MICHIGAN STATE UNIVERSITY

ByteDance

Automated Embedding Size Search in Deep Recommender Systems

Haochen Liu[1], Xiangyu Zhao[1], Chong Wang[2], Xiaobing Liu[2] and Jiliang Tang[1]

1: Data Science and Engineering Lab, Michigan State University

2: Applied Machine Learning, Bytedance

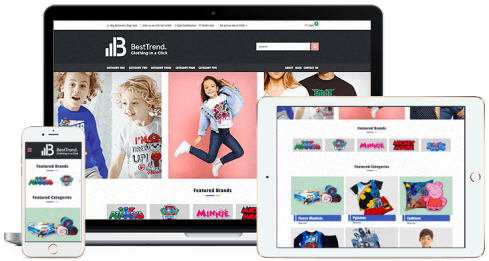
Presenter: Haochen Liu

July, 2020

Recommender Systems

- Goal
  - Predict a user's preference for an item

- Applications



E-commerce websites

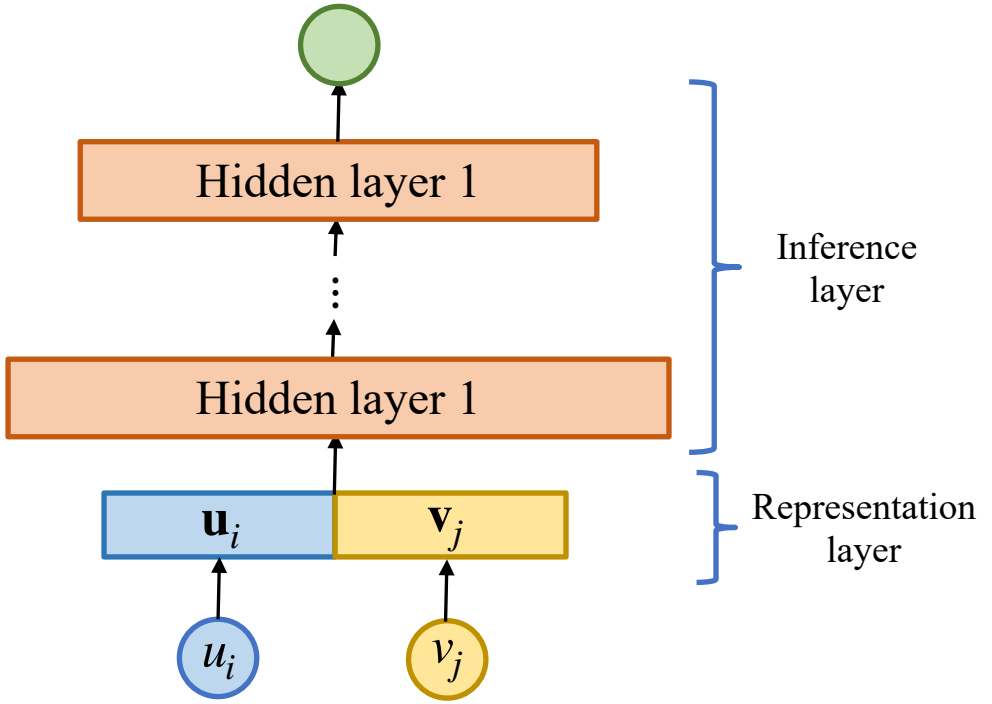
Music/video platforms

Social media

Deep Recommender Systems

- Advantages
  - ✓ Learning the feature representations of users and items
  - ✓ Modeling the non-linear relationships between users and items

