Ein Bild, das Text, Schrift, Visitenkarte, Logo enthält.

Automatisch generierte Beschreibung

Duale Hochschule Baden-Württemberg Center for Advanced Studies

Laborarbeit

Modul Big Data Engineering (W3M20027)

Verfasser: Cenk Yagkan ()

Felix Schneider ()

Niko Sauter (9755750)

Nikola Mihajlovski ()

Professor: Prof. Dr.-Ing. Dennis Pfisterer

Abgabedatum: 19.01.2024

Inhalt

[1 Use Case Beschreibung 1](#_Toc153893065)

[2 Aufbau 1](#_Toc153893066)

[2.1 Generation 1](#_Toc153893067)

[2.2 Ingestion 1](#_Toc153893068)

[2.3 Stream Processing Layer 1](#_Toc153893069)

[2.4 Serving Layer 1](#_Toc153893070)

[3 Reflexion 1](#_Toc153893071)

# Use Case Beschreibung

Twitter-Clone -> Auswertung verwendeter Wörter

[Sentiment140 dataset with 1.6 million tweets (kaggle.com)](https://www.kaggle.com/datasets/kazanova/sentiment140/data)

# Aufbau

Dieses Kapitel beginnt mit einer Beschreibung der Architektur. Die Unterkapitel gehen daraufhin auf die einzelnen Bestandteile ein.

Die gesamte Applikation wird über ein Helm-Chart und Skaffold gesteuert?

[Big Data Engineering - Application Deployment (farberg.de)](https://farberg.de/talks/big-data-eng/?03c%20-%20Application%20Deployment.md#/20)

Big Data Lifecycle inkl. unserer gewählten Ansätze und Produkte?

## Generation

Posts etc. werden generiert

In unserem Beispiel wird die Echtzeitgenerierung simuliert. Ein Python-Programm schreibt vorgegebene Posts aus einer Textdatei periodisch in den Ingestion Layer.

Das simulierte Quellsystem pusht die Daten zu Big Data Analyse Systemen. Dieses Vorgehen ist besonders geeignet für Real-Time-Anforderungen, wie es unser Use Case verlangt.

Publisher und Subscriber: [Big Data Engineering - Big Data: Data Ingestion Layer (farberg.de)](https://farberg.de/talks/big-data-eng/?04d%20-%20Data%20Ingestion%20Layer.md#/24)

Logfiles haben folgendes Format (Eine Zeile ist ein JSON-File):

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Datum | User | Zeit |

## Ingestion

Hierfür wird Apache Kafka auf HDFS verwendet um als Big Data Message Queue System die Daten in fast Real-Time aufzunehmen und die Weiterverarbeitung zu ermöglichen. Dies ist für unseren Use Case von der Verarbeitungsgeschwindigkeit ausreichend und zusätzlich haben wir noch ein paar historische Daten zur Verfügung. Die Entkopplung von Erzeuger und Abnehmer nutzen wir so ebenfalls.

Kafka: [Big Data Engineering - Big Data: Data Ingestion Layer (farberg.de)](https://farberg.de/talks/big-data-eng/?04d%20-%20Data%20Ingestion%20Layer.md#/21)

## Stream Processing Layer

Als Trigger für unsere Microbatches wählen wir ein festes Intervall. Es werden die Zwischenergebnisse der einzelnen Microbatches aufsummiert. Diese werden einer Datenbank gespeichert. Structured Streaming wird somit angewendet. Apache Spark wird verwendet, da es diese Microbatch-Anforderungen gut umsetzen kann. Ebenfalls wird YARN als Vermittler zwischen der Speicher-Schicht mit Kafka und Spark eingesetzt.

[Big Data Engineering - Big Data: Spark: Structured Streaming (farberg.de)](https://farberg.de/talks/big-data-eng/?05d%20-%20Structured%20Streaming.md#/8)

## Serving Layer

Als Output-Mode wählen wir „Update Mode“. Aufgrund der Datenaggregation auf Tagesbasis können wir Append als Möglichkeit ausschließen.

MariaDB? Schema? ConfigMap um Datenbank zu konfigurieren: [Big Data Engineering - Application Orchestration (farberg.de)](https://farberg.de/talks/big-data-eng/?03b%20-%20Application%20Orchestration.md#/27)

|  |  |  |
| --- | --- | --- |
| Wort | Datum | Anzahl |
| #Elon | 18.12.2023 | 2 |
| #Tesla | 18.12.2023 | 5 |
| #Elon | 19.12.2023 | 15 |

Man könnte Wordcloud aus Daten in der Datenbank erstellen.

# Reflexion

Herausforderungen, Lernerfahrungen