk #6 (karakter 2125-2625) | awal di tengah kalimat: ya | akhir di tengah kalimat: ya | memotong tabel: tidak

```
memerlukan GPU dan energi yang signifikan.
(cid:127) Interpretabilitas: keputusan model deep learning sulit dijelaskan (black box).
6. Metrik Evaluasi
Beberapa metrik yang sering dipakai: akurasi, presisi, recall, F1-score untuk klasifikasi; serta
MAE dan RMSE untuk regresi. Pemilihan metrik harus sesuai tujuan; misalnya pada deteksi
penyakit, recall yang tinggi sering lebih penting daripada akurasi semata.
7. Kesimpulan
Machine learning dan deep learning menjadi tulang punggung AI modern. Keber
```

Chunk #7 (karakter 2550-2732) | awal di tengah kalimat: ya | akhir di tengah kalimat: tidak | memotong tabel: tidak

```
Machine learning dan deep learning menjadi tulang punggung AI modern. Keberhasilan
penerapannya bergantung pada data yang baik, pemilihan model yang tepat, dan evaluasi
yang cermat.
```

Ukuran 1000 karakter

Chunk #1 (karakter 0-1000) | awal di tengah kalimat: tidak | akhir di tengah kalimat: ya | memotong tabel: tidak

```
Machine Learning dan Deep
Learning
Cara mesin belajar dari data
1. Machine Learning
Machine Learning (ML) adalah pendekatan AI di mana komputer belajar pola dari data untuk
membuat prediksi atau keputusan, tanpa aturan yang ditulis secara manual. Semakin
berkualitas dan banyak data, umumnya semakin baik performa model.
Alur kerja umum ML: (1) mengumpulkan data, (2) membersihkan dan menyiapkan data, (3)
memilih model, (4) melatih model, (5) mengevaluasi, dan (6) menerapkan serta memantau
model.
2. Tiga Paradigma Pembelajaran
Paradigma Penjelasan Contoh
Supervised Learning Belajar dari data berlabel (input dan Klasifikasi email spam, prediksi
jawaban benar). harga rumah.
Unsupervised Menemukan struktur tersembunyi pada Segmentasi pelanggan, deteksi
Learning data tanpa label. anomali.
Reinforcement Agen belajar lewat percobaan dengan Game, robot, optimasi strategi.
Learning hadiah dan hukuman.
3. Algoritma yang Umum Digunakan
(cid:127) Regresi linear dan regresi logistik
(cid:127) Pohon k
```

Chunk #3 (karakter 1700-2700) | awal di tengah kalimat: ya | akhir di tengah kalimat: ya | memotong tabel: ya

```
ar; unggul pada teks, gambar, dan
audio.
GAN dan Diffusion Pembuatan gambar dan konten sintetis.
5. Tantangan dalam Pelatihan Model
(cid:127) Overfitting: model terlalu hafal data latih sehingga buruk pada data baru.
(cid:127) Underfitting: model terlalu sederhana untuk menangkap pola.
(cid:127) Kualitas data: data yang bias, kotor, atau tidak seimbang menurunkan kualitas model.
(cid:127) Kebutuhan komputasi: model besar memerlukan GPU dan energi yang signifikan.
(cid:127) Interpretabilitas: keputusan model deep learning sulit dijelaskan (black box).
6. Metrik Evaluasi
Beberapa metrik yang sering dipakai: akurasi, presisi, recall, F1-score untuk klasifikasi; serta
MAE dan RMSE untuk regresi. Pemilihan metrik harus sesuai tujuan; misalnya pada deteksi
penyakit, recall yang tinggi sering lebih penting daripada akurasi semata.
```

```
7. Kesimpulan
Machine learning dan deep learning menjadi tulang punggung AI modern. Keberhasilan
penerapannya bergantung pada data yang baik, pemilihan model yang t
```

Chunk #4 (karakter 2550-2732) | awal di tengah kalimat: ya | akhir di tengah kalimat: tidak | memotong tabel: tidak

```
Machine learning dan deep learning menjadi tulang punggung AI modern. Keberhasilan
penerapannya bergantung pada data yang baik, pemilihan model yang tepat, dan evaluasi
yang cermat.
```