- Architecture
  - Representation layer
  - Inference layer



Motivations

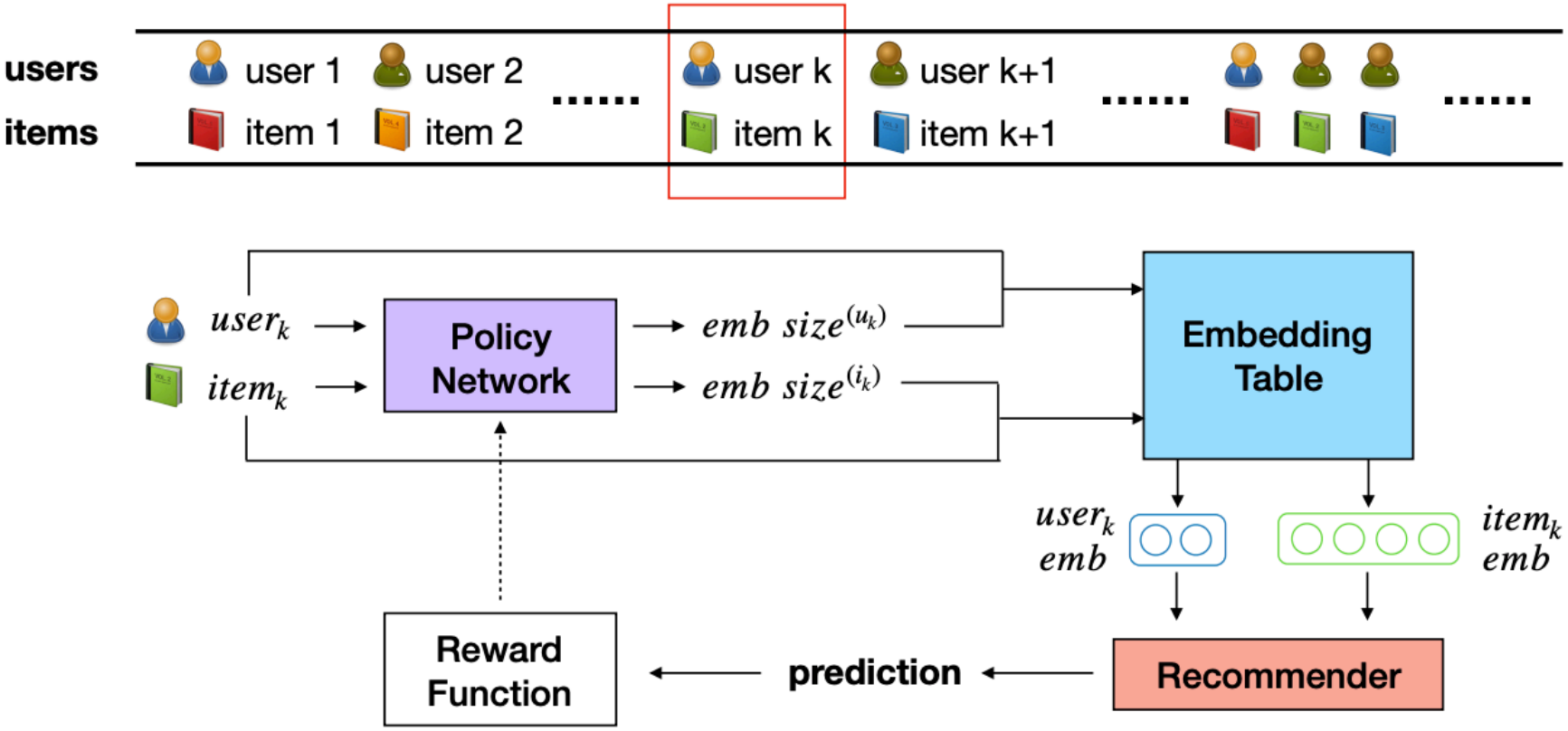
- Dynamically search the embedding sizes for different users and items

- Why
  - Users and items have highly varied frequencies
  - The frequency of a user/item changes dynamically
  - More efficient in memory

