Knowledge Tracing Machines
Towards an unification of DKT, IRT & PFA

Jill-Jênn Vie

Optimizing Human Learning, June 12, 2018

Predicting student performance

Data

A population of students answering questions

Goal

- Measure their knowledge
- Potentially optimize their learning

Assumption

Good model for prediction $\rightarrow$ Good adaptive policy for teaching

Learning outcomes of this tutorial

- Logistic regression is amazing
- Factorization machines are even more amazing
- Recurrent neural networks are boring (but useful)

Families of models

- Factorization Machines (Rendle 2012)
  - Multidimensional Item Response Theory
  - Logistic Regression
    - Item Response Theory
    - Performance Factor Analysis
- Recurrent Neural Networks
  - Deep Knowledge Tracing (Piech et al. 2015)

Steffen Rendle (2012). "Factorization Machines with libFM". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 3.3, 57:1–57:22. DOI: 10.1145/2168752.2168771

Chris Piech et al. (2015). "Deep knowledge tracing". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 505–513

Problems

Weak generalization

Filling the blanks: some students did not attempt all questions

Strong generalization

Cold-start: some new students are not in the train set

Dummy 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 ???

user	item	correct
1	1	1
1	2	0
2	1	0
2	1	1
2	2	0

Task 1: Item Response Theory

Learn abilities θ_i for each user i
Learn easiness e_j for each item j such that:

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

Logistic regression

Learn $\boldsymbol{w}$ such that $\text{logit } Pr(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle$

Graphically: IRT as logistic regression

Encoding of "User i answered Item j":

	Users	Items
$\boldsymbol{w}$	$\boxed{\theta_i}$	$\boxed{e_j}$
	U_i	I_j
$\boldsymbol{x}$	1	1

$$\text{logit } Pr(\text{User } i \text{ Item } j \text{ OK}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle = \theta_i + e_j$$

Time to experiment

Cf. `README.md` @ `https://github.com/jilljenn/ktm`

```
git clone https://github.com/jilljenn/ktm
cd ktm
python3 -m venv venv    # Python 2 OK
. venv/bin/activate
pip install -r requirements.txt

git clone https://github.com/srendle/libfm
cd libfm
git reset --hard 91f8504a15120ef6815d6e10cc7dee42eebaab0f
make all
```

Encoding

```
python encode.py --users --items
# Creates data/dummy/X-ui.npz
```

Users			Items		
U_0	U_1	U_2	I_0	I_1	I_2
0	1	0	0	1	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	1	0	1	0
0	0	1	0	0	1

Then logistic regression can be run on the sparse features:

```
python lr.py data/dummy/X-ui.npz
```

Oh, there's a problem

```
python encode.py --users --items
python lr.py data/dummy/X-ui.npz
```

	Users			Items				
	U_0	U_1	U_2	I_0	I_1	I_2	y_{pred}	y
User 1 Item 1 OK	0	1	0	0	1	0	0.575135	1
User 1 Item 2 NOK	0	1	0	0	0	1	0.395036	0
User 2 Item 1 NOK	0	0	1	0	1	0	0.545417	0
User 2 Item 1 OK	0	0	1	0	1	0	0.545417	1
User 2 Item 2 NOK	0	0	1	0	0	1	0.366595	0

We predict the same thing when there are several attempts.

Count successes and failures

Keep track of what the student has done before:

user	item	skill	correct	wins	fails
1	1	1	1	0	0
1	2	2	0	0	0
2	1	1	0	0	0
2	1	1	1	0	1
2	2	2	0	0	0

Task 2: Performance Factor Analysis

W_{ik}: how many successes of user i over skill k (F_{ik}: #failures)

Learn β_k, γ_k, δ_k for each skill k such that:

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

```
python encode.py --skills --wins --fails
```

Skills			Wins			Fails		
S_0	S_1	S_2	S_0	S_1	S_2	S_0	S_1	S_2
0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0

Better!

```
python encode.py --skills --wins --fails

python lr.py data/dummy/X-swf.npz
```

	Skills			Wins			Fails				
	S_0	S_1	S_2	S_0	S_1	S_2	S_0	S_1	S_2	y_{pred}	y
User 1 Item 1 OK	0	1	0	0	0	0	0	0	0	0.544	1
User 1 Item 2 NOK	0	0	1	0	0	0	0	0	0	0.381	0
User 2 Item 1 NOK	0	1	0	0	0	0	0	0	0	0.544	0
User 2 Item 1 OK	0	1	0	0	0	0	0	1	0	0.633	1
User 2 Item 2 NOK	0	0	1	0	0	0	0	0	0	0.381	0

Task 3: a new model (but still logistic regression)

```
python encode.py --items --skills --wins --fails

python lr.py data/dummy/X-iswf.npz
```

Here comes a new challenger

How to model side information in recommender systems?

