Lab 3 - Ukur Hit@k

backend: neural | overlap 15% | koleksi ukuran 500 dan 1000

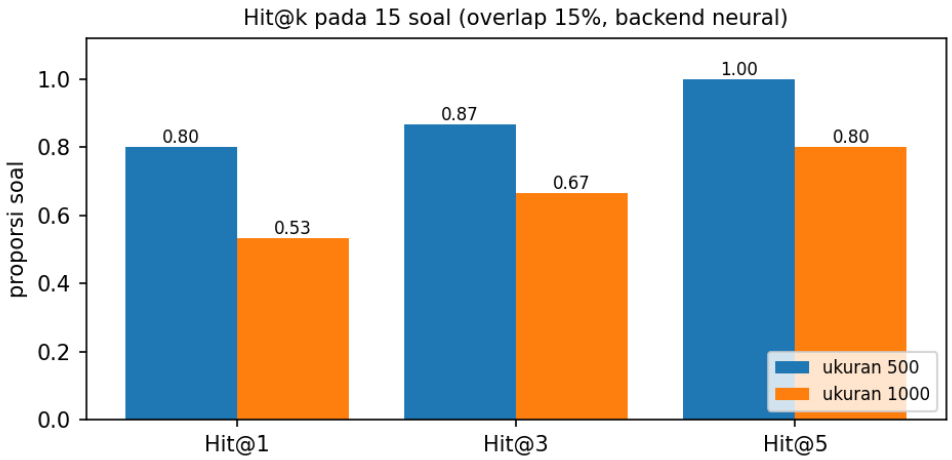
Langkah 1. 15 pertanyaan uji

Soal dibuat dari 3 dokumen di folder dokumen/ (5 soal per dokumen). Sebuah chunk dianggap benar bila memuat semua kata kunci jawaban. Sebanyak 7 soal bernada langsung dan 8 soal diparafrase (kata-katanya sengaja berbeda dari teks).

No	Jenis	Pertanyaan	Kata kunci	File
1	langsung	Siapa yang memperkenalkan istilah kecerdasan buatan pada Konferensi Dartmouth?	McCarthy	1
2	langsung	Siapa yang mengusulkan Tes Turing?	Alan Turing	1
3	parafrase	Mesin buatan IBM yang menundukkan Kasparov di papan catur bernama apa?	Deep Blue	1
4	parafrase	Jenis AI yang hanya mampu mengerjakan satu tugas spesifik disebut apa?	Narrow AI	1
5	langsung	Apa itu Natural Language Processing?	pembuatan bahasa manusia	1
6	langsung	Apa itu overfitting?	hafal data latih	2
7	parafrase	Algoritma mana yang dipakai untuk membagi data ke dalam kelompok-kelompok?	K-Means	2
8	langsung	Arsitektur apa yang dipakai untuk pengolahan gambar dan video?	CNN	2
9	parafrase	Apa sebutan untuk model yang terlalu sederhana sehingga gagal menangkap pola?	Underfitting	2
10	langsung	Arsitektur apa yang menjadi dasar model bahasa besar?	Transformer	2
11	parafrase	Apa istilah untuk informasi keliru yang disampaikan model bahasa dengan nada meyakinkan?	Halusinasi	3
12	langsung	Apa yang dimaksud dengan pelatihan ulang keterampilan?	reskilling	3
13	parafrase	Mengapa keputusan penting tidak boleh diserahkan sepenuhnya kepada mesin?	Pengawasan manusia	3
14	parafrase	Teknologi apa yang menjalankan model langsung di perangkat milik pengguna?	on-device	3
15	parafrase	Isu etika apa yang muncul dari pemakaian karya orang lain sebagai bahan pelatihan model?	Hak cipta	3

Langkah 2. Hasil Hit@k

Koleksi	Hit@1	Hit@3	Hit@5	Jumlah chunk
ukuran 500	0.800	0.867	1.000	46
ukuran 1000	0.533	0.667	0.800	25



Hit@1 dan Hit@5 menurut jenis soal

Jenis soal	Jumlah	ukuran 500 (Hit@1 / Hit@5)	ukuran 1000 (Hit@1 / Hit@5)
langsung	7	0.71 / 1.00	0.43 / 0.71
parafrase	8	0.88 / 1.00	0.62 / 0.88

Peringkat chunk pertama yang memuat jawaban, per soal

Angka 1 berarti chunk teratas sudah benar. Angka > 5 atau '-' berarti gagal pada Hit@5.

