Recommender systems
Introduction

Agenda

1. Admin
2. Recommender systems in real world
3. Overview of recommenders
4. Basic recommenders
5. Idea behind Amazon, Netflix and YouTube recommenders

About me

- Piotr Zioło
- PhD in functional analysis at Adam Mickiewicz University (2011)
- Polish Academy of Sciences (2011-2016)
  - Glycol dehydration modeling
  - Optimization of fuel quality laboratory system
  - Algotrading
  - Data imputation in forest databases
- RoomSage (2017-)
  - Bidding optimization on Google Ads
  - Personalized recommenders for hotels
  - Cooperation with Princeton University
- Allegro Machine Learning Research (2021-)

Class info

- 25th of February – 10th of June
- Moodle:
  [Systemy rekomendacyjne (ćw.) (2021/22) (P. Zioło)]
- Code and presentations:
  https://github.com/PiotrZiolo/recommender-systems-class
- Discussion of problems: Moodle forum („Pytania i dyskusje”)
- Contact:
  - Email: pziolo@amu.edu.pl
  - Forum on Moodle
  - Chat in Teams

Class plan

- Dataset preparation – Numpy, Pandas refresher
- Basic recommenders and evaluation
- Content-based recommenders
- Amazon recommender
- Neighborhood methods
- Dimensionality reduction and matrix factorization
- Collaborative filtering (user-to-user, item-to-item) – Netflix recommender
- Optimizers (algorithms finding maximum/minimum of a function)
- PyTorch, neural nets refresher
- Neural collaborative filtering
- YouTube recommender

Environment

- Python 3.8
- Mostly work in Jupyter Notebooks
- PyCharm
- Anaconda: https://www.anaconda.com/products/individual
- Git
- Conda environment in the repository

Assessment

- Quizes in Moodle after most classes – 50% of total points
- 2 projects – 50% of total points

Grade	Criterion
5.0	Above 90%
4.5	Above 80%
4.0	Above 70%
3.5	Above 60%
3.0	Above 50%
2.0	Less than 50%

- Retake – projects can be resent and only the new score counts
- Project solutions will be posted on GitHub
- Best 3 results in every project receive +1 to the grade

Literature

Aggarwal C., Recommender Systems: The Textbook, Springer, 2016

Falk K., Practical Recommender Systems, Manning, 2019

James G., Witten D., Hastie T., Tibshirani T., An Introduction to Statistical Learning, Springer, 2013

Trask A., Grokking Deep Learning, Manning, 2019

Linden G., Smith B., York J., Amazon.com Recommendations: Item-to-Item Collaborative Filtering, IEEE Internet Computing, 2003

Smith B., Linden G., The Test of Time, Two Decades of Recommender Systems at Amazon.com, IEEE Internet Computing, 2017

Sarwar B.. Karypis G., Konstan J., Riedl J., Item-Based Collaborative Filtering Recommendation Algorithms, Proc. 10th International World Wide Web Conference, 2001

Koren Y., Bell R., Volinsky C., Matrix factorization techniques for recommender systems, Computer, 2009

Gomez-Uribe C., Hunt N., The Netflix Recommender System: Algorithms, Business Value, and Innovation, ACM Trans. Manage. Inf. Syst., 2015

He X., Liao L., Zhang H., Nie L., Hu X., Chua T., Neural collaborative filtering, International World Wide Web Conference Committee, 2017

Covington P., Adams J., Sargin E., Deep Neural Networks for YouTube Recommendations, RecSys ’16, 2016

Agenda

1. Admin
2. **Recommender systems in real world**
3. Overview of recommenders
4. Basic recommenders
5. Idea behind Amazon, Netflix
   and YouTube recommenders

Examples of recommender systems

eCommerce:

- Amazon
- Allegro

Multimedia:

