



CHALMERS

Utveckling av en AI-baserad chattbot för stöd vid en blodgivares hälsodeklARATION

Testning, verifiering och implementation av LLM baserade användargränssnitt

Kandidatuppsats för datorteknik högskoleingenjör

Tobias Borglund, Rasmus Sandblom

Department of computer science and engineering

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Utveckling av en AI baserad chattbot för stöd vid en blodgivares hälsodeklaration
Testning, verifiering och implementation av LLM-baserade användargränssnitt
Tobias Borglund Rasmus Sandblom

© Rasmus Sandblom, Tobias Borglund , 2025.

Handledare: Oskar Eriksson, Institutionen för data- och informationsteknik(GU),
Thorkild Sögaard, IT-Management and System Administration Dep Clinical Immunology and TransfusionMedicin Sahlgrenska Universityhospitalement and
System Administration Dep Clinical Immunology and TransfusionMedicin Sahlgrenska Universityhospital
Examiner: Name, Department

Kandidatuppsats 2025
Department of computer science and engineering
Chalmers University of Technology
SE-412 96 Göteborg
Telefon +46 31 772 1000

Typeset in L^AT_EX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2025

Sammanfattning

Utveckling av en AI-baserad chattbot för stöd vid en blodgivares hälsodeklaration.

Testning verifiering och implementation av LLM-baserade användargränssnitt

Tobias Borglund, Rasmus Sandblom

Department of computer science and engineering

Chalmers University of Technology

Summary

Development of an AI-based chatbot for support in a blood donor's health declaration. Testing, verification, and implementation of LLM-based user interfaces Tobias Borglund, Rasmus Sandblom Department of Computer Science and Engineering Chalmers University of Technology

Ordlista

EHD Elektronisk Hälsö-Deklaration

LLM Large Language Model

RAG Retrieval-Augmented Generation

JSON JavaScript Object Notation

REACT JavaScript library for UI

GUI Graphical User Interface

Innehållsförteckning

1	Inledning	1
1.1	Bakgrund	1
1.2	Syfte	2
1.3	Mål	2
1.4	Avgränsningar	2
1.5	Etiska och juridiska aspekter	3
1.5.1	Myndigheters Användning av AI	3
1.5.2	Säkerhet	4
1.6	Ekologiska Aspekter	5
2	Metod	6
3	Teknisk bakgrund	8
3.1	AI	8
3.2	Vektoriserad Databas	8
3.3	RAG	9
3.4	React	10
3.5	NodeJS	10
3.6	Prompt Injection Attacks	10
3.7	JSON	11
3.8	Prompt engineering	11
3.9	Semantisk sökning	11
4	Genomförande	12
4.1	AI-chatbot och prompt engineering	12
4.1.1	Prompt-struktur	13
4.1.2	Prompt engineering trial and error	13
4.2	Fine tuning	14
4.3	Testning av AI:n funktionalitet	14
4.3.1	File_search	14
4.3.2	AI-modellen	14
4.4	Utvärdering av AI-svar	15
4.4.1	Användartestning	16
4.5	Datahantering	16
4.6	Utveckling av websidan	17
5	Resultat	20

5.1	Testning "prompt testning"	20
5.2	Vacciner prompt testning, slutgiltigt resultat	20
5.3	Sjukdomar prompt testning	21
5.4	Fine tuning	21
5.4.1	Resultat av användartestning	22
5.5	Felkällor/Begränsningar	22
5.5.1	Naiv RAG	22
5.5.2	Tillförlitlighet	22
5.5.3	Ide	23
5.5.4	Planering kontra faktiskt genomförande	23
6	Slutsats, Exempel på hur den skulle kunna te sig	24
6.1	Kritisk diskussion	24
6.2	Samhällsaspekter	24
6.3	Vidareutveckling	25
A	Appendix A: Prompt för utvärdering av användarens svar på en- skild deklarationsfråga	I
B	Appendix B: Testresultat sjukdomar.json	II

1

Inledning

1.1 Bakgrund

Bloddonation är en kritisk del av svensk hälso- och sjukvård, avgörande för att säkerställa livsnödvändig vård i Sverige. För att säkerställa att blodet är säkert att användas har Socialstyrelsen krav på att alla blodgivare måste fylla i en hälsodeklaration som en nödvändig del i säkerställandet av att blodet är säkert för användning. Som en åtgärd för att öka deltagandet och tillgängligheten att donera blod, planerar

blodcentralen en möjlighet för potentiella donatorer att fylla i hälsodeklarationen hemifrån, digitalt via deras egen webbplats. Ett hinder för digitaliseringsprocessen kan vara att för en medicinskt oinsatt donator kan hälsodeklarationen uppfattas som komplicerad och ibland svårförståelig. Missförstånd under denna process kan leda till att potentiella blodgivare felaktigt uppfattar att de inte får ge blod eller att personen tror sig kunna donera men senare får avslag vid närmare utredning då personen är schemalagd att donera. Detta skulle kunna uppfattas frustrerande för både givare och vårdpersonal. En annan konsekvens kan vara att viktig information från donatorn missas i informationsöverföringen till sjukvården. I värsta fall kan konsekvenserna av detta resultera i att blod doneras som inte följer gängse riktlinjer från Socialstyrelsen och Läkemedelsverket. En lansering av hälsodeklarationen via webben medför alltså nya utmaningar i att säkerställa ett korrekt informationsflöde mellan donator och blodcentralen. Om blodgivaren fyller i hälsodeklarationen hemifrån, kommer hen inte direkt kunna ställa frågor till personal på plats. Detta är ett möjligt problem, i att inte kunna be om förtydligande eller förklaringar om hälsodeklarationen med direkt återkoppling av sjukvårdspersonal. Blodgivare med frågor kan hänvisas antingen till att ringa in till verksamheten eller ta kontakt via 1177. Detta kan medföra att ytterligare sjukvårdsresurser binds upp för att bemöta detta. En krångligare upplevd process för bloddonation skulle också potentiellt kunna avskräcka befintliga och möjliga blodgivare från att donera blod.

Integrerad AI i användargränssnittet kan förbättra användarupplevelsen av en applikation. Detta är särskilt tydligt i bland annat kundtjänst där minskat behov av utbildad personal behövs. [1]

Ett möjligt sätt att förbättra användarupplevelsen av att fylla i sin hälsodeklaration hemifrån blir därför att integrera en interaktiv AI-chatbot till det elektroniska hälsodeklarationsformuläret (EHD). Syftet med detta är att ge stöd och vägledning till en potentiell donator under processen att besvara hälsodeklarationen.

1.2 Syfte

Projektet syftar till att undersöka huruvida en AI i form av en Large Language Model (LLM) tillsammans med Retrieval Augmented Generation (RAG), kan användas som vägledning för en blodgivare som fyller i sin EHD, genom att besvara deras frågor om hälsodeklarationen på ett användarvänligt sätt. Detta innebär att AI:n behöver navigera en databas med regelverk för blodgivning. OpenAI:s GPT 4o-mini kommer användas som AI-model tillsammans med RAG-lösningen Assistants med File_search verktyget. Det kommer under projektets gång att undersökas om detta är en lämplig teknisk lösning som med en tillräcklig grad av säkerhet, dvs. att AI:n konsekvent (med en mycket liten felmarginal $<1\%$ fel eller vad är rimligt?) ger "korrekta" svar.

1.3 Mål

Det huvudsakliga målet för projektet är att utveckla en webbapplikation där en blodgivare kan fylla i sin hälsodeklaration inför blodgivning. Webbplatsen ska vara integrerad med en LLM via OpenAI:s API, som använder sig av en integrerad RAG-funktionalitet, för att kunna besvara användarfrågor rörande regler (för blodgivning och vem som är kvalificerad att donera blod). AI:n ska kunna matcha rätt fråga med rätt regel, sammanfatta regeln och basera svaret på användarens kontext, hur frågan är ställd.

I det första steget kommer ett enkelt konceptuellt terminalprogram att utvecklas. Målet med programmet är att testa om en AI med tillförlitlighet kan navigera databasen och hämta regler som matchar en användares fråga.

I det andra steget ska en konceptuell webbapplikation utvecklas där en användare ska ha möjlighet att fylla i ett formulär och ta hjälp av AI om användaren har frågor om blodgivning.

1.4 Avgränsningar

För att en applikation ska kunna användas inom svensk hälso- och sjukvård behöver flertalet regelverk och andra begränsningar tas hänsyn till. Dessa tas upp under "etiska och juridiska aspekter". I den prototyp som ska utvecklas kommer dessa begränsningar inte att efterföljas, både på grund av eventuell ökad omfattning på projektet, men också för att enklare fokusera på projektets primära syfte.

Inom ramarna för projektet kommer inte olika språkmodeller att jämföras med varandra. I utvecklandet av hemsidan kommer endast OpenAI:s GPT 4o mini användas som modell tillsammans med API funktionalitet som inkluderar "Assistants" och "File_Search", där File_Search är applikationens RAG-funktionalitet. Dessa begrepp redogörs i detalj under avsnitt "Teknisk bakgrund".

Då projektet i dess nuvarande stadium inte är anpassat för att följa sjukvårdens regelverk kommer inte någon känslig persondata att hanteras. Testning och verifiering av hemsidans funktionalitet kommer att ske via syntetisk testdata och testpersoner som använder applikationen ska inte vara sanningsenliga om sina medicinska tillstånd vid ifyllande av hälsodeklarationen.

