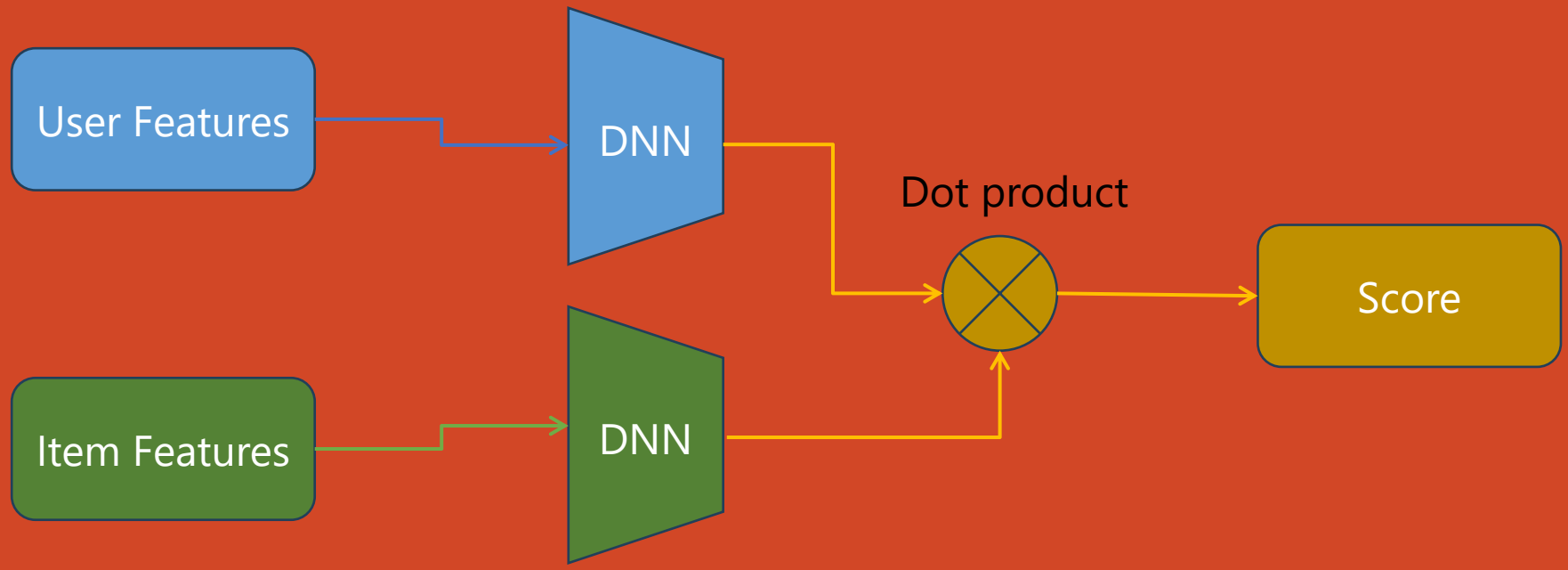
Two-Tower Embedding Model

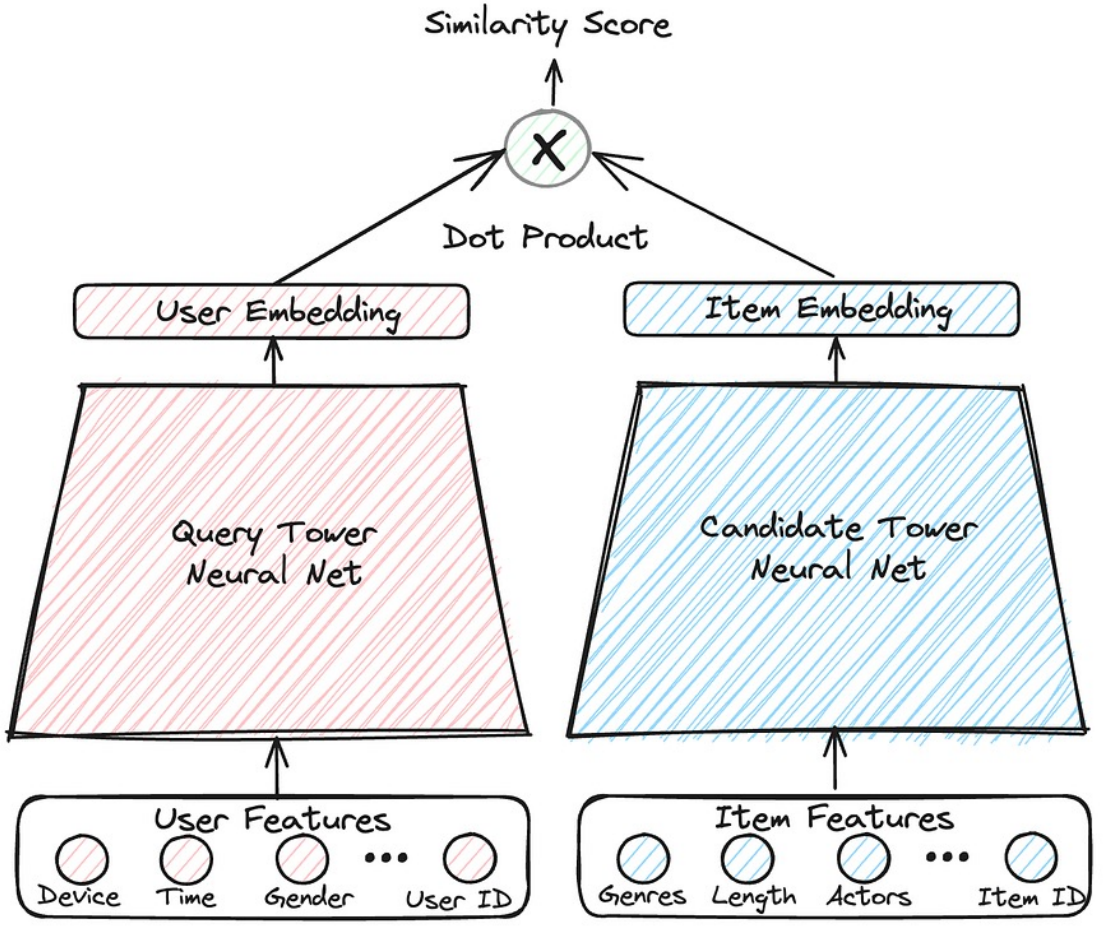
A Simplified Approach to Similarity Learning