Ukuran 1500 karakter

Chunk #1 (karakter 0-1500) | awal di tengah kalimat: tidak | akhir di tengah kalimat: ya | memotong tabel: tidak

```
Machine Learning dan Deep
Learning
Cara mesin belajar dari data
1. Machine Learning
Machine Learning (ML) adalah pendekatan AI di mana komputer belajar pola dari data untuk
membuat prediksi atau keputusan, tanpa aturan yang ditulis secara manual. Semakin
berkualitas dan banyak data, umumnya semakin baik performa model.
Alur kerja umum ML: (1) mengumpulkan data, (2) membersihkan dan menyiapkan data, (3)
memilih model, (4) melatih model, (5) mengevaluasi, dan (6) menerapkan serta memantau
model.
2. Tiga Paradigma Pembelajaran
Paradigma Penjelasan Contoh
Supervised Learning Belajar dari data berlabel (input dan Klasifikasi email spam, prediksi
jawaban benar). harga rumah.
Unsupervised Menemukan struktur tersembunyi pada Segmentasi pelanggan, deteksi
Learning data tanpa label. anomali.
Reinforcement Agen belajar lewat percobaan dengan Game, robot, optimasi strategi.
Learning hadiah dan hukuman.
3. Algoritma yang Umum Digunakan
(cid:127) Regresi linear dan regresi logistik
(cid:127) Pohon keputusan (decision tree) dan random forest
(cid:127) Support Vector Machine (SVM)
(cid:127) K-Nearest Neighbors (KNN)
(cid:127) K-Means untuk pengelompokan (clustering)
(cid:127) Jaringan saraf tiruan (neural network)
4. Deep Learning
Deep Learning adalah subbidang ML yang menggunakan jaringan saraf tiruan dengan
banyak lapisan. Setiap lapisan mempelajari representasi data yang makin abstrak; misalnya
pada gambar, lapisan awal mengenali tepi, lapisan tengah mengenali bentuk, dan lapisan
akhir me
```

Chunk #2 (karakter 1275-2732) | awal di tengah kalimat: ya | akhir di tengah kalimat: tidak | memotong tabel: tidak

```
enggunakan jaringan saraf tiruan dengan
banyak lapisan. Setiap lapisan mempelajari representasi data yang makin abstrak; misalnya
pada gambar, lapisan awal mengenali tepi, lapisan tengah mengenali bentuk, dan lapisan
akhir mengenali objek utuh.
Arsitektur Kegunaan Utama
CNN (Convolutional Neural Network) Pengolahan gambar dan video.
RNN / LSTM Data berurutan seperti teks dan deret waktu.
Transformer Dasar model bahasa besar; unggul pada teks, gambar, dan
audio.
GAN dan Diffusion Pembuatan gambar dan konten sintetis.
5. Tantangan dalam Pelatihan Model
(cid:127) Overfitting: model terlalu hafal data latih sehingga buruk pada data baru.
(cid:127) Underfitting: model terlalu sederhana untuk menangkap pola.
(cid:127) Kualitas data: data yang bias, kotor, atau tidak seimbang menurunkan kualitas model.
(cid:127) Kebutuhan komputasi: model besar memerlukan GPU dan energi yang signifikan.
(cid:127) Interpretabilitas: keputusan model deep learning sulit dijelaskan (black box).
6. Metrik Evaluasi
Beberapa metrik yang sering dipakai: akurasi, presisi, recall, F1-score untuk klasifikasi; serta
MAE dan RMSE untuk regresi. Pemilihan metrik harus sesuai tujuan; misalnya pada deteksi
penyakit, recall yang tinggi sering lebih penting daripada akurasi semata.
7. Kesimpulan
Machine learning dan deep learning menjadi tulang punggung AI modern. Keberhasilan
penerapannya bergantung pada data yang baik, pemilihan model yang tepat, dan evaluasi
yang cermat.
```