- YouTube
- Netflix
- Spotify

Social media

- Facebook

News

- Google Discover

Job recommendations

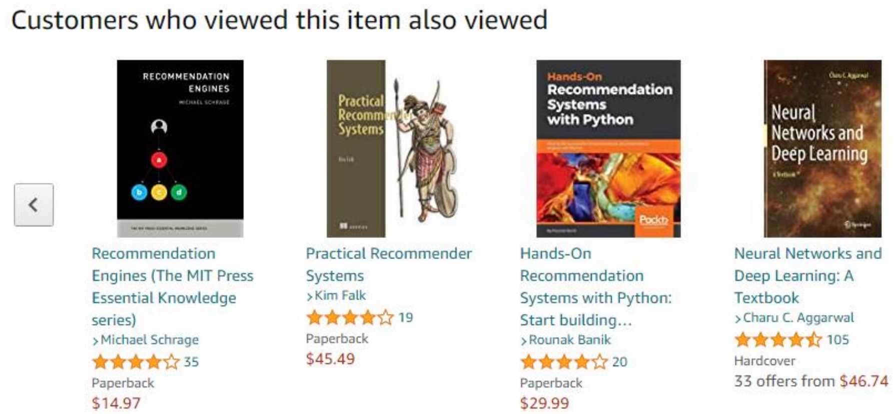
- OLX

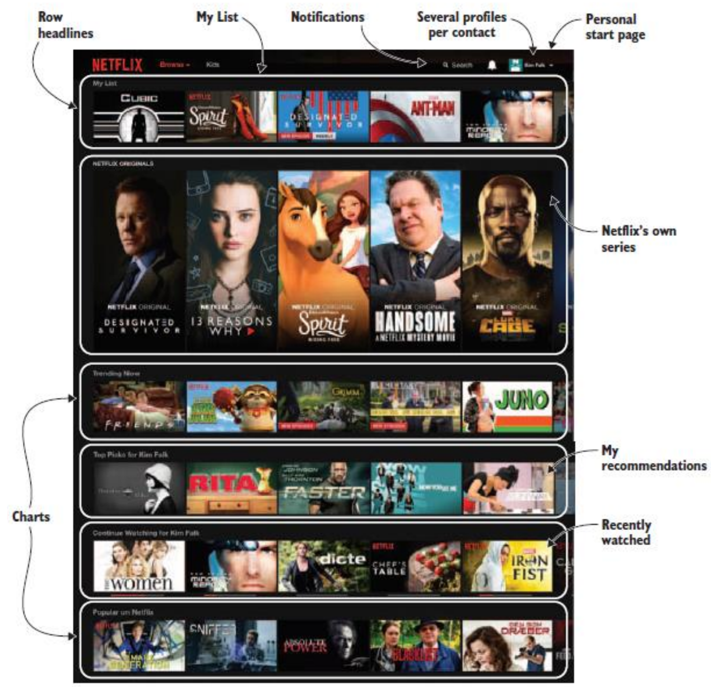
(E-)Learning

E-Government

Tourism

Personal assistants





The Netflix Prize

- Competition held between 2006 and 2009
- Sparked large interest in the field
- Data set of 100 480 507 ratings that 480 189 users gave to 17 770 movies
- Each training rating is a quadruplet of the form <user, movie, date of grade, grade>
- The goal was to achieve the lowest RMSE on grade predictions on a test set
- 1 mln $ prize for beating the benchmark by 10% (target RMSE=0.8572)
- 50k$ every year for the best result if the final goal not achieved
- Overall over 50 thousands teams took part
- Won by BellKor's Pragmatic Chaos on the 18 of September, 2009 (RMSE=0.8567)
- Koren Y., Bell R., Volinsky C., Matrix factorization techniques for recommender systems, Computer, 2009
- https://en.wikipedia.org/wiki/Netflix_Prize

Agenda

1. Admin
2. Recommender systems in real world
3. **Overview of recommenders**
4. Basic recommenders
5. Idea behind Amazon, Netflix and YouTube recommenders

Definition and goals

Definition: Recommender system
A recommender system calculates and provides relevant content to the user based on knowledge of the user, content, and interactions between users and items.

