Book Recommender using NLP

Carsten Lydeking

Zealand Business College

Oral Exam – AI and ML, 4th Semester

Motivation and Problem

- **Growing privacy and cost concerns** with cloud-based systems
- Traditional systems require **user profiles and data collection**
- **Goal:** Build a fully offline, content-based book recommender
- **Research Question:**

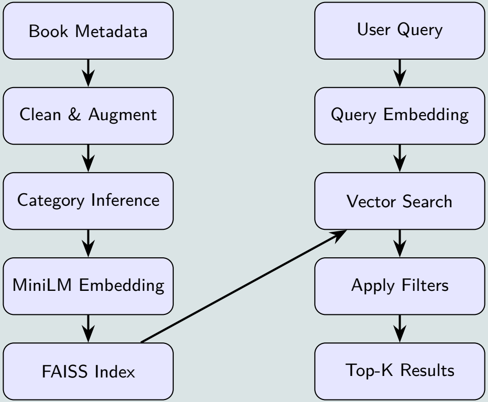
How can a local ML model recommend books based on natural language descriptions?

- **Key Requirements:** No external APIs, no user tracking, semantic understanding

Core Components:

- **Data Pipeline:** Clean & augment book metadata
- **Category Inference:** Zero-shot classification + fallback rules
- **Semantic Embedding:** `MiniLM` sentence transformers
- **Vector Search:** `FAISS` similarity matching
- **Local UI:** `Streamlit` interface

Key Innovation: Fully local semantic search without cloud dependencies

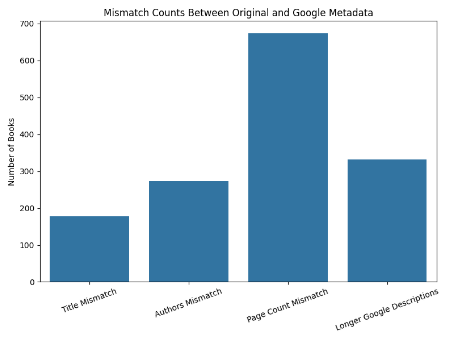


Data Quality Challenges:

- Started with **6,800 books** from multiple sources
- **Issues found:**
  - Missing author/category information
  - Very short descriptions (< 9 words)
  - Inconsistent categorization across sources

Data Engineering Solutions:

- **API enrichment:** OpenLibrary & Google Books
- **Quality filtering:** Remove inadequate descriptions
- **Final dataset:** 5,160 high-quality books



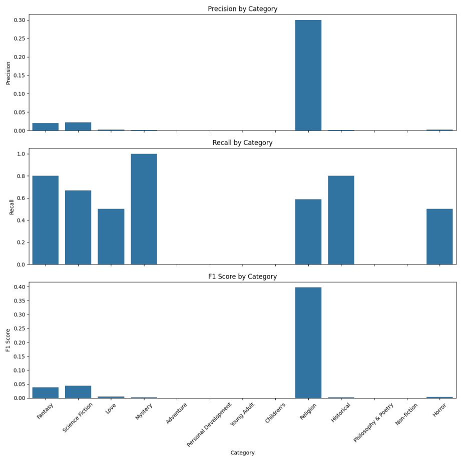
Data inconsistencies across sources required systematic cleaning and validation

Two-Tier Classification Approach:

- **Primary:** Zero-shot classification with `BART-MNLI`
  - No training data required
  - 13 predefined book categories
  - Confidence scoring for predictions
- **Fallback:** Rule-based keyword matching
  - When confidence $<$ threshold
  - Genre-specific keyword patterns

Quality Control:

- Description length $\geq$ 200 chars
- Average confidence $\geq$ 0.2
- Maximum confidence $\geq$ 0.4



Results focus on high-confidence predictions rather than perfect recall across all categories

Semantic Embedding & Vector Search

Sentence Embedding with `MiniLM`:

- **Model:** `all-MiniLM-L6-v2`
- **Input format:**
  "Title: ... Author: ... Description: ..."
- **Output:** 384-dimensional vectors
- **Advantage:** Semantic similarity beyond keywords

Vector Search with `FAISS`:

- **Index:** 5,160 book embeddings
- **Search:** L2 distance (exact search)
- **Performance:** $<$ 10ms query time
- **Local:** No external dependencies

Query: "space exploration novels"

↓

MiniLM Transformer

↓

Vector: $\vec{v} \in \mathbb{R}^{384}$

↓

FAISS L2 Distance Search

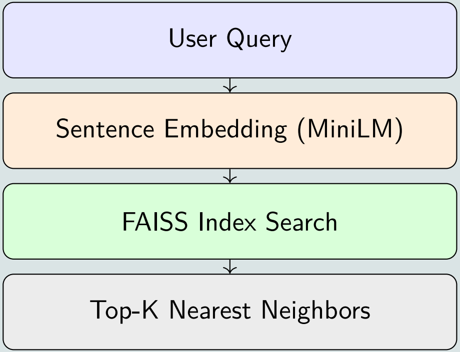
↓

Top-K Similar Books

End-to-end semantic search in $<$ 200ms

Vector Similarity Search with FAISS

- Used **Facebook AI Similarity Search (FAISS)** library
- Performs **L2 distance** search in embedding space
- Index built with:
  - 5160 book embeddings ($\vec{v} \in \mathbb{R}^{384}$)
  - Exact search (IndexFlatL2)
- At runtime:
  - User query is embedded
  - Top-k nearest neighbors retrieved
  - Results shown in UI
- **Fully local**, fast search