Kesimpulan

Pemotongan per karakter tidak mengenal batas kalimat maupun tabel. Hasil seluruh chunk (overlap 15%) - ukuran 500: awal di tengah kalimat 6/7 (86%), akhir di tengah kalimat 6/7 (86%), memotong tabel 3/7 (43%); ukuran 1000: awal di tengah kalimat 3/4 (75%), akhir di tengah kalimat 3/4 (75%), memotong tabel 2/4 (50%); ukuran 1500: awal di tengah kalimat 1/2 (50%), akhir di tengah kalimat 1/2 (50%), memotong tabel 0/2 (0%). Proporsi chunk yang berakhir di tengah kalimat berbeda antar ukuran (50% sampai 86%). Chunk yang dimulai di tengah kalimat berkisar 50% sampai 86%; chunk pertama selalu dimulai bersih. Catatan: dokumen ini pendek sehingga ukuran 500, 1000, 1500 hanya menghasilkan kurang dari 10 chunk; persentasenya mudah bergeser oleh satu chunk saja. Dari 2 tabel, pemotongan tabel terjadi pada ukuran 500, 1000. Perbaikan yang layak dicoba: pemotongan berbasis kalimat/paragraf dan memperlakukan tabel sebagai satu chunk utuh.

Lampiran: kode lab1_chunking.py

```
"""Lab 1 - Eksperimen chunking (15 menit).

1. Baca satu PDF dengan baca_pdf() dan cetak 500 karakter pertama.
2. Potong dengan potong() memakai tiga ukuran (500, 1000, 1500) dan overlap 10-20%.
3. Lihat 3 chunk acak tiap ukuran: terpotong di tengah kalimat atau tabel?

Versi ini dijalankan untuk ketiga PDF AI. Satu laporan PDF dibuat per dokumen:
    Lab1_Eksperimen_Chunking_1.pdf, _2.pdf, _3.pdf
Jalankan semua : python lab1_chunking.py
Jalankan satu  : python lab1_chunking.py 2_Machine_Learning_dan_Deep_Learning.pdf
"""
import os
import random
import re
import sys

sys.path.insert(0, os.path.dirname(os.path.abspath(__file__)))
from laporan_utils import Laporan
from rag_utils import FOLDER_DOKUMEN, baca_pdf, norm, potong, potong_dengan_posisi

BASE = os.path.dirname(os.path.abspath(__file__))

# Ketiga PDF yang dipakai. Letakkan di FOLDER_DOKUMEN, di folder skrip ini, atau di folder kerja.
DAFTAR_PDF = [
    "1_Pengenalan_Kecerdasan_Buatan.pdf",
    "2_Machine_Learning_dan_Deep_Learning.pdf",
    "3_Etika_Risiko_dan_Masa_Depan_AI.pdf",
]

# PDF ini tidak memakai penanda "Tabel N." / "Sumber:", jadi wilayah tabel ditentukan lewat
# teks baris judul kolom (awal) dan teks akhir baris terakhir tabel (akhir).
PENANDA_TABEL = {
    "1_Pengenalan_Kecerdasan_Buatan.pdf": [
        ("Periode Peristiwa Penting", "secara luas."),
        ("Sektor Contoh Penerapan", "pembuatan musik dan gambar."),
    ],
    "2_Machine_Learning_dan_Deep_Learning.pdf": [
        ("Paradigma Penjelasan Contoh", "hadiah dan hukuman."),
        ("Arsitektur Kegunaan Utama", "konten sintetis."),
    ],
    "3_Etika_Risiko_dan_Masa_Depan_AI.pdf": [
        ("Isu Penjelasan", "pelatihan."),
    ],
}

UKURAN = [500, 1000, 1500]
OVERLAP = [0.10, 0.15, 0.20]
OVERLAP_UTAMA = 0.15
SEED = 42


# ------------------------------------------------------------------------------ utilitas

def cari_pdf(nama):
    """Cari file PDF di FOLDER_DOKUMEN, folder skrip, lalu folder kerja."""
    kandidat = [os.path.join(d, nama) for d in (FOLDER_DOKUMEN, BASE, os.getcwd())]
    for k in kandidat:
        if os.path.isfile(k):
            return k
    raise FileNotFoundError(f"{nama} tidak ditemukan. Sudah dicari di:\n  " + "\n  ".join(kandidat))


def bersihkan_footer(teks):
    """Buang baris nomor halaman ('Halaman 1') yang ikut terbaca di tengah isi.
    Dilakukan sebelum pemotongan, sehingga semua posisi karakter konsisten."""
    baru, n = re.subn(r"^[ \t]*Halaman \d+[ \t]*\n?", "", teks, flags=re.MULTILINE)
    return baru, n


