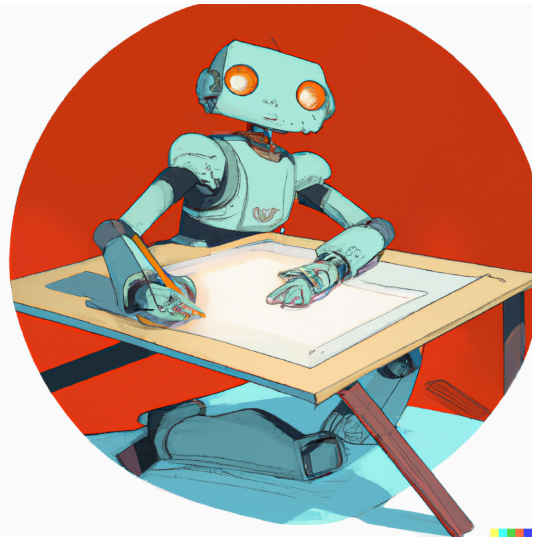


Projektidé

Modulärisering av AI-modell (AI-toolchain)

Mattias Festin
Emelie Chandni Jutvik
2024-05-28

Pitch

Föreställ dig en AI-lösning som är enkel att integrera men kraftfull nog att lösa komplexa affärsproblem. Vår AI-toolchain, tillgänglig som ett NuGet- eller npm-paket, automatiserar databehandling, modellval, träning och verifiering, vilket minskar utvecklingstiden och säkerställer att kundens data förblir skyddad. Distribuerbar både on-premise och i molnet, stödjer den avancerade tekniker som transfer learning och federated learning. Få snabba prediktioner och insikter direkt från dina affärssystem med vår plug-and-play AI-lösning – allt utan att behöva bli en AI-expert.

Syfte

- Utveckla ett verktyg (toolchain) som förenklar inlärningsprocess och arbetsflöde för att implementera AI-modeller i Decerno:s projekt.
- Kunna erbjuda en specialistlösning som skiljer Decerno från andra aktörer på marknaden.
- Kunna erbjuda anställda på Decerno att vara delaktig i utvecklingen av en teknikmodul som ligger i framkant.
- Verktyget fungerar som kvalitetssäkring av framtagen AI-modell då utvecklingsstegen i toolchain:en ställer krav på arbetsflödet samt data.

Nytta

- Fler utvecklare inom Decerno får kunskap om AI
- Fler utvecklare inom Decerno kan implementera AI-modeller i sina projekt
- Utvecklingstiden för att implementera AI minskas
- Möjligheten till att bygga kundspecifika lösningar
- Möjligheten till att kapsla in lösningen och kunna säkerställa att kundens data inte lämnar deras system

