ONDERZOEKSVOORSTEL

Toepassen van een tijdelijke grafiek transformatie op object traceringsdata voor het trainen van een LLM.

Bachelorproef, 20242025

Maarten Van der Schueren

E-mail: maarten.vanderschueren@student.hogent.be
Co-promotor: B. Peirens (Tracked, bart.peirens@tracked.be)

Samenvatting

In deze bachelorproef onderzoeken we hoe grafiekmodellering kan bijdragen aan efficiënte klachtenafhandeling binnen bedrijfsprocessen. Door gebruik te maken van een large language model (LLM) gekoppeld aan een chatbot willen we bedrijven in staat stellen sneller en nauwkeuriger de oorzaken van problemen te achterhalen, waardoor kostbaar handmatig werk wordt verminderd. Dit door een vraag te stellen aan de chatbot waarbij hij via het grafiekmodel een ongewone gebeurtenis kan ophalen. Het onderzoek richt zich op twee fases: de omzetting van EPCIS-events naar een grafiekmodel in Cosmos DB met behulp van de Gremlin API, en het trainen van een LLM om te analyseren waar, wanneer, wat en hoe gebeurtenissen plaatsvinden. Dit proces helpt bij het opsporen van anomalieën, met als doel schaalbare en performante oplossingen te bieden voor bedrijfs en nationale use cases. Uiteindelijk streven we naar het optimaliseren van klantenervaring en operationele efficiëntie.

Keuzerichting: AI & Data Engineering
Sleutelwoorden: LLM, OTG, EPCIS, CosmosDB, Assistant Agents

Inhoudsopgave

1. Inleiding

Voor deze bachelorproef werk ik samen met Tracked, een compentence center binnen de Cronos groep gespecialiseerd in traceerbaarheid van de toeleveringsketen in de industrie, en gaan we aan de slag met traceringsdata van ArcelorMittal Gent.

Binnen ArcelorMittal Gent bestaan er veel verschillende processen die verspreid zijn over meerdere afdelingen. Hierbij bevat elke afdeling een deel van het proces om van grondstof tot een afgewerkt product te komen. Tussen elke afdeling bevindt zich een deel van de toeleveringsketen, waarbij het resultaat van de ene afdeling het beginpunt is voor de volgende afdeling. Dit zorgt ervoor dat iedere stap zijn invloed heeft op de kwaliteit van het eindresultaat.

Elke afdeling is opgebouwd rond een specifiek proces, zoals de hoogoven die ruwe grondstoffen omzet in ruw ijzer en de staalfabriek die ruw ijzer in staal verandert. Omdat deze afdelingen zeer specifieke processen hebben, hebben ze ook zeer specifieke behoeften en datamodellen. Dit maakt analyse of onderzoek over afdelingen heen zeer complex en tijdsrovend.

De scope van deze bachelorproef is tweedelig, waarbij deel één eruit bestaat om samen met Tracked vanuit een gestandariseerd datamodel die zichtbaarheid events bijhoudt om te zetten naar een grafiekmodel. Deel twee bouwt verder voort op dit grafiekmodel om dan een LLM (large language model) te trainen die eenvoudig informatie eruit kan halen en teruggeven.

De centrale vraag die hier gesteld wordt is: “Hoe we efficiënt en snel grafiekmodellering kunnen toepassen om een LLM te ontwikkelen die in staat is om het waar, wanneer, wat en hoe van gebeurtenissen binnen een proces vast te stellen, ter ondersteuning van klachtenafhandeling bij productfouten.“

2. Literatuurstudie

2.1. Cosmos DB

Cosmos DB is een NoSQLdatabase van Microsoft. Het biedt een lage latentie, multiquery-API die eenvoudig grote hoeveelheden data kan verwerken en heeft een grote beschikbaarheid zegt **Put2020<empty citation>**, wat zeer belangrijk is in ons project. Daarnaast is CosmosDB horizontaal schaalbaar, wat betekent dat we op hoogtepunten tot een miljoen lees- en schrijfaanvragen kunnen verwerken door het benodigde aantal servers toe te voegen. De hoge beschikbaarheid wordt gegarandeerd door replicatie, waardoor we snel kunnen overschakelen als er een probleem is in onze database.