def wilayah_tabel(teks, penanda):
    """Rentang karakter tiap tabel: dari teks judul kolom sampai akhir baris terakhir tabel.
    Penanda yang tidak ditemukan dilaporkan, bukan diam-diam dilewati."""
    hasil = []
    for awal, akhir in penanda:
        a = teks.find(awal)
        if a < 0:
            print(f"  [PERINGATAN] awal tabel tidak ditemukan: {awal!r}")
            continue
        z = teks.find(akhir, a)
        if z < 0:
```

```
            print(f"  [PERINGATAN] akhir tabel tidak ditemukan: {akhir!r}")
            continue
        hasil.append((a, z + len(akhir)))
    return hasil


def batas_bersih(teks, pos):
    """True bila posisi `pos` jatuh di batas satuan teks yang wajar (bukan di tengah kalimat).

    Newline TIDAK otomatis dianggap batas, karena PDF memecah baris di tengah kalimat.
    Batas bersih bila: awal/akhir dokumen; ATAU tidak memotong kata dan teks sebelumnya
    berakhir . ! ? : ; ATAU bagian berikutnya diawali butir daftar ('- ') / judul bernomor
    ('3. ') sehingga teks sebelumnya adalah butir atau judul yang sudah selesai."""
    if pos <= 0 or pos >= len(teks):
        return True
    kiri, kanan = teks[:pos], teks[pos:]
    if kiri[-1] not in " \n" and kanan[0] not in " \n":
        return False                                    # memotong di tengah kata
    kiri_s, kanan_s = kiri.rstrip(), kanan.lstrip()
    if not kiri_s or not kanan_s:
        return True
    kata_akhir = kiri_s.split()[-1]
    if re.fullmatch(r"\d+\.", kata_akhir):
        return False                                    # nomor judul ("3.") yang terpisah dari judulnya
    if kiri_s[-1] in ".!?:":
        return True
    return bool(re.match(r"(-\s|\d+\.\s)", kanan_s))


def awal_di_tengah_kalimat(teks, mulai):
    return not batas_bersih(teks, mulai)


def akhir_di_tengah_kalimat(teks, akhir):
    return not batas_bersih(teks, akhir)


def memotong_tabel(wil, mulai, akhir):
    """True bila chunk beririsan dengan tabel tetapi tidak memuat seluruh tabel."""
    for a, b in wil:
        beririsan = mulai < b and akhir > a
        utuh = mulai <= a and akhir >= b
        if beririsan and not utuh:
            return True
    return False


def pct(x, n):
    return f"{x}/{n} ({x / n:.0%})" if n else "0/0"


# ------------------------------------------------------------------------------ narasi dari data

def catatan_kualitas(k, n_footer):
    """Penilaian kualitas teks yang dihitung dari angka, bukan ditulis tetap."""
    masalah = []
    if k["Karakter rusak (U+FFFD)"]:
        masalah.append(f"{k['Karakter rusak (U+FFFD)']} karakter rusak (U+FFFD)")
    if k["Sisa kode glyph (cid:)"]:
        masalah.append(f"{k['Sisa kode glyph (cid:)']} kode glyph '(cid:N)' (simbol gagal
    diterjemahkan)")
    if n_footer:
        masalah.append(f"{n_footer} baris nomor halaman 'Halaman N' yang menyelip di tengah isi "
                       "(sudah dibuang sebelum pemotongan)")
    if not masalah:
        return "Teks terbaca bersih: tidak ada karakter rusak, kode glyph, atau sisa footer."
    return "Temuan kualitas teks: " + "; ".join(masalah) + "."


def buat_kesimpulan(statistik_angka, jumlah_tabel):
    """Kesimpulan yang seluruh klaimnya dihitung dari statistik, jadi benar untuk PDF apa pun."""
    rincian = "; ".join(
        f"ukuran {u}: awal di tengah kalimat {pct(a, n)}, akhir di tengah kalimat {pct(z, n)}, "
        f"memotong tabel {pct(t, n)}" for u, n, a, z, t in statistik_angka)
