Neural Recsys and Side Information

STAT3009 Recommender Systems

by **Ben Dai** (CUHK-STAT)
on **November 21, 2024**

» Recall: Neural Networks

Using TF2.0 to Implement Your Own Model

- M — Define your Model mathematically
  - * Motivation, EDA, input/output, parameters/hyperparameters, etc.
- T — Translate your model into a neural network
- F — Loss function, regularization
- OPT — Optimizer, cross-validation, early stopping, etc.

Implementation

- Build — Define the model using **Keras**
  - * Layers and path from input to output
- Compile — Compile your model with keras.losses, keras.optimizers, and keras.metrics
- Fit — Train your model with data and hyperparameters
- Pred — Make predictions using model.predict

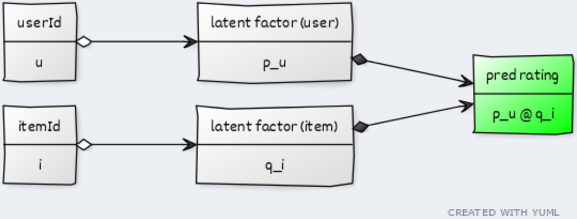
Can we reformulate MF as a neural network?

» Steps: MF to NN

- **MF** model:

$$\hat{r}_{ui} = \mathbf{p}_u^\top \mathbf{q}_i$$

  - Input user-item pair: (u, i) $\rightarrow$ Output inner product: $\mathbf{p}_u^\top \mathbf{q}_i$
- Translate your model to a neural network



» MF: Cold-Start Issue

Cold-Start Issue:

Problem Completely new users/items in recommender systems

Prediction Existing methods:

- User/Item-Average Unable to compute a meaningful average
- Correlation Cannot compute distance
- MF/SVD For example, if user u has no ratings in the system ($\mathcal{I}_u = \emptyset$):

Conclusion For cold-start users/items, most existing methods **fail** to provide a solution.
We NEED side information!

» MovieLens: Side Information

YES, we have side information.

For example, in the MovieLens dataset:

User USER_ID, AGE, GENDER, OCCUPATION, ZIP_CODE, RATING_MEAN, RATING_COUNT, RATING_QUANTILE, etc.

Item ITEM_ID, DATE, GENRE, RATING_MEAN, RATING_COUNT, RATING_QUANTILE, etc.

Other DATE

Key Message:

* Exploratory Data Analysis (EDA) in MovieLens indicates that side information is critical.
* Side information is promising for solving cold-start issues.
* How to incorporate side information to build a new recommender system.

» Conclusion from EDA (MovieLens)

Insights from EDA in MovieLens:

- **Side information** or user/item features are critical for predicting ratings: $f(\mathbf{x}_u, \mathbf{z}_i) \to r_{ui}$
- Personalization/itemization remains important even for users/items with similar features—MF terms are necessary: $f(\mathbf{x}_u, \mathbf{z}_i) + \mathbf{p}_u^\intercal \mathbf{q}_i \to r_{ui}$

→ **Joint effect** of features should be considered in recommender system modeling:

$$f\big(\mathbf{p}(u, \mathbf{x}_u), \mathbf{q}(i, \mathbf{z}_i)\big) \to r_{ui}$$

» MovieLens: Side Information

Update our setting:

Rating [userID, itemID, rating]: (u, i, r_{ui})

Side Info
- User [continuous_features, categorical_features]
- Item [continuous_features, categorical_features]

Examples
- Continuous [AGE, RATING_MEAN, RATING_COUNT, RATING_QUANTILE]
- Categorical USER_ID + [GENDER, OCCUPATION, ZIPCODE]

Train [userID, itemID, userFeatures, itemFeatures, rating]

Test [userID, itemID, userFeatures, itemFeatures, ?]

» Pre-processing: Side Information

Cont Continuous Features

- Type → float
- Process: Standardize features
- Python: SKLEARN.PREPROCESSING.STANDARDSCALER

Cate Categorical Features

- Type → int
- Process: Label encoding
- Python: SKLEARN.PREPROCESSING.LABELENCODER

Cont/Cate Can be continuous or categorical

- Example: ”year” in item_feature
- EDA: Use EDA to decide: continuous effect or group effect
- Both: Or include in both

» Continuous/Categorical Features

Cont Continuous features can be directly used as inputs.

