Data Mining Recommendation Systems Part 2

Joseph Burdis
Fall 2024
CUNY Graduate Center

Topic List

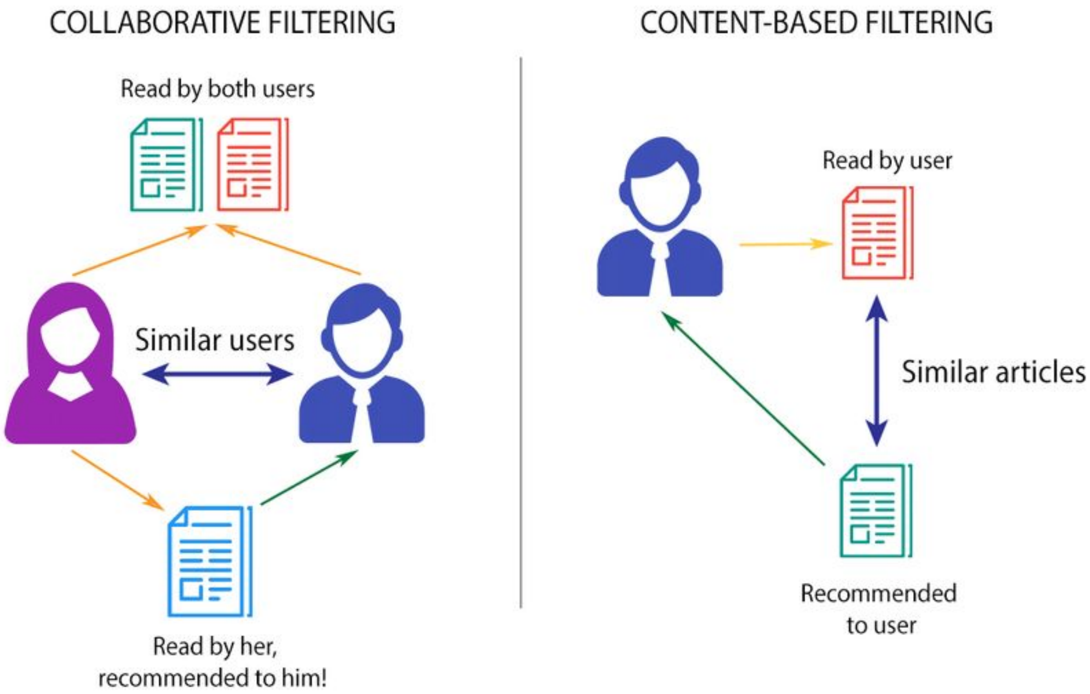
- Midterm
- Recap of Content-based and Collaborative Filtering
- Intro to ML in Recommendation Systems
- Click-Through Rate Models
- Deep Learning for Recommendation Systems

Midterm Topics

- Association Rules vs Predictive Modeling:
    - E.g. What questions does each answer? Similarities and differences? Which can handle information about a product directly? Which can handle broad questions like which items are bought together often?
- What are the map and reduce steps in distributed file systems (Hadoop)? How is this used for frequent itemsets?
- Experimental Design for Modeling
    - How do you mitigate overfitting and risk that your model will do badly when used on new data ("in production")?
    - Match these partitioning methods to use with the following modeling tasks?
- Describe the PageRank algorithm. Describe BM25. What are the goals of each and how might they be used together to build a search engine? What is tf-idf and bag-of-words?
- Transformers, Generative AI and LLMs
    - Describe Retrieval Augmented Generation. How are LLMs and embedding models used? How could you combine BM25 and embedding models to improved the retrieval process?
    - Encoder vs Decoder. Which is used for generative AI / LLMs? Which is used for document similarity? Masking in encoder vs decoder. What is the purpose of the masked attention mechanism for the decoder?
- Collaborative filtering vs content-based filtering.

Recap of Content-based and Collaborative Filtering

Recap



https://www.researchgate.net/figure/Content-based-filtering-vs-Collaborative-filtering-Source_fig5_323726564

Collaborative Filtering Recap

A technique that makes automatic predictions about the interests of a user by collecting preferences from many users (collaborating).

Data Used: Relies solely on user-item interaction data (e.g., ratings, clicks).