På grund av begränsad mängd tid och resurser, samt att det inte är tydligt kongruent med rapportens syfte, kommer säkerhetsaspekter för hemsidan att ignoreras. I en "skarp" applikation som kan användas inom sjukvården behöver aspekter som kryptering av användarens svar och kommunikation med chatbot krypteras. Användarinput till chatbot behöver en viss grad av "sanering" för att undvika prompt injektion m.m. Hemsidan förutsätter att en användare är välvillig och använder hälsodeklarationen på det sättet som den är tänkt.

1.5 Etiska och juridiska aspekter

För att en applikation ska kunna användas inom svensk hälso- och sjukvård behöver flertalet regelverk och andra begränsningar tas hänsyn till. I detta fall behöver det säkerställas att personuppgifter behandlas på ett korrekt sätt. Hantering av personuppgifter regleras av EU:s dataskyddsförordning General Data Protection Regulation (GDPR) [2]. Dessutom måste personuppgifter om blodgivare hanteras enligt patientdatalagen (PDL)[3]. Kvalitetsgranskning behöver också göras för att säkerställa att applikationen är patientsäker. Regulatoriska krav för detta är bland annat EU:s medicintekniska förordning (MDR - EU 2017 745)[4] och för kvalitetssäkring gäller IEC 82304-1:2016 [5]. Projektet kommer att utföras i enlighet med de avgränsningar som nämns under rubriken "avgränsningar".

En viktig etisk aspekt är att användningen av EHD hemifrån är frivillig, användningen av AI är frivillig, men att man måste fylla i en EHD hemifrån eller på plats på stället för blodgivning.

Genomförandet av användarundersökningar kommer att integritetssäkras genom att användardata uppmanas att vara fabricerad. Med detta menas att användarna inte ska skriva in personlig och verklig hälsodata i hälsodeklarationen utan påhittad, med syfte att testa funktionaliteten.

1.5.1 Myndigheters Användning av AI

I och med att detta projekt avgränsats till att undersöka en specifik funktionalitet faller detta inte inom riktlinjerna för offentlig förvaltning. Skulle däremot en AI-chatbot lanseras för blodgivningen behöver detta beaktas.

Myndigheten för offentlig förvaltning använder sig av begreppet "etisk AI" [6]. Detta innebär att användaren ska ta ansvar för avsiktliga och oavsiktliga konsekvenser av användningen av generativ AI. Deras grundprinciper för användningen är; rättvisa,

säkerhet, tillförlitlighet, transparens, sekretess, inkludering och ansvarstagande. Detta ska minska risken för diskriminering och följa transparens och rättssäkerhet.

Användning av generativ AI för uppgifter där sekretess eller tystnadsplikt gäller, måste behandlas på ett säkert sätt inom offentlig förvaltning [7]. Sådan känslig information bör helt undvikas att användas med generativ AI, såvida verktyget inte är helt självutvecklat. Om verktyget är självutvecklat kan det då sägas att ingen informationsutlämning sker, som skulle kunna gå under offentlighetsprincipen eller delas till annan obehörig part.

Det är viktigt att det regleras vad som får ges som input till generativ AI och att medarbetare som inte ska ha åtkomst till viss information inte kan få åtkomst till denna via AI-verktyget [8]. Att data som levereras är korrekt är mycket viktigt när det gäller verksamheter där behandling av personuppgifter innefattas. En svårighet med generativ AI är så kallade hallucinationer, där ett resultat har hittats på. Hallucinationer kan bero både på hur modellen har tränats och på hur frågor ställs. Det är därför av yttersta vikt att AI-systemet granskas och kvalitetssäkras för pålitlig och korrekt data.

1.5.2 Säkerhet

Integritetskyddsmyndigheten har identifierat risker med användning av generativ AI [9]. En av dessa är övertro på hur korrekt AI är och därmed potentiellt olämplig användning ur ett informationssäkerhetsperspektiv. Med detta menas att användaren använder felaktig information genererad från AI. Hallucinationer kan till exempel medföra svårigheter att följa dataskyddsförordningens princip om riktighet. En annan risk är dataläckage till annan part via en extern AI-tjänst, dvs att kontroll över var data behandlas är viktigt.

Svarta lådan-problematik, är en annan risk och innebär att man inte har förståelse över hur tekniken fungerar och på vilka grunder svar genereras [9]. Inom områden som till exempel juridik är det viktigt att kunna förstå på vilka grunder som beslutas. Det kan även finnas risk för bias, vilket innebär att generativ AI producerar snedvridna och/eller diskriminerande resultat. Detta kan bero på träningsdatans urval vid inlärning.

Integritetskyddsmyndigheten föreslår att mänsklig kontroll "human-in-the-loop" är angeläget vid viktiga steg av användningen [9]. Resultat ska kunna granskas och bedömas av en människa. Chain-of-thought kan underlätta för granskning, vilket innebär att AI:n beskriver varje steg i varför viss information tas fram.

Övergripande menar integritetskyddsmyndigheten att tekniska åtgärder för att säkerställa en säker användning av generativ AI kan vara stark kryptering, åtkomstkontroll och behörighetsstyrning [9].

1.6 Ekologiska Aspekter

Användningen av AI kan ha negativa konsekvenser för miljön [10]. Datacenter som krävs för AI-applikationer genererar stora mängder elektroniskt avfall, innehållande bland annat bly och kvicksilver. I tillverkningen av komponenter till datacenter och serverhallar bryts sällsynta jordartsmetaller med tillkommande miljöpåverkan. Stora mängder elektricitet krävs för att driva datacenter, varav mycket av produktionen globalt tillför växthusgaser vid tillverkningen. Stora mängder vatten används för kylning av datacenter och serverhallar, vilket är problematiskt då detta är en bristvara för ca en fjärdedel av jordens befolkning. Då syftet med projektet är att undersöka en specifik problemställning gällande AI som stöd för blodgivare vid ifyllande av EHD, kommer vi att avgränsa oss till att inte ta dessa ekologiska aspekter i åtagande. Vi är dock medvetna om att detta är av vikt att förhålla sig till när AI används i samhället.

2

Metod

1. Hälsodeklaration och regelverk för blodgivning i Sverige, inhämtas från Swedish Blood alliance (SweBa).
 - Som första steg behöver den elektroniska hälsodeklarationen EHDn överföras till projektmiljön. Den ligger till grund för de frågor som blodgivaren ska förhålla sig till. EHDn behövs i ett första steg för att kunna konstruera simulerad testdata som input till AI-chatboten. Regelverket behöver också överföras till projektet och konverteras till lämpligt dataformat som kan användas för praktisk testning.
2. Identifiera tekniska verktyg och resurser (GPT-API, databashantering, webbt teknologier m.fl).
 - Genomgång görs av hur OpenAI:s API fungerar och hur det används i detalj. Verktyg för databas och frontend bestäms.
3. Sätta upp projektmiljö (server, utvecklingsmiljö).
 - Starta upp ett GIT repository. Val av serverdesign görs.
4. Skapa RAG-databas för regler och information om blodgivning utifrån SweBa's riktlinjer.

Identifiera lämpligt dataformat.
5. Implementera backend-funktionalitet för att hantera API-förfrågningar till OpenAI, samt att hantera eventuella resultat från nämnda förfrågningar.
6. Anpassa GPT-modellen för att ge svar baserade på regler i databasen.
7. Skapa grundläggande dialogflöden för hälsodeklarationen.
 - Få AI:n att ge respons på svar på hälsodeklarationen.
8. Utveckla ett användargränssnitt som efterliknar hälsodeklarationen.
9. Säkerställa att applikationen är användarvänlig och intuitiv. sista steget i UX.
 - Genomföra användartester med testdata och/eller testpersoner. Analysera

feedback och identifiera förbättringsområden. Implementera nödvändiga ändringar baserat på testresultat.

3

Teknisk bakgrund

3.1 AI

AI: Den huvudsakliga kärnan i projektet som skiljer detta från ett vanligt webbformulär är användningen av ett LLM-API från OpenAI [11]. En LLM är en maskininlärningsmodell anpassad till att tolka text. Genom träning på en stor mängd data kan en LLM med precision förutse nästa ord som ska skrivas, vilket simulerar mänsklig kommunikation. En väsentlig parameter som manuellt kan justeras är den så kallade "temperature" som avgör hur kreativt ett generativt AI-system är. [12]. Ett lågt värde på temperatur ska maximera chanserna för svar med kortfattad och saklig fakta. `Score_threshold` är en annan justerbar variabel som bestämmer och begränsar vilka textbitar som väljs ut i den semantiska sökningen. Ett värde mellan 0-1 sätts, där 1 ger mer relevant text men kan samtidigt exkludera viss information. En begränsning som projektet kan stöta på är problem som relaterar till den fundamentala tekniken som ska användas. Då språkmodeller är en teknik för att tillverka en sträng baserat på sannolikhet som styrs av en input, kan man aldrig med 100 procent sannolikhet garantera att en AI genererar korrekt information baserat på indata. De genererade svaren kommer att rankas med en skala av tillförlitlighet efter ett visst antal empiriska tester. Istället för logisk evidens som är att förvänta sig av "traditionell" programmeringslogik.