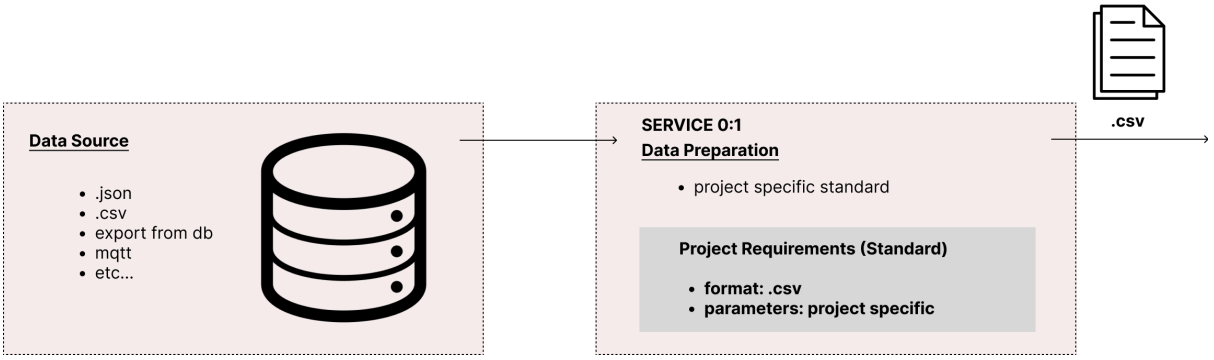
Värde

- En modul/toolchain som kan tillämpas inom flera projekt möjliggör att kostnader för vidareutveckling och förädling kan fördelas mellan projekten
- Produkten kan användas som säljargument och visa upp Decerno som ett företag som utvecklat en unik toolchain som förenklar vägen från idé till realisering av AI-modell
- Produkten kan utökas enkelt progressivt med mer modeller för att lösa mer problem i framtiden och per behov.
- Data pipeline processen kan enkelt förändras och guardrails kan läggas till.

Befintlig kodbas

Modul 1

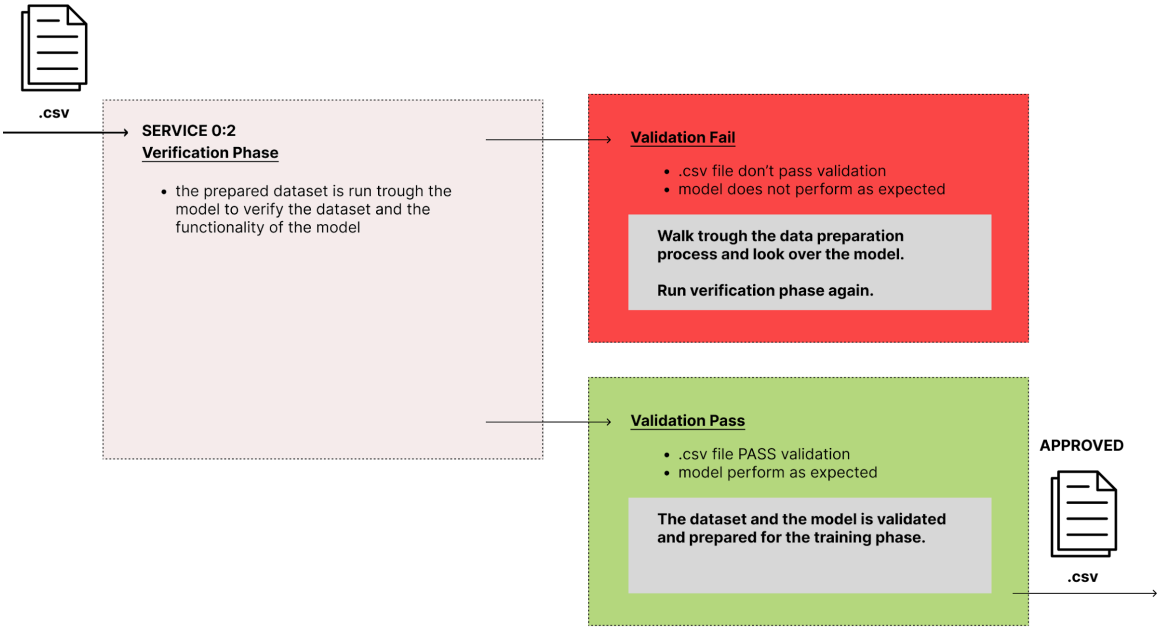
Hanterar olika typer av indata-format och levererar en csv-fil med relevant data för beräkningen.