No	Jenis	Peringkat (ukuran 500)	Peringkat (ukuran 1000)
1	langsung	1	1
2	langsung	1	5
3	parafrase	1	1
4	parafrase	1	9
5	langsung	4	2
6	langsung	1	1
7	parafrase	1	1
8	langsung	2	15
9	parafrase	1	1
10	langsung	1	14
11	parafrase	1	3
12	langsung	1	1
13	parafrase	1	1
14	parafrase	5	4
15	parafrase	1	1

Diagnosis soal yang gagal (k=5)

Ukuran	No	Pertanyaan	Diagnosis
1000	4	Jenis AI yang hanya mampu mengerjakan satu tugas spesifik disebut apa?	chunk jawaban ada tetapi di peringkat 9 (> 5); pertanyaan berbagi 4 kata bermakna dengan chunk jawaban: jenis, mampu, satu, tugas
1000	8	Arsitektur apa yang dipakai untuk pengolahan gambar dan video?	chunk jawaban ada tetapi di peringkat 15 (> 5); pertanyaan berbagi 4 kata bermakna dengan chunk jawaban: arsitektur, gambar, pengolahan, video

Ukuran	No	Pertanyaan	Diagnosis
1000	10	Arsitektur apa yang menjadi dasar model bahasa besar?	chunk jawaban ada tetapi di peringkat 14 (> 5); pertanyaan berbagi 4 kata bermakna dengan chunk jawaban: arsitektur, bahasa, besar, dasar

Langkah 3. Kesimpulan

Ukuran 500 karakter lebih baik: Hit@1/3/5 = 0.80/0.87/1.00 dibanding 0.53/0.67/0.80 pada ukuran 1000. Pada k=5, pertanyaan yang gagal: ukuran 500 -> tidak ada; ukuran 1000 -> no. 4, no. 8, no. 10. Dugaan penyebab (dari diagnosis otomatis): ukuran 1000 no. 4 - chunk jawaban ada tetapi di peringkat 9 (> 5); pertanyaan berbagi 4 kata bermakna dengan chunk jawaban: jenis, mampu, satu, tugas; ukuran 1000 no. 8 - chunk jawaban ada tetapi di peringkat 15 (> 5); pertanyaan berbagi 4 kata bermakna dengan chunk jawaban: arsitektur, gambar, pengolahan, video; ukuran 1000 no. 10 - chunk jawaban ada tetapi di peringkat 14 (> 5); pertanyaan berbagi 4 kata bermakna dengan chunk jawaban: arsitektur, bahasa, besar, dasar. Catatan: korpus hanya 3 dokumen kecil (ukuran 500: 46 chunk, peluang kebetulan pada k=5 sekitar 11%; ukuran 1000: 25 chunk, peluang kebetulan pada k=5 sekitar 20%), sehingga satu soal saja menggeser skor sekitar 7%; gunakan korpus lebih besar sebelum menarik kesimpulan kuat.