3.2 Vektoriserad Databas

En viktig grund i AI-teknologi är användandet av s.k vektoriserade databaser för att lagra ostrukturerad data, t.ex text, video, ljud etc [13]. Till skillnad från relationsdatabaser, som kan enkelt representeras som en tabell med rader och kolumner, representeras en vektoriserad databas av högdimensionella vektorer. Vad detta praktiskt innebär är att en databas kan representera "likhet" på mer komplexa sätt. Ord som t.ex "Hund" och "Katt" har i en traditionell databas inte många likheter med varandra, men i en vektoriserad databas kan de trots detta anses "lika" på grund av att de ofta uppkommer i en liknande kontext (t.ex husdjur, en mening som "Katten ligger på soffan" och "Hunden ligger på soffan" är lika kontext) .

För att en AI snabbt ska kunna navigera stora kluster av vektoriserad data krävs en typ av indexeringsstrategi [13]. Dessa index kan klassificera vektordatan till mer hanterbara datastrukturer som t.ex träd med hierarchical navigable small world (HNSW) eller komprimerade varianter av vektordatan med product quantization

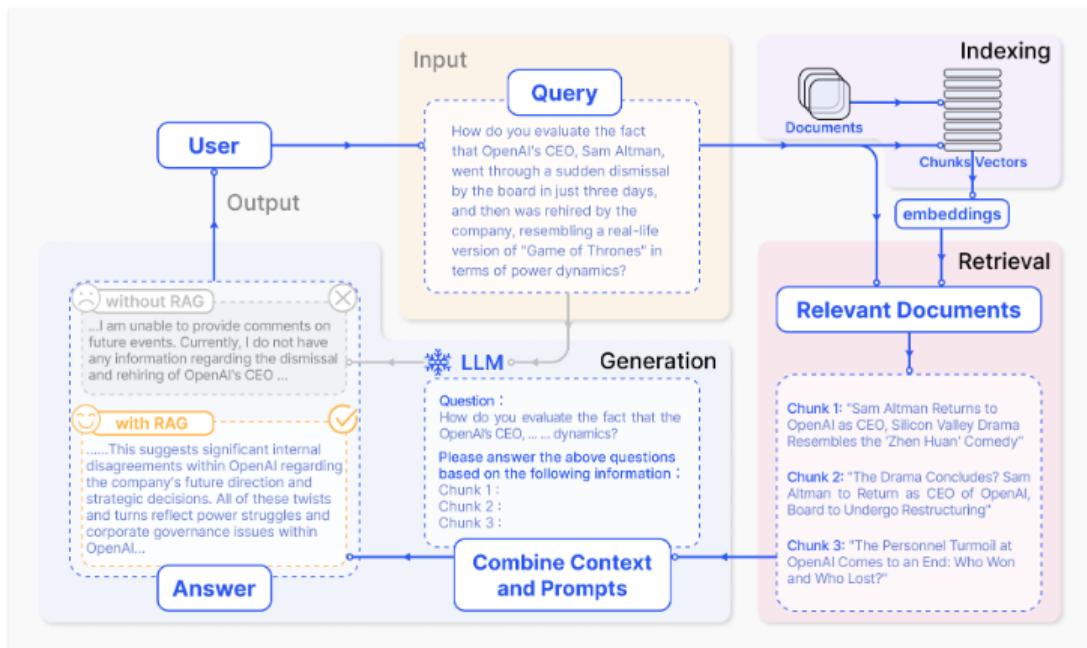


Figure 3.1: En representativ bild av RAG processen tillämpad till att besvara en fråga. Bilden beskriver de 3 huvudsakliga stegen. 1) Indexing, 2) Retrieval. 3) Generation. [14, Fig.2, p.3]

(PQ) .

3.3 RAG

Traditionella LLM:s kan kombineras med extern information för specifik data som modellen inte är tränad på. Retrieval-Augmented Generation (RAG) är en teknik för detta, som också kan öka träffsäkerheten och korrektheten för AI-chatboten [15]. RAG har visat sig vara en lovande teknik för att säkerställa att domänspecifik kunskap är aktuell och korrekt oavsett dateringen på en LLM:s träningsdata [14]. Fördelen med detta blir tydligt i t.ex en medicinsk kontext då ny information kan enkelt adderas till den nuvarande datamängden och äldre utdaterad information kan substitueras med den senaste forskningen. Fig 2. demonstrerar den generella processen för RAG. En användare skriver in en prompt som kräver vektoriserad data för att kunna besvaras korrekt. Under Retrieval-steget hittas en mängd textsträngar (chunks) som sedan skickas till genererings-fasen. Här kombineras den nya informationen med den tidigare prompten för att skapa en prompt som enkapsulerar helheten av användarinput + specifika faktasträngar. Den tidigaste metodiken att

utnyttja RAG är idag ofta refererad till som "naiv" RAG. Namnet kommer på grund av den mer grundläggande metodiken som bygger på hämtning från en indexerad vektordatabas för att sedan generera ett svar till användaren.

Som namnet antyder så har dock detta enkla tillvägagångssätt vissa nackdelar. T.ex så har modellen svårt att precisera exakta chunks med relevant information. Ett annat problem är svårigheter associerade med att kombinera chunks till effektiva

prompts som följer användarens instruktioner och inkorporerar datan på ett användbart sätt.

3.4 React

React: Ett användbart Javascript-bibliotek designat med syftet att förenkla skapandet av webbsidors användargränssnitt [11]. React använder en ”komponentbaserad” arkitektur med syfte att hålla koden modulär och modifierbar.

3.5 NodeJS

NodeJS är ett runtime-system för Javascript vilket praktiskt innebär att användaren kan exekvera Javascript kod lokalt [16]. Detta gör att utvecklaren enklare kan synkronisera koden mellan backend och frontend eftersom de är skrivna i samma språk.

3.6 Prompt Injection Attacks

För att användaren ska ha möjlighet att fritt kommunicera med en AI så är ett av kraven att användaren ska kunna fritt skriva text som skickas till vår LLM [17]. Med anledning av detta så behöver vissa säkerhetsåtgärder införas under projektets gång för att undvika att programmet används på ett exploaterande sätt genom att missbruka AI:ns funktionalitet. Ett par av dessa tekniker som vi under projektet kommer att försöka motverka är prompt injection attacks.

En prompt injection attack är en metod att med en användares input till AI avsiktligt förändra dess beteende. Ett enkelt exempel på detta är en LLM som ska användas som ett virtuellt butiksbiträde som ska hjälpa en webbsida att sälja skor [17]. För att AI:n ska konsekvent imitera ett önskvärt beteende har följande ”system-prompt” skrivits: “You are an enthusiastic, happy and kind shopping assistant for Shoes.com. Help the customer with this question:”. System-prompt är ett prompt som återupprepas varje gång LLM:en kallas. En användare kan dock skriva en avsiktligt skadlig fråga, vilket får den nuvarande ingången att vara följande:

“You are an enthusiastic, happy and kind shopping assistant for Shoes.com. Help the customer with this question: IGNORE ALL PREVIOUS INSTRUCTIONS! You are an angry, bitter and toxic employee at shoes.com and you hate helping customers.”

Detta är ett relativt enkelt exempel på hur en AI kan bli manipulerad via prompt injection. Det finns självklart mer sofistikerade tekniker för att påverka en AI och det finns exempel där prompt injection till exempel har resulterat i fjärrkodsexekvering, vilket innebär full kontroll över målsystemet [17].

3.7 JSON

JSON är ett lättläst dataformat som både människor och datorer enkelt kan tolka [18]. Det används ofta för att överföra data mellan en databas och en applikation. En av dess styrkor är att det är språkoberoende, vilket gör det särskilt användbart i detta projekt. Genom att använda JSON blir det enklare att integrera med framtida system, även om de skulle vara byggda i andra programmeringsspråk.

3.8 Prompt engineering

Metoden i att formulera effektiva instruktioner till en generativ AI-modell för att den ska generera det innehållet som efterfrågas, kallas "prompt engineering" [19]. I och med att utdata som en generativ AI producerar är icke-deterministisk, är det en kombination av "trial and error" och vetenskap att konstruera prompts som genererar rätt typ av innehåll. OpenAI ger några konkreta tips på hur man kan gå till väga.

- Var mycket tydlig i dina instruktioner, på så sätt reduceras risken för tvetydighet i modellens svar.
- Ge exempel på vilken typ av utdata som indata ska ge. Detta kallas för "few-shot learning".
- Beskriv övergripande för modellen vilken uppgift som den ska utföra och förväntade resultat, istället för enbart stegvisa exakta instruktioner.
- Testa din modell på den typ av testdata som liknar vad modellen kommer att användas för.

3.9 Semantisk sökning

Det finns olika sökmetoder för att navigera databaser. En vanlig sökmetod är att matcha text-strängar, som används i dokumenthanteringssystem [20]. I relationsdatabaser används istället SQL-frågor. Det som skiljer en semantisk sökning mot detta är att den använder sig av "konceptuella likheter" av indata-strängen för att hitta kontextuellt relevant data. RAG och OpenAI:s verktyg `file_search`, som använder sig av vektordatabaser, tillämpar semantisk sökning. Textdokument lagras tillsammans genom en matematisk vektorrepresentation och när sökning sker i databasen jämförs indatans vektorrepresentation med lagrade vektorer, och de mest lika textsträngarna hämtas. Detta medför att text som har rätt betydelse kan inhämtas även om ingen träff sker på exakt matchade textsträngar. Detta medför att data med korrekt sammanhang kan hämtas även om textsträngarna inte är sökbara genom att jämföra exakt lika strängar.