Goals

- Business – maximize sales/watch time
- Relevance – provide meaningful choices among millions of options
- Diversity – expose long-tail products
- Serendipity – match people with products they might not even be aware of

Challenges

- High hit ratio/accuracy
- Scalability
- Fast updating (after each user action)
- Sparsity (even 99.9% unknown interactions)
- Cold-start problem (user and entire system)
- Implicit feedback
- Long-tail
- Changing preferences
- Context awareness
- Attack resistance

Mathematical formulation

User-item rating/interaction matrix $R \in \mathbb{R}^{M \times N}$:

$$R = \begin{bmatrix} r_{1,1} & ? & r_{1,3} & \cdots & r_{1,N} \\ r_{2,1} & r_{2,2} & ? & \cdots & ? \\ ? & r_{3,2} & ? & \cdots & ? \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{M,1} & ? & r_{M,3} & \cdots & r_{M,N} \end{bmatrix}$$

where $r_{u,i}$ denotes the interaction of user u with item i. This interaction might be:

- a boolean indicating that a user bought or watched an item,
- a number of given items bought by a user,
- a rating the user has assigned to the item.

In mathematical terms the recommender has to predict $\hat{r}_{u,i}$ - the expected value of $r_{u,i}$ using previous interactions, user characteristics, item features and context information (time, external events etc.).

Input data

Interactions

- Bought/watched or not
- Number of bought items
- Rating

User characteristics

- Gender
- Age
- Location
- Personal interests

Item features

- Category/genre
- Price
- Reviews
- Movie length
- Actors
- Producer/seller/channel

Context

- Time of day, week, year
- What the user bought/watched just before
- Time since the last interaction
- External events

Explicit/implicit feedback

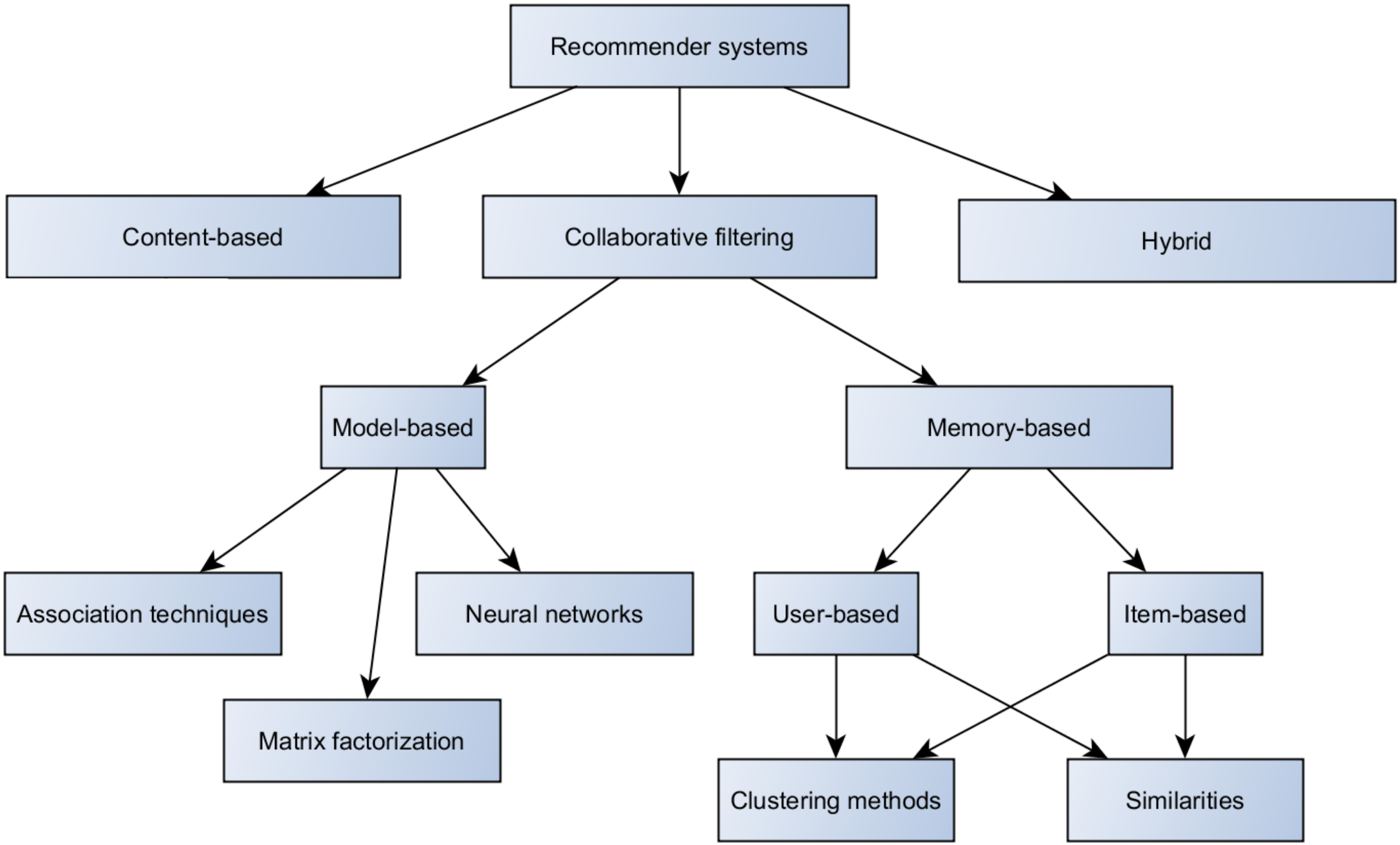
	GLADITOR	GODFTHER	BEN-HUR	GOODFELLAS	SCARFACE	SPARTACUS
U_1	1			5		2
U_2		5			4	
U_3	5	3		1		
U_4			3			4
U_5				3	5	
U_6	5		4			