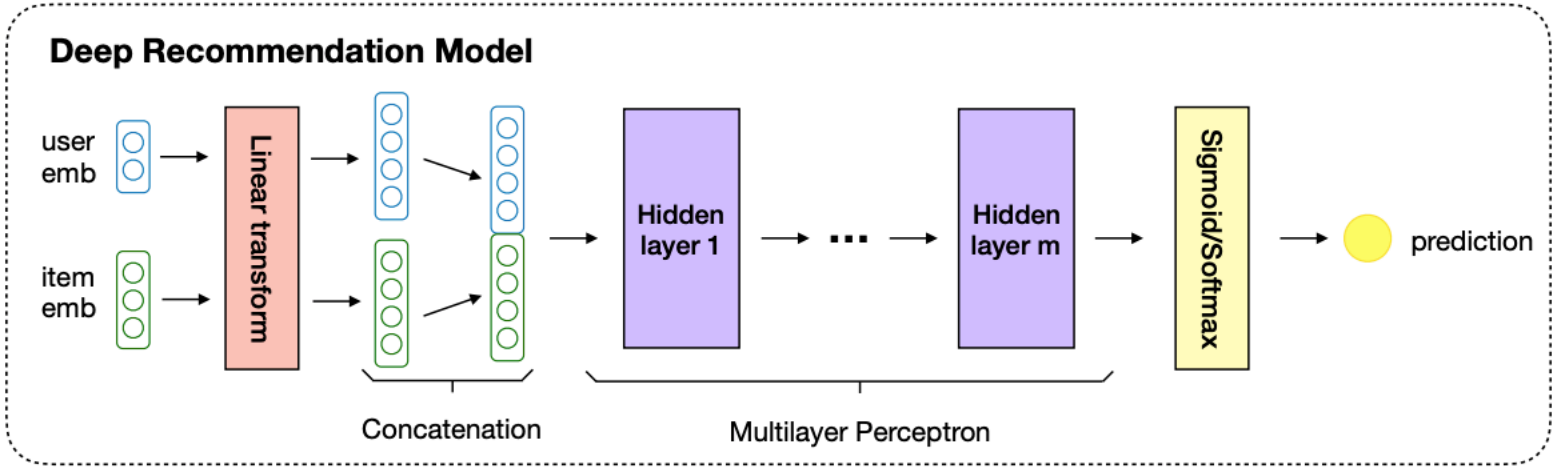
Introduction

- Challenges
  - Automatic search
  - Non-differentiability
  - Soft selection
- Our Design
  - Embedding Size Adjustment Policy Network (ESAPN)
  - Reinforcement learning (RL)
  - Hard selection

Overview


users
items
user 1
item 1
user 2
item 2
user k
item k
user k+1
item k+1
$user_k$
$item_k$
Policy Network
$emb\ size^{(u_k)}$
$emb\ size^{(i_k)}$
Embedding Table
$user_k$ emb
$item_k$ emb
Recommender
prediction
Reward Function

Deep Recommendation Model



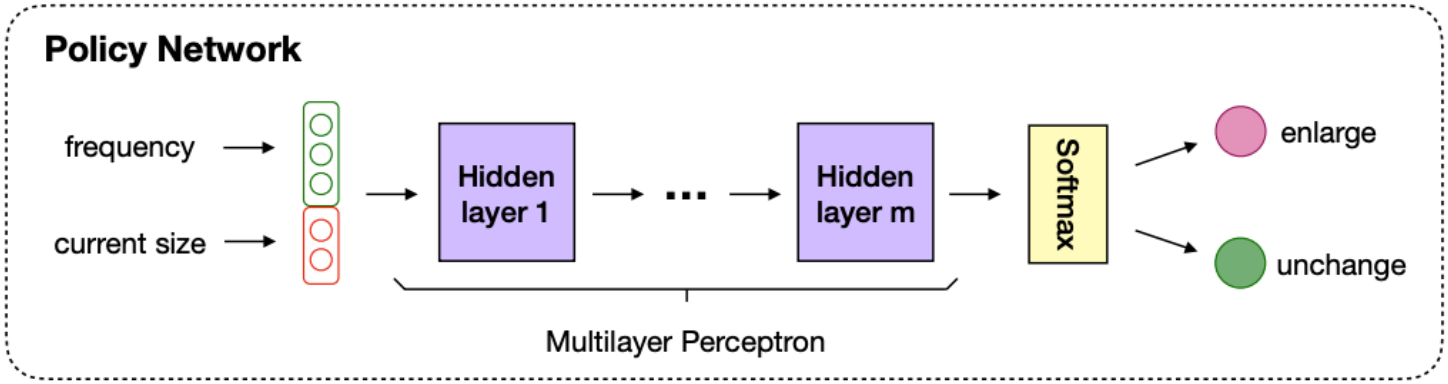
- Candidate embedding sizes

$$D = \{d_1, d_2, \ldots, d_n\} \qquad d_1 < d_2 < \cdots < d_n$$

- Linear transformations

$$\mathbf{e}_2 = W_{1\to 2}\mathbf{e}_1 + b_{1\to 2}$$
$$\mathbf{e}_3 = W_{2\to 3}\mathbf{e}_2 + b_{2\to 3}$$
$$\ldots$$
$$\mathbf{e}_n = W_{n-1\to n}\mathbf{e}_{n-1} + b_{n-1\to n}$$