    kal = [f"Pemotongan per karakter tidak mengenal batas kalimat maupun tabel. Hasil seluruh chunk "
           f"(overlap {int(OVERLAP_UTAMA * 100)}%) - {rincian}."]

    pz = [z / n for _, n, _, z, _ in statistik_angka if n]
    pa = [a / n for _, n, a, _, _ in statistik_angka if n]
    if pz:
        if min(pz) > 0.5:
            kal.append("Pada setiap ukuran, mayoritas chunk berakhir di tengah kalimat.")
        elif max(pz) < 0.5:
```

```
            kal.append("Pada setiap ukuran, kurang dari separuh chunk berakhir di tengah kalimat.")
        else:
            kal.append("Proporsi chunk yang berakhir di tengah kalimat berbeda antar ukuran "
                       f"({min(pz):.0%} sampai {max(pz):.0%}).")
        kal.append(f"Chunk yang dimulai di tengah kalimat berkisar {min(pa):.0%} sampai {max(pa):.0%}; "
                   "chunk pertama selalu dimulai bersih.")
        if max(pz) - min(pz) <= 0.15:
            kal.append("Persentase chunk yang terpotong tidak banyak berubah saat ukuran diperbesar; "
                       "yang berubah adalah jumlah chunk dan banyaknya topik per chunk.")
    kecil = [u for u, n, *_ in statistik_angka if n < 10]
    if kecil:
        kal.append(f"Catatan: dokumen ini pendek sehingga ukuran {', '.join(map(str, kecil))} hanya
    menghasilkan "
                   "kurang dari 10 chunk; persentasenya mudah bergeser oleh satu chunk saja.")

    if jumlah_tabel == 0:
        kal.append("Tidak ada tabel yang terdeteksi pada dokumen ini.")
    else:
        kena = [str(u) for u, _, _, _, t in statistik_angka if t > 0]
        if len(kena) == len(statistik_angka):
            kal.append(f"Dari {jumlah_tabel} tabel, minimal satu terpotong oleh sedikitnya satu chunk
    pada "
                       "setiap ukuran; baris tabel yang terpisah dari judul kolomnya sulit dipahami.")
        elif kena:
            kal.append(f"Dari {jumlah_tabel} tabel, pemotongan tabel terjadi pada ukuran {',
    '.join(kena)}.")
        else:
            kal.append(f"Dari {jumlah_tabel} tabel, tidak ada yang terpotong pada pengaturan ini.")
    kal.append("Perbaikan yang layak dicoba: pemotongan berbasis kalimat/paragraf dan memperlakukan "
               "tabel sebagai satu chunk utuh.")
    return " ".join(kal)


# ---------------------------------------------------------------------------- satu dokumen

def jalankan(nama_pdf, nomor):
    pdf = cari_pdf(nama_pdf)
    print("\n" + "#" * 70 + f"\n# {nama_pdf}\n" + "#" * 70)

    # ---- Langkah 1
    mentah = baca_pdf(pdf)
    teks, n_footer = bersihkan_footer(mentah)
    if not teks.strip():
        raise ValueError(f"{nama_pdf}: tidak ada teks yang terbaca (mungkin PDF hasil scan).")
    n_huruf = sum(c.isalpha() for c in teks)
    kualitas = {
        "File": os.path.basename(pdf),
        "Total karakter": len(teks),
        "Karakter rusak (U+FFFD)": teks.count("\ufffd"),
        "Sisa kode glyph (cid:)": teks.count("(cid:"),
        "Persentase huruf": f"{n_huruf / len(teks):.0%}",
    }
    print("=== Langkah 1: 500 karakter pertama ===")
    print(teks[:500])
    print(kualitas)

    # ---- Langkah 2
    print("\n=== Langkah 2: jumlah chunk ===")
    tabel_jumlah = [["Ukuran (karakter)"] + [f"overlap {int(o * 100)}%" for o in OVERLAP]]
    for u in UKURAN:
        baris = [u]
        for o in OVERLAP:
            n = len(potong(teks, u, o))
            baris.append(n)
            print(f"ukuran={u:5d} overlap={o:.0%} -> {n} chunk")
        tabel_jumlah.append(baris)

    # ---- Langkah 3
    wil = wilayah_tabel(teks, PENANDA_TABEL.get(nama_pdf, []))