Modul 2

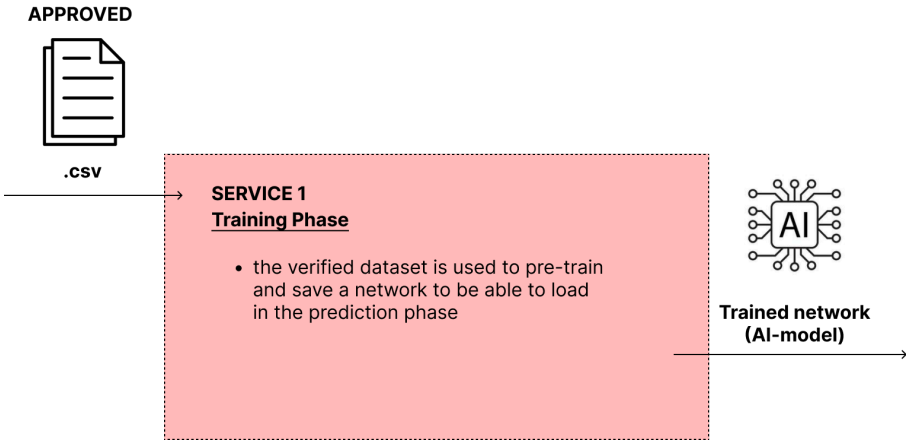
Läser csv-filen från modul 1 och förbereder denna data för att kunna läsas av ett neuralt nätverk. Den förberedda datan och nätverket genomgår ett valideringstest:

- Godkänd validering —> data och modell är redo för att skickas till modul 3
- Icke godkänd validering —> modellen behöver justeras och körs genom valideringstest tills dess att önskat resultat uppnås



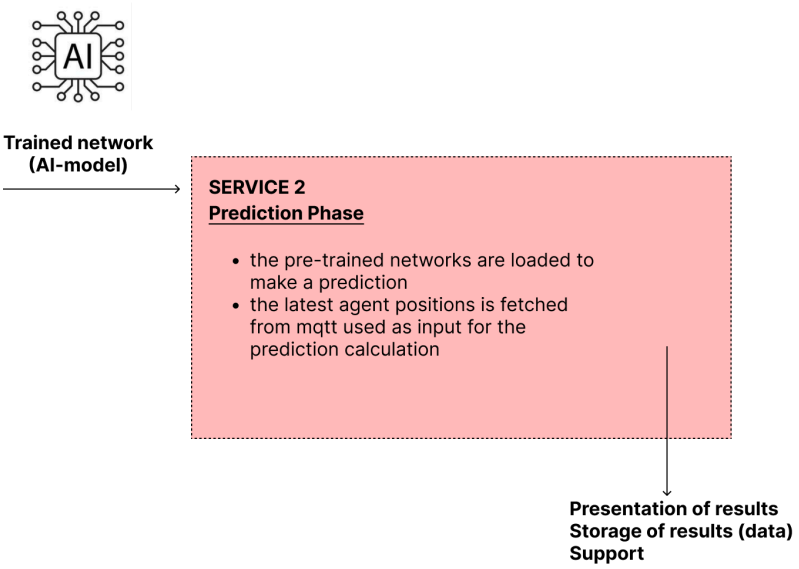
Modul 3

Det neurala nätverket tränas och sparas.



Modul 4

Det sparade/tränade neurala nätverket (från modul 3) nyttjas för att göra en prediktion. Resultatet skickas i form av rådata. Presentationen av resultatet och jämförelsen med verklig data görs lokalt. Tanken är att denna data kan skickas till ett lagringsutrymme som vidare kan läsas in i BI-rapport, webb-applikation, o. dyl.



Paketering

MVP (Minimum value product)

- **Hantering av olika typer av indata-filer (.json, .csv till att börja med)**
- **Användargränssnitt som hanterar:**
    - Uppladdning av indata-fil
    - Val av neuralt nätverk
    - Specifikation av in-parametrar som ska användas för beräkningen (string)
    - Specifikation av ut-parameter (string)
    - Inställning av hyper-parametrar
    - Presentation av verifieringstest (plot)
    - Möjligheten att starta träning av ett nätverk
    - Output —> tränad modell

Vidareutveckling

- **Utveckla ett prediktionssteg (modul 4) där vi automatiserar flödet för att hämta indata och göra en prediktion i realtid**
- **Stegvis automatisering av manuella funktioner som sker via användargränssnittet i MVP:**
    - Val av neuralt nätverk
    - Val av in-parametrar till modellen
    - Hyperparameter-justering
    - Verifiering av modellen
    - Start av träningsfasen
    - Utveckla ett test där modellen själv kan avgöra minsta möjliga datamängd som krävs för att göra en god nog prediktion
- **Implementation av flera olika typer av nätverk (idag befintliga lösningen finns ANN, RNN , Transformer)**
- **Tillägg av hantering av flera olika datakällor (text, bilder, osv. I befintlig lösning hanteras endast indata i form av rådata.)**

Framtida Vision

I framtiden planerar vi att tillhandahålla denna lösning som ett NuGet- eller npm-paket med en klientdel som anropar en serverdel.