(a) Ordered ratings

	GLADIATOR	GODFATHER	BEN-HUR	GOODFELLAS	SCARFACE	SPARTACUS
U_1	1			1		1
U_2		1			1	
U_3	1	1		1		
U_4			1			1
U_5				1	1	
U_6	1		1			

(b) Unary ratings

Classification



Agenda

1. Admin
2. Recommender systems in real world
3. Overview of recommenders
4. **Basic recommenders**
5. Idea behind Amazon, Netflix and YouTube recommenders

Non-personalized – baseline

Most popular

- Choose the item bought/watched the most and recommend it to every user

Highest rated

- Choose the item rated highest and recommend it to every user

Random

- Choose an item at random and recommend it to the user

Context-aware most popular

- Create a model predicting the most popular item based on context data (e.g. day of week, Black Friday)
- Choose the most appropriate offer for the given context and recommend it to the user

Personalized – baseline

Repeat

- Recommend the item the user is buying most often

Most popular in clusters

- Cluster users based on their features (for instance with K-means)
- For every cluster find the most popular item
- For a given user recommend the most popular item for their cluster

Nearest neighbours

- For every user prepare the interaction vector p_u (respective row from the interaction matrix)
- Find nearest neighbours of a given user in the space of those vectors
- Calculate popularity/average ratings for items bought/watched by those neighbours
- Out of items bought/watched by those neighbours recommend the most popular/highest rated item the user hasn't yet bought/watched

Agenda

1. Admin
2. Recommender systems in real world
3. Overview of recommenders
4. Basic recommenders
5. **Idea behind Amazon, Netflix and YouTube recommenders**

Amazon recommender

Amazon recommender

- Calculate conditional probabilities – if a user bought item X what is the chance they will buy item Y
- For a given user take the set of all his purchases S
- For every other item look up its probability conditional on items from S
- Recommend items with the highest conditional probabilities

$$E_{XY} = \sum_{c\in X}\left[1-(1-P_Y)^{|c|}\right] = \sum_{c\in X}\left[1-\sum_{k=0}^{|c|}\binom{|c|}{k}(-P_Y)^k\right]$$

$$= \sum_{c\in X}\left[1-\left[1+\sum_{k=1}^{|c|}\binom{|c|}{k}(-P_Y)^k\right]\right] = \sum_{c\in X}\sum_{k=1}^{|c|}(-1)^{k+1}\binom{|c|}{k}P_Y^k$$

$$= \sum_{c\in X}\sum_{k=1}^{\infty}(-1)^{k+1}\binom{|c|}{k}P_Y^k \qquad \left(\text{since } \binom{|c|}{k} = 0 \text{ for } k > |c|\right)$$

$$= \sum_{k=1}^{\infty}\sum_{c\in X}(-1)^{k+1}\binom{|c|}{k}P_Y^k \qquad (\text{Fubini's theorem})$$

$$= \sum_{k=1}^{\infty}\alpha_k(X)P_Y^k \qquad \text{where } \alpha_k(X) = \sum_{c\in X}(-1)^{k+1}\binom{|c|}{k}.$$

Figure 1. The derivation of the expected number of customers who bought both items X and Y, accounting for multiple opportunities for each X-buyer to buy Y.

