One Person, One Model, One World: Learning Continual User Representation without Forgetting

SIGIR2021

Data&Code: https://github.com/fajieyuan/SIGIR2021_Conure

Fajie Yuan (Westlake University, Tencent), Guoxiao Zhang (Tencent),
Alexandros Karatzoglou (Google Research), Joemon Jose (University of Glasgow),
Beibei Kong (Tencent), Yudong Li (Tencent)
Outline

- Motivation
- Related Work
- Conure
- Experiments
Our Motivation

A person has different roles to play in life! But all these roles may have some commonalities, such as personalization, habits, preference.

Our Focus:
Whether we can build a user representation model that could keep learning throughout all sequential tasks without forgetting.
A person has different roles to play in life! But all these roles may have some commonalities, such as personalization, habits, preference.
Using Lifelong learning techniques to solve recommendation tasks

Keypoints
• Necessity and possibility why lifelong learning for UR learning?
• Lifelong learning paradigm throughout all tasks.
• Performance gain for tasks have certain correlations.
Outline

- Motivation
- Related Work
- Conure
- Experiments
• **Classical UR models** (works well but is specific to only one task)

- SASRec (Kang et al ICDM2018)
- NextItNet (Yuan et al WSDM2019)
- DSSM (Huang et al CIKM2013)
- Grec (Yuan et al WWW2020)
- GRU4Rec (Hidasi et al ICLR2016)
PeterRec (Two-stage Transfer Learning):

(a) pre-training

(b) fine-tuning
• PeterRec (Finetuning):

\[
p(\mathbf{x}^\mu; \Theta) = \prod_{i=1}^{n} p(x_i^\mu | x_1^\mu, \ldots, x_{i-1}^\mu; \Theta)
\]

\[
p(\mathbf{x}_\Delta^\mu; \Theta) = \prod_{i=1}^{m} p(x_{\Delta i}^\mu | \hat{x}^\mu; \Theta)
\]
• Transfer Learning Paradigm Comparisons:

(a) Standard TF
(b) PeterRec
(b) Conure
(b) MTL

Lifelong learning without parameter preserving
Outline

- Motivation
- Related Work
- Conure
- Experiments
Catastrophic Forgetting:

Parameter Changes

Last hidden Vector Changes

(a) Only training $T_1$

(b) After Training $T_2$

(c) Only training $T_1$

(d) After training $T_2$
Over-parameterization:

- (1) the more parameters are pruned, the worse it performs
- (2) performing retraining on the pruned network (i.e., “pr70+retrain”) regains its original accuracy quickly
- (3) smaller models (i.e., (b)) are also highly over-parameterized
- Conure architecture and learning process.

Conure is conceptually very simple, easy to implement, and applicable to various sequential encoder networks.
Outline

- Motivation
- Related Work
- Conure
- Experiments
Datasets:

TTL: https://drive.google.com/file/d/1imhHU5ivh6oMEtEW-RwVc4OsDgn-xOaP/view
ML: https://drive.google.com/file/d/1-KmnZFaOdH11keLYVcgkf-kW_BaM266/view

Table 1: Number of instances. The number of distinct items $|X|$ in $T_1$ for TTL and ML is $646K$ and $54K$ ($K = 1000$), respectively. The number of labels $|Y|$ is $18K$, $8K$, $8$, $2$, $6$, respectively from $T_2$ to $T_6$ in TTL, and $26K$, $16K$, respectively from $T_2$ to $T_3$ in ML. $M = 1000K$.

<table>
<thead>
<tr>
<th>Data</th>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$T_3$</th>
<th>$T_4$</th>
<th>$T_5$</th>
<th>$T_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>TTL</td>
<td>$1.47M$</td>
<td>$2.70M$</td>
<td>$0.27M$</td>
<td>$1.47M$</td>
<td>$1.47M$</td>
<td>$1.02M$</td>
</tr>
<tr>
<td>ML</td>
<td>$0.74M$</td>
<td>$3.06M$</td>
<td>$0.82M$</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Results:

Table 2: Accuracy comparison. #B is the number of backbone networks. The left and right of ‘||’ represent TTL and ML, respectively. Conure—denotes Conure that has not experienced the pruning operation after training on the current task. The worse and best results are marked by ‘∨’ and ‘Δ’, respectively.