    print(f"\nTabel terdeteksi: {len(wil)} -> {wil}")
    rnd = random.Random(SEED)
    cuplikan = {}
    statistik = [["Ukuran", "Jumlah chunk", "Awal di tengah kalimat",
                  "Akhir di tengah kalimat", "Memotong tabel"]]
    statistik_angka = []
    for u in UKURAN:
        pot = potong_dengan_posisi(teks, u, OVERLAP_UTAMA)
        n = len(pot)
        a = sum(awal_di_tengah_kalimat(teks, m) for m, _, _ in pot)
        z = sum(akhir_di_tengah_kalimat(teks, e) for _, e, _ in pot)
        t = sum(memotong_tabel(wil, m, e) for m, e, _ in pot)
        statistik.append([u, n, pct(a, n), pct(z, n), pct(t, n)])
        statistik_angka.append((u, n, a, z, t))
```

```
        pilih = sorted(rnd.sample(range(n), min(3, n)))
        cuplikan[u] = [(i + 1, pot[i][0], pot[i][1], pot[i][2],
                        awal_di_tengah_kalimat(teks, pot[i][0]),
                        akhir_di_tengah_kalimat(teks, pot[i][1]),
                        memotong_tabel(wil, pot[i][0], pot[i][1])) for i in pilih]
        print(f"\n=== Langkah 3: ukuran {u} ({len(pilih)} chunk acak) ===")
        for no, m, e, isi, aw, ak, tb in cuplikan[u]:
            print(f"--- chunk #{no} [{m}:{e}] awal_tengah={aw} akhir_tengah={ak} potong_tabel={tb}")
            print(isi)

    catatan = catatan_kualitas(kualitas, n_footer)
    kesimpulan = buat_kesimpulan(statistik_angka, len(wil))
    print("\nKualitas:", catatan)
    print("Kesimpulan:", kesimpulan)

    # ---- Laporan PDF
    keluar = os.path.join(BASE, f"Lab1_Eksperimen_Chunking_{nomor}.pdf")
    L = Laporan(keluar, f"Lab 1 - Eksperimen Chunking ({nomor}/{len(DAFTAR_PDF)})",
                f"{os.path.basename(pdf)} | Python (pdfplumber + fungsi potong)")
    L.h1("Langkah 1. Membaca PDF dan memeriksa teks")
    L.p(f"File yang dibaca: {os.path.basename(pdf)}. Berikut 500 karakter pertama hasil baca_pdf():")
    L.kode(teks[:500])
    L.spasi()
    L.tabel([["Pemeriksaan", "Hasil"]] + [[k, v] for k, v in kualitas.items()], [6 * 28.35, 8 * 28.35])
    L.p(catatan)
    L.h1("Langkah 2. Jumlah chunk tiap pengaturan")
    L.p("Fungsi potong(teks, ukuran, overlap) memakai jendela geser per karakter; langkah = ukuran x (1 -
     overlap).")
    L.tabel(tabel_jumlah)
    L.cap("Tabel 1. Jumlah chunk untuk 3 ukuran x 3 overlap. Makin besar ukuran, makin sedikit chunk; "
          "makin besar overlap, makin banyak chunk.")
    L.h1(f"Langkah 3. Tiga chunk acak per ukuran (overlap {int(OVERLAP_UTAMA * 100)}%, seed {SEED})")
    L.tabel(statistik)
    L.cap("Tabel 2. Statistik seluruh chunk pada tiap ukuran (bukan hanya yang dicuplik).")
    for u in UKURAN:
        L.h2(f"Ukuran {u} karakter")
        for no, m, e, isi, aw, ak, tb in cuplikan[u]:
            L.p(f"Chunk #{no} (karakter {m}-{e}) | awal di tengah kalimat: {'ya' if aw else 'tidak'} | "
                f"akhir di tengah kalimat: {'ya' if ak else 'tidak'} | memotong tabel: {'ya' if tb else
    'tidak'}")
            L.kode(isi)
            L.spasi(4)
    L.h1("Kesimpulan")
    L.p(kesimpulan)
    L.halaman_baru()
    L.h1("Lampiran: kode lab1_chunking.py")
    L.kode(open(__file__, encoding="utf-8").read())
    L.simpan()
    print(f"\nLaporan: {os.path.basename(keluar)}")


def main():
    # Argumen opsional: nama satu/lebih PDF. Tanpa argumen, ketiga PDF diproses.
    daftar = sys.argv[1:] or DAFTAR_PDF
    for i, nama in enumerate(daftar, 1):
        jalankan(os.path.basename(nama), i)


if __name__ == "__main__":
    main()
```