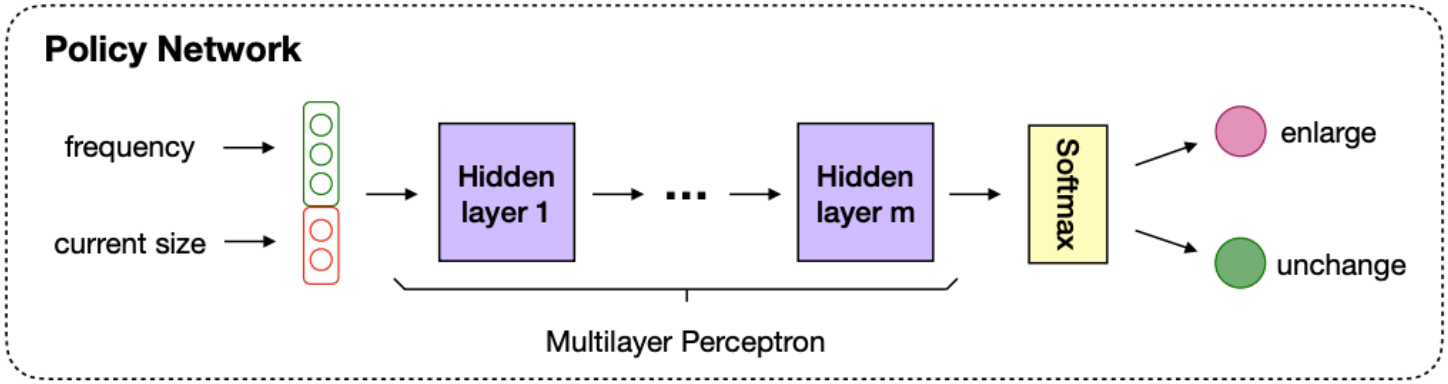
Policy Network



- Environment
  The deep recommendation model

- State
  $s = (f, e)$
  f : frequency e : current embedding size

Policy Network



- Action
  - Enlarge
  - Unchange
- Reward

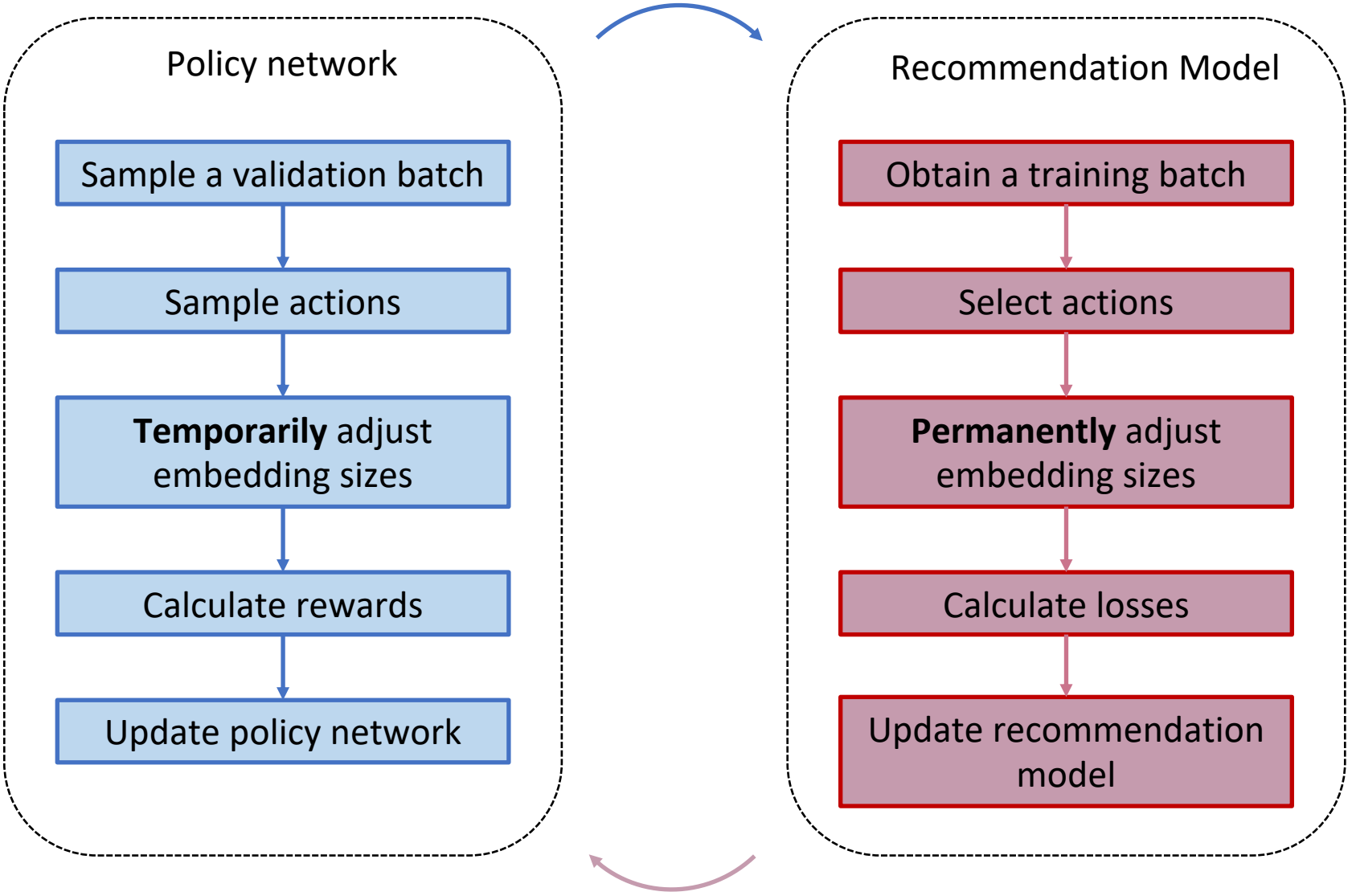
$$L^{(u)} = (L_1^{(u)}, \ldots, L_T^{(u)})$$

