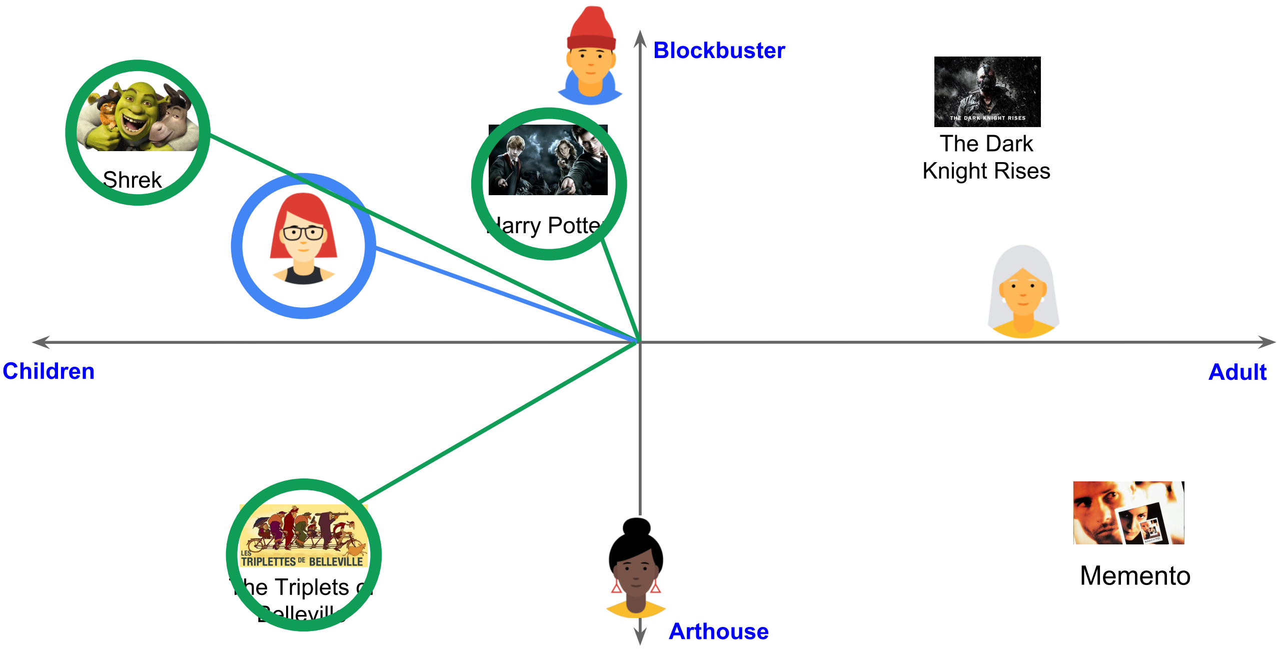
Retrieval

Suppose you have an embedding model. Given a user, how would you decide which items to recommend?

At serve time, given a query, you start by doing one of the following:

- For a matrix factorization model, the query (or user) embedding is known statically, and the system can simply look it up from the user embedding matrix.
- For a DNN model, the system computes the query embedding $\psi(x)$ at serve time by running the network on the feature vector x.

Once you have the query embedding q, search for item embeddings V_j that are close to q in the embedding space. This is a nearest neighbor problem. For example, you can return the top k items according to the similarity score $s(q, V_j)$.



You can use a similar approach in related-item recommendations. For example, when the user is watching a YouTube video, the system can first look up the embedding of that item, and then look for embeddings of other items V_j that are close in the embedding space.

Large-scale Retrieval

To compute the nearest neighbors in the embedding space, the system can exhaustively score every potential candidate. Exhaustive scoring can be expensive for very large corpora, but you can use either of the following strategies to make it more efficient:

- If the query embedding is known statically, the system can perform exhaustive scoring offline, precomputing and storing a list of the top candidates for each query. This is a common practice for related-item recommendation.
- Use approximate nearest neighbors.

Previous
← Softmax Training (/machine-learning/recommendation/dnn/training)

Next
Scoring (/machine-learning/recommendation/dnn/scoring) →

Last updated 2020-02-10 UTC.