UserSim: User Simulation via Supervised Generative Adversarial Network

Xiangyu Zhao\textsuperscript{1}, Long Xia\textsuperscript{2}, Lixin Zou\textsuperscript{3}
Hui Liu\textsuperscript{1}, Dawei Yin\textsuperscript{3}, Jiliang Tang\textsuperscript{1}

\textsuperscript{1}: Data Science and Engineering Lab, Michigan State University
\textsuperscript{2}: York University \hspace{1cm} \textsuperscript{3}: Baidu
Reinforcement Learning for Recommendations

- Increasing interests in applying Reinforcement Learning for recommendations

- Advantages
  - Continuously updating the recommendation strategies during the interactions
  - Maximizing the long-term reward from users
Real-time Feedback

- The most practical and precise way is online A/B test

- Online A/B test is inefficient and expensive
  - Taking several weeks to collect sufficient data
  - Numerous engineering efforts
  - Bad user experience
Simulating users’ real-time feedback is challenging
- Underlying distribution of item sequences is extremely complex
- Data available to each user is rather limited
- Learning the data distribution
- Generating indistinguishable logs based on users' browsing history

**Generator**

User's Preference → Prior Items → Encoder → Decoder → Output Layer → Next Item

- $h_1 \rightarrow \ldots \rightarrow h_N$
- $e_1 f_1 \rightarrow e_N f_N$
- $P^E$
- $f_{1:N} \rightarrow G_\theta(s)$
- Supervised component
- **Input layer**
  - $e_n$: item’s identifier embedding
  - $f_n$: user’s feedback embedding

\[ I_n = \text{concat}(e_n, f_n) \]
GRU layer:

- Capturing user’s preference from the sequence of items

$$z_t = \sigma(W_z E_t + U_z h_{t-1})$$
$$r_t = \sigma(W_r E_t + U_r h_{t-1})$$
$$h_t = (1 - z_t) h_{t-1} + z_t \hat{h}_t$$
$$\hat{h}_t = \tanh(W E_t + U (r_t \cdot h_{t-1}))$$
### Decoder

#### Goal:
- Predicting the item to be recommended
Discriminator

- Distinguishing real/fake items
- Predicting user’s feedback

2K feedback
- K feedback for real items
- K feedback for fake items

\[ p_{model}(l_j|s, a) = \frac{\exp(l_j)}{\sum_{k=1}^{2K} \exp(l_k)} \]
$L_D = L_{D^{unsup}} + \alpha \cdot L_{D^{sup}}$

$L_{D^{unsup}} = -\mathbb{E}_{s,a \sim p_{data}} \left[ \log p_{model}(l_k|s, a, k \leq K) \right]$
$+ \lambda \cdot \mathbb{E}_{s,a \sim p_{data}} \left[ \log p_{model}(l_k|s, G_{\theta}(s), K < k \leq 2K) \right]$

$L_{D^{sup}} = -\left\{ \mathbb{E}_{s,a \sim p_{data}} \log D_{\phi}(s, a) \right\} + \mathbb{E}_{s \sim p_{data}} \log D_{\phi}(s, G_{\theta}(s))$

$D_{\phi}(s, G_{\theta}(s)) = \sum_{k=K+1}^{2K} p_{model}(l_k|s, G_{\theta}(s))$

$D_{\phi}(s, a) = \sum_{k=1}^{K} p_{model}(l_k|s, a)$

**Optimization**

- **Discriminator**
Optimization

Generator

$$L_{G}^{\text{unsup}} = \mathbb{E}_{s \sim p_{\text{data}}} \left[ \log D_{\phi}(s, G_{\theta}(s)) \right]$$

$$L_{G} = L_{G}^{\text{unsup}} + \beta \cdot L_{G}^{\text{sup}}$$

$$L_{G}^{\text{sup}} = \mathbb{E}_{s, a \sim p_{\text{data}}} \left[ a - G_{\theta}(s) \right]^{2}$$
Experimental Settings

- Public benchmark datasets
  - Netflix and JD.com
  - 70%: training/validation set
  - 30%: test set

- 4 types of feedback
  - Real-positive
  - Real-negative
  - Fake-positive
  - Fake-negative
  - Real: real item from data
  - Fake: fake item from generator

<table>
<thead>
<tr>
<th>Object</th>
<th>Netflix Prize</th>
<th>JD.com</th>
</tr>
</thead>
<tbody>
<tr>
<td># user (session)</td>
<td>480,189</td>
<td>283,228</td>
</tr>
<tr>
<td># item</td>
<td>17,770</td>
<td>1,355,255</td>
</tr>
<tr>
<td># interaction</td>
<td>100,480,507</td>
<td>97,713,660</td>
</tr>
<tr>
<td># ave. length</td>
<td>209</td>
<td>345</td>
</tr>
<tr>
<td># feedback</td>
<td>rating 1~5</td>
<td>skip, click</td>
</tr>
</tbody>
</table>

4~5: positive
1~3: negative
click: positive
skip: negative
Overall Performance

- Metric: F1-score
- Baselines: LR, UserSim-d, RecSim, RecoGym, Virtual-Taobao, GAN-PW, IRecGAN
- Generator can learn the item distribution, and generate fake items
- Discriminator can distinguish real and fake items, and predict user’s feedback
### RL-based Recommender Training

#### Metric
- average reward of a session

#### Baselines
- Historical Logs, IRecGAN

#### UserSim
- Converges to the similar avg_reward with the one upon historical data
- Performs much more stably than the one trained based upon IRecGAN

**On-policy RL algorithms such as SARSA cannot be directly trained on historical data**
Conclusion

- We propose a novel user simulator based on Generative Adversarial Network
  - Generating real-time feedback like real users
  - Pre-training and evaluating new recommendation algorithms before launching them online

zhaoxi35@msu.edu