$$R^{(u)} = \frac{1}{T}\sum_{t=1}^{T} L_t^{(u)} - L$$

$$L^{(i)} = (L_1^{(i)}, \ldots, L_T^{(i)})$$

$$R^{(i)} = \frac{1}{T}\sum_{t=1}^{T} L_t^{(i)} - L$$

Optimization



Experimental Settings

- Datasets
  - MovieLens 20M Dataset (ml-20m)
  - MovieLens Latest Dataset (ml-latest)

- Candidate Embedding Sizes

$$D = \{2, 4, 8, 16, 64, 128\}$$

Experimental Results

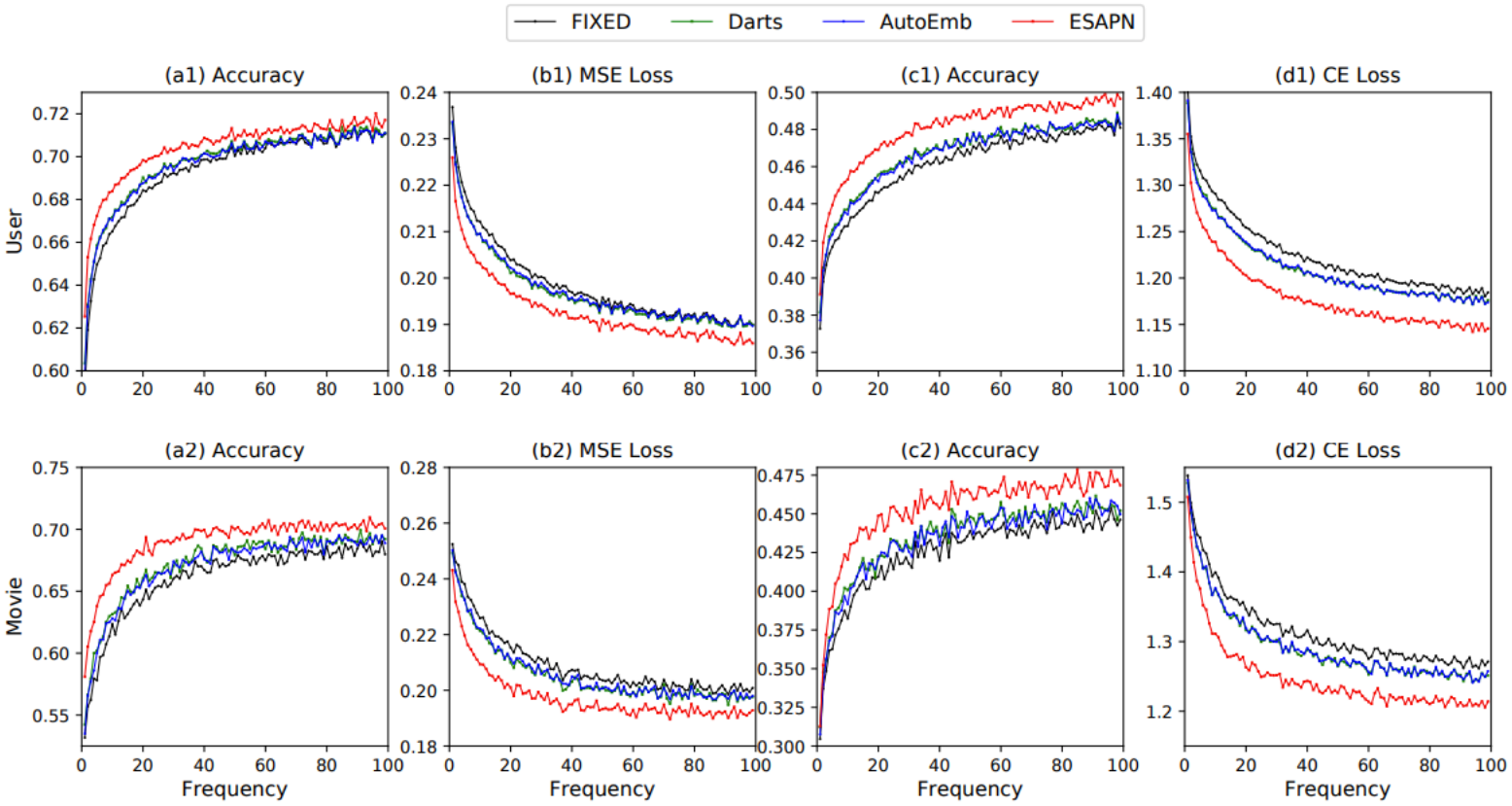
- Baselines
  - FIXED
  - DARTS
  - AutoEmb
- Tasks
  - Binary Classification
  - Multiclass Classification