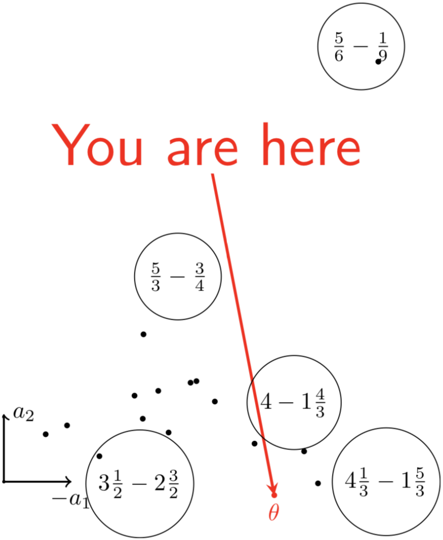
Logistic Regression

Learn a bias for each feature (each user, item, etc.)

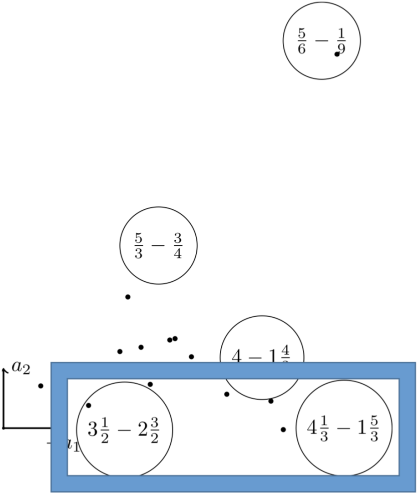
Factorization Machines

Learn a bias and an embedding for each feature

The power of embeddings



Interpreting the components



Items that discriminate only over one dimension

$3\frac{1}{2} - 2\frac{3}{2}$

$b = 0.13$
$-a_1 = 2.01$
$a_2 = -0.03$

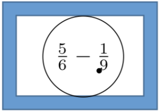
$4\frac{1}{3} - 2\frac{4}{3}$

$b = -0.46$
$-a_1 = 4.65$
$a_2 = -0.02$

$4\frac{1}{3} - 1\frac{5}{3}$

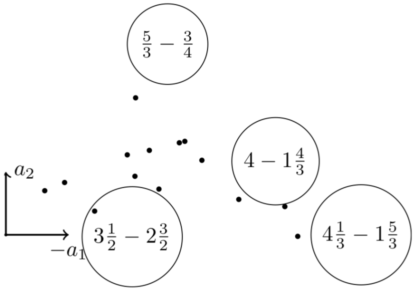
$b = -1.99$
$-a_1 = 5.66$
$a_2 = 0.00$

Interpreting the components

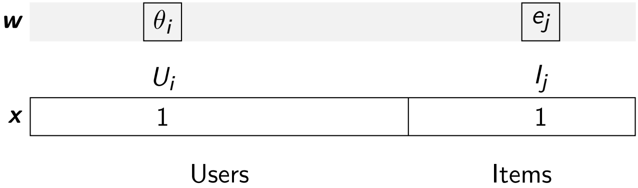


Items that highly discriminate over both dimensions

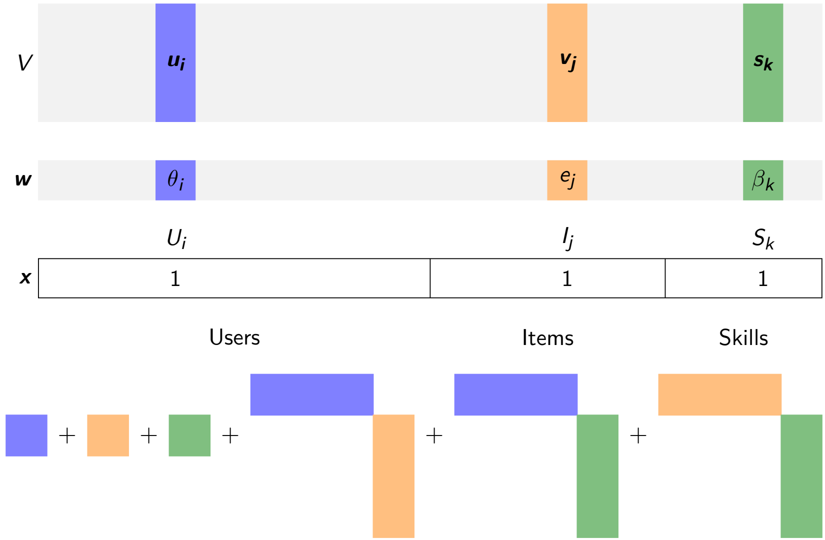
$\frac{3}{4} - \frac{3}{8}$	$\frac{5}{6} - \frac{1}{9}$
$b = 1.09$	$b = -0.28$
$-a_1 = 5.54$	$-a_1 = 5.29$
$a_2 = 6.22$	$a_2 = 6.44$



Graphically: logistic regression



Graphically: factorization machines



Formally: factorization machines

Learn bias w_k and embedding $\boldsymbol{v_k}$ for each feature k such that:

$$\operatorname{logit} p(\boldsymbol{x}) = \mu + \underbrace{\sum_{k=1}^{N} w_k x_k}_{\text{logistic regression}} + \underbrace{\sum_{1 \leq k < l \leq N} x_k x_l \langle \boldsymbol{v_k}, \boldsymbol{v_l} \rangle}_{\text{pairwise interactions}}$$