Netflix recommender

Netflix recommender

- Take the interaction matrix $R = [r_{u,i}]$
- Pose the predictive problem in the form
$$\hat{r}_{u,i} = \sum_{k=1}^{D} p_{u,k}\, q_{i,k}$$
- Solve the above problem for vectors $\vec{p}_u$ and $\vec{q}_i$ using Stochastic Gradient Descent (SGD) or Alternating Least Squares (ALS)
- For a given user calculate the score for every item as $\vec{p}_u^T \cdot \vec{q}_i$ and recommend the items with the highest score

Add user and item biases

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T p_u$$

Add time dependencies

$$\hat{r}_{ui}(t) = \mu + b_i(t) + b_u(t) + q_i^T p_u(t)$$

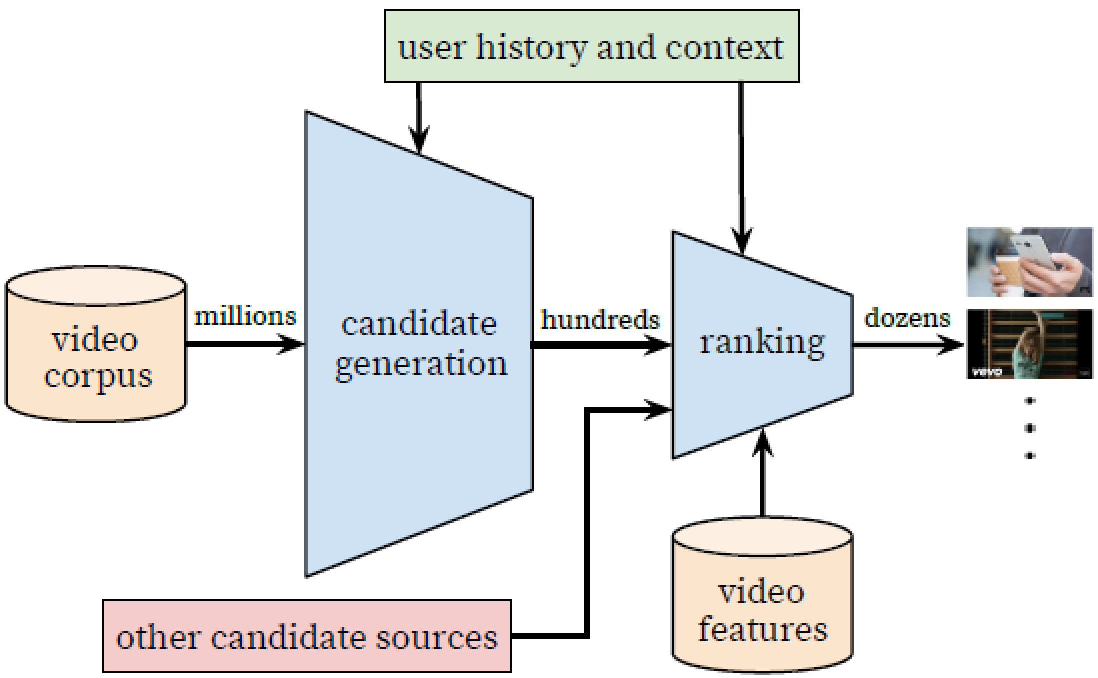
Add user and item features

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T [p_u + |N(u)|^{-0.5} \sum_{i \in N(u)} x_i + \sum_{a \in A(u)} y_a]$$