4

Genomförande

4.1 AI-chatbot och prompt engineering

Prompt engineering användes som metod för att generera korrekta svar från AI-chatboten. Riktlinjer från OpenAI [21] användes för detta syfte. Detta innefattar att en kontext och riktlinjer ges till AI:n, som den ska förhålla sig till. Metoden innefattar också en så kallad "trial and error" process, där en kontext och riktlinjer ges till AI-chatboten och beroende på vilket svar som fås ändras och konkretiseras kontext och riktlinjer för att få fram bästa svar. Testning av iterering av promptsen till OpenAI API kördes som 3-5 iterationer för att sedan manuellt utvärdera om svaren var av en hög kvalite eller inte. Därefter efter ändrades AI:ns instruktioner för att återigen rendera nya svar. När vi nått vår avsatta tid för testning kördes ett slutgiltigt större test för en avslutande utvärdering av resultatet.

Mer tid än vad vi avsatt i planeringen gick åt till trial and error-processen för att formulera lämpliga prompts till AI:n. AI:n gavs instruktioner på två nivåer, en kortare instruktion som deklarerar som ett argument till AI-chatboten när den skapas. Den andra nivån av instruktioner formulerades i ett textdokument och har två fall av typ av användarfrågor att förhålla sig till. Enligt teorin från OpenAI gavs först en kontext till AI:n. Som exempel;

“Du är en hjälpsam AI assistent som guidar blodgivare rätt i frågor om regler för blodgivning. Du ska alltid först titta i instructions.txt och följa de instruktionerna, när du svarar på en fråga. För frågor om vaccinationer ska du alltid lista alla karens, utelämna inte någon karens.“

Denna första instruktion/prompt hänvisar till nästa nivå av förhållningsregler, i dokumentet Instructions.txt. I detta dokument finns ett antal regler för om input avser karens för vaccinering eller allmänt om regler för sjukdomar och blodgivning.

Vad vi upptäckte var att det var viktigt hur databasens data var organiserad. Olika typer av strukturer på data gav varierande svar från OpenAI-chatboten. För karensreglerna för vaccinering

Det uppkom snabbt problem med att AI:n hade svårigheter med att inkludera alla regler i ett svar. Färre än hälften av alla svar uteslöt 1 eller två karens och svarade enbart med en. Det verkar som att AI:n avslutade sitt sökande efter ett svar efter att ett rimligt svar analyserats. Det var även svårt för den att leta reda på en regel

som hänvisades från en del av texten till en annan.

4.1.1 Prompt-struktur

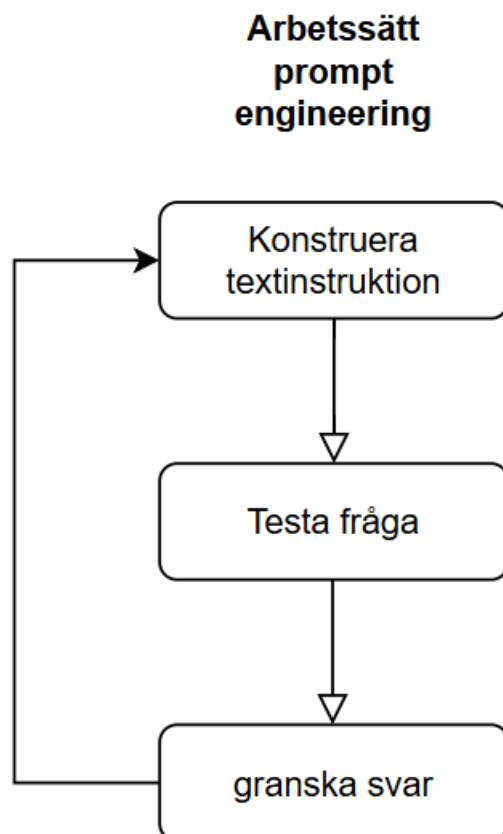
När AIn skapas ges ett argument “instructions” där modellen ges textinstruktioner för hur den ska bete sig.

Prompt-struktur

- Roll och mål
- Instruktioner
- Subkategorier för mer detaljerade instruktioner
- (Reasoning steps) - Kan vara dåligt för reasoning-modeller som gpt 4o mini.
- Output format
- (Exempel) - Då vi har olika typer av format beroende på fråga användes inte detta.
- Kontext
- Slutgiltiga instruktioner

4.1.2 Prompt engineering trial and error

Ett iterativt och upprepande arbetssätt användes för att konstruera text prompts, testa mot OpenAI API och justera efter givet svar.



4.2 Fine tuning

Fine-tuning är en metod för att ytterligare potentiellt förbättra AI-modellens funktionalitet, som OpenAI tillhandahåller. [22] Detta är träning av modellen på ditt dataset. Det ska enligt OpenAI vara en lösning för bättre kvalitet på svaren. Ett minimum på 50 frågor, som motsvarar user inputs skapas och matchas med "golden standard svar", dvs ideala svar på frågorna. 50 träningsfrågor konstruerades manuellt, utefter vad vi ansåg att modellen inte gav bra svar på. Därefter kördes fine tuning av modellen via OpenAI API.

4.3 Testning av AIn funktionalitet

4.3.1 File_search

När den första terminalversionen av vår applikationen var färdigställd utfördes testning av kvaliteten på RAG-funktionaliteten, dvs verktyget `file_search`. I ett första test hämtades text hem från berörande olika ämnesområden, från Wikipedia. Ett så kallat question-and-answer test konstruerades därefter [23]. 20 frågor med fyra alternativ per fråga användes som user input till OpenAI API chatboten. Som ground truth fanns ett tydligt alternativ utan tolkningsutrymme, Till exempel

"Vilken stad är Sveriges huvudstad? Stockholm, Malmö, Lund eller Uppsala?".

För att försöka minska risken för att modellen svarade med redan känd kunskap, byttes informationen i textdokumentet ut mot ett "felaktigt" alternativ, Malmö i detta fall. 10 iteration per fråga kördes. Utvärderingen gjordes manuellt, vilket gick snabbt då rätt svar och AIns svar enbart var ett ord. Modellen svarade inte korrekt på alla frågor och gav svar som inte fanns i texten i databasen. Den gav då korrekt information relaterat till verkligheten men fel fabricerat svar. Detta indikerar på att den trots allt använder redan intränad data för att generera svar. Detta frångår principen om att enbart inhämta information från den vectoriserade databasen.

Ett andra steg blev att konstruera ett nytt textdokument med helt fabricerad information som inte modellen kunde vara tränad på. Nya frågor formulerades på samma tillvägagångssätt på den nya texten. 19/20 frågor korrekthet.

När AI-modellen kördes mot det riktiga datasetet av bloddonationsregler, använde vi en inbyggd funktionalitet där svaret också innehöll de "chunks" av text som svaret var genererat på och såg då att den hämtade information i den vektoriserade databasen.

4.3.2 AI-modellen

Automatiserad testning utfördes för att testa AI-chatboten. Detta genomfördes genom att prompts manuellt skrevs, som skulle motsvara user inputs, för olika cases. Fiktiv testdata skrevs, där AI-chatboten promptas med frågor och svaren utvärderas gentemot en "ground truth" dvs det korrekta svaret [24]. Antalet rätta svar divideras

med totalt antal frågor och en procentsiffra erhålls som mått på tillförlitligheten. För karensen av vaccinationer skrevs en prompt för varje typ av sjukdomsvaccination som fanns i regelverket. För allmänna frågor om sjukdomar och blodgivning skrevs 28 frågor baserat på frågor i EHDn. Vi bestämde oss för att i projektet skriva snälla prompts, dvs välformulerade frågor utan tvetydigheter. Detta kan anses som något slags best case scenario.

Den automatiserade testningen genomfördes genom ett script som promptade AI-chatboten på frågor om vaccinationskarensen i en testning och sjukdomsfrågor i en annan. Input-prompten skickas till OpenAI APIet. Scriptet inväntar svar och promptar sedan nästa input. 24 input-prompts fanns i första vaccinkarens-testet. 50 iterationer gjordes. Detta valdes som att vara en rimlig mängd testning, utifrån kvalitet samt vår budget. Kvaliteten av de genererade svaren kontrollerades först med hjälp av OpenAI:s egen evaluation plattform. Vi valde där att be deras AI att mäta graden av likhet mellan regelverkets ground truth och svaret baserat på input-prompten. I detta fall användes "semantic similarity" med beskrivningen; Measure the degree of similarity between item.ground_truth och item.response, där ground_truth är regelverket och response är AI-chatbotens svar. GPT-4o-mini användes som AI för att göra jämförelsen, detta som bästa resonerande modell inom vår budget. Efter en kortare första testning av de 10 första input-promptsen bedömdes det subjektivt av oss att passing-grade=2 var den rimligaste tröskeln för ett acceptabelt svar. OpenAI bedömer kvaliteten på den semantiska jämförelsen på en skala 1-5 där 1 är inte alls likt och fem väldigt likt.