Lampiran: kode lab3_hit_at_k.py

```
"""Lab 3 - Ukur Hit@k (25 menit).

1. 15 pertanyaan uji beserta kata kunci jawaban dari dokumen.
2. hit_at_k untuk k = 1, 3, 5 pada dua koleksi dengan ukuran chunk berbeda.
3. Kesimpulan 3-4 kalimat: ukuran terbaik, pertanyaan yang gagal, dugaan penyebab.

Versi ini memakai ketiga PDF tentang kecerdasan buatan (lihat DAFTAR_PDF).
"""
import os
import re
import shutil
import sys

import matplotlib
matplotlib.use("Agg")
import matplotlib.pyplot as plt

sys.path.insert(0, os.path.dirname(os.path.abspath(__file__)))
from laporan_utils import Laporan
from rag_utils import FOLDER_DOKUMEN, baca_pdf, bangun_indeks, query

BASE = os.path.dirname(os.path.abspath(__file__))
OVERLAP = 0.15
UKURAN = (500, 1000)          # dua koleksi yang dibandingkan
KS = (1, 3, 5)

P1 = "1_Pengenalan_Kecerdasan_Buatan.pdf"
P2 = "2_Machine_Learning_dan_Deep_Learning.pdf"
P3 = "3_Etika_Risiko_dan_Masa_Depan_AI.pdf"
DAFTAR_PDF = [P1, P2, P3]

# q = pertanyaan, kw = kata kunci yang harus ada (semuanya) di chunk jawaban,
# jenis = langsung (kata pertanyaan mirip teks) / parafrase (kata sengaja berbeda),
# file = PDF tempat jawaban berada. 5 soal per dokumen.
SOAL = [
    {"q": "Siapa yang memperkenalkan istilah kecerdasan buatan pada Konferensi Dartmouth?",
     "kw": ["McCarthy"], "jenis": "langsung", "file": P1},
    {"q": "Siapa yang mengusulkan Tes Turing?",
     "kw": ["Alan Turing"], "jenis": "langsung", "file": P1},
    {"q": "Mesin buatan IBM yang menundukkan Kasparov di papan catur bernama apa?",
     "kw": ["Deep Blue"], "jenis": "parafrase", "file": P1},
    {"q": "Jenis AI yang hanya mampu mengerjakan satu tugas spesifik disebut apa?",
     "kw": ["Narrow AI"], "jenis": "parafrase", "file": P1},
    {"q": "Apa itu Natural Language Processing?",
     "kw": ["pembuatan bahasa manusia"], "jenis": "langsung", "file": P1},
    {"q": "Apa itu overfitting?",
     "kw": ["hafal data latih"], "jenis": "langsung", "file": P2},
    {"q": "Algoritma mana yang dipakai untuk membagi data ke dalam kelompok-kelompok?",
     "kw": ["K-Means"], "jenis": "parafrase", "file": P2},
    {"q": "Arsitektur apa yang dipakai untuk pengolahan gambar dan video?",
     "kw": ["CNN"], "jenis": "langsung", "file": P2},
    {"q": "Apa sebutan untuk model yang terlalu sederhana sehingga gagal menangkap pola?",
     "kw": ["Underfitting"], "jenis": "parafrase", "file": P2},
    {"q": "Arsitektur apa yang menjadi dasar model bahasa besar?",
     "kw": ["Transformer"], "jenis": "langsung", "file": P2},
    {"q": "Apa istilah untuk informasi keliru yang disampaikan model bahasa dengan nada meyakinkan?",
     "kw": ["Halusinasi"], "jenis": "parafrase", "file": P3},
    {"q": "Apa yang dimaksud dengan pelatihan ulang keterampilan?",
     "kw": ["reskilling"], "jenis": "langsung", "file": P3},
    {"q": "Mengapa keputusan penting tidak boleh diserahkan sepenuhnya kepada mesin?",
     "kw": ["Pengawasan manusia"], "jenis": "parafrase", "file": P3},
    {"q": "Teknologi apa yang menjalankan model langsung di perangkat milik pengguna?",
     "kw": ["on-device"], "jenis": "parafrase", "file": P3},
    {"q": "Isu etika apa yang muncul dari pemakaian karya orang lain sebagai bahan pelatihan model?",
