Lab 3 - Ukur Hit@k

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

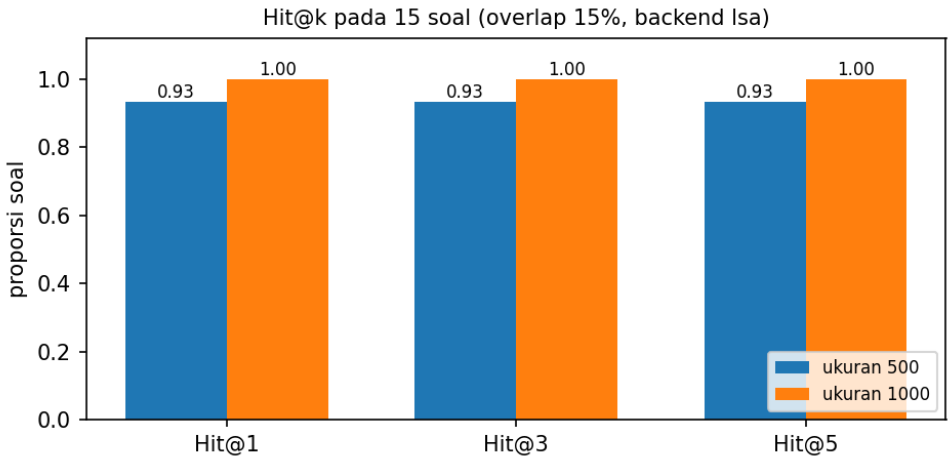
Langkah 1. Lima belas pertanyaan uji

Soal dibuat dari tiga dokumen di folder 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
1	langsung	Siapa yang memperkenalkan RAG?	Lewis
2	langsung	Berapa dimensi embedding yang dihasilkan model MiniLM?	384
3	parafrase	Seberapa besar tumpang tindih antar potongan teks yang disarankan?	10 sampai 20 persen
4	langsung	Pustaka apa buatan Meta untuk pencarian tetangga terdekat?	FAISS
5	parafrase	Jika 12 dari 15 soal terjawab pada lima hasil teratas, berapa skornya?	0,8
6	langsung	Berapa rentang nilai cosine similarity?	-1 dan 1
7	parafrase	Bagaimana menyatukan ketepatan dan kelengkapan menjadi satu angka?	rata-rata harmonik
8	parafrase	Mana yang diutamakan untuk skrining pasien: tidak melewatkan kasus atau menghindari alarm palsu?	deteksi penyakit
9	parafrase	Berapa banyak lipatan yang biasa dipakai pada validasi silang?	5 atau 10
10	parafrase	Teknik apa yang membuat sebagian bobot menjadi tepat nol?	Lasso
11	langsung	Undang-undang nomor berapa yang mengatur pelindungan data pribadi?	27 Tahun 2022
12	parafrase	Bagaimana melindungi sistem dari instruksi jahat yang disusupkan ke dokumen?	memisahkan instruksi
13	parafrase	Kasus perekrutan yang merendahkan pelamar perempuan dialami perusahaan apa?	Amazon
14	langsung	Siapa yang memperkenalkan model card?	Mitchell
15	langsung	Apa kekurangan chunk berukuran 1500 karakter?	boros

Langkah 2. Hasil Hit@k

Koleksi	Hit@1	Hit@3	Hit@5	Jumlah chunk
ukuran 500	0.933	0.933	0.933	25
ukuran 1000	1.000	1.000	1.000	13



Hit@5 menurut jenis soal

Jenis soal	Jumlah	ukuran 500	ukuran 1000
langsung	7	1.000	1.000
parafrase	8	0.875	1.000

Peringkat chunk pertama yang memuat jawaban, per soal

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

No	Jenis	Peringkat (ukuran 500)	Peringkat (ukuran 1000)
1	langsung	1	1
2	langsung	1	1
3	parafrase	1	1
4	langsung	1	1
5	parafrase	1	1
6	langsung	1	1
7	parafrase	18	1
8	parafrase	1	1
9	parafrase	1	1
10	parafrase	1	1
11	langsung	1	1
12	parafrase	1	1
13	parafrase	1	1
14	langsung	1	1
15	langsung	1	1

Diagnosis soal yang gagal (k=5)

Ukuran	No	Pertanyaan	Diagnosis
500	7	Bagaimana menyatukan ketepatan dan kelengkapan menjadi satu angka?	chunk jawaban ada tetapi baru muncul di peringkat 18 (> 5)