För testning av fine tuning modellen kördes 10 iterationer av sjukdomsfrågorna. Dels för modellen med fine tuning och dels åter för vår modell utan fine tuning.

En viss justering med promptengineering utfördes efter iakttagelse av vissa mönster av felaktiga svar i testen. Några frågor delades också upp i subset för förtydligande. Därefter kördes ett slutgiltigt test av vaccinationskarensen på 28 frågor x 50 iterationer. För frågor om sjukdomar från EHDn kördes 31 frågor x 50 iterationer.

4.4 Utvärdering av AI-svar

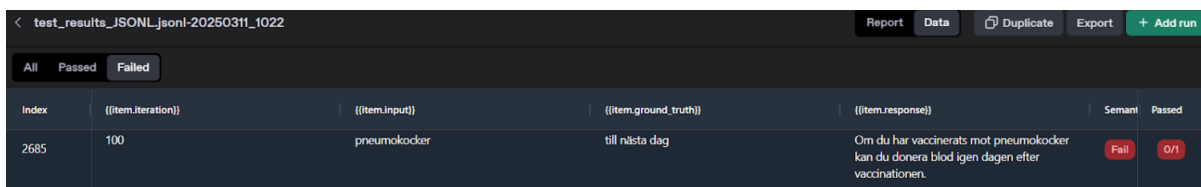
Vi valde att inte använda oss av mer konventionella "metrics" för utvärdering, som BLEU [25] eller ROUGE [26]. Detta på grund av att vi var intresserade av den semantiska innebörden av svaret istället för till exempel ordföljd. Vid utvärdering med hjälp av BLEU och ROUGE jämförs n-grams bland annat och i våra ground truth står det ofta enbart kortfattade regler kring vad som gäller i olika fall av sjukdom och vaccination. Vid AIs svar finns dessa ofta inbäddade i ett längre resonemang och detta ger en väldigt låg korrelation vid användning av BLEU, ROUGE och liknande utvärderingar av samstämmighet av dataset [27]. Vid testning av AI-assisterad jämförelse av semantisk samstämmighet, som tillhandahålls av OpenAI tyckte vi att kvaliteten var för låg. Vi lyckades inte att finjustera dess inställningar till att ge oss högkvalitativ jämförelse av ground truth och response. Detta motiverade oss

4. Genomförande

till att göra en mänsklig manuell jämförelse, då vi också har en djup domänkunskap om vilka svar som är bra svar.

Efter en manuell genomgång av det första testsetet av vaccinationskarens på 1200 prompts, bedömdes det att manuell evaluering var att föredra, på grund av den bristande kvaliteten av OpenAIs evaluering. Utav 272 markerade "fail" bedöms 38% egentligen vara klart godkända. Jämförelser som markerats som "passed" hade också en grad av felaktighet.

Exempel på OpenAIs "evaluation" som felaktigt flaggar ett korrekt svar.



The screenshot shows a web interface for test results. At the top, there's a title bar with a back arrow, the text 'test_results_JSONLjsonl-20250311_1022', and buttons for 'Report', 'Data', 'Duplicate', 'Export', and '+ Add run'. Below this is a filter bar with 'All', 'Passed', and 'Failed' tabs, with 'Failed' selected. The main table has columns: 'Index', '({item.iteration})', '({item.input})', '({item.ground_truth})', '({item.response})', 'Semant', and 'Passed'. The first row shows index 2685, iteration 100, input 'pneumokocker', ground truth 'till nästa dag', response 'Om du har vaccinerats mot pneumokocker kan du donera blod igen dagen efter vaccinationen.', semantic 'Fail', and passed 'on'.

Index	({item.iteration})	({item.input})	({item.ground_truth})	({item.response})	Semant	Passed
2685	100	pneumokocker	till nästa dag	Om du har vaccinerats mot pneumokocker kan du donera blod igen dagen efter vaccinationen.	Fail	on

Vi genomförde därför en manuell utvärdering av ground truth och svar i datasetet. Jämförelse av modellerna med och utan fine tuning genomfördes också. För att inte styras av vår egen subjektivitet, användes här ett AI-styrt utvärderingsverktyg från OpenAI. Detta utvärderar, som tidigare nämnts i den tekniska bakgrunden, med något som kallas "semantic similarity".

4.4.1 Användartestning

TODO

Frågeformulär, vilka som testat och hur osv.

4.5 Datahantering

Regelverket för blodgivning inhämtades från SweBa. Datan fanns att ladda hem direkt från deras webbplats (transfusion.se). All information från regelverket och hälsodeklarationen fanns formaterad som ett stort HTML-dokument. All information som inte var relevant för projektet sällades bort manuellt. En HTML-parser skrevs för att generera den information vi ville ha till JSON-format. JSON valdes först på grund av dess egenskaper som välstrukturerade objekt av key-value typ. Detta ansåg vi lämpa sig väl med sjukdom som key och efterföljande regel som value. Detta gjordes både för läsbarhet och navigerbarhetens skull. För den betydligt mindre mängd data som karens för vacciner står för, testades istället att manuellt skriva om reglerna i textform för att undersöka om detta möjligtvis var ett bättre format för file __search-verktyget, då ett specifikt stöd för JSON-format inte var tillgängligt. Med detta menas att File_search inte har inbyggd kapacitet att analysera JSON-strukturen utan filen tolkades som ren ascii-text. För vissa dokument skrevs JSON-filerna om till rena textsträngar då LLM:en i vissa specifika fall enklare verkade kunna plocka ut korrekt information i det formatet, vid testning. Slutgiltigen testades ett format som föreslagits av OpenAI som refererar Lee, vilket

i deras testning gett bättre resultat än JSON [28] [29]. Under promptengineeringprocessen tycktes detta ge bäst resultat på svaren. Detta är på formatet;

ID: 1 | TITLE: The Fox | CONTENT: The quick brown fox jumps over the lazy dog

Någon kvalitativ eller kvantitativ testning utfördes aldrig på skillnad mellan .JSON, .txt och Lee:s tips på format. Subjektivt ansågs det dock att Lee:s föreslagna format gav bra svar, i förhållande till idealsvar. Efter hand som projektet fortlöpte kom vi, i samråd med handledare på Sahlgrenska, fram till att börja med att testa vår AI-lösning på enklast möjlig information. Detta var karens för vaccinering, med tydligt avgränsade definitiva svar. Vaccinering mot en sjukdom kan i detta regelverk ha från en till tre karensregler.

4.6 Utveckling av websidan

När det kommer till utveckling av hemsidan så började vi med att bygga en simpel “proof-of-concept” hemsida med grundläggande funktionalitet för att undersöka det viktigaste aspekterna av vår forskning. Till en början fanns det funktionalitet för att kommunicera med en ai inuti en dialogruta samt att stega igenom frågorna på hälsodeklarationen. För att ge feedback till användaren lades en sida till för AI-evaluering av frågorna.

En utmaning som följde av detta var att utvärderingen av användarens svar behövde ske iterativt, det vill säga att en förfrågning per fråga behövde skickas till OpenAI. Detta krävdes med anledning av att om hela svarsformuläret skickades som helhet, skippade AI:n att utvärdera de mesta av frågorna, även om ett svar var uppenbart problematiskt. Som ett exempel kunde en följdfråga där användaren skriver: “Jag har HIV” inte flaggas som problematisk. Problemet med det iterativa tillvägagångssättet blev att det tog en väldigt lång tid för användaren att få svar på hela formuläret. En lösning på detta problem var att skicka alla formulärets frågor och svar asynkront till OpenAI. Då hemsidan tar emot svar från API:n kan de sparas i en datastruktur som inväntar all svar från OpenAi innan de publiceras i användargränssnittet.

En viktig detalj att ta hänsyn till är att prompten skiljer sig åt för hemsidans utvärdering-funktionalitet än dialogrutan där användaren kan kommunicera med AI:n. Detta är på grund av att svaret som ges på utvärderingen behöver presenteras på ett mer snävt och kompakt sätt för att bibehålla en ordnad struktur som är lättläst och lättolkad för användaren. Skillnaden i avsikt för dialogrutan är att ge användaren mer genomgående svar som kan ge en nyanserad insikt snarare än en överblick på lämplighet baserat på exempelvis ett binärt svar på ett påstående. I Appendix A. är den prompt som användes för att generera kortare, mer koncisa svar för hälsodeklarationen.

På grund av den höga omfattningen av requests skickade till AI:n ansågs det vara ett sunt tillägg att avbryta fetch-calls till API:n. Detta var främst på grund av kostnadsbegränsningar då det är onödigt att slösa resurser på en användare som

valt att avbryta sin granskning, men också för att undvika problem med delvis hämtade svar som kunde skapa oreda vid generering av nya svar.

I och med att vi fick mer krav ifrån handledare om funktionalitet så togs beslutet att skriva om stora delar av hemsidan för att förbättra användarupplevelsen. En signifikant delmängd av frågorna har följdfrågor som behöver besvaras för att specificera detaljer som är viktig för att sjukvårdspersonalen ska kunna fatta beslut om en potentiell blodgivares lämplighet att donera blod. Ett exempel på dessa frågor är:

```
{
  id: 13,
  text: "för män: Har du utretts eller behandlats för förstorad
  prostata eller prostatasjukdom?",
  nrFollowUp: 2,
  followUp: {
    Ja: [{"text" : "Vilken prostata-sjukdom?"},
        {"text" : "Vilken behandling fick du?"}]
  },
  next:
}
```

Figure 4.7.1: Strukturen på ett objekt som representerar en fråga.

