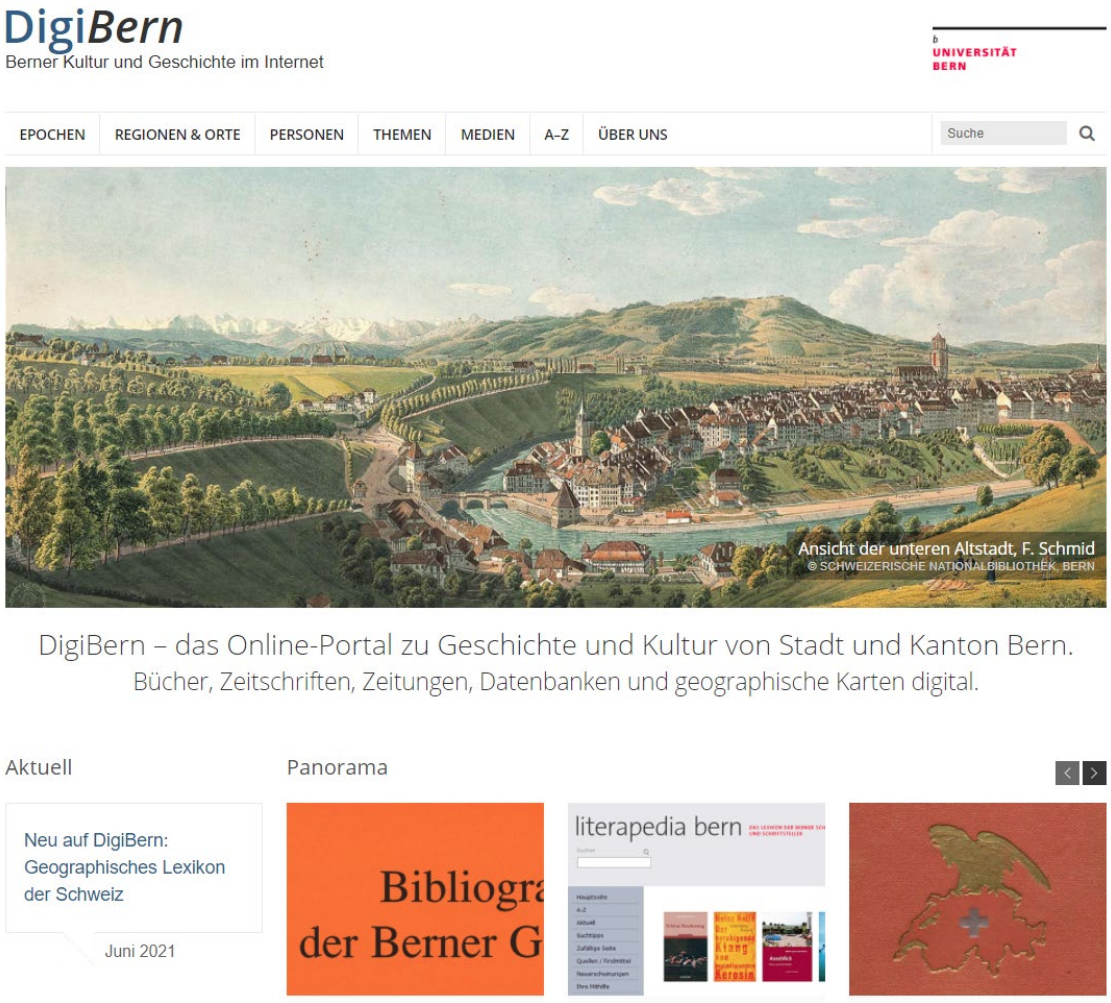
Volltextsuchmaschine für digitalisierte Bernensia

Semesterprojekt CAS Big Data

Kathi Woitas, Universitätsbibliothek Bern

2021-10-05

Worum geht es?



Worum geht es?

DigiBern

- Online-Portal zu Geschichte und Kultur von Stadt und Kanton Bern = *Bernensia*
- „Verzeichnis“ über digitale/digitalisierte Bücher, Zeitschriften, Zeitungen, Datenbanken und Karten
- Technologie: Drupal WCMS

Herausforderung

- Erschliessung nur über Kategorien und Website-Suche
- Volltexte liegen auf anderen Plattformen (e-rara, e-periodica) oder direkt im WCMS
- Keine Volltext-Suche über die Digitalisate möglich

Motivation

- zielgerichtetes Retrieval der Digitalisate verbessern

Zielsetzung

Implementierung einer Volltextsuchmaschine über die „externen“ Digitalisate von DigiBern

- Proof-of-Concept: funktionierende Suche per API
- Aggregation und Vorverarbeitung der Volltexte und Metadaten

Ausgangsdaten (ca. 570 e-rara, ca. 1000 e-periodica)

- Bibliografische Metadaten: XML per OAI-Schnittstellen
- Volltexte: Websites (e-rara: PDF und z.T. TXT; e-periodica: PDF)
- beides: HTTP

Implementierung + Konfiguration Search Engine

- Elasticsearch (Document Store mit Apache Lucene Suchindex)
- Explizites Mapping der Datenfelder mit Analyzern

Implementierung – Komponenten

Distributed Data Processing	Apache Spark (PySpark) • DataFrame API • Spark ML
Data Storage	AWS S3
Document Store/Index	Elasticsearch
Text Extraktion/OCR	Apache Tika

Implementierung

Set up «Lokal»

- Jupyter/all-spark-notebook in Docker Container, Pyspark
- AWS S3
- Tika als freier Web-Dienst verfügbar
- (Docker Container Elasticsearch ausprobiert, Laptop zu schwach)

Set up «Cloud»

- AWS Lightsail Ubuntu Instance (Docker Compose) mit Jupyter/all-spark-notebook, Elasticsearch, Tika, Kibana, etc.
- AWS S3

Data Preprocessing

Metadaten:

- Download als XML, Transformation zu JSON und Upload zu S3
- Merge mit Volltexten, Upload zu S3
- Jupyter Notebook

Volltexte als PDF – zusätzlich:

- Download nach S3: Shell Script
- Text-Extraktion und Upload TXT zu S3: Shell Script
- OCR (testweise, verworfen)

Volltexte als TXT

- Download
- Merge mit Metadaten in JSON, Upload zu S3: s.o.

Text Processing & Ingest Elasticsearch

Spark/PySpark:

- Load JSON von S3 in Spark DataFrame
- Transformationen auf Felder
- NER mit NLTK und spaCy via UDF

Ingestion Elasticsearch

- Mappings
- direkt von Spark DataFrame mit *Elasticsearch for Apache Hadoop and Spark* "spark.jars.packages", "org.elasticsearch:elasticsearch-spark-30_2.12:7.12.0"

Alternativen, Tuning

Code

- Pipeline, Notebooks -> Python Scripts

Data Sources

- i.M. manuell, da Inkremente in grösseren Abständen -> CRON job

Data Preprocessing & Text Processing

- Text-Extraktion distributieren
- OCR-Modul mit Ground-Truth-Erstellung und Training einbinden
- spezifischere NLP/NER einsetzen

+ Steigende Datenvarianz bei Einbezug von mehr Quellen…

Vielen Dank für die Aufmerksamkeit!

Kathi Woitas
kathi.woitas@students.bfh.ch

Universitätsbibliothek Bern
Digital Scholarship Services