2.2. Grafiek modellering

In een grafiekdatabase slaan we gegevens op in de vorm van vertices (knopen) en edges (relaties), hierdoor kunnen we complexe verbanden leggen zoals in ons geval de productieproces-

sen. Cosmos DB ondersteunt ook grafiekmodellering met behulp van Gremlin API. Gremlin is gebasseerd op Apache tinkerpop's Gremlin, dit is een krachtig grafiekverwerkingsframework waarbij we grote grafieken kunnen opslaan, modelleren en doorzoeken via gremlin traversal language (**Microsoft2024**). Dit gebeurt met een snelheid van milliseconden waardoor we een snelle verwerkingstijd zullen kunnen neerzetten. Daarnaast is Gremlin ook schaalbaar en kan het consistentie level gekozen worden om een balans te vinden tussen consistentie, beschikbaarheid en latentie. Binnen zo'n grafiek model hebben we vertices (of nodes) die een persoon, plaats of event beschrijven zoals in ons geval bijvoorbeeld een slab die wordt verplaatst van A naar B.

2.2.1. Verschil transformatie en aggregatie

Binnen ons project kunnen de bestaansvormen van producten veranderen. Zo hebben we transformatie, dit is een deel van het proces waarbij de verandering van het product onomkeerbaar is. Daarnaast hebben we aggregatie waarbij we wel terug naar de oorspronkelijke vorm zouden kunnen gaan indien nodig. In onze grafiekmodellering moeten we gebruik maken van tijdelijke edges om ook terug te kunnen kijken naar wat voor het transformatie- of aggregatieproces gebeurd is, om zo anomalieën terug te vinden (**JaewookByun2020**).

2.3. Retrieval Augmented Generation

Om onze chatbot te optimaliseren gaan we gebruik maken van RAG. Dit zorgt ervoor dat de antwoorden die de chatbot zal geven niet outdated zijn en een geldige bron bevatten (zoals onze eigen data) (**IBM2023**). Binnen een LLM waarbij je RAG gebruikt zal de user een prompt geven, daarna gaat het LLM vragen aan RAG om alle bronnen (data) te geven over het gevraagde onderwerp. Zodra hij jouw prompt samen heeft gevoegd met de data die erover beschikbaar is geeft hij een correct antwoord terug. In ons geval kunnen we bijvoorbeeld vragen waar de container staal was op een bepaalde datum, en op dat moment vraagt het LLM aan RAG alle informatie over die container waarna hij een geschikt antwoord kan geven. Deze methode wordt ook gebruikt in onder andere ChatGPT en andere LLM's om ervoor te zorgen dat je het model zelf niet herhaaldelijk moet trainen, maar een up-to-date dataset aan koppelt. Kort gezegd is RAG een AI framework dat data gebruikt buiten zijn eigen getrainde data om te voorkomen dat je het model moet blijven updaten.

2.3.1. RAG vs Finetuning

Bij RAG gebruiken we een soort database waarbij het model up-to-date blijft en geldige bronnen gebruikt. Dit is voordelig in ons project omdat er dagelijks heel veel staal geproduceerd wordt en dus veel verschillende processen zijn. Daarnaast hebben we finetuning waarbij we onze eigen stijl kunnen geven aan het model door bepaalde vakjargon toe te voegen of het model sneller te maken (**IBM2024**). Maar er moet wel zorgvuldig met finetuning omgegaan worden, want het kan snel veel rekenkracht vragen of zelfs voor overfitting zorgen waarbij de trainingsdata goed presteert, maar ongeziene data niet zegt **EASIIO2022<empty citation>**. Een combinatie van deze 2 methodes zou in ons scenario ideaal zijn. Dit komt doordat ArcelorMittal veel vakjargon hanteert, zoals cokes en slabs. Hierdoor kunnen we ons model specialiseren. Daarnaast moet de data die we gebruiken specifiek en up-to-date zijn, zodat we altijd van elk moment op de hoogte blijven

2.4. EPCIS

Voor dit onderzoek moet alles voldoen aan de EPCIS (Electronic Product Code Information Services) waarden, dit zijn de wat, wanneer, waar, waarom en hoe. Deze waarden zijn ontwikkeld door GS1 om gegevens over beweging, status en verandering van een item in de toeleveringsketen (supply chain) vast te leggen en te delen binnen en buiten het bedrijf (**Devins**). "Met behulp van deze waarden kunnen we real-life objecten omzetten in elektronisch opgeslagen informatie, waarna we dit kunnen communiceren met eindgebruikers." zegt **Devins<empty citation>**. Door deze normen toe te passen kunnen we de traceerbaarheid van het product per proces garanderen inclusief de gewenste parameters die opgeslagen worden in ons grafiekmodel zoals tijd (wanneer) en temperatuur (hoe) waar nodig.