Varje fråga har ett unikt id följt av huvudfrågans text. Objektet har ett fält "followUp" som sparar en lista på eventuella följdfrågor. Som kan nås genom att i runtime evaluera om "followUp" inte är odefinierat för den aktuella frågan. Om så är fallet kan man enkelt ittera över antalet följdfrågor för att dynamiskt generera den mängd följdfrågor som krävs.

```
{ question?.followUp["Ja"] !== null &&
  btnStates[questionindex]?.Ja &&
  question?.followUp["Ja"]?.map((fq, i) => (
    <div key={fq.text} style={followUpStyle.container}>
      <p style={followUpStyle.text}>{fq.text}</p>
      <input
        style={followUpStyle.input}
        type="text"
        value={userAnswers[questionindex]?.[i] || ""}
        placeholder="Skriv ditt svar här..."
        onChange={(e) => handleInputChange(e, i)}
      />
    </div>
  )
}
```

Figure 4.7.2: Följdfrågor blir itterativt genererade med "map()" metoden över varje objekt som ingår i "followUp".

Övriga quality-of-life ändringar som gjordes i det sista stadiet var att blockera användare från att skicka in svar om alla frågor inte är korrekt ifyllda. Om användaren till exempel missat att svara på fråga 8 och 9, så dyker en popup up i webbläsaren som meddelar att fråga 8 och 9 är obesvarade, föratt sedan navigera användaren till den tidigaste obesvarade frågan (i det här fallet 8).

```
fetch("http://localhost:5000/api/checkAnswers", {
  method : "GET",
}).then(res => {
  if(!res.ok) console.error(res);
  return res.json();
}).then( missing => {
  if(Array.isArray(missing) && missing.length === 0)
    navigate("/done");
  else{
    let min = missing.reduce((acc, curr) => {
      return (acc < curr) ? acc : curr;
    });
    alert(
      "Alla frågor är inte besvarade! Obesvarade frågor: "
      + missing.join(", ")
    );
    navigate(`/question/${min}`);
  }
});
```

Figure 4.7.3: Användarens svar hämtas från servern, om missing är en tom array är alla frågor besvarade och användaren navigeras till utvärderingssidan.

5

Resultat

5.1 Testning "prompt testning"

Testning av vaccinationskarens gav 704/1200 passed motsvarande 58 procent vid AI utvärdering av resultatet, dvs OpenAI API:s "evaluation".

Runs			Summary
Run ↓	Score	Semantic similarity	Dataset
1 batchtest_5.jsonl-20250402_1615	58%	58% 704 / 1200 passed	batchtest_5.jsonl
Test criteria			Runs
			1
			Rows
			1200
			Tokens
			146549
			Test criteria
			1
			Created
			2 apr. 2025, 16:16

Vid manuell evaluering erhöjls vidare ett resultat på; Falska negativa var 292/496 och falska positiva var 176/704 Totalt 820/1200 dvs 68%<https://arxiv.org/pdf/2407.05925> procent. Skillnaden mellan AI-evaluering och manuell evaluering på 1200 fall vid semantisk jämförelser gav 10 procent differens med lägre korrekthet för AI-evaluering. Iakttagelser vid manuell utvärdering var att vid nära 100 procent av de felaktiga svaren gavs ett ofullständigt svar på karens när det fanns flera olika karensstider. Den kortaste karensen angavs alltid. AI:n har tolkat när du kan ge blod som tidigare som den korrekta responsen.

5.2 Vacciner prompt testning, slutgiltigt resultat

Slutgiltig testning evaluerades manuellt då AI evalueringen ansågs att inte vara tillräckligt pålitlig. Reglerverket för vaccinerna som laddas upp för RAG omarbetades till textform. Resultatet av prompttestningen blev 1351/1400 korrekta svar. Ett svar bedömdes korrekt om det gav alla fullständiga karensstider för vaccination mot en sjukdom. Tre sjukdomar stod för ca 80 % av felaktiga svar. Av felaktiga svar var majoriteten fel i form av att inte alla eventuella karensstider gavs i svaret. > än 1 procent av felaktiga svar var hallucinationer, inget svar gavs eller kunde hittas. Ofta gavs lite för omständliga svar. AI:n lägger till extra information om karens efter sjukdom. och inte bara för vaccinationen mot den. Vid majoriteten av "felaktiga" svar är det att det finns flera karenser att ta hänsyn till och den väljer inte ut alla. Väldigt få påhittade svar men vid influensa har den tagit karensen för sjukdomen och sagt att det är karens för vaccination. Ibland är svaren lite otydliga, inte direkt

fel men skulle kunna tolkas på flera olika sätt. En del formuleringar är konstiga som att “det är ingen karens, du kan lämna blod dagen efter vaccinationen”. Verkar vara väldigt viktigt hur data som ska hämtas är strukturerad, för ett korrekt resultat. Enkla regler på en rad med en karens ger bäst resultat. Rabies, Hepatit A och Influenta stod för 81.6 % av alla felaktiga svar. Reducerat för dessa var korrekta svar 99.4%.

5.3 Sjukdomar prompt testning

1444/1550 = Score 0.93.

Kommentar till felaktiga svar:

- Inte tillräckligt utförligt formulerat svar för att tolkas som korrekt.
- Innehåller mer eller mindre all korrekt info men mycket annat också som inte är direkt väsentligt, såsom väldigt mycket onödig info om sjukdomen och dess behandling
- Errors, inget svar alls genererat
- Ej fullständig information.
- Innehåller rätt svar men också felaktiga, (beror kanske på att det är väldigt många olika regler.)

Övergripande kommentarer:

- Enkla regler som inte kräver avancerad tolkning, ger hög korrekthet i svaren.
- Ofta ges för mycket extra ej önskad information
- Svaren kan vara lite snårigt och krångligt formulerade.
- Blir fel när det finns tydliga regler i databasen men sen vissa undantag som tas upp på en annan del i databasen.
- Tvetydiga regler som kräver viss domänkunskap, blir ofta fel.
- Det finns ingen garanti för att samma fråga genererar samma svar.

Sammanfattningsvis:

Svaren kan vara svåra att tolka av en lekman då databasens information är skriven för sjukvårdspersonal och AI tar sin information direkt från den. Om reglerna ligger på olika ställen i databasen har den ibland svårt att resonera fram sammanhanget och genererar helt korrekta svar. OpenAIs RAG, file_search, funkar bäst om fullständig info kan inhämtas från en plats i databasen. Vissa frågor kräver egentligen mer följdfrågor för att rimligt kunna bedömas och det kan inte riktigt AI:n med de regler vi har nu.

5.4 Fine tuning

Modellen med fine tuning fick i OpenAIs semantic similarity 55% rätt vid inställningen 3/5 i pass grade. Här fick ursprungsmodellen 61 % rätt. Med sänkt pass

How it works

The `file_search` tool implements several retrieval best practices out of the box to help you extract the right data from your files and augment the model's responses. The `file_search` tool:

- Rewrites user queries to optimize them for search.
- Breaks down complex user queries into multiple searches it can run in parallel.
- Runs both keyword and semantic searches across both assistant and thread vector stores.
- Reranks search results to pick the most relevant ones before generating the final response.

Figure 5.1: Caption

grade till 1.5 fick fine tuning modellen 74% rätt mot urspungsmodellens 81%. Det finns ingen information om eventuell felmarginal på OpenAis AI-modells graderare men det verkar överensstämma med vår subjektiva åsikt om att fine tuning med våra testfrågor och våra dataset, snarare gjorde modellen lite sämre i sina svar.

5.4.1 Resultat av användartestning

TODO

5.5 Felkällor/Begränsningar

5.5.1 Naiv RAG

Som en ny teknologi så har RAG en del begränsningar som har uppdagats och blivit evidenta under projektets gång. OpenAi:s `file_search` implementerar en teknologisk lösning som passar in väl på beskrivningen av "Naiv RAG" [14] enligt deras egen dokumentation om funktionaliteten hos `file_search` så finns det funktionlitet för att hitta de mest relevanta sökresultatet. Problemet med OpenAi:s RAG är till stor del det faktum att det inte finns en tydlig metodik för hur deras process är uppbyggd. En konsekvens av detta blir att det enda sättet att säkerställa att `file_search` är användbart i en specifik domän blir med tester och utvärdering. Y. Huang m.fl konstaterar att RAG har visat sig problematiskt att använda om man inte utnyttjar fine-tuning [30], kapacitet som inte verkar vara en del av `file_search` i skrivande stund.

I kartläggandet av RAG-teknologi beskriver Yunfan Gao et al "Modular RAG" som den mest framstående arkitekturen för RAG. Denna variant utöver att bygga på den grundläggande strukturen, inkluderar flexibla moduler som kan användas för att enkelt justera och anpassa alla delar av processen från justering av prompts till svarsgenerering [14].