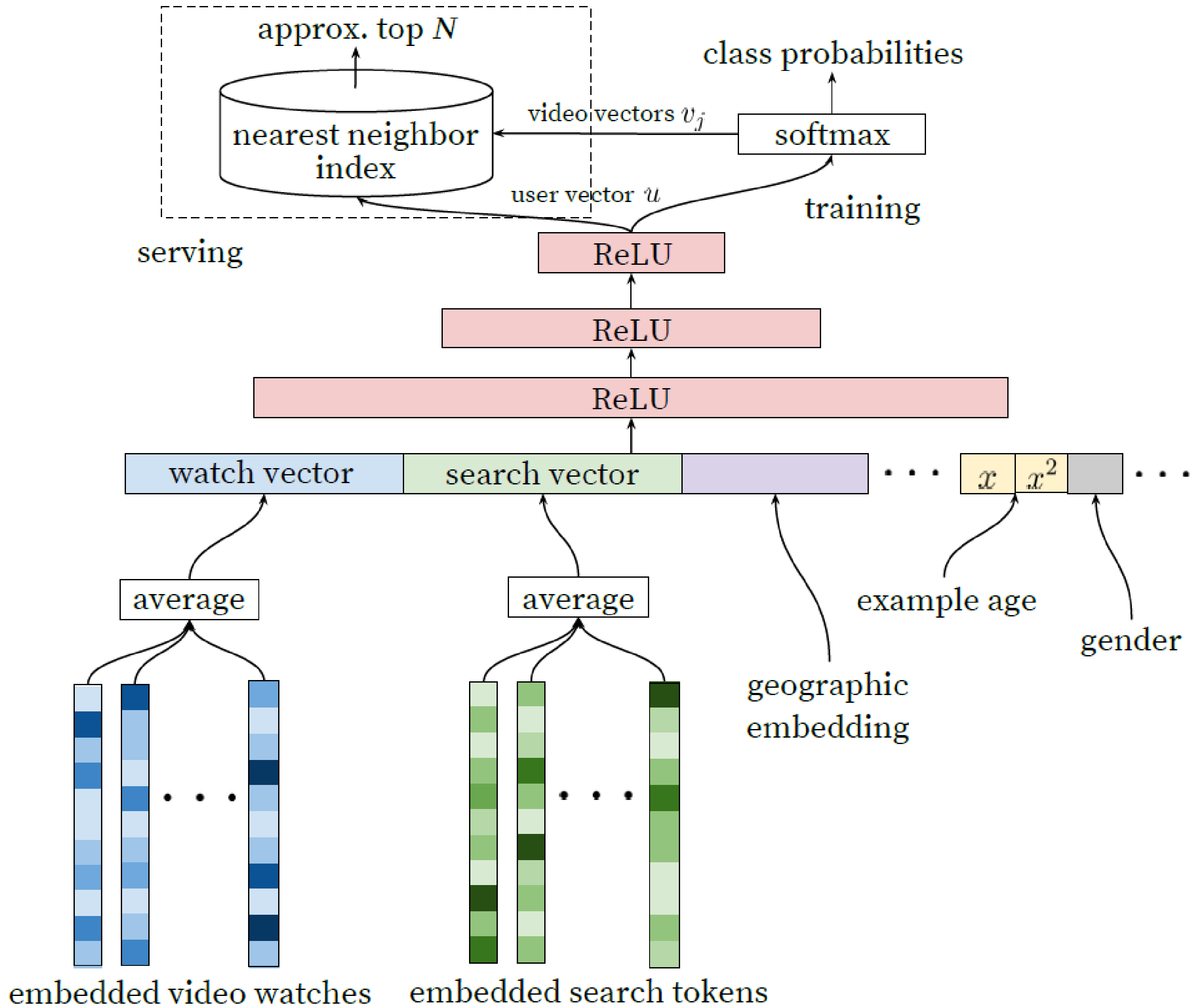
YouTube recommender

YouTube recommender

- Use Deep Candidate Generation Network to choose a set of hundreds of candidate videos for a given user
- Use Deep Ranking Network to score each of those candidates separately for this user
- Show recommended videos in the order of the above score



YouTube recommender – Deep Candidate Generation network



YouTube recommender – Deep Ranking network