Particular cases

- Multidimensional item response theory: $\operatorname{logit} p = \langle \boldsymbol{u_i}, \boldsymbol{v_j} \rangle + e_j$
- SPARFA: $\boldsymbol{v_j} > \mathbf{0}$ and $\boldsymbol{v_j}$ sparse
- GenMA: $\boldsymbol{v_j}$ is constrained by the zeroes of a q-matrix

Andrew S Lan et al. (2014). "Sparse factor analysis for learning and content analytics". In: *The Journal of Machine Learning Research* 15.1, pp. 1959–2008

Jill-Jênn Vie et al. (2016). "Adaptive Testing Using a General Diagnostic Model". In: *European Conference on Technology Enhanced Learning*. Springer, pp. 331–339

Assistments 2009 dataset

278608 attempts of 4163 students over 196457 items on 124 skills.

- Download `http://jiji.cat/weasel2018/data.csv`
- Put it in `data/assistments09`

`python fm.py data/assistments09/X-ui.npz`
etc. or `make big`

AUC	users + items	skills + w + f	items + skills + w + f
LR	0.734 (IRT) 2s	0.651 (PFA) 9s	0.737 23s
FM	0.730 2min9s	0.652 43s	0.739 2min30s

Results obtained with FM $d = 20$

Deep Factorization Machines

Learn layers $W^{(\ell)}$ and $b^{(\ell)}$ such that:

$$\begin{aligned}
\boldsymbol{a}^0(\boldsymbol{x}) &= (\boldsymbol{v}_{\texttt{user}}, \boldsymbol{v}_{\texttt{item}}, \boldsymbol{v}_{\texttt{skill}}, \ldots) \\
\boldsymbol{a}^{(\ell+1)}(\boldsymbol{x}) &= \mathrm{ReLU}(W^{(\ell)}\boldsymbol{a}^{(\ell)}(\boldsymbol{x}) + \boldsymbol{b}^{(\ell)}) \quad \ell = 0, \ldots, L-1 \\
y_{DNN}(\boldsymbol{x}) &= \mathrm{ReLU}(W^{(L)}\boldsymbol{a}^{(L)}(\boldsymbol{x}) + \boldsymbol{b}^{(L)})
\end{aligned}$$

$$\operatorname{logit} p(\boldsymbol{x}) = y_{FM}(\boldsymbol{x}) + y_{DNN}(\boldsymbol{x})$$

When trained, performance was lower than Bayesian FMs.

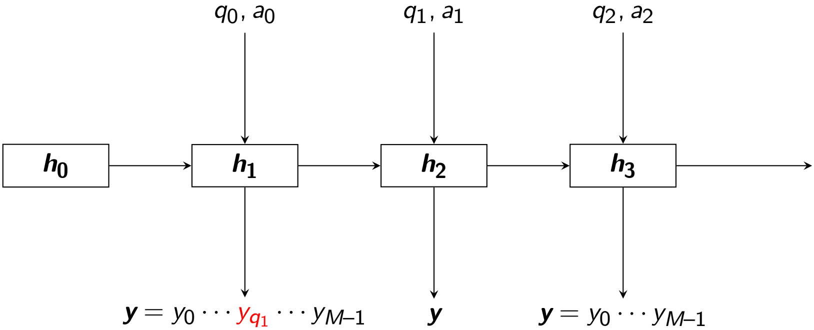
Jill-Jênn Vie (2018). "Deep Factorization Machines for Knowledge Tracing". In: *The 13th Workshop on Innovative Use of NLP for Building Educational Applications*. URL: https://arxiv.org/abs/1805.00356

What 'bout recurrent neural networks?

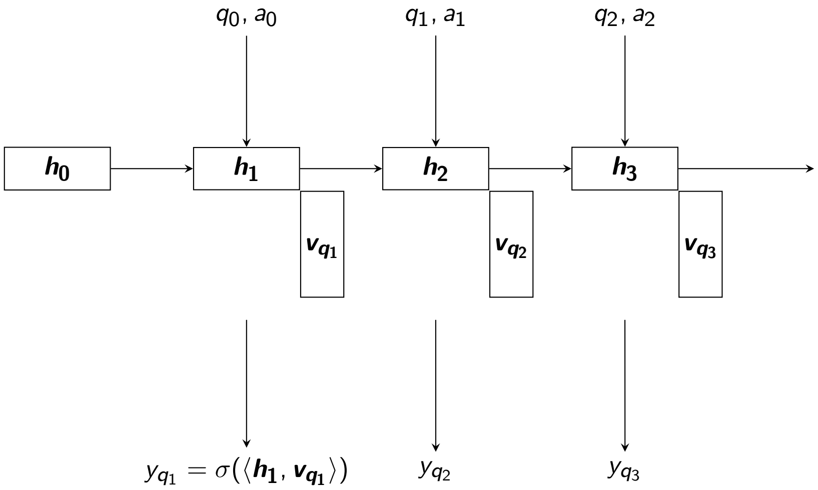
Deep Knowledge Tracing: model the problem