Engineer

Retrieval-augmented generation

at generere svar ud fra dokumenter hentet til spørgsmålet

Hvad det er

Retrieval-augmented generation er en måde at besvare et spørgsmål med en sprogmodel på i to trin. Først søger applikationen i en samling dokumenter efter de passager, der mest sandsynligt rummer svaret. Derefter lægger den passagerne ind i modellens input ved siden af spørgsmålet, og modellen skriver sit svar ud fra dem i stedet for ud fra det, den opsugede under træningen. Navnet, som som regel forkortes til RAG, stammer fra en forskningsartikel fra 2020 om vidensintensive sprogopgaver som at besvare åbne spørgsmål.

Artiklen beskrev to slags hukommelse. En forudtrænet model gemmer det, den har lært, i sine parametre, og det kaldte forfatterne parametrisk hukommelse. Et separat indeks over dokumenter, som en trænet søgemodel, en såkaldt retriever, søger i, er ikke-parametrisk hukommelse. Deres system koblede en forudtrænet sequence-to-sequence-model til et tæt vektorindeks over Wikipedia, og de sammenlignede to måder at bruge det fundne på: den ene lader hele svaret bygge på de samme fundne passager, den anden kan trække på forskellige passager for hvert token, den skriver. På tre benchmarks for besvarelse af spørgsmål i åbne domæner satte systemet nye rekorder, og forfatterne fandt, at dets genererede tekst var mere specifik, mere varieret og mere faktuelt korrekt end teksten fra en sammenlignelig model uden søgning.

En applikation kan bygge det samme mønster af dele, den allerede har: et søgeindeks over sine egne dokumenter og en stor sprogmodel, den kalder over et API. Søgedelen er almindelig søgning, og begge dens familier dukker op her. Sparse retrieval matcher ord, sådan som fuldtekstsøgning rangerer dokumenter med en scoringsfunktion som BM25. Dense retrieval gør spørgsmålet og hver passage til vektorer af tal og returnerer de passager, hvis vektorer ligger tættest på, og det er vektorsøgning. En artikel fra 2020 om dense passage retrieval fandt, at en tæt retriever, der var trænet på et beskedent antal par af spørgsmål

og passager, slog en stærk BM25-baseline med 9 til 19 procentpoint, målt på hvor ofte den rigtige passage var blandt de første tyve resultater. Et system kan køre begge dele og flette de to lister.

Brug begrebet i en beslutning

Brug »Hvad er retrieval-augmented generation?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Jeres data samlet, hvor AI-agenter kan nå dem, med én udgave af hver kunde, uden at nogen ser mere, end de må, og med hvert svar sporet tilbage til kilden.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har gjort en geodatapipeline 50 gange hurtigere — 50x hurtigere geografiske datapipelines

Har gjort en geodatapipeline 50 gange hurtigere Kontakt os →

Hvorfor det betyder noget

En sprogmodel kender kun det, der stod i dens træningstekst, og den tekst slutter på en fast dato og har aldrig rummet jeres kontrakter, jeres prislister eller referatet fra sidste uge. RAG-artiklen nævner to begrænsninger ved viden, der ligger i parametre: det er svært at vise, hvor et svar kom fra, og svært at opdatere den. Søgningen tager hånd om begge dele. Svaret kan henvise til de passager, modellen fik, så en læser kan åbne kilden og tjekke den, og ny viden når frem til modellen i samme øjeblik, et dokument kommer i indekset, helt uden en træningskørsel.

Det placerer også spørgsmålet om, hvem der må se hvad, der, hvor det hører hjemme. En søgning kan køre med rettighederne hos den, der spørger, så modellen kun får vist passager, som den person selv kunne have åbnet. I en model, der var trænet på de samme dokumenter, findes den grænse ikke: alt, hvad den har lært, er tilgængeligt for alle, der bruger den.

Prisen er, at svaret aldrig bliver bedre end det, søgningen returnerer. En model, der får de forkerte passager, skriver et flydende svar ud fra dem, og en model, der intet brugbart får, svarer måske ud fra hukommelsen alligevel. Det meste af arbejdet i et system bygget på retrieval-augmented generation ligger derfor i dataene og søgningen, ikke i modellen.

Sådan ser det ud i praksis

Sporet nedenfor følger ét spørgsmål gennem sådan et system. Søgningen kører med den spørgende brugers rettigheder, hver passage kommer med den fil og den side, den stammer fra, og instruktionerne beder modellen svare ud fra passagerne og intet andet:

```
spørgsmål: Hvilke leveringsbetingelser gælder for ordre 4471?
bruger:    salg-3 (rolle: salg, region: nord)

fundet, filtreret til det, salg-3 må læse:
  [1] kontrakter/rammeaftale-2025.pdf, side 4
      "Varer leveres til køberens lager,
       medmindre ordren angiver andet."
  [2] ordrer/4471.txt, linje 12
      "Leveringsbetingelser: jf. rammeaftalen."

instruktioner:
  Svar kun ud fra passagerne nedenfor, og henvis til dem som [n].
  Hvis de ikke besvarer spørgsmålet, så sig det.

svar:
  Ordre 4471 henviser til rammeaftalen [2], hvorefter
  varer leveres til køberens lager [1].
```

Hver del af sporet er data, systemet kan tjekke. En test kan bekræfte, at de rigtige passager kom tilbage for et kendt spørgsmål, et script kan bekræfte, at hvert `[n]` i svaret henviser til en passage, der faktisk blev fundet, og en person, der læser svaret, kan slå op på side 4 i kontrakten.