<table>
<thead>
<tr>
<th>Model</th>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$T_3$</th>
<th>$T_4$</th>
<th>$T_5$</th>
<th>$T_6$</th>
<th>#B</th>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$T_3$</th>
<th>#B</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN</td>
<td>0.0104</td>
<td>0.0154</td>
<td>0.0231</td>
<td>0.7131</td>
<td>0.8908</td>
<td>0.6003</td>
<td>6</td>
<td>0.0276</td>
<td>0.0175</td>
<td>0.0313</td>
<td>3</td>
</tr>
<tr>
<td>SinMo</td>
<td>0.0473</td>
<td>0.0144</td>
<td>0.0161</td>
<td>0.7068</td>
<td>0.8998</td>
<td>0.5805</td>
<td>6</td>
<td>0.0637</td>
<td>0.0160</td>
<td>0.0259</td>
<td>3</td>
</tr>
<tr>
<td>SinMoAll</td>
<td>0.0009$^∨$</td>
<td>0.0079$^∨$</td>
<td>0.0124$^∨$</td>
<td>0.5640$^∨$</td>
<td>0.7314$^∨$</td>
<td>0.6160</td>
<td>1</td>
<td>0.0038$^∨$</td>
<td>0.0145$^∨$</td>
<td>0.0310</td>
<td>1</td>
</tr>
<tr>
<td>FineSmax</td>
<td>0.0473</td>
<td>0.0160</td>
<td>0.0262</td>
<td>0.6798</td>
<td>0.8997</td>
<td>0.6070</td>
<td>1</td>
<td>0.0637</td>
<td>0.0150</td>
<td>0.0262</td>
<td>1</td>
</tr>
<tr>
<td>FineAll</td>
<td>0.0473</td>
<td>0.0172</td>
<td>0.0271</td>
<td>0.7160$^Δ$</td>
<td>0.9053</td>
<td>0.6132</td>
<td>6</td>
<td>0.0637</td>
<td>0.0189</td>
<td>0.0325</td>
<td>3</td>
</tr>
<tr>
<td>PeterRec</td>
<td>0.0473</td>
<td>0.0173</td>
<td>0.0275</td>
<td>0.7137</td>
<td>0.9053</td>
<td>0.6156</td>
<td>1</td>
<td>0.0637</td>
<td>0.0182</td>
<td>0.0308</td>
<td>1</td>
</tr>
<tr>
<td>MTL</td>
<td>-</td>
<td>0.0151</td>
<td>0.0172</td>
<td>0.7094</td>
<td>0.8979</td>
<td>0.6027</td>
<td>1</td>
<td>-</td>
<td>0.0167</td>
<td>0.0276</td>
<td>1</td>
</tr>
<tr>
<td>Conure−</td>
<td>0.0473</td>
<td>0.0174</td>
<td>0.0286</td>
<td>0.7139</td>
<td>0.9051</td>
<td>0.6180</td>
<td>-</td>
<td>0.0637</td>
<td>0.0183</td>
<td>0.0347</td>
<td>-</td>
</tr>
<tr>
<td>Conure</td>
<td>0.0480$^Δ$</td>
<td>0.0177$^Δ$</td>
<td>0.0287$^Δ$</td>
<td>0.7146</td>
<td>0.9068$^Δ$</td>
<td>0.6185$^Δ$</td>
<td>1</td>
<td>0.0656$^Δ$</td>
<td>0.0197$^Δ$</td>
<td>0.0353$^Δ$</td>
<td>1</td>
</tr>
</tbody>
</table>

— (1) Conure largely outperforms other models on T3 because of the positive transfer from T1 and T2
— (2) Conure, PeterRec and FineAll largely outperforms SimMo because of the positive transfer from T1
— (3) SinMoAll performs much worse on most tasks (except the last one) because of catastrophic forgetting
Ablation study - T2 for T3:

Table 3: Impact of T₂ on T₃. Conure_noT₂ denotes training Conure on T₃ after T₁. Conure_noT₂ and Conure both are the Conure-versions. TTL20% and ML20% denote the 20/80 train/test split.

<table>
<thead>
<tr>
<th></th>
<th>TTL</th>
<th>TTL20%</th>
<th>ML</th>
<th>ML20%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conure_noT₂</td>
<td>0.0277</td>
<td>0.0245</td>
<td>0.0334</td>
<td>0.0295</td>
</tr>
<tr>
<td>Conure</td>
<td>0.0286</td>
<td>0.0261</td>
<td>0.0347</td>
<td>0.0309</td>
</tr>
<tr>
<td>Impro.</td>
<td>3.2%</td>
<td>6.5%</td>
<td>3.9%</td>
<td>4.7%</td>
</tr>
</tbody>
</table>

— (1) Without training T2, Conure shows worse results, e.g., -6.5% on TTL20%
Ablation study- Task order:

Table 4: Impact of task orders. Order1 is the original order as mentioned in Section 5.1. KC, KT and Life denotes the clicking dataset, the thumbs-up dataset and the life status dataset of Kandian, respectively. Results on $T_1$ are omitted due to the same accuracy. The left and right of ‘||’ are results of Conure- and Conure, respectively.