Komponenter:

- **Klientdel:**
    - Ett API-wrapper mot servern som gör det enkelt för utvecklare att integrera AI-funktionalitet i sina applikationer utan att behöva hantera den underliggande komplexiteten.
- **Serverdel:**
    - Kör de specialiserade AI-modellerna och hanterar tunga beräkningar. Serverdelen kan vara en wrapper eller fasad mot AI API-molntjänster, vilket möjliggör flexibilitet och skalbarhet.
    - Designad för att kunna köras både on-premise och i klustrade miljöer samt i molnet, vilket ger kunderna valfrihet i hur de vill distribuera och använda AI-lösningarna.

Syfte:

- Att kunna erbjuda en "out of the box"-lösning som är enkel att använda.
- Att integrera med affärssystemens databaser eller filytor med dokument för att automatiskt klassificera och prognostisera data.

Förbättrad AI-process

Översikt: Vi utvecklar ett AI-system som använder olika specialiserade AI-modeller för att lösa specifika problem. Systemet består av flera moduler som samarbetar för att behandla indata, välja rätt AI-modell, generera en output och säkerställa att resultatet är godtagbart.

Steg-för-steg Process:

1. **Indatamodul:**
    - **Funktion:** Behandlar och förbereder indata så att det passar den valda AI-modellen.
    - **Detaljer:** Hanterar olika typer av indata-format (t.ex. CSV, JSON) och transformerar dessa till ett format som är kompatibelt med AI-modellen. Om indata behöver en specifik representation, som text som behöver transformeras till vektoer, utförs detta i detta steg. Vi kan använda tekniker som embeddings för detta.
2. **Konduktör-AI:**
    - **Funktion:** Klassificerar indatat och väljer rätt för-tränad AI-modell för uppgiften.
    - **Detaljer:** En algoritm för klassificering avgör vilken AI-modell som är mest lämplig baserat på indatan:s natur. Exempelvis kan en RNN användas för tidsserier, en DNN för klassificeringsproblem, och en LLM för textbaserade uppgifter.
3. **AI-Exekvering:**
    - **Funktion:** Den valda AI-modellen körs och genererar en output.
    - **Detaljer:** Den valda AI-modellen tar förberedd indata och bearbetar den för att producera ett resultat.

4. **Review-AI:**
    - **Funktion:** Utvärderar resultatet från AI-modellen och avgör om det är godtagbart.
    - **Detaljer:** En sekundär AI-modell granskar resultatet för att säkerställa att det uppfyller förväntningarna. Om resultatet inte är tillfredsställande itereras processen med en annan AI-modell eller justeringar görs för att förbättra utfallet.
5. **Output-AI:**
    - **Funktion:** Formaterar resultatet i rätt format för presentation.
    - **Detaljer:** Beroende på behovet kan resultatet presenteras som text, rådata, bild, eller annan önskad form. Output-AI ser till att presentationen är korrekt och användbar.

Exempel på hjälpfunktioner:

- **Vektordatabas:** För långtidsminne och effektiv lagring av embeddings, vilket är användbart för LLM-modeller som behöver hålla reda på historiska data.

Plan för MVP

Fokus: Steg 2 till 4

- **Inställningar och Konfiguration:** Använd JSON-format för att ställa in parametrar och metadata för varje AI-modell och deras specifika användningsområden.
- **Mål:** Skapa en flexibel och effektiv pipeline som kan hantera olika typer av indata och välja den mest lämpliga AI-modellen för att generera tillförlitliga resultat.