ml-20m				
Models	Binary		Multiclass	
	Accuracy (%)	MSE Loss	Accuracy (%)	CE Loss
FIXED	72.13	0.1845	49.45	1.1517
DARTS	72.18	0.1836	49.85	1.1423
AutoEmb	72.27	0.1828	49.94	1.1399
ESAPN	72.98	0.1785	51.10	1.1126

ml-latest				
Models	Binary		Multiclass	
	Accuracy (%)	MSE Loss	Accuracy (%)	CE Loss
FIXED	72.13	0.1845	50.01	1.1414
DARTS	72.22	0.1834	50.46	1.1304
AutoEmb	72.36	0.1823	50.47	1.1311
ESAPN	72.88	0.1790	51.11	1.1147

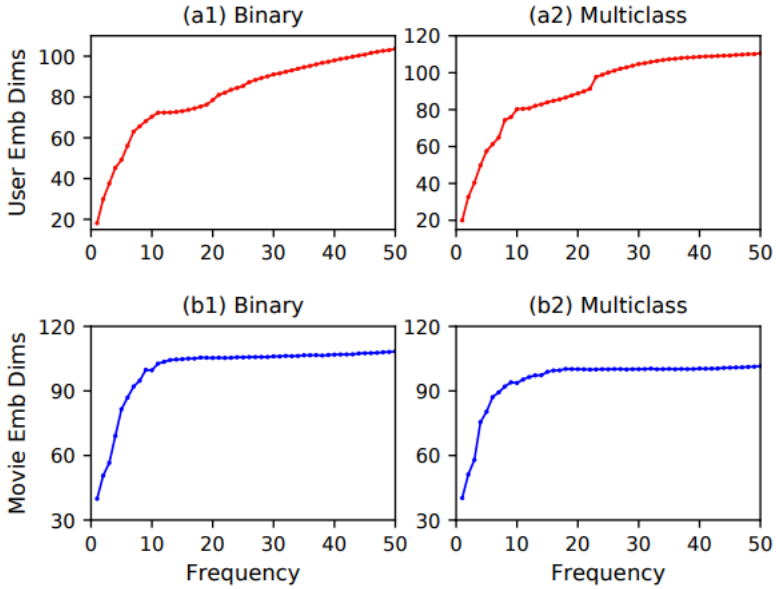
Experimental Results

- Performance Comparison with Frequency



Experimental Results

- Performance Comparison with Frequency



- Memory Consumption Comparison

	2	4	8	16	64	128	Total Dim	FIXED	Ratio
user (ml-20m)	1,041	10,854	18,146	30,836	22,830	54,786	9,157,770	17,727,104	51.66%
movie (ml-20m)	17,032	216	380	623	2,237	6,790	1,060,224	3,491,584	30.37%
user (ml-latest)	62,808	83,190	31,450	22,778	31,613	51,389	9,675,448	36,253,184	26.69%
movie (ml-latest)	45,430	2,089	2,006	1,678	1,545	5,350	925,792	7,436,544	12.45%

Future Works

- Incorporate other information to determine an appropriate embedding size

- Automatically design the network structure in the inference layer

Thanks!

Haochen Liu: http://www.cse.msu.edu/~liuhaoc1/

DSE Lab: http://dse.cse.msu.edu/