Langkah 3. Kesimpulan

Ukuran 1000 karakter lebih baik: Hit@1/3/5 = 1.00/1.00/1.00 dibanding 0.93/0.93/0.93 pada ukuran 500. Pada k=5, pertanyaan yang gagal: ukuran 500 -> no. 7; ukuran 1000 -> tidak ada. Dugaan penyebab: kosakata pertanyaan berbeda dari kosakata dokumen (parafrase, mis. istilah Indonesia vs istilah Inggris di teks), dan chunk kecil memuat lebih sedikit kata konteks sehingga skor kemiripannya kalah dari chunk lain yang topiknya lebih umum. Catatan: skor mendekati sempurna karena korpus hanya 3 dokumen kecil (13-25 chunk) dengan topik yang jelas berbeda; pada PDF yang lebih besar dan lebih mirip antar dokumen, Hit@k diperkirakan turun dan perbedaan antar ukuran chunk akan lebih terlihat.

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.
"""
import os
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, cocok, hit_at_k, norm, peringkat_hit

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

SOAL = [  # q = pertanyaan, kw = kata kunci yang harus ada di chunk jawaban, jenis = langsung / parafrase
    {"q": "Siapa yang memperkenalkan RAG?", "kw": ["Lewis"], "jenis": "langsung"},
    {"q": "Berapa dimensi embedding yang dihasilkan model MiniLM?", "kw": ["384"], "jenis": "langsung"},
    {"q": "Seberapa besar tumpang tindih antar potongan teks yang disarankan?", "kw": ["10 sampai 20
    persen"], "jenis": "parafrase"},
    {"q": "Pustaka apa buatan Meta untuk pencarian tetangga terdekat?", "kw": ["FAISS"], "jenis":
    "langsung"},
    {"q": "Jika 12 dari 15 soal terjawab pada lima hasil teratas, berapa skornya?", "kw": ["0,8"],
    "jenis": "parafrase"},
    {"q": "Berapa rentang nilai cosine similarity?", "kw": ["-1 dan 1"], "jenis": "langsung"},
    {"q": "Bagaimana menyatukan ketepatan dan kelengkapan menjadi satu angka?", "kw": ["rata-rata
    harmonik"], "jenis": "parafrase"},
    {"q": "Mana yang diutamakan untuk skrining pasien: tidak melewatkan kasus atau menghindari alarm
    palsu?", "kw": ["deteksi penyakit"], "jenis": "parafrase"},
    {"q": "Berapa banyak lipatan yang biasa dipakai pada validasi silang?", "kw": ["5 atau 10"], "jenis":
     "parafrase"},
    {"q": "Teknik apa yang membuat sebagian bobot menjadi tepat nol?", "kw": ["Lasso"], "jenis":
    "parafrase"},
    {"q": "Undang-undang nomor berapa yang mengatur pelindungan data pribadi?", "kw": ["27 Tahun 2022"],
    "jenis": "langsung"},
    {"q": "Bagaimana melindungi sistem dari instruksi jahat yang disusupkan ke dokumen?", "kw":
    ["memisahkan instruksi"], "jenis": "parafrase"},
    {"q": "Kasus perekrutan yang merendahkan pelamar perempuan dialami perusahaan apa?", "kw":
    ["Amazon"], "jenis": "parafrase"},
    {"q": "Siapa yang memperkenalkan model card?", "kw": ["Mitchell"], "jenis": "langsung"},
    {"q": "Apa kekurangan chunk berukuran 1500 karakter?", "kw": ["boros"], "jenis": "langsung"},  #
    jawaban ada di tabel
]


def diagnosis(idx, soal, k):
    """Tebak penyebab gagal secara otomatis dari data (bukan menebak sembarang)."""
    r = peringkat_hit(idx, soal)
    if r is None:
        return "kata kunci terbelah di batas chunk (tidak ada satu chunk pun yang memuatnya utuh)"
    return f"chunk jawaban ada tetapi baru muncul di peringkat {r} (> {k})"


def main():
    # sanity check: setiap kata kunci memang ada di dokumen asli
    korpus = norm(" ".join(baca_pdf(os.path.join(FOLDER_DOKUMEN, f))
                           for f in sorted(os.listdir(FOLDER_DOKUMEN)) if f.endswith(".pdf")))
    for s in SOAL:
        assert all(norm(k) in korpus for k in s["kw"]), f"kata kunci tidak ada di dokumen: {s}"

    koleksi, hasil = {}, {}
    for u in UKURAN:
        print(f"\n=== Koleksi ukuran {u} ===")
        koleksi[u] = bangun_indeks(ukuran=u, overlap=OVERLAP, backend="auto")
        hasil[u] = {k: hit_at_k(koleksi[u], SOAL, k) for k in KS}
        print({f"Hit@{k}": round(v, 3) for k, v in hasil[u].items()})
    backend = koleksi[UKURAN[0]].backend