Implementation Strengths:

- **Privacy-preserving** by design
- **Semantic understanding** beyond keyword matching
- **Lightweight** - runs on consumer hardware
- **Modular architecture** for easy extension

Current Limitations:

- **No personalization** - stateless by design
- **Dataset scope** - 5,160 books vs. commercial scale
- **Cold start problem** for new books
- **No feedback learning** - static recommendations

Future Research Directions:

- **Hybrid approach:** Combine content-based with collaborative filtering
- **Better embeddings:** Experiment with domain-specific models
- **Privacy-preserving personalization:** Local user preference learning
- **Multi-modal features:** Include cover images, genre embeddings

"Demonstrates that local-first ML - or using a more popular term - edge AI, can provide meaningful semantic recommendations without compromising user privacy or requiring cloud infrastructure."

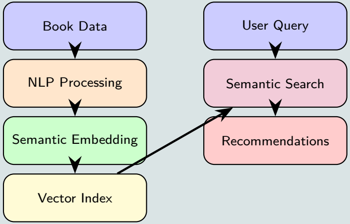
Research Question Answered:

How can a local ML model recommend books based on natural language descriptions?

Solution Implemented:

- **Semantic embeddings** with MiniLM transformers
- **Vector similarity search** using FAISS
- **Zero-shot classification** for categorization
- **Privacy-first design** - fully local processing
- **Key Contributions:** *Proof-of-concept that modern NLP enables practical, privacy-preserving recommendation systems*

System Architecture:



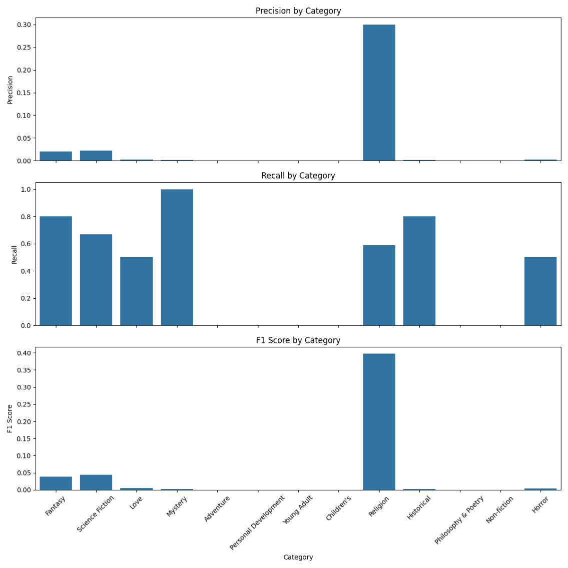
Impact & Applications:

- Educational tool for privacy-aware ML
- Foundation for local-first recommendation systems
- Demonstrates transformer accessibility on consumer hardware

Thank you for your attention!

Questions?

Backup: Per-Category Metrics



Backup: Fallback Keywords

- Fallback used when zero-shot model confidence was low
- Example keywords:
  - **Fantasy**: magic, wizard, dragon
  - **Science Fiction**: space, AI, dystopia
  - **Love**: romance, passion, relationship
  - **Mystery**: detective, clue, crime

Backup: Performance

- Embedding time per book: $\approx$ 2 ms (batch embedding)
- Query embedding: $\approx$ 50-200 ms
- FAISS search: $<$ 10 ms
- UI render time: $\approx$ 1-2 seconds (including image loading)
- All processing fully local on consumer-grade laptop

Example Query 1:

"Books about artificial intelligence and ethics"

Top Results:

- "The Alignment Problem" - Brian Christian
- "Life 3.0" - Max Tegmark
- "Weapons of Math Destruction" - Cathy O'Neil

Example Query 2:

"mystery novels with unreliable narrators"

Top Results:

- "Gone Girl" - Gillian Flynn
- "The Girl on the Train" - Paula Hawkins
- "In the Woods" - Tana French

What This Demonstrates:

- **Semantic understanding** beyond keywords
- **Abstract concept matching** (ethics, unreliable narrators)
- **Cross-genre discovery** potential

Evaluation Challenges:

- No ground truth for "perfect" recommendations
- Subjective nature of book preferences
- **Solution:** Focus on semantic relevance rather than prediction accuracy

Discussion Starters:

- How could one evaluate recommendation quality
- Books with sparse descriptions
- Could this approach work for other domains?