Group 9: Mohamed Achour & Emile Le Gallic | date: 10.01.2025

What is the Two-Tower Embedding Model?

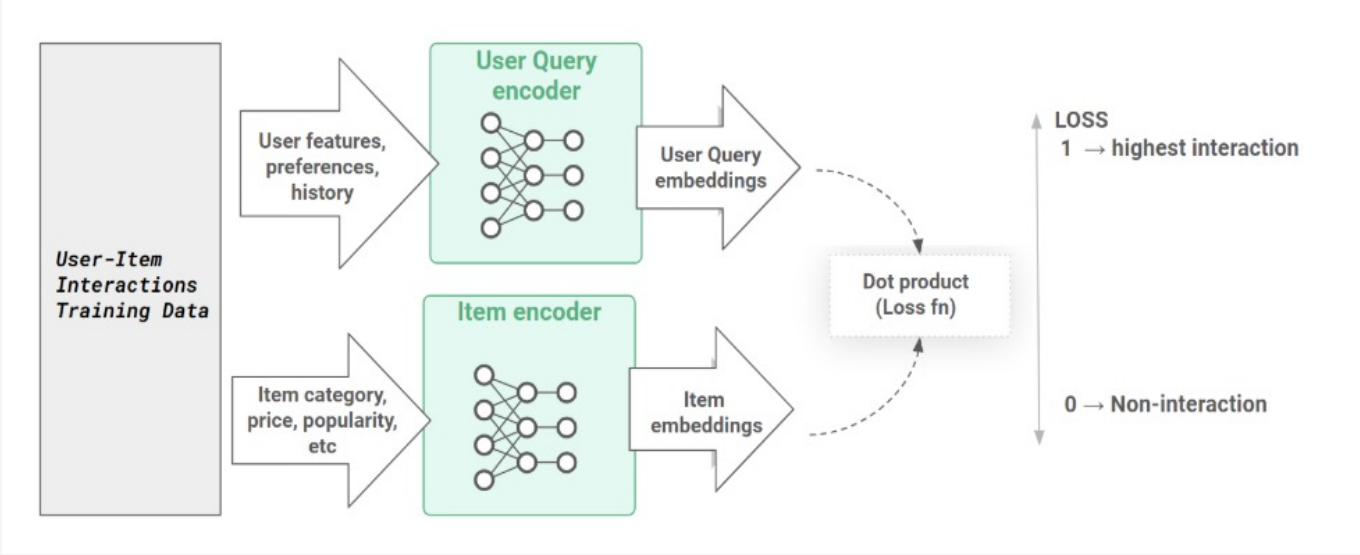
- The Two-Tower Embedding Model is a neural network architecture used for learning representations of two different inputs (e.g., queries and documents) in a shared embedding space. It offers a unique approach to handle complex data relationships
- Most two-tower architectures are used for personalized search or recommendations, where there are queries to match with items
- One of the first applications of this networks for recommendations can be found in a paper about YouTube.



Source: link

Architecture Overview

- **Two-Tower Structure:**
  Separate neural networks (towers) for two input types (e.g., user queries and documents).
- **Embedding Generation:**
  Each tower encodes its input into a fixed-size embedding vector.
- **Similarity Calculation:**
  Embeddings from both towers are compared using a similarity metric (e.g., cosine similarity or dot product).
- **Decoupled Training:**
  Towers are trained simultaneously but independently, ensuring flexibility.



Source: link

Advantages and challenges

Advantages:

Scalability: Since embeddings for one tower can be precomputed and stored, the model scales efficiently to large datasets. During inference, only the embeddings for the query or user need to be computed, reducing computational overhead.

Flexibility: Adaptable to various data types (text, images, etc.) and supports diverse use cases like search ranking, recommendation systems, and cross-modal retrieval.

Efficiency: By comparing embeddings in a low-dimensional space, the Two-Tower Model significantly reduces the complexity of similarity computations. This is especially beneficial in scenarios requiring real-time recommendations

Challenges:

Alignment: Ensuring embeddings from the two towers reside in the same semantic space can be challenging and this poor alignment can lead to inaccurate similarity scores

Training complexity: The choice of the loss function and training strategies is critical to achieve meaningful embeddings.

Cold start: If one tower (e.g., user data) relies on embeddings generated from sparse or unseen inputs, the model may perform poorly.

Integration of the Two-Tower Model in our Project

Architecture of the system:

- User tower: Encodes the user's Spotify profile data (playlist features, audio attributes) into an embedding.
- Candidate Tower: Encodes the profiles of other Spotify users into embeddings.
- These embeddings are compared in a shared space to identify similar users. Both towers are trained to map inputs to the same semantic space.

Similarity Metric:

- Using cosine_similarity to efficiently compute similarity scores between embeddings

Dynamic Updates:

- The Two-Tower Model allows new profiles to be integrated seamlessly, improving the recommendation system over time.