**Skills**

**Programmiersprachen:**

* Python, R, C++, Java, SQL, JavaScript

**Technologien / Tools / Plattformen:**

* Pandas, Git, HTML, CSS, Docker, AWS EC2, AWS Sagemaker, Azure, Tableau, Power BI, Airflow, Git Actions

**Python Programmierung / Pakete:**

* **Data Science (Analysis):** pandas, scikit-learn, statsmodels, numpy, scipy, pytorch
* **Natural Language Processing / AI:** pytorch, Regex, openai, langchain, transformers, huggingface-hub, FAISS, tiktoken, bitsandbytes, peft, torchtext, torchvision, flash-attn
* **(Advanced) Clustering & Segmentation:** K-means, DBSCAN, hierarchical clustering.  
  Advanced: CLIQUE, SUBCLU, SCHISM, PROCLUS, DOC, P3C, ENCLUS, HICS, LSA
* **Automatisierung & Web Scraping:** scrapy, selenium, beautifulsoup4, lxml, urllib3, puppeteer, puppeteer-extra-plugin-stealth, Git Actions
* **Visualisierung & Plotting:** matplotlib, seaborn, pillow
* **Data Pipelines & Workflow Automatisierung:** Airflow, AWS EC2, Docker

**Statistische / ML-Modelle:**

* Large Language Models, Knowledge Models, Neural Networks, Dimensionality Reduction
* GPT fine-tuning, Mixtral 8x7b fine-tuning (with LoRa)
* Model evaluation with custom metrics
* Model training with SFTTrainer (transformers, trl, flash-attn)
* Linear Models, Generalized Linear Models, Mixed Effects Models
* Topic Models (PLSA, LDA, CTM, STM) for discovering hidden Topics/Schemas in Text
* Time Series Modeling, Discriminant Analysis, KNN Classification
* Clustering (K-means, DBSCAN, hierarchical clustering)

**Sprachen:**

* Deutsch (Muttersprache), Englisch (C1), Spanisch (B2)

**Projekthistorie**

Sparmahl   
**04/2024 - bis jetzt**

**Hybrides Produktklassifizierungsmodell mithilfe von einem Large Language-, Knowledge- und Vision Model**

(< 10 Mitarbeiter)

**Beschreibung:** In diesem Projekt haben wir ein innovatives, hybrides Produktklassifizierungsmodell entwickelt, das maschinelles Lernen nutzt, um Supermarktprodukte und Rezeptzutaten präzise in über 1.000 Kategorien einzuordnen. Durch den Einsatz von Natural Language Processing (NLP) und Deep Learning können wir selbst subtile Unterschiede zwischen Produkten erkennen und diese im richtigen Kontext einordnen. Dabei werden unter anderem ein Large Language-, Knowledge- und Vision Model benutzt um in unterschiedlichen Phasen Produkte zu klassifizieren. Außerdem wurde ein Vektorraum mit über 25.000 Produkten, wobei jedes Produkt als Vektor in 3072 Dimensionen dargestellt wird, erstellt und ein KNN Approach angewendet.  
  
**Tools & Libraries:**

* Programmiersprache: Python
* Machine Learning: OpenAI GPT-3.5 Turbo API, GPT-4o-mini API, RAG
* Pakete: Pandas, FAISS, langchain, fuzzy\_search

**01/2024 - bis jetzt**

**Automatisierte Crawler Erstellung für alle deutschen Supermarktketten**

**SparMahl**(Internet und Informationstechnologie, < 10 Mitarbeiter)

**Beschreibung:** In diesem innovativen Projekt wurde eine umfassende automatisierte Lösung zur Erstellung von Crawlern entwickelt, die ortsspezifische Angebote für über 30.000 Filialen führender deutscher Supermarktketten wie Rewe, Aldi Süd, Aldi Nord, Edeka, Penny, Netto, Kaufland und Lidl erfasst. Angesichts der Vielfalt an Angeboten und Produkten, die jeder Supermarkt bereitstellt, ist ein flexibler und skalierbarer Ansatz für das Scraping unerlässlich. Jeder Supermarkt verfügt über einzigartige Strukturen und Designs, weshalb für jeden einzelnen eine speziell angepasste Crawler-Lösung implementiert wurde. Durch die Automatisierung der Datenakquise können alle relevanten Angebote effizient erfasst werden, in der Regel innerhalb von 30 Minuten, während für Rewe aufgrund seiner Komplexität etwas mehr Zeit benötigt wird. Dieses Verfahren ermöglicht es, wertvolle Marktdaten ohne jegliche Kosten zu sammeln, was einen signifikanten Vorteil in der wettbewerbsintensiven Einzelhandelslandschaft darstellt.  
**Methoden:**

* **Crawler-Entwicklung:** Implementierung von spezialisierten Web- und API-Crawlern, die gezielt spezifische Angebote von den jeweiligen Supermarkt-Websites scrapen. Jeder Crawler wurde auf die individuellen Gegebenheiten der Supermärkte abgestimmt, um eine präzise Datenerfassung zu gewährleisten.
* **Automatisierung:** Nutzung von GitHub Actions zur vollständigen Automatisierung des Scraping-Prozesses. Dies stellt sicher, dass die Datenakquise ohne zusätzliche Kosten für die laufende Datenerfassung erfolgt.