Det forarbejde, det kræver, er emnet for siden om virksomhedsdata klar til AI-agenter her på sitet. Den beskriver, hvordan gamle PDF'er, scannede dokumenter og mailarkiver bliver til søgbar tekst med kilden gemt ved siden af, og hvordan hver række gemmer, hvor den kom fra og hvornår, nemlig systemet, indlæsningen, filen og sidetallet, så en agent kan henvise til posten bag et tal, og et menneske kan åbne den. Siden lægger også reglerne i dataene og ikke i prompten: agenten arbejder med brugerens privilegier, aldrig en administrators, og hver læsning logges med den person, den er foretaget på vegne af.

Hvor det går galt

Søgningen rammer ved siden af, og modellen svarer alligevel. Når den rigtige passage ikke bliver fundet, får modellen stadig noget og skriver stadig et svar. Mål søgningen for sig: skriv rigtige spørgsmål ned sammen med de passager, der besvarer dem, og tjek, hvor ofte de passager er blandt de resultater, modellen får. Bed modellen sige det, når passagerne ikke besvarer spørgsmålet, og tag spørgsmål af den slags med blandt testsagerne.

Passager, der skærer svaret over. Når dokumenter deles i stykker af fast længde, bliver en tabel skilt fra sin overskrift og en klausul fra sin undtagelse, og den ene halvdel vildleder, når den bliver fundet alene. Del langs den struktur, dokumentet allerede har, dets afsnit, sider og tabelrækker, og hold dokumentets titel og placering sammen med hvert stykke.

Indekset er ligeglad med, hvem der spørger. Et indeks, der er bygget af en tjeneste, som kan læse alt, returnerer passager fra dokumenter, som den spørgende ikke må åbne, og modellen opsummerer dem for vedkommende. Filtrer på brugerens rettigheder inde i søgningen, i indekset eller databasen, før noget når modellen. En regel i prompten er ikke adgangskontrol.

Instruktioner plantet i et dokument. Fundet tekst er tekst, som en anden har skrevet, og modellen kan ikke pålideligt skelne en passage fra en instruktion. NIST's profil for generativ AI beskriver indirekte prompt injection som angreb på applikationer bygget på sprogmodeller gennem data, der sandsynligvis bliver hentet frem, og en artikel fra 2023 demonstrerede den slags angreb mod rigtige systemer. Behandl hver fundet passage som input, der ikke er til at stole på, lad ingen passage udløse en handling uden et tjek i kode, og giv de værktøjer, modellen kan kalde, ikke flere rettigheder, end brugeren selv har.

Et indeks, der halter bagefter. Et dokument bliver ændret eller slettet, og indekset rummer stadig de gamle passager, så systemet svarer selvsikkert ud fra en kontrakt, der ikke længere gælder. Opdater indekset, hver gang en kilde ændrer sig, fjern passager, når deres dokument bliver slettet, og vis datoen for hver kilde ved siden af svaret.

Kildehenvisninger, der ikke siger det, svaret påstår. En model kan henvise til passage [2] for en sætning, som passage [2] ikke underbygger, og NIST bemærker, at genererede kildehenvisninger selv kan være konfabulerede. Tjek i

kode, at hver henvisning peger på en passage, der blev fundet, og lad mennesker tage stikprøver af svar mod deres kilder som en fast del af testarbejdet.

Læs mere

- Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks — artiklen fra 2020, der introducerede RAG-modellerne og sammenlignede de to varianter på besvarelse af spørgsmål og generering af tekst.
- Dense Passage Retrieval for Open-Domain Question Answering — en tæt retriever, trænet på par af spørgsmål og passager og målt mod BM25 på at finde den passage, der rummer svaret.
- NIST AI 600-1: Generative Artificial Intelligence Profile — NIST's risikoprofil for generativ AI, med konfabulation, databeskyttelse og indirekte prompt injection.
- Not what you've signed up for: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection — hvordan instruktioner gemt i fundne data overtager en applikation, med angreb på rigtige systemer.

Ordliste

Vælg ud fra jeres faktiske vilkår

Medbring et repræsentativt datasæt, de spørgsmål det skal besvare, og den aktuelle flaskehals. Vi kan aftale målinger af kvalitet, gennemløb og gendannelse.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er retrieval-augmented generation?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

welcome@engineer.company +45 50 10 61 84 København, Danmark

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/retrieval-augmented-generation/