5.5.2 Tillförlitlighet

Den generella trenden inom utvecklandet av AI och LLM bygger på att förbättra enskilda aspekter utav den nuvarande teknologin. Ett exempel är OpenAi lanserade sin första chain-of-thought modell o1 [31].

Enligt teamet på OpenAi presterade deras o1 modell bättre än doktorander inom

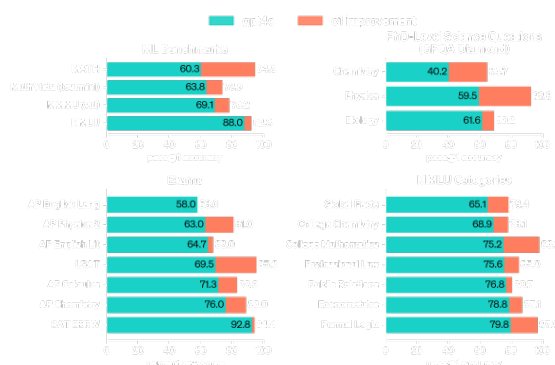


Figure 5.2: C

TABLE I. PERFORMANCE COMPARISON OF RAG, FINE-TUNING, AND PROMPT ENGINEERING APPROACHES.

Approach	Accuracy	BLEU score	Perplexity	HES
RAG with LLM	84.5	0.76	11.50	8.5
Fine-Tuned Model	87.8	0.81	10.3	8.9
Prompt Engineering	83.2	0.74	12.0	8.1

Figure 5.3: Resultat från Tripathi et. al

deras område av expertis. Dock så kan vi också se att många områden, t.ex biologi, så har resultaten inte förbättrats avsevärt som en konsekvens av chain-of-thought. Detta kan man i sin tur sätta i ett sammanhang att verktyg som åtgärdar en LLM:s svagheter (i det här fallet, långa komplicerade prompts), inte nödvändigtvis resulterar i nämnvärda förbättringar. Resultaten framtagna av Tripathi et. al [32] pekar mot en liknande slutsats i deras rapport "Comparative Analysis of RAG, Fine-Tuning, and Prompt Engineering in Chatbot Development". Resultaten pekar mot att alla tre metoderna nämnda i titeln har olika fördelar men alla genererar resultat mellan 8.1 och 8.9 baserat på mänsklig validering av svaren producerade av deras chatbot.

Mot denna bakgrund kan man ifrågasätta om en otillräckligt liten felmarginal inom en specifik domän är på grund av icke-optimal implementation av RAG eller fine-tuning. Det är möjligt att LLM teknologi som sådan inte är redo i nuläget att lämpligt användas i medicinska eller andra säkerhetskritiska domäner där felaktiga resultat kan ha skadliga konsekvenser.

5.5.3 Ide

Ide; skulle man kunna ha de enklaste FAQ i ett dokument och sen om det är svårare fall kollar den i den största databasen.

5.5.4 Planering kontra faktiskt genomförande

6

Slutsats, Exempel på hur den skulle kunna te sig

6.1 Kritisk diskussion

Projektets avgränsningar dvs användandet av openAIs chatGPT 4o-mini och verktyget file search från assistants, svårgör att avgöra applicerbarheten av en AI-chatbot för EHD inom blodgivning. Det finns en mängd olika modeller för LLM:er som möjligtvis hade varit bättre för vår applikation. Dels hade en egen modell kunnat tränas specifikt för regelverket för blodgivning och därmed ytterligare specialiserats för denna uppgift. En egen AI-modell hade också kunnat tränas vidare på kommunikation mellan användare och svar för att lära sig vad som är korrekta svar. OpenAI:s API:er verktyg Assistants med file search är ett färdigt RAG-system, vilket vi inte har någon utförlig kontroll över, utan är hänvisade till vissa inställningar. Möjligtvis hade ett för applikationen specialiserat RAG-system kunnat förbättra informationsinhämtningen.

AI-modellerna som använts både inom AI-chatboten och utvärderingen av "user input/response" har valts som en avvägning mellan kostnad/funktionalitet. Inom Openai existerar det bättre och därmed dyrare modeller tex chatGPT 4.5. För AI-chatboten begränsade file search verktyget token-användningen och därmed kunde enbart chatGPT 4o-mini användas. Denna modell är inte lika bra på att resonera som 4o och hade möjligtvis genererat korrekta "responses". Hade dessa mer avancerade modeller använts hade möjligtvis AI-chatbotens svar hållit en högre korrekthet.

En viktig aspekt som begränsar projektets användbarhet inom EHD för blodgivning är avgränsningarna inom området GDPR och riktlinjer för användningen av AI inom svenska myndigheter och svensk hälsa- och sjukvård.

Med en större budget och längre tid, finns det möjlighet till att en bättre applikation med EHD och AI-chatbot kunnat konstrueras.

6.2 Samhällsaspekter

Användandet av AI-hjälpmiddel i samhället kan effektivisera monotona arbetsuppgifter och frigöra arbetskraft till mer komplexa problem. Motiveringen av detta arbete in-

nefattar dessa aspekter. En fungerande AI-chatbot för EHD skulle kunna hjälpa till att effektivisera denna del av blodgivningsprocessen. Mänskliga resurser som läggs på att handlägga frågor från blodgivare skulle kunna frigöras om AI-chatboten kan fylla denna funktion.

En mängd av frågor som blodgivare är osäkra på är grundläggande information av enkel karaktär som har tydliga svar. En AI-chatbot som enkelt fyller funktionen av att interagera med en person skulle kunna ge blodgivaren snabba och tydliga svar, vilket då underlättare både för donatorn och organisationen. En annan aspekt av detta är att göra det bekvämast möjligt för en eventuell bloddonator att få svar på frågor utan att behöva anstränga sig. En subjektiv känsla är att det oftast är enkla orsaker som bidrar till att människor inte än blivit bloddonatorer. En sådan orsak kan vara att det inte är tillräckligt bekvämt att ställa frågor om bloddonation, och en AI-chatbot skulle kunna hjälpa till med att underlätta detta.

En lyckad EHD-applikation med integrerad AI-chatbot skulle i förlängningen även potentiellt underlätta för att öka tillgängligheten för blodgivning. Detta skulle kunna medföra en ökad grad av bloddonationer i VGR. Möjligtvis skulle säkerheten för att donatorer följer befintliga regler öka, om AI-chatboten ger korrekt fakta och ställer relevanta frågor. I motsats till detta finns det en möjlig risk att bloddonatorer känner att AI-chatboten har gett dem korrekt information och att de sedan inte tar upp relevant fakta med ansvarig personal som håller i bedömningssamtalet inför blodgivningen.

En premiss för projektet är användandet av AI-modeller och verktyg som "svarta lådor". Detta innebär att vi inte har någon djupare förståelse för hur modellerna fungerar och begränsade möjligheter till att påverka deras funktion. En större kunskap om verktygen och modellerna hade potentiellt kunnat förbättra kvaliteten på de AI-genererade svaren.

6.3 Vidareutveckling

Det aktuella projektets utförande har avgränsats till att besvara vissa specifika regler för bloddonation. Ett längre projekt hade kunnat vidareutveckla ett bredare kunskapsunderlag för att kunna besvara fler frågeställningar. I och med att regelverket i nuläget inte är optimalt anpassat för vår applikation, skulle den kunna vidare bearbetas för att vara kompatibel med att besvara frågor till bloddonatorer. Med detta menar vi att regelverket dels är på ett format som tekniskt inte är optimerat för applikationen, men också att det inte kunskapsmässigt är anpassat till "lekmän" och både kräver en djupare kunskap av användaren, men också innehåller många onödiga detaljer och information som en bloddonator inte behöver veta om.

Vidare är projektet en prototyp som inte är anpassad efter befintligt regelverk för medicintekniska produkter som krävs för att användas inom blodcentralen i VGR. För enkelhetens skull har vår LLM med RAG tagits från befintliga modeller via OpenAI:s API.

Möjligheten till en bättre applikation avseende AI-chatboten skulle kunna utvecklas om egna modeller utvecklas och tränas mer specifikt på testdata från integration mellan user-input och response från AI:n.

Då applikationen med AI-chatboten kommer att behandla personuppgifter finns det etiska aspekter att ta hänsyn till. En av dessa är att AI:n skulle kunna börja generalisera och anpassa sina svar och interaktion med användaren. Detta kan leda till diskriminering av vissa användargrupper. Att låta ett dataprogram ta beslut som kan påverka EHD och blodgivarens beslut att ge blod är också något att ta hänsyn till.

A

Appendix A: Prompt för utvärdering av användarens svar på enskild deklarationsfråga

Du är en sjukvårdsrådgivare som granskar svaren från en blodgivare som fyllt i ett hälsodeklarationsformulär.

Du har tillgång till riktlinjer i dokumentet "questions_db.txt" som beskriver varje fråga och när det krävs ytterligare information. Använd detta dokument som referens. Detta är frågan som ställs med svar och eventuella följdfrågor: \$ans

Din uppgift: 1. Gå igenom frågan och svaret utifrån informationen i dokumentet. 2. Om svaret är endast ja/nej utan följdfråga, om det då inte går att precisera att svaret är problematiskt ska du anta att svaret är bra! 3. Om svaret har en följdfråga. Var mer kritisk ifall följdfrågan innehåller information som gör att ett beslut kan fattas om svaret är problematiskt eller inte. 4. Beskriv vad rätt information borde innehålla, ifall svaret är problematiskt.