    ranks = {u: [peringkat_hit(koleksi[u], s) for s in SOAL] for u in UKURAN}

    # ---- kesimpulan otomatis
    kunci = lambda u: (hasil[u][5], hasil[u][3], hasil[u][1])
    a, b = UKURAN
```

```
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 = {}
for u in UKURAN:
    for i in gagal[u]:
        penyebab[(u, i)] = diagnosis(koleksi[u], SOAL[i - 1], 5)
print("\nDiagnosis gagal (k=5):")
for (u, i), c in penyebab.items():
    print(f"  ukuran {u}, no. {i} ({SOAL[i-1]['q']}): {c}")

# hit@k per jenis soal (langsung vs parafrase)
per_jenis = {}
for jenis in ("langsung", "parafrase"):
    sub = [s for s in SOAL if s["jenis"] == jenis]
    per_jenis[jenis] = (len(sub), {u: hit_at_k(koleksi[u], sub, 5) for u in UKURAN})

# kalimat 3: dugaan penyebab berdasarkan jenis diagnosis
if penyebab:
    terbelah = [x for x in penyebab.values() if x.startswith("kata kunci terbelah")]
    peringkat = [x for x in penyebab.values() if x.startswith("chunk jawaban ada")]
    dugaan = []
    if peringkat:
        dugaan.append("kosakata pertanyaan berbeda dari kosakata dokumen (parafrase, mis. istilah
Indonesia "
                      "vs istilah Inggris di teks), dan chunk kecil memuat lebih sedikit kata konteks
 sehingga "
                      "skor kemiripannya kalah dari chunk lain yang topiknya lebih umum")
    if terbelah:
        dugaan.append("kata kunci jawaban terbelah di batas chunk dan overlap belum cukup
menyelamatkannya")
    s3 = "Dugaan penyebab: " + "; serta ".join(dugaan) + "."
else:
    s3 = "Tidak ada kegagalan pada k=5, sehingga tidak ada penyebab yang perlu dianalisis."
s4 = ("Catatan: skor mendekati sempurna karena korpus hanya 3 dokumen kecil (13-25 chunk) dengan
topik yang "
      "jelas berbeda; pada PDF yang lebih besar dan lebih mirip antar dokumen, Hit@k diperkirakan
turun dan "
      "perbedaan antar ukuran chunk akan lebih terlihat.")
kesimpulan = " ".join([s1, s2, s3, s4])
print("\nKesimpulan:", kesimpulan)

# grafik
fig, ax = plt.subplots(figsize=(6.4, 3.2), dpi=150)
lebar = 0.35
for j, u in enumerate(UKURAN):
    xs = [i + (j - 0.5) * 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 15 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("Langkah 1. Lima belas pertanyaan uji")
L.p("Soal dibuat dari tiga dokumen di folder 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).")
L.tabel([["No", "Jenis", "Pertanyaan", "Kata kunci"]] +
        [[i, s["jenis"], s["q"], ", ".join(s["kw"])] for i, s in enumerate(SOAL, 1)],
        [1 * W, 2 * W, 10 * W, 4 * 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] + [len(koleksi[u].chunks)] for u in
    UKURAN],
            [4 * W, 2.5 * W, 2.5 * W, 2.5 * W, 3 * W])
    L.gambar(gambar, 13)
    L.h2("Hit@5 menurut jenis soal")
    L.tabel([["Jenis soal", "Jumlah"] + [f"ukuran {u}" for u in UKURAN]] +
            [[j, n] + [f"{v[u]:.3f}" for u in UKURAN] for j, (n, v) in per_jenis.items()],
            [4 * W, 2.5 * W, 3 * W, 3 * W])
    L.h2("Peringkat chunk pertama yang memuat jawaban, per soal")
    L.p("Angka 1 berarti chunk teratas sudah benar. Angka > 5 berarti gagal pada Hit@5.")
    L.tabel([["No", "Jenis", f"Peringkat (ukuran {UKURAN[0]})", f"Peringkat (ukuran {UKURAN[1]})"]] +
            [[i, s["jenis"], ranks[UKURAN[0]][i - 1], ranks[UKURAN[1]][i - 1]] 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, 7 * W,
    8 * W])
    L.h1("Langkah 3. Kesimpulan")
    L.p(kesimpulan)
    L.halaman_baru()
    L.h1("Lampiran: kode lab3_hit_at_k.py")
    L.kode(open(__file__, encoding="utf-8").read())
    L.simpan()
    os.remove(gambar)
    print("\nLaporan: Lab3_Hit_at_k.pdf")


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