     "kw": ["Hak cipta"], "jenis": "parafrase", "file": P3},
]


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

def siapkan_dokumen():
    """Pastikan ketiga PDF ada di FOLDER_DOKUMEN; salin dari folder skrip/kerja bila belum ada."""
    os.makedirs(FOLDER_DOKUMEN, exist_ok=True)
    for nama in DAFTAR_PDF:
        tujuan = os.path.join(FOLDER_DOKUMEN, nama)
        if os.path.isfile(tujuan):
            continue
        for d in (BASE, os.getcwd()):
            asal = os.path.join(d, nama)
            if os.path.isfile(asal):
```

```
                shutil.copy(asal, tujuan)
                print(f"  disalin: {nama} -> {FOLDER_DOKUMEN}")
                break
        else:
            print(f"  [PERINGATAN] {nama} tidak ditemukan di {FOLDER_DOKUMEN}, {BASE}, maupun
    {os.getcwd()}")


def rapikan(t):
    """Buang baris footer 'Halaman N', ratakan spasi/baris baru, huruf kecil."""
    t = re.sub(r"^[ \t]*Halaman \d+[ \t]*$", " ", t, flags=re.MULTILINE)
    return " ".join(t.split()).lower()


def cocok_kw(teks, kw):
    """True bila SEMUA kata kunci ada di teks (abaikan huruf besar/kecil dan pemisah baris)."""
    t = rapikan(teks)
    return all(k.lower() in t for k in kw)


STOPWORD = {"yang", "untuk", "dengan", "dalam", "pada", "dari", "atau", "dapat", "adalah", "apakah",
    "apa",
            "siapa", "mana", "bagaimana", "mengapa", "tidak", "juga", "sebagai", "oleh", "kepada",
    "akan",
            "data", "model", "itu", "ini", "serta", "agar", "karena", "hanya", "lebih"}


def kata(t):
    """Kata bermakna (>=4 huruf, bukan kata tugas) untuk menghitung kemiripan kosakata."""
    return set(re.findall(r"[a-z]{4,}", rapikan(t))) - STOPWORD


def peringkat(idx, s):
    """Peringkat chunk pertama yang memuat semua kata kunci di SELURUH indeks (None bila tidak ada)."""
    for r, a in enumerate(query(idx, s["q"], len(idx.chunks)), 1):
        if cocok_kw(a["teks"], s["kw"]):
            return r
    return None


def hit_at_k(rank_list, k):
    """Proporsi soal yang chunk relevannya muncul di k teratas."""
    return sum(r is not None and r <= k for r in rank_list) / len(rank_list)


def diagnosis(idx, s, r, k=5):
    """Penyebab gagal, dihitung dari data indeks (bukan tebakan)."""
    ada = [c for c in idx.chunks if cocok_kw(c, s["kw"])]
    if not ada:
        return "kata kunci terbelah di batas chunk (tidak ada satu chunk pun yang memuatnya utuh)"
    bersama = kata(s["q"]) & kata(ada[0])
    info = (f"pertanyaan berbagi {len(bersama)} kata bermakna dengan chunk jawaban"
            + (f": {', '.join(sorted(bersama))}" if bersama else ""))
    return f"chunk jawaban ada tetapi di peringkat {r} (> {k}); {info}"


# ---------------------------------------------------------------------------- utama

def main():
    print("=== Persiapan dokumen ===")
    siapkan_dokumen()

    # Sanity check 1: setiap kata kunci ada di PDF yang dimaksud
    teks_file = {}
    for f in DAFTAR_PDF:
        p = os.path.join(FOLDER_DOKUMEN, f)
        teks_file[f] = baca_pdf(p) if os.path.isfile(p) else ""
    salah = [s for s in SOAL if not cocok_kw(teks_file.get(s["file"], ""), s["kw"])]
    for s in salah:
        print(f"  [PERINGATAN] kata kunci {s['kw']} tidak ada di {s['file']}: {s['q']}")
    if salah:
        raise SystemExit("Perbaiki kata kunci pada SOAL lebih dulu; soal tersebut tidak mungkin
    terjawab.")

    koleksi, ranks, hasil = {}, {}, {}
    for u in UKURAN:
        print(f"\n=== Koleksi ukuran {u} ===")
        koleksi[u] = bangun_indeks(ukuran=u, overlap=OVERLAP, backend="auto")
        ranks[u] = [peringkat(koleksi[u], s) for s in SOAL]
        hasil[u] = {k: hit_at_k(ranks[u], k) for k in KS}
        print({f"Hit@{k}": round(v, 3) for k, v in hasil[u].items()})
        # Sanity check 2: kata kunci utuh dalam minimal satu chunk (batas atas Hit@k)
        terbelah = [i for i, s in enumerate(SOAL, 1) if not any(cocok_kw(c, s["kw"]) for c in
```

```
koleksi[u].chunks)]
    if terbelah:
        print(f"  [PERINGATAN] soal no. {terbelah} tidak punya chunk yang memuat kata kunci utuh "
              f"pada ukuran {u}; Hit@k tidak mungkin 1.0")
backend = koleksi[UKURAN[0]].backend
n_chunk = {u: len(koleksi[u].chunks) for u in UKURAN}
n_soal = len(SOAL)
n_lang = sum(s["jenis"] == "langsung" for s in SOAL)
n_para = n_soal - n_lang

# ---- kesimpulan: setiap klaim dihitung dari data
a, b = UKURAN
kunci = lambda u: (hasil[u][5], hasil[u][3], hasil[u][1])
terbaik = a if kunci(a) > kunci(b) else b if kunci(b) > kunci(a) else None
fmt = lambda u: "/".join(f"{hasil[u][k]:.2f}" for k in KS)
if terbaik:
    lain = b if terbaik == a else a
    s1 = (f"Ukuran {terbaik} karakter lebih baik: Hit@1/3/5 = {fmt(terbaik)} dibanding {fmt(lain)}
pada "
          f"ukuran {lain}.")
else:
    s1 = f"Kedua ukuran seimbang: Hit@1/3/5 = {fmt(a)} (ukuran {a}) dan {fmt(b)} (ukuran {b})."

gagal = {u: [i for i, r in enumerate(ranks[u], 1) if r is None or r > 5] for u in UKURAN}
s2 = "Pada k=5, pertanyaan yang gagal: " + "; ".join(
    f"ukuran {u} -> " + (", ".join(f"no. {i}" for i in gagal[u]) if gagal[u] else "tidak ada")
    for u in UKURAN) + "."

penyebab = {(u, i): diagnosis(koleksi[u], SOAL[i - 1], ranks[u][i - 1])
            for u in UKURAN for i in gagal[u]}
print("\nDiagnosis gagal (k=5):")
for (u, i), c in penyebab.items():
    print(f"  ukuran {u}, no. {i} ({SOAL[i - 1]['q']}): {c}")
if penyebab:
    s3 = "Dugaan penyebab (dari diagnosis otomatis): " + "; ".join(
        f"ukuran {u} no. {i} - {c}" for (u, i), c in penyebab.items()) + "."
else:
    s3 = "Tidak ada kegagalan pada k=5, sehingga tidak ada penyebab yang perlu dianalisis."

# per jenis soal, Hit@5 dan Hit@1
per_jenis = {}
for jenis in ("langsung", "parafrase"):
    idxs = [i for i, s in enumerate(SOAL) if s["jenis"] == jenis]
    per_jenis[jenis] = (len(idxs), {u: (hit_at_k([ranks[u][i] for i in idxs], 1),
                                        hit_at_k([ranks[u][i] for i in idxs], 5)) for u in UKURAN})

acak = "; ".join(f"ukuran {u}: {n_chunk[u]} chunk, peluang kebetulan pada k=5 sekitar {min(1, 5 /
n_chunk[u]):.0%}"
                 for u in UKURAN)
s4 = f"Catatan: korpus hanya {len(DAFTAR_PDF)} dokumen kecil ({acak})"
if all(hasil[u][5] >= 0.999 for u in UKURAN):
    s4 += ", sehingga Hit@5 yang sempurna belum membuktikan banyak; perbedaan antar ukuran lebih
berarti " \
          "dilihat pada Hit@1 dan pada korpus yang lebih besar."
else:
    s4 += ", sehingga satu soal saja menggeser skor sekitar " \