Utvecklingsplan:

1. Implementera indatamodulen för att korrekt behandla och förbereda data.
2. Bygga konduktör-AI för att klassificera indatat och välja rätt AI-modell.
3. Skapa mekanismen för AI-exekvering och resultatgenerering.

Framtida Utveckling:

- **Recursion och Augmentation:** Introducera mekanismer för att iterativt förbättra resultat genom att använda olika AI-modeller och justeringar i processen.
- **Transfer Learning:** En teknik där en modell tränad på en stor datamängd återanvänds och finjusteras för en specifik uppgift med mindre datamängd. Detta minskar träningskostnader och tid genom att dra nytta av befintlig kunskap.
- **Federated Learning:** En metod där modeller tränas över flera decentraliserade enheter eller servrar som innehåller lokala dataexempel, utan att dela data mellan enheterna. Detta förbättrar datasekretessen och möjliggör träning på känsliga data utan att de behöver överföras.

Nyttan med mindre AI-modeller

Energieffektivitet och miljöfördelar

I takt med att användningen av artificiell intelligens (AI) ökar, blir det allt viktigare att överväga de miljömässiga och energimässiga effekterna av dessa teknologier. Mindre AI-modeller erbjuder en rad fördelar när det gäller energieffektivitet och miljöpåverkan, vilket gör dem till ett attraktivt alternativ för hållbar utveckling.

Energieffektivitet

1. **Minskad Beräkningskraft:**
    - Mindre AI-modeller kräver betydligt mindre beräkningskraft jämfört med större modeller. Detta innebär att de kan köras på enklare hårdvara, vilket minskar energiförbrukningen per operation. Till exempel, en mindre modell kan ofta köras på en vanlig dator eller en mobil enhet, istället för att behöva kraftfulla GPU-kluster.
2. **Snabbare Träning och Inferens:**
    - Mindre modeller tränas och körs snabbare än större modeller. Detta reducerar den tid som datorer och servrar behöver vara igång, vilket i sin tur minskar den totala energikonsumtionen. Kortare träningstider innebär även att utvecklare kan iterera snabbare och effektivare, vilket ytterligare optimerar resursanvändningen.

Miljöfördelar

1. **Lägre Koldioxidutsläpp:**
    - Mindre energiförbrukning direkt översätts till lägre koldioxidutsläpp. Eftersom datacenter som kör AI-modeller ofta drivs av elektricitet från fossila bränslen, innebär varje watt som sparas en minskning av koldioxidutsläppen. Genom att använda mindre AI-modeller kan företag och organisationer minska sitt koldioxidavtryck och bidra till en mer hållbar framtid.
2. **Minskade Krav på Hårdvara:**
    - Mindre modeller kräver mindre avancerad och mindre energiintensiv hårdvara. Detta inte bara reducerar den omedelbara energiförbrukningen, utan minskar också behovet av att producera och underhålla högpresterande datacenter och hårdvarukomponenter, vilket ytterligare minskar miljöpåverkan.
3. **Lokal Bearbetning:**
    - Mindre modeller möjliggör lokal bearbetning av data, vilket minskar behovet av att skicka stora datamängder över nätverk till centrala datacenter. Lokal bearbetning minskar nätverksbelastningen och den energianvändning som är förknippad med datatransporter, vilket bidrar till en mer energieffektiv infrastruktur.

Praktiska Tillämpningar

- **Edge Computing:**
    - Mindre AI-modeller är idealiska för edge computing, där bearbetning sker nära datakällan, exempelvis på IoT-enheter eller smartphones. Detta minskar behovet av centraliserade resurser och ger realtidsanalys med minimal miljöpåverkan.

- **Optimering av Resurser:**
    - I scenarier där resurser är begränsade, som i utvecklingsländer eller avlägsna områden, kan mindre AI-modeller tillhandahålla avancerad teknik utan att belasta energisystemen.