2.4.1. Waarom EPCIS?

Huidige Legacy Systemen maken gebruik van ERP, POS, WMS,...Vaak lopen deze niet real-time wat nadelig is als we direct iets willen weten over een product en hoge beschikbaarheid verwachten (**Vieweger**). Doordat elk bedrijf eenzelfde norm gebruikt kunnen we dit schaalbaar houden en makkelijk uitbereiden.

3. Methodologie

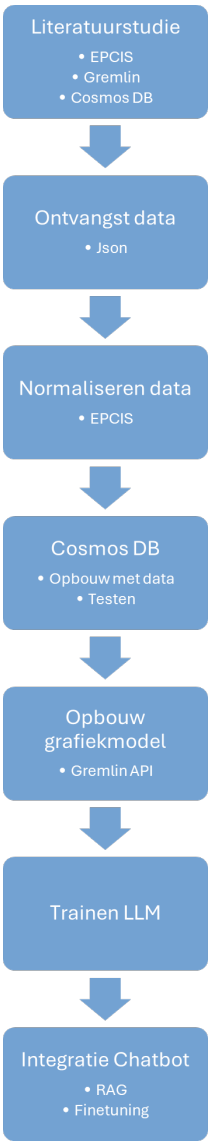
Eerst en vooral gaan we aan de slag met een json bestand dat data bevat van ArcelorMittal waar events in opgeslagen staan. Daarna zetten we dit om naar Cosmos DB (de door Tracked gebruikte database service) en kunnen we via Gremlin API werken om een grafiek op te zetten met tijdsgebonden edges. Dit dient ervoor om te zorgen dat we in tijd terug kunnen om de historiek op te vragen en te kijken waar, wanneer en hoe een proces gebeurd is. Daardoor kunnen we achteraf bij onder andere klachtenafhandeling terug kijken in het grafiekmodel wat er mogelijks anders was of fout is gelopen.

Dit grafiekmodel moet ook voldoen aan de normen volgens EPCIS en schaalbaar zijn, daarom gaan we zorgen dat de gebruikte data binnen het grafiekmodel zoals een node een id krijgt in de plaats van een statische naam. Daardoor kan Tracked in latere fases of andere business cases dezelfde technieken gebruiken zonder veel aanpassingen te moeten doen op grote schaal.

Zodra we een duidelijk grafiekmodel hebben waarmee we makkelijk in het verleden kunnen kijken, kunnen we dit trainen met copilot van microsoft. Daarna gaan we op zoek naar de beste methode om dit te implementeren met onder andere hulp van de RAG methode en finetuning. Daarnaast gaan we ook nog op zoek naar een geschikt zoekalgoritme om zo snel mogelijk op de juiste data terecht te komen. Daardoor kunnen we dan snel zoeken naar waar mogelijks anomalieën zijn in een proces door te vergelijken met andere batches die hetzelfde proces hebben doorlopen. Als laatste kunnen we dan via copilot makkelijk een implementatie starten voor onder andere Microsoft Teams waarbij iemand van de klantenservice kan vragen wat er fout is gelopen en dat de chatbot zegt waar, wanneer en hoe er mogelijks een fout is geweest in het proces.

4. Verwacht resultaat, conclusie

In dit onderzoek maken we startlaag aan een groter project waarbij het mogelijk is om uit te bereiden naar bijvoorbeeld een planningstool om de meest efficiënte route te zoeken. Door deze chatbot verwachten we dat medewerkers van ArcelorMittal eenvoudig een vraag kan stellen en een antwoord kan krijgen op een korte tijd. Zonder deze optie moeten ze elk deel van het bedrijf handmatig opbellen en vragen wat er gebeurd is, dat zijn zaken die zeer tijdsslopend zijn en veel geld kosten. Er wordt gehoopt dat we een schaalbaar systeem kunnen opstellen en dit performant kunnen houden waardoor het gebruikt kan worden op nationaal en/of internationaal niveau. Met de implementatie van dit systeem wordt er gestreefd naar de optimalisatie van klanttevredenheid en besparingen binnen een bedrijf, in dit geval ArcelorMittal Gent.



Figuur 1: Flowchart