Knowledge Tracing: Predicting & Optimizing Human Learning

Jill-Jênn Vie  Hisashi Kashima

AIP-IITH workshop, March 15, 2019
Modeling learning over time

Combining representations (users & items)
  - Dimension 1 user2bias
  - Dimension n user2vec

Adaptive strategies for testing & optimizing human learning
  - If we can understand how human learns
  - We can learn a policy to teach better
## Related applications

### Crowdsourcing

Data: worker $i$ labels item $j$ with class $k$

What is the true label of all items?

### Mixture of experts, ensemble methods

Modeling which algorithm suits which features

### Machine teaching

Feed the best sequence of samples to train a known algorithm
When exercises are too easy (or difficult), students get bored (or discouraged).

To personalize assessment,
→ need a model of how people respond to exercises.
Learning low-rank representations of users and items
# Students try exercises

## Math Learning

<table>
<thead>
<tr>
<th>Items</th>
<th>5 − 5 = ?</th>
<th>17 − 3 = ?</th>
<th>13 − 7 = ?</th>
</tr>
</thead>
<tbody>
<tr>
<td>New student</td>
<td>○</td>
<td>○</td>
<td>×</td>
</tr>
</tbody>
</table>

## Language Learning

<table>
<thead>
<tr>
<th>correct:</th>
<th>She</th>
<th>is</th>
<th>my</th>
<th>mother</th>
<th>and</th>
<th>he</th>
<th>is</th>
<th>my</th>
<th>father</th>
</tr>
</thead>
<tbody>
<tr>
<td>student:</td>
<td>she</td>
<td>is</td>
<td>my</td>
<td>mader</td>
<td>and</td>
<td>he</td>
<td>is</td>
<td>my</td>
<td>fhader</td>
</tr>
<tr>
<td>label:</td>
<td>○</td>
<td>○</td>
<td>×</td>
<td>×</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>

## Challenges
- Users can attempt a same item multiple times
- Users learn over time
- People can make mistakes that do not reflect their knowledge
Predicting student performance: knowledge tracing

Data

A population of users answering items
- Events: “User $i$ answered item $j$ correctly/incorrectly”

Side information
- If we know the skills required to solve each item e.g., $+$, $\times$
- Device used by the student, etc.

Goal: classification problem

Predict the performance of new users on existing items
Metric: AUC

Method

Learn parameters of questions from historical data e.g., difficulty
Measure parameters of new students e.g., expertise
Our small dataset

- User 1 answered Item 1 correct
- User 1 answered Item 2 incorrect
- User 2 answered Item 1 incorrect
- User 2 answered Item 1 correct
- User 2 answered Item 2 ???

<table>
<thead>
<tr>
<th>user</th>
<th>item</th>
<th>correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>???</td>
</tr>
</tbody>
</table>

dummy.csv
Our approach

- Encode data to sparse features

<table>
<thead>
<tr>
<th>user</th>
<th>item</th>
<th>correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>???</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>???</td>
</tr>
</tbody>
</table>

**data.csv**

<table>
<thead>
<tr>
<th>user</th>
<th>item</th>
<th>correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>???</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>???</td>
</tr>
</tbody>
</table>

**sparse matrix X**

- Run logistic regression or factorization machines

⇒ recover existing models or better models
Simplest baseline: Item Response Theory (Rasch, 1960)

Learn abilities \( \theta_i \) for each user \( i \)
Learn easiness \( e_j \) for each item \( j \) such that:

\[
Pr(\text{User } i \text{ Item } j \text{ OK}) = \sigma(\theta_i + e_j) \quad \sigma : x \mapsto 1/(1 + \exp(-x))
\]

\[
\logit Pr(\text{User } i \text{ Item } j \text{ OK}) = \theta_i + e_j
\]

Really popular model, used for the PISA assessment

Can be encoded as logistic regression

Learn \( w \) such that \( \logit Pr(x) = \langle w, x \rangle + b \)
Graphically: IRT as logistic regression

Encoding “User \( i \) answered Item \( j \)” with **sparse features:**

\[
\langle \mathbf{w}, \mathbf{x} \rangle = \theta_i + e_j = \text{logit} \ Pr(\text{User } i \text{ Item } j \text{ OK})
\]
Oh, there’s a problem