Types of CF:

- **Memory-Based CF:**
    a. User-based and item-based nearest neighbor methods.
    b. Covered in simple examples earlier in last week's slides and top of notebook
- **Model-Based CF:**
    a. Matrix factorization, singular value decomposition (SVD).
    b. **Covered at end of slides and pyspark section of notebook (ALS)**

Collaborative Filtering: Matrix Factorization

Matrix factorization is a simple embedding model. Given the feedback matrix A $\in R^{m \times n}$, where m is the number of users (or queries) and n is the number of items, the model learns:

- A user embedding matrix $U \in \mathbb{R}^{m \times d}$, where row i is the embedding for user i.
- An item embedding matrix $V \in \mathbb{R}^{n \times d}$, where row j is the embedding for item j.

https://developers.google.com/machine-learning/recommendation/collaborative/matrix

Pros and Cons of Collaborative Filtering

- \+ No feature selection needed
- \+ No domain knowledge needed
- \- Cold start: how to find match for new users/items
- \- Sparsity: rating matrix is sparse degrading recommendation quality
- \- Popularity bias: tend to recommend popular items
- \- Cannot recommend items to someone with unique taste (grey sheep problem)

ML for Recommendation Systems

ML for Recommendation System

Best of both worlds!

Data Used: Incorporate additional features beyond user-item interactions, such as user demographics, item attributes, and contextual information.

Capabilities:

- **Feature Integration:** Can handle various data types and incorporate side information.
- **Modeling Complex Patterns:** Capture nonlinear relationships and interactions between features.
- **Cold Start Problem Mitigation:** Can mitigate cold start by using available features

Click-Through Rate Models

Definition of CTR: Traditionally predict CTR = (Number of Clicks) / (Number of Impressions) and measures how often users click on recommended items.

Importance in Recommendations:

- **User Engagement Metric:** High CTR indicates that recommendations are relevant and engaging.
- **Business Impact:** Directly affects revenue in ad-based and e-commerce platforms.
- **Feedback Loop:** User clicks provide data to improve future recommendations.

Applications:

- **Personalized Advertising:** Serving ads that users are more likely to click.
- **Content Recommendations:** Suggesting articles, videos, or products to users.

Common Algorithms:

1. **Logistic Regression:** Interpretable coefficients, fast to train.
2. **Decision Trees:** Easy to visualize, can handle nonlinear relationships.
3. **Random Forests:** Reduces overfitting, handles large datasets.
4. **Gradient Boosting Machines (e.g., XGBoost, LightGBM):** High predictive accuracy, handles missing data well.

DeepFM (Deep Factorization Machines)

DeepFM: Combines the strengths of Factorization Machines (Efficiently model pairwise feature interactions) and DL (Capture high-order, nonlinear interactions) in a unified model.

Factorization Machines:

$$y_{\mathrm{FM}} = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle v_i, v_j \rangle x_i x_j$$

where v_i and v_j are latent vectors.

DeepFM Architecture:

- **Shared Embedding Layer:**
  - Embeddings are used by both FM and deep components.
- **FM Component (Wide Part):**
  - Captures low-order feature interactions efficiently.
- **Deep Component:**
  - Stacks multiple layers to model high-order interactions.

Advantages:

- **No Manual Feature Engineering:** Automatically captures feature interactions.
- **Efficient Learning:** Shares embeddings between components, reducing parameters.

Attention Factorization Machines

To be updated

FT-Transformer

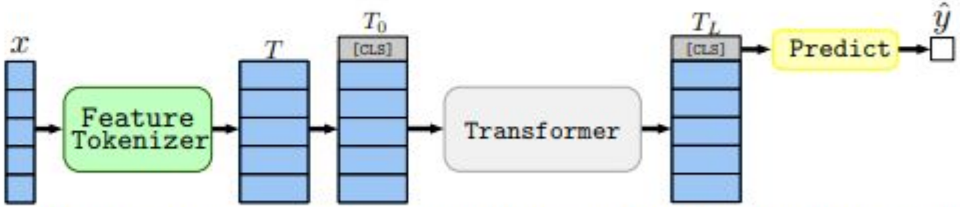


Figure 1: The FT-Transformer architecture. Firstly, Feature Tokenizer transforms features to embeddings. The embeddings are then processed by the Transformer module and the final representation of the [CLS] token is used for prediction.

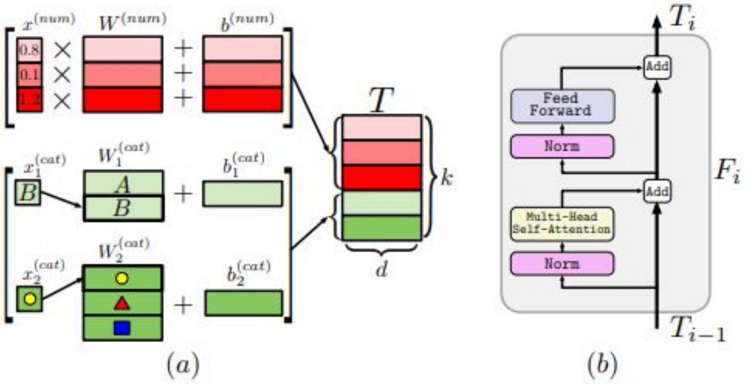


Figure 2: (a) Feature Tokenizer; in the example, there are three numerical and two categorical features; (b) One Transformer layer.

https://arxiv.org/abs/2106.11959