Cate Utilizing categorical features;

Recall the concept of **Matrix Factorization (MF)**:

id USER_ID $\to \mathbf{p}_u$ ITEM_ID $\to \mathbf{q}_i$

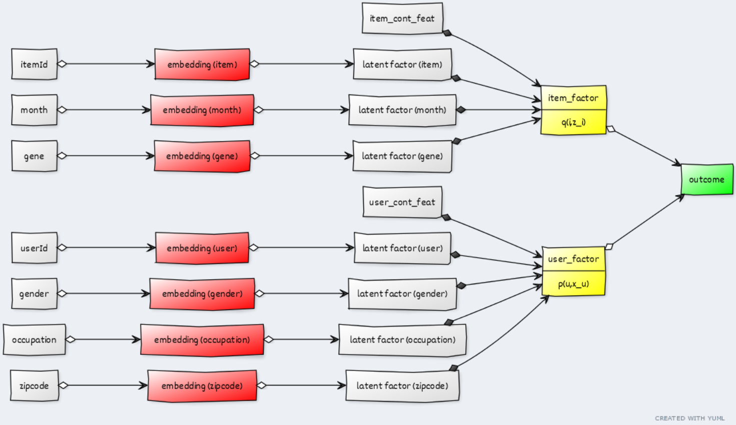
We can apply the **MF** approach to other categorical features, illustrated here with gender.

Encoding M $\to 0$, F $\to 1$ (sklearn.preprocessing.LabelEncoder)

Embedding M $\to 0 \to \mathbf{w}_0$; F $\to 1 \to \mathbf{w}_1$

» LinearRS

Plot Network:



Code Implementation in Colab

» Two-tower neural nets: overview

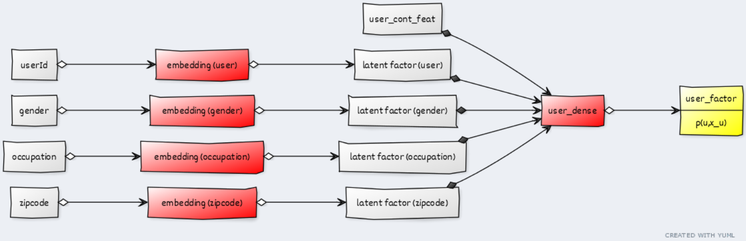
Motivation:

nl nonlinear modeling for features
within&btw user/item feats

effect with-in effects of user/item features:
E.g. (GENDER + OCCUPATION) @ ITEMID $\to$ rating

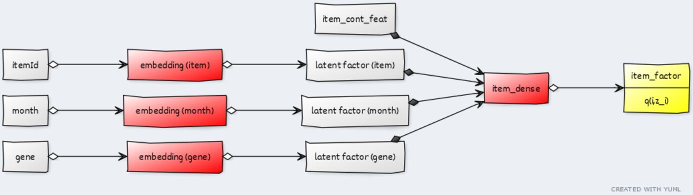
» Two-tower neural nets: user-tower

Plot **user-tower**



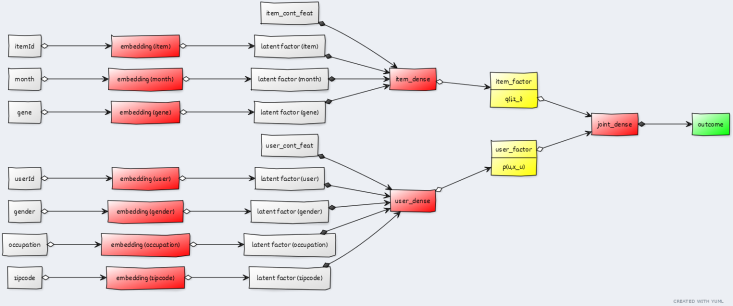
» Two-tower neural nets: item-tower

Plot **item-tower**



» Two-tower neural nets

Plot Dense layer to model the joint effect:



M The final formulation is given as:

$$\rho\left(\mathbf{p}(u, \mathbf{x}_u), \mathbf{q}(i, \mathbf{z}_i)\right) \rightarrow r_{ui}$$

» Side information in industry (Optional)

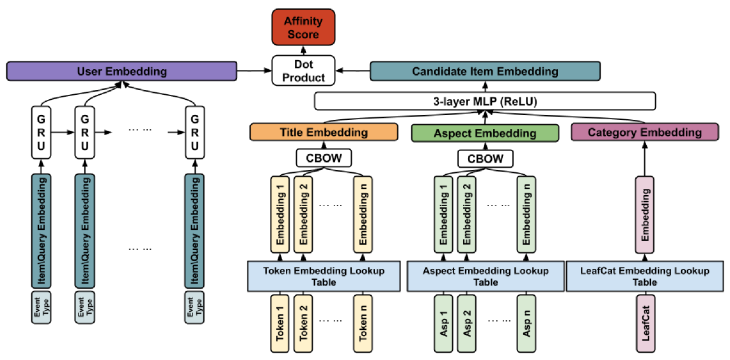
- **Text** Searching query; reviews; description; comments;
- **Image** profile for users; images for items;
- **Network** social network for users; item networks
- **Dynamic** behavior sequence; historical series

The general idea is mapping side information as numerical vectors, and feed into a two-tower based model.

- **Text** $\rightarrow$ embedding; Word2Vec; recurrent layers
- **Image** $\rightarrow$ convolutional layers
- **Network** $\rightarrow$ embedding; Node2Vec; graph convolutional layers

» Two-tower models in industry (Optional)

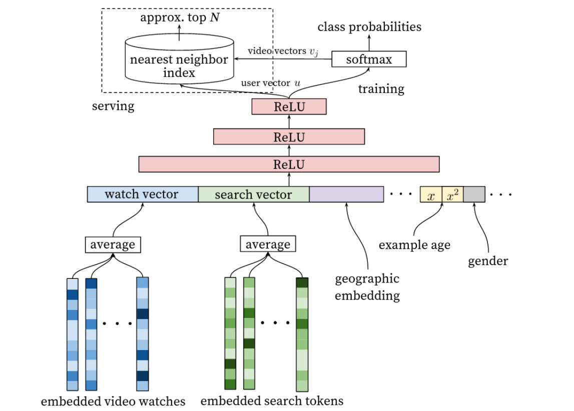
eBay **Model**



Ref Wang, T., Brovman, Y. M., & Madhvanath, S. (2021). *Personalized embedding-based e-commerce recommendations at ebay.*

» Two-tower models in industry (Optional)

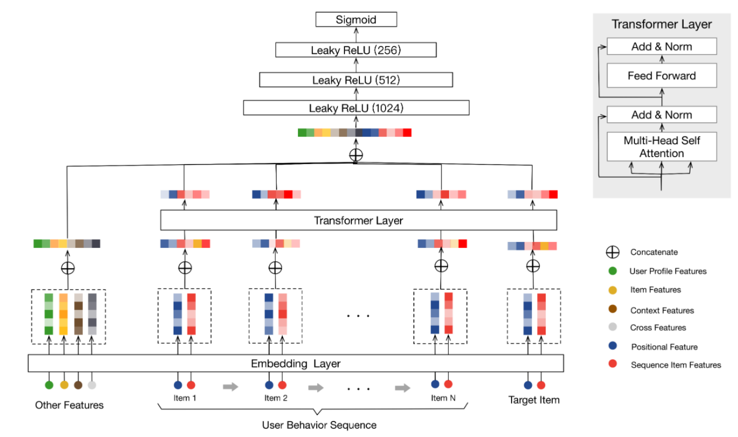
Youtube **Model**



Ref Covington, P., Adams, J., & Sargin, E. (2016). *Deep neural networks for youtube recommendations.*

» Two-tower models in industry (Optional)

Alibaba **Model**



Ref Chen, Q., Zhao, H., Li, W., Huang, P., & Ou, W. (2019). *Behavior sequence transformer for e-commerce recommendation in alibaba.*