<table>
<thead>
<tr>
<th>Orders</th>
<th>KC</th>
<th>KT</th>
<th>Life</th>
<th>KC</th>
<th>KT</th>
<th>Life</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order1</td>
<td>0.0174</td>
<td>0.0286</td>
<td>0.6180</td>
<td>0.0177</td>
<td>0.0287</td>
<td>0.6185</td>
</tr>
<tr>
<td>Order2</td>
<td>0.0174</td>
<td>0.0289</td>
<td>0.6154</td>
<td>0.0177</td>
<td>0.0290</td>
<td>0.6152</td>
</tr>
<tr>
<td>Order3</td>
<td>0.0174</td>
<td>0.0289</td>
<td>0.6145</td>
<td>0.0177</td>
<td>0.0287</td>
<td>0.6149</td>
</tr>
</tbody>
</table>

(1) Conure is not sensitive to the task order.
Ablation study:

Table 5: Pruning and retraining both the embedding & convolutional layers. The left & right of ‘||’ are tasks on TTL & ML.

<table>
<thead>
<tr>
<th>Models</th>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$T_3$</th>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$T_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conure-</td>
<td>0.0473</td>
<td>0.0175</td>
<td>0.0290</td>
<td>0.0637</td>
<td>0.0191</td>
<td>0.0341</td>
</tr>
<tr>
<td>Conure</td>
<td>0.0474</td>
<td>0.0177</td>
<td>0.0295</td>
<td>0.0645</td>
<td>0.0196</td>
<td>0.0347</td>
</tr>
</tbody>
</table>

— (1) pruning also works for the embedding layer.

Table 6: Results by specifying Conure with Transformer as the backbone network. The left and right of ‘||’ represent tasks on TTL and ML, respectively. ‘Mo’, ‘FA’, ‘C’-, ‘C’, denotes Models, FineAll, Conure- and Conure, respectively.

<table>
<thead>
<tr>
<th>Mo</th>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$T_3$</th>
<th>#B</th>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$T_3$</th>
<th>#B</th>
</tr>
</thead>
<tbody>
<tr>
<td>FA</td>
<td>0.0510</td>
<td>0.0161</td>
<td>0.0243</td>
<td>3</td>
<td>0.0654</td>
<td>0.0193</td>
<td>0.0321</td>
<td>3</td>
</tr>
<tr>
<td>C-</td>
<td>0.0510</td>
<td>0.0177</td>
<td>0.0288</td>
<td>-</td>
<td>0.0654</td>
<td>0.0198</td>
<td>0.0345</td>
<td>-</td>
</tr>
<tr>
<td>C</td>
<td>0.0513</td>
<td>0.0179</td>
<td>0.0289</td>
<td>1</td>
<td>0.0662</td>
<td>0.0200</td>
<td>0.0357</td>
<td>1</td>
</tr>
</tbody>
</table>

— (1) Conure is not restricted to specialized sequential encoder.
— (2) Conure with the Transformer backbone works a bit better than it with NextItNet.
Contributions:

— (1) providing the first lifelong learning paradigm for user representations.
— (2) providing insights for forgetting and redundancy issues in user representation models
— (3) designing Conure, the first lifelong learning algorithm - simple and easy to implement
— (4) instantiating Conure with NextItNet and Transformer backbones
— (5) Extensive experiments with SOTA performance with many new discoveries and insights
新用户推荐效果：左侧是A场景用户行为，右侧是B场景下预测出来的用户行为。A B为不同的推荐引擎。该用户在B场景下均为新用户或者只有少量的点击被仅用作评测。
未来怎么还是你的！詹姆斯10大不讲武德时刻，哈登羡慕詹姆斯大个子！
上半场开局詹姆斯7次出手，命中3球，其中三分球2中1。赛后詹姆斯表示，开场手感不好，命中率只有42.8%，但他在最后一节找回了感觉。

皮蓬生涯最恐怖一战！对手：真不敢追，谁追谁死啊！
科比对皮蓬说：趁我看不清，如果当年我身后能有几个总决赛，我就上！

一对一，两个外线！大火锅，詹姆斯就是这样征服施罗德的吗？
比赛进入最后20秒，詹姆斯在右侧底角面对施罗德，他突然连续运球，然后一个加速突破到篮下，面对施罗德的防守，詹姆斯一个勾手得分。