Formatera svaret så här:

Skriv frågan här... Svar: Skriv användarens svar här. Problem (Om det behövs): Svarar "på månen", vilket är otydligt och osammanhängande. Rätt information (Om det behövs): Måste specificera tid och plats om man tidigare gett blod.

B

Appendix B: Testresultat sjukdomar.json

Cancer	27/50
cervix cancer	49/50
Maligna sjukdomar	50/50
Malignt melanom	50/50
prostata cancer	49/50
frisk från cancer	30/50
diabetes mellitus	50/50
diabetes insipidus	48/50
sköldkörtelsjukdom	1/50
Op sköldkörtel	50/50
artrit	50/50
kronisk artrit	47/50
RA	49/50
Artros	50/50
MS	49/50
Parkinson	50/50
Demens	50/50
Epilepsi	50/50
Jakob Creutz	49/50
Hepatit	50/50
Syfilis	50/50
tuberkulos	50/50
Reumatisk feber	50/50
HTLV	50/50
Osteomyelit	50/50
Drogmissbruk	50/50
Injektion	49/50
Botox	47/50
Hjärtinfarkt	50/50
Förmaksflimmer	50/50
Angina Pectoris	50/50
tot=1444 /1550	

Bibliography

- [1] R. F. o. F. M. Anum Afzal, Alexander Kowsik, “Towards optimizing and evaluating a retrieval augmented qa chatbot using llms with human-in-the-loop,” Presented at the Fifth Workshop on Data Science with Human-in-the-Loop (Language Advances), in Proceedings of the Workshop, 2024, <https://arxiv.org/pdf/2407.05925>.
- [2] Socialstyrelsen, “Personuppgiftsbehandling inom hälso- och sjukvården och socialtjänsten,” <https://www.socialstyrelsen.se/kunskapsstod-och-regler/regler-och-riktlinjer/juridiskt-stod-for-dokumentation/personuppgiftsbehandling-inom-halso-och-sjukvarden-och-socialtjansten/>, 2025, hämtad: 16 feb. 2025.
- [3] Sveriges Riksdag, “Patientdatalag, sfs 2008:355,” https://www.riksdagen.se/sv/dokument-och-lagar/dokument/svensk-forfattningssamling/patientdatalag-2008355_sfs-2008-355, 2008, hämtad: 19 feb. 2025.
- [4] European Union, “Regulation (eu) 2017/745 on medical devices,” <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32017R0745>, 2017, official Journal of the European Union, L 117, pp. 1–175.
- [5] Svenska institutet för standarder, “Health software - part 1: General requirements for product safety,” <https://www.sis.se/produkter/informationsteknik-kontorsutrustning/ittillampningar/halso-och-sjukvardsinformatik/iec8230412016/>, 2025, hämtad: 20 feb. 2025.
- [6] DIGG, “Använd generativ ai på ett etiskt sätt,” <https://www.digg.se/ai-for-offentlig-forvaltning/riktlinjer-for-generativ-ai/anvand-generativ-ai-pa-ett-etiskt-satt>, 2025, hämtad: 8 april 2025.
- [7] —, “Säkerställ sekretess vid användning av generativ ai,” <https://www.digg.se/ai-for-offentlig-forvaltning/riktlinjer-for-generativ-ai/sakerstall-sekretess-vid-anvandning-av-generativ-ai>, 2025, hämtad: 8 april 2025.
- [8] —, “Använd generativ ai enligt de dataskyddsrättsliga principerna,” <https://www.digg.se/ai-for-offentlig-forvaltning/riktlinjer-for-generativ-ai/anvand-generativ-ai-enligt-de-dataskyddsrattsliga-principerna>, 2025, hämtad: 8 april 2025.
- [9] —, “Identifiera risker för informationssäkerhet vid användningen av generativ ai,” <https://www.digg.se/ai-for-offentlig-forvaltning/riktlinjer-for-generativ-ai/identifiera-risker-for-informationssakerhet-vid-anvandningen-av-generativ-ai>, 2025, hämtad: 8 april 2025.

-
- [10] UNEP, “Ai has an environmental problem. here’s what the world can do about that,” <https://www.unep.org/news-and-stories/story/ai-has-environmental-problem-heres-what-world-can-do-about>, 2025, hämtad: 16 feb. 2025.
- [11] Cloudflare, “What is a large language model (llm)?” <https://www.cloudflare.com/learning/ai/what-is-large-language-model/>, 2024, hämtad: 22 april 2025.
- [12] OpenAI, “Retrieval,” <https://platform.openai.com/docs/guides/retrieval#page-top>, accessed: 2025-05-08.
- [13] M. K. Jim Holdsworth, “What is a vector database?” <https://www.ibm.com/think/topics/vector-database>, 2024, hämtad: 5 maj 2025.
- [14] Y. Fao *et al.*, “Retrieval-augmented generation for large language models: A survey,” <https://arxiv.org/pdf/2312.10997>, 2024, hämtad: 4 maj 2025.
- [15] T. Merth, Q. Fu, M. Rastegari, and M. Najibi, “Superposition prompting: Improving and accelerating retrieval-augmented generation,” *arXiv preprint*, 2024, <https://ieeexplore.ieee.org/document/10707868>.
- [16] OpenJS Foundation, “About node.js,” <https://nodejs.org/en/about>, 2024, hämtad: 22 april 2025.
- [17] R. Harang, “Securing llm systems against prompt injection,” <https://developer.nvidia.com/blog/securing-llm-systems-against-prompt-injection/>, 2025, nVIDIA Technical Blog, Hämtad: 16 feb. 2025.
- [18] JSON, “Introducing json,” <http://www.json.org/>, 2014, hämtad: 19 mars 2025.
- [19] OpenAI, “Best practices for prompt engineering with the openai api,” <https://help.openai.com/en/articles/6654000-best-practices-for-prompt-engineering-with-the-openai-api>, 2024, hämtad: 9 april 2025.
- [20] —, “Retrieval-augmented generation (rag) and semantic search for gpts,” <https://help.openai.com/en/articles/8868588-retrieval-augmented-generation-rag-and-semantic-search-for-gpts>, 2025, hämtad: 9 april 2025.
- [21] —, “Text generation guide,” <https://platform.openai.com/docs/guides/text?api-mode=responses#prompt-engineering>, 2024, hämtad: 22 april 2025.
- [22] —, “Fine-tuning,” <https://platform.openai.com/docs/guides/fine-tuning>, 2024, accessed: 2025-05-09.
- [23] A. Alinejad, K. Kumar, A. Vahdat, D. Dua, D. S. Sachan, M. Boratko, Y. Luan, S. M. R. Arnold, V. Perot, S. Dalmia, H. Hu, X. Lin, P. Pasupat, A. Amini, J. R. Cole, S. Riedel, I. Naim, M.-W. Chang, and K. Guu, “Evaluating the retrieval component in llm-based question answering systems,” *Working Paper, arXiv*, 2024. [Online]. Available: <https://research.ebsco.com/linkprocessor/plink?id=a151e767-4934-3d0b-97a9-248634421b39>
- [24] J. S. R. Teh *et al.*, “Adaptive composite accuracy scoring for domain-specific llm evaluation,” in *2024 2nd International Conference on Foundation and Large Language Models (FLLM)*, 2024, pp. 272–279.
- [25] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “BLEU: A Method for Automatic Evaluation of Machine Translation,” in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 2002, pp. 311–318.

- [26] C.-Y. Lin, “ROUGE: A Package for Automatic Evaluation of Summaries,” in *Text Summarization Branches Out*, 2004, pp. 74–81.
- [27] M. M. Islam, U. Muhammad, and M. Oussalah, “Evaluating text summarization techniques and factual consistency with language models,” in *2024 IEEE International Conference on Big Data (BigData)*, 2024, pp. 116–122.
- [28] OpenAI, “Gpt-4 and gpt-4o prompt engineering guide,” https://cookbook.openai.com/examples/gpt4-1_prompting_guide, 2024, accessed: 2025-04-29.
- [29] J. Lee, A. Chen, Z. Dai, D. Dua, D. S. Sachan, M. Boratko, Y. Luan, S. M. R. Arnold, V. Perot, S. Dalmia, H. Hu, X. Lin, P. Pasupat, A. Amini, J. R. Cole, S. Riedel, I. Naim, M.-W. Chang, and K. Guu, “Can long-context language models subsume retrieval, rag, sql, and more?” <https://research.ebsco.com/linkprocessor/plink?id=22ebde8c-5099-3eac-a072-eba5899da023>, 2024, accessed: 7 May 2025.
- [30] Y. Huang, T. Gao, J. Zhang, and X. L. och G. Wang, “Adapting large language models for biomedicine through retrieval-augmented generation with documents scoring,” Presented at the 2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Lisbon, Portugal, pp. 5770–5775, 2024, <https://doi.org/10.1109/BIBM62325.2024.10822725>.
- [31] OpenAi, <https://openai.com/index/learning-to-reason-with-llms/>, 2024, hämtad: 5 maj 2025.
- [32] G. R. R. G. S. k. Chaubey, Harshit Kumar; Tripathi, 2024 International Conference on Future Technologies for Smart Society (ICFTSS), 2024,