**Tools & Libraries:**

* **Programmiersprachen und Bibliotheken:** Python, Requests, BeautifulSoup (bs4), Scrapy, Selenium, lxml, urllib3, Puppeteer, Puppeteer-extra-plugin-stealth
* **Entwicklungstools:** Regex für die Datenextraktion sowie Burp Suite und Postman für das Testen und Debuggen der Crawler und der APIs
* **Automatisierung:** GitHub Actions zur Planung und Ausführung der Scraping-Jobs

**03/2024 - 09/2024**

**Fine Tuning von LLMs zur Generierung von Fußball Spielberichten**

**Olympia-Verlag GmbH (Kicker App)**(Medien und Verlage, 50-250 Mitarbeiter)

**Beschreibung:** In diesem Projekt wurde das Fine-Tuning von Large Language Models (LLMs) untersucht, um qualitativ hochwertige Fußballspielberichte zu generieren. Es wurde ein proprietäres und ein open-source Modell mit GPT-3.5 und Mixtral 8x7b augewählt und Fine getuned, um Spielberichte für die Bundesliga und Oberliga im Stil des Kickers zu generieren. Ziel war es zu überprüfen, ob KI-Modelle in der Lage sind, ansprechende und informative Spielberichte zu erstellen, die den Erwartungen von Lesern und Sportfans gerecht werden. Die Berichte waren den journalistischen überlegen.  
**Methoden:**

* **Fine-Tuning mit Input-Output-Pairs:** Die GPT Modelle wurden mithilfe von Input-Output-Paaren trainiert, die eigenständig mit Crawlen und Queries erstellt wurden sind
* **Fine-Tuning mit LoRa (Low-Rank Adaptation):** Mixtral 8x7b wurde mithilfte von LoRa (eine Parameter Efficient Fine Tune Variante) fine getuned um viele Ressourcen zu sparen
* **Abschließende Umfrage:** Die Qualität der generierten Berichte wurde durch eine Umfrage, die sowohl Experten als auch Laien einbezog, bewertet, um Feedback zu erhalten und die Ergebnisse zu validieren.

**Tools & Libraries:**

* **Modelle:** GPT-3.5, Mixtral 8x7b
* **Trainingsumgebungen:** Microsoft Azure, Google Colab für LoRa-Fine-Tuning
* **Bibliotheken:** Transformers, PyTorch, SFTTrainer, LoRa, pandas, regex, requests,..

**Entwicklung eines fortschrittlichen neuronalen Netzwerkmodells für die Textgenerierung**

**Beschreibung:** In diesem Projekt haben wir ein umfassendes neuronales Netzwerkmodell entwickelt, das verschiedene moderne Architekturen integriert, um die Textgenerierung zu verbessern. Der Fokus liegt auf der Implementierung eines Multi-Layer Long Short-Term Memory (LSTM) Netzwerks für die Zeichenebene, der Entwicklung eines Multi-Head Attention Moduls, wie es in modernen Sprachmodellen verwendet wird, und der Einführung fortgeschrittener Schichten in Convolutional Networks. Dieses Projekt zielt darauf ab, die Stärken dieser Techniken zu kombinieren, um ein leistungsfähiges Modell für die Textgenerierung zu erstellen, das den Stil und die Struktur des Trainingsdatensatzes erlernen kann.  
**Komponenten:**

1. **Long Short-Term Memory (LSTM) Netzwerk:**
   * **Ziel:** Implementierung eines mehrschichtigen LSTM-Netzwerks, das darauf trainiert wird, das nächste Zeichen in einer Sequenz basierend auf einem gegebenen Textdatei-Input vorherzusagen.
   * **Methoden:** Verwendung von Rückkopplungsmechanismen, um sich Informationen über längere Sequenzen hinweg zu merken. Die LSTM-Architektur besteht aus Gedächtniszellen und Gates, die die Informationsübertragung steuern.
   * **Anwendung:** Nach dem Training kann das Modell genutzt werden, um Zeichen für Zeichen neuen Text zu generieren, der dem Stil des ursprünglichen Datensatzes ähnelt.
2. **Multi-Head Attention:**
   * **Ziel:** Entwicklung eines Multi-Head Attention Moduls, das in Large Language Models (LLMs) wie GPT-4 verwendet wird, um die Verarbeitung von Sequenzen zu verbessern und Kontextinformationen effektiver zu nutzen.
   * **Methoden:** Implementierung von Mechanismen, die es dem Modell ermöglichen, verschiedene Teile der Eingabesequenz parallel zu gewichten und zu kombinieren. Dies ermöglicht eine verbesserte Fokussierung auf relevante Teile des Textes während der Verarbeitung.
   * **Anwendung:** Die Integration des Multi-Head Attention Moduls in die LSTM-Architektur ermöglicht es dem Modell, sowohl die sequenzielle Struktur als auch den kontextuellen Inhalt der Texte zu erfassen.
3. **Convolutional Networks:**
   * **Ziel:** Implementierung fortschrittlicher Schichten, die in Convolutional Networks häufig verwendet werden, um die Effizienz und Leistung der Textverarbeitung zu steigern.
   * **Methoden:** Entwicklung und Integration verschiedener Schichten wie Convolutional Layer, Pooling Layer und Aktivierungsfunktionen in die bestehende Architektur. Diese Schichten verbessern die Fähigkeit des Modells, relevante Merkmale in den Eingabedaten zu extrahieren.
   * **Anwendung:** Die Kombination dieser Schichten mit LSTM und Multi-Head Attention trägt dazu bei, die Genauigkeit und Effizienz des Modells bei der Textgenerierung zu erhöhen.

**Tools & Libraries:**

* Programmiersprache: Python
* Frameworks: NumPy, PyTorch
* Entwicklungsumgebung: Jupyter Notebook
* **10/2022 - 12/2023**
* **Erstellung und Pflege von Datenpipelines**
* **Ionity GmbH**(Automobil und Fahrzeugbau, 50-250 Mitarbeiter)
* \*\*Projektbeschreibung: Erstellung und Pflege von Datenpipelines bei Ionity\*\*  
    
  Im Rahmen meiner Tätigkeit bei Ionity war ich maßgeblich an der Erstellung und Pflege mehrerer komplexer Datenpipelines beteiligt, die eine zentrale Rolle in der Datenverarbeitung und -analyse spielten. Eines meiner Hauptprojekte bestand in der Zusammenführung aller Kundendaten aus unterschiedlichen Datenquellen, wie CRM-Systemen und IoT-Daten, zu einer konsistenten Datenbasis. Hierbei habe ich eine hochperformante Datenpipeline entwickelt, die vollständig in Python programmiert wurde.  
    
  Um dies zu realisieren, habe ich Apache Airflow zur Orchestrierung der Datenprozesse und AWS EC2 zur Bereitstellung von Rechenkapazitäten genutzt. EC2 ermöglichte dabei eine skalierbare Infrastruktur, die es uns erlaubte, große Datenmengen effizient zu verarbeiten. Die Daten wurden in einem Data Lake auf AWS S3 gespeichert, was die Verwaltung und Verarbeitung der strukturierten und unstrukturierten Daten wesentlich vereinfachte.  
    
  Im Zuge der Pipeline-Entwicklung habe ich umfassende Datenmodellierungen durchgeführt, um die Daten aus den verschiedenen Quellen für analytische Zwecke optimal aufzubereiten. Dazu gehörten unter anderem die Definition von Entity-Relationship-Modellen (ER-Modelle) sowie die Normalisierung und Transformation der Daten für weitere Analysen.  
    
  Weitere wichtige Projekte waren die Implementierung und Pflege der Ladevorgangsdaten- und Fehlerdaten-Pipeline.   
    
  Neben der reinen Datenintegration gehörte auch die Verwaltung von Zugriffsrechten, die Implementierung von Sicherheitsmechanismen, das Monitoring der Pipeline sowie die regelmäßige Optimierung der Performance zu meinen Aufgaben. Zudem wurden ETL-Prozesse (Extract, Transform, Load) und Partitionierungsstrategien für eine effiziente Datenverarbeitung implementiert.  
    
  Durch die Automatisierung und Wartung dieser Pipelines konnte Ionity wertvolle Erkenntnisse aus den Daten gewinnen, was die Geschäftsprozesse optimierte und die Entscheidungsfindung verbesserte.  
    
  Verwendete Technologien und Tools:  
  - Python  
  - Apache Airflow  
  - AWS (EC2, S3, Sagemaker,..)  
  - SQL  
  - Datenmodellierung (ER-Diagramme, Normalisierung)  
  - API-Integration  
  - ETL-Prozesse
* **07/2023 - 10/2023**
* **Customer Segmentation**
* **Ionity GmbH**(Automobil und Fahrzeugbau, 50-250 Mitarbeiter)
* **Kategorisierung der Ionity-Kunden in verschiedene Segmente zur Entwicklung maßgeschneiderter Abonnementmodelle**: Ionity betreibt ein weitreichendes Netzwerk von Schnellladesäulen für Elektroautos in ganz Europa und bedient dadurch eine heterogene Kundschaft mit unterschiedlichen Bedürfnissen. Ziel des Projekts war es, die Kunden in spezifische Gruppen zu unterteilen, um geeignete Abonnementstufen zu entwickeln, die besser auf die individuellen Anforderungen und Nutzungsmuster der Nutzer abgestimmt sind. Die Kategorisierung erfolgte mithilfe verschiedener Clustering-Methoden sowie durch Dimensionsreduzierung mittels PCA (Hauptkomponentenanalyse)