          f"{1 / n_soal:.0%}; gunakan korpus lebih besar sebelum menarik kesimpulan kuat."
kesimpulan = " ".join([s1, s2, s3, s4])
print("\nKesimpulan:", kesimpulan)

# ---- grafik
fig, ax = plt.subplots(figsize=(6.4, 3.2), dpi=150)
lebar = 0.8 / len(UKURAN)
for j, u in enumerate(UKURAN):
    xs = [i + (j - (len(UKURAN) - 1) / 2) * lebar for i in range(len(KS))]
    bars = ax.bar(xs, [hasil[u][k] for k in KS], lebar, label=f"ukuran {u}")
    ax.bar_label(bars, fmt="%.2f", fontsize=8)
ax.set_xticks(range(len(KS)), [f"Hit@{k}" for k in KS])
ax.set_ylim(0, 1.12)
ax.set_ylabel("proporsi soal")
ax.set_title(f"Hit@k pada {n_soal} soal (overlap {int(OVERLAP * 100)}%, backend {backend})",
fontsize=10)
ax.legend(loc="lower right", fontsize=8)
fig.tight_layout()
gambar = os.path.join(BASE, "_grafik_hit_at_k.png")
fig.savefig(gambar)
plt.close(fig)

# ---- laporan PDF
W = 28.35
L = Laporan(os.path.join(BASE, "Lab3_Hit_at_k.pdf"), "Lab 3 - Ukur Hit@k",
            f"backend: {backend} | overlap {int(OVERLAP * 100)}% | koleksi ukuran {UKURAN[0]} dan
{UKURAN[1]}")
```

```
    L.h1(f"Langkah 1. {n_soal} pertanyaan uji")
    L.p(f"Soal dibuat dari {len(DAFTAR_PDF)} dokumen di folder dokumen/ (5 soal per dokumen). Sebuah
    chunk dianggap "
        f"benar bila memuat semua kata kunci jawaban. Sebanyak {n_lang} soal bernada langsung dan
    {n_para} soal "
        "diparafrase (kata-katanya sengaja berbeda dari teks).")
    L.tabel([["No", "Jenis", "Pertanyaan", "Kata kunci", "File"]] +
            [[i, s["jenis"], s["q"], ", ".join(s["kw"]), s["file"].split("_")[0]] for i, s in
    enumerate(SOAL, 1)],
            [1 * W, 2 * W, 9 * W, 3.5 * W, 1.5 * W])
    L.h1("Langkah 2. Hasil Hit@k")
    L.tabel([["Koleksi"] + [f"Hit@{k}" for k in KS] + ["Jumlah chunk"]] +
            [[f"ukuran {u}"] + [f"{hasil[u][k]:.3f}" for k in KS] + [n_chunk[u]] for u in UKURAN],
            [4 * W, 2.5 * W, 2.5 * W, 2.5 * W, 3 * W])
    L.gambar(gambar, 13)
    L.h2("Hit@1 dan Hit@5 menurut jenis soal")
    L.tabel([["Jenis soal", "Jumlah"] + [f"ukuran {u} (Hit@1 / Hit@5)" for u in UKURAN]] +
            [[j, n] + [f"{v[u][0]:.2f} / {v[u][1]:.2f}" for u in UKURAN] for j, (n, v) in
    per_jenis.items()],
            [3.5 * W, 2 * W, 5 * W, 5 * W])
    L.h2("Peringkat chunk pertama yang memuat jawaban, per soal")
    L.p("Angka 1 berarti chunk teratas sudah benar. Angka > 5 atau '-' berarti gagal pada Hit@5.")
    L.tabel([["No", "Jenis", f"Peringkat (ukuran {UKURAN[0]})", f"Peringkat (ukuran {UKURAN[1]})"]] +
            [[i, s["jenis"]] + [ranks[u][i - 1] if ranks[u][i - 1] is not None else "-" for u in UKURAN]
             for i, s in enumerate(SOAL, 1)],
            [1.2 * W, 3 * W, 5 * W, 5 * W])
    if penyebab:
        L.h2("Diagnosis soal yang gagal (k=5)")
        L.tabel([["Ukuran", "No", "Pertanyaan", "Diagnosis"]] +
                [[u, i, SOAL[i - 1]["q"], c] for (u, i), c in penyebab.items()], [1.5 * W, 1 * W, 6 * W,
    8.5 * W])
    L.h1("Langkah 3. Kesimpulan")
    L.p(kesimpulan)
    L.halaman_baru()
    L.h1(f"Lampiran: kode {os.path.basename(__file__)}")
    L.kode(open(__file__, encoding="utf-8").read())
    L.simpan()
    os.remove(gambar)
    print("\nLaporan: Lab3_Hit_at_k.pdf")


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