<table>
<thead>
<tr>
<th></th>
<th>Users</th>
<th>Items</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$U_0$</td>
<td>$U_1$</td>
<td>$U_2$</td>
<td>$I_0$</td>
<td>$I_1$</td>
<td>$I_2$</td>
<td>$y_{\text{pred}}$</td>
<td>$y$</td>
<td></td>
</tr>
<tr>
<td>User 1 Item 1 OK</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.575135</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>User 1 Item 2 NOK</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.395036</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>User 2 Item 1 NOK</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.545417</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>User 2 Item 1 OK</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.545417</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>User 2 Item 2 NOK</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.366595</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

We predict the same thing when there are several attempts.
Performance Factor Analysis (Pavlik et al., 2009)

Keep counters over time:
\( W_{ik} (F_{ik}) \): how many successes (failures) of user \( i \) over skill \( k \)

- keep counters over time:
- \( W_{ik} \) (\( F_{ik} \)): how many successes (failures) of user \( i \) over skill \( k \)

\[
\text{logit } Pr(\text{User } i \text{ Item } j \text{ OK}) = \sum_{\text{Skill } k \text{ of Item } j} \beta_k + W_{ik} \gamma_k + F_{ik} \delta_k
\]

### Table

<table>
<thead>
<tr>
<th>Skills</th>
<th>Wins</th>
<th>Fails</th>
<th>y_{pred}</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>S_0</td>
<td>S_1</td>
<td>S_2</td>
<td>0.544</td>
<td>1</td>
</tr>
<tr>
<td>S_0</td>
<td>S_1</td>
<td>S_2</td>
<td>0.381</td>
<td>0</td>
</tr>
<tr>
<td>S_0</td>
<td>S_1</td>
<td>S_2</td>
<td>0.544</td>
<td>0</td>
</tr>
<tr>
<td>S_0</td>
<td>S_1</td>
<td>S_2</td>
<td>0.633</td>
<td>1</td>
</tr>
<tr>
<td>S_0</td>
<td>S_1</td>
<td>S_2</td>
<td>0.381</td>
<td>0</td>
</tr>
</tbody>
</table>
Model 3: a new model (but still logistic regression)

346860 attempts of 4217 students over 26688 items on 123 skills.

<table>
<thead>
<tr>
<th>model</th>
<th>dim</th>
<th>AUC</th>
<th>improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>KTM: items, skills, wins, fails</td>
<td>0</td>
<td>0.746</td>
<td>+0.06</td>
</tr>
<tr>
<td>IRT: users, items</td>
<td>0</td>
<td>0.691</td>
<td></td>
</tr>
<tr>
<td>PFA: skills, wins, fails</td>
<td>0</td>
<td>0.685</td>
<td>+0.07</td>
</tr>
<tr>
<td>AFM: skills, attempts</td>
<td>0</td>
<td>0.616</td>
<td></td>
</tr>
</tbody>
</table>
Here comes a new challenger

How to model pairwise interactions with side information?

**Logistic Regression**
Learn a 1-dim bias for each feature (each user, item, etc.)

**Factorization Machines**
Learn a 1-dim bias and a k-dim embedding for each feature
How to model pairwise interactions with side information?

If you know user $i$ attempted item $j$ on mobile (not desktop) How to model it?

$y$: score of event “user $i$ solves correctly item $j$”

<table>
<thead>
<tr>
<th>IRT</th>
<th>$y = \theta_i + e_j$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multidimensional IRT (similar to collaborative filtering)</td>
<td>$y = \theta_i + e_j + \langle \mathbf{v}<em>{user	ext{ }i}, \mathbf{v}</em>{item	ext{ }j} \rangle$</td>
</tr>
<tr>
<td>With side information</td>
<td>$y = \theta_i + e_j + w_{mobile} + \langle \mathbf{v}<em>{user	ext{ }i}, \mathbf{v}</em>{item	ext{ }j} \rangle + \langle \mathbf{v}<em>{user	ext{ }i}, \mathbf{v}</em>{mobile} \rangle + \langle \mathbf{v}<em>{item	ext{ }j}, \mathbf{v}</em>{mobile} \rangle$</td>
</tr>
</tbody>
</table>
Graphically: logistic regression

\[
\begin{align*}
\mathbf{w} & \quad \theta_i \\
\mathbf{x} & \quad 1
\end{align*}
\]

\[
\begin{align*}
\mathbf{U}_i & \\
\mathbf{l}_j & 1
\end{align*}
\]

abilities easinesses

Users Items
Graphically: factorization machines

\[
\begin{array}{c|c|c|c}
\text{Users} & \text{Items} & \text{Skills} \\
\hline
U_i & I_j & S_k \\
\hline
0 & 0 & 0 \\
0 & 0 & 1 \\
0 & 1 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{array}
\]
Formally: factorization machines

Each user, item, skill $k$ is modeled by bias $w_k$ and embedding $v_k$.

$$\logit p(x) = \mu + \sum_{k=1}^{N} w_k x_k + \sum_{1 \leq k < l \leq N} x_k x_l \langle v_k, v_l \rangle$$

$$= \mu + \langle w, x \rangle + \frac{1}{2} \left( \|Vx\|^2 - I^T (V \circ V)(x \circ x) \right)$$

Training using MCMC

Priors: $w_k \sim \mathcal{N}(\mu_0, 1/\lambda_0)$  \quad $v_k \sim \mathcal{N}(\mu, \Lambda^{-1})$

Hyperpriors: $\mu_0, \ldots, \mu_n \sim \mathcal{N}(0, 1), \lambda_0, \ldots, \lambda_n \sim \Gamma(1, 1) = U(0, 1)$

Algorithm 1 MCMC implementation of FMs

```c
for each iteration do
    Sample hyperp. $(\lambda_i, \mu_i)_i$ from posterior using Gibbs sampling
    Sample weights $w$
    Sample vectors $V$
    Sample predictions $y$
end for
```

Implementation in C++ (libFM) with Python wrapper (pyWFM).

## Datasets

### Fraction
500 middle-school students, 20 fraction subtraction questions, 8 skills (full matrix)

### Assistments
346860 attempts of 4217 students over 26688 math items on 123 skills (sparsity 0.997)

### Berkeley
On a MOOC of Computer Science, 562201 attempts of 1730 students over 234 items of 29 categories
Existing work on Assistments

<table>
<thead>
<tr>
<th>Model</th>
<th>Basically</th>
<th>Original AUC</th>
<th>Fixed AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bayesian Knowledge Tracing (Corbett and Anderson 1994)</td>
<td>Hidden Markov Model</td>
<td>0.67</td>
<td>0.63</td>
</tr>
<tr>
<td>Deep Knowledge Tracing (Piech et al. 2015)</td>
<td>Recurrent Neural Network</td>
<td>0.86</td>
<td>0.75</td>
</tr>
<tr>
<td>Item Response Theory (Rasch 1960)</td>
<td>Online Logistic Regression</td>
<td>0.76</td>
<td></td>
</tr>
<tr>
<td>(Wilson et al., 2016)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Knowledge Tracing Machines</td>
<td>Factorization Machines</td>
<td>0.82</td>
<td></td>
</tr>
</tbody>
</table>

AUC results on the Assistments dataset

<table>
<thead>
<tr>
<th>model</th>
<th>dim</th>
<th>AUC</th>
<th>improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>KTM: items, skills, wins, fails, extra</td>
<td>5</td>
<td>0.819</td>
<td></td>
</tr>
<tr>
<td>KTM: items, skills, wins, fails, extra</td>
<td>0</td>
<td>0.815</td>
<td>+0.05</td>
</tr>
<tr>
<td>KTM: items, skills, wins, fails</td>
<td>10</td>
<td>0.767</td>
<td></td>
</tr>
<tr>
<td>KTM: items, skills, wins, fails</td>
<td>0</td>
<td>0.759</td>
<td>+0.02</td>
</tr>
<tr>
<td><em>DKT</em> (Wilson et al., 2016)</td>
<td>100</td>
<td>0.743</td>
<td>+0.05</td>
</tr>
<tr>
<td>IRT: users, items</td>
<td>0</td>
<td>0.691</td>
<td></td>
</tr>
<tr>
<td>PFA: skills, wins, fails</td>
<td>0</td>
<td>0.685</td>
<td>+0.07</td>
</tr>
<tr>
<td>AFM: skills, attempts</td>
<td>0</td>
<td>0.616</td>
<td></td>
</tr>
</tbody>
</table>
Bonus: interpreting the learned embeddings
What 'bout recurrent neural networks?

Deep Knowledge Tracing: knowledge tracing as sequence prediction

- Each student on skill $q_t$ has performance $a_t$
- How to predict outcomes $y$ on every skill $k$?
- Spoiler: by measuring the evolution of a latent state $h_t$


Our approach: encoder-decoder

$$\begin{align*}
  h_t &= Encoder(h_{t-1}, x_{t}^{in}) \\
  p_t &= Decoder(h_t, x_{t}^{out})
\end{align*}$$

$t = 1, \ldots, T$
Graphically: deep knowledge tracing

$h_0 \rightarrow h_1 \rightarrow h_2 \rightarrow h_3$

$q_0, a_0 \rightarrow \rightarrow \rightarrow q_1, a_1 \rightarrow \rightarrow \rightarrow q_2, a_2 \rightarrow \rightarrow \rightarrow$

$y = y_0 \cdots y_{q_1} \cdots y_{M-1}$
DKT seen as encoder-decoder

\[ y_{q_1} = \sigma(\langle h_1, v_{q_1} \rangle) \]

\[ y_{q_2} \]

\[ y_{q_3} \]
Results on Fraction dataset

500 middle-school students, 20 Fraction subtraction questions, 8 skills (full matrix)

<table>
<thead>
<tr>
<th>Model</th>
<th>Encoder</th>
<th>Decoder</th>
<th>$x_t^{out}$</th>
<th>ACC</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours</td>
<td>GRU $d = 2$</td>
<td>bias</td>
<td>iswf</td>
<td>0.880</td>
<td>0.944</td>
</tr>
<tr>
<td>KTM</td>
<td>counter</td>
<td>bias</td>
<td>iswf</td>
<td>0.853</td>
<td>0.918</td>
</tr>
<tr>
<td>PFA</td>
<td>counter</td>
<td>bias</td>
<td>swf</td>
<td>0.854</td>
<td>0.917</td>
</tr>
<tr>
<td>Ours</td>
<td>$\emptyset$</td>
<td>bias</td>
<td>iswf</td>
<td>0.849</td>
<td>0.917</td>
</tr>
<tr>
<td>Ours</td>
<td>GRU $d = 50$</td>
<td>$\emptyset$</td>
<td>iswf</td>
<td>0.814</td>
<td>0.880</td>
</tr>
<tr>
<td>DKT</td>
<td>GRU $d = 2$</td>
<td>$d = 2$</td>
<td>$s$</td>
<td>0.772</td>
<td>0.844</td>
</tr>
<tr>
<td>Ours</td>
<td>GRU $d = 2$</td>
<td>$\emptyset$</td>
<td>iswf</td>
<td>0.751</td>
<td>0.800</td>
</tr>
</tbody>
</table>
562201 attempts of 1730 students over 234 CS-related items of 29 categories.

<table>
<thead>
<tr>
<th>Model</th>
<th>Encoder</th>
<th>Decoder</th>
<th>$x_t^{out}$</th>
<th>ACC</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours</td>
<td>GRU $d = 50$</td>
<td>bias</td>
<td>iswf</td>
<td>0.707</td>
<td>0.778</td>
</tr>
<tr>
<td>KTM</td>
<td>counter</td>
<td>bias</td>
<td>iswf</td>
<td>0.704</td>
<td>0.775</td>
</tr>
<tr>
<td>Ours</td>
<td>$\emptyset$</td>
<td>bias</td>
<td>iswf</td>
<td>0.700</td>
<td>0.770</td>
</tr>
<tr>
<td>DKT</td>
<td>GRU $d = 50$</td>
<td>$d = 50$</td>
<td>s</td>
<td>0.684</td>
<td>0.751</td>
</tr>
<tr>
<td>Ours</td>
<td>GRU $d = 100$</td>
<td>$\emptyset$</td>
<td></td>
<td>0.682</td>
<td>0.750</td>
</tr>
<tr>
<td>PFA</td>
<td>counter</td>
<td>bias</td>
<td>swf</td>
<td>0.630</td>
<td>0.683</td>
</tr>
<tr>
<td>DKT</td>
<td>GRU $d = 2$</td>
<td>$d = 2$</td>
<td>s</td>
<td>0.637</td>
<td>0.656</td>
</tr>
</tbody>
</table>

Factorization machines unify many existing EDM models

- Side information improves performance more than higher $d$
- We can visualize learning (and provide feedback to learners)

They can be combined with deep neural networks

- Unidimensional decoders perform better
- But simple counters are good enough encoders

Then we can optimize learning

- Increase success rate of the student
  (Clement et al., JEDM 2015)
- Identify something that the student does not know
  (Teng et al., ICDM 2018, Seznec et al., AISTATS 2019)
- See more on https://humanlearn.io
Merci ! Do you have any questions?

https://jilljenn.github.io

I’m interested in:

- predicting student performance
- optimizing human learning using reinforcement learning
- (manga) recommender systems

We are organizing a workshop on June 3–4, 2019 Optimizing Human Learning (Kingston, Jamaica) colocated with Intelligent Tutoring Systems, ITS 2019 CFP open until April 16, 2019: https://humanlearn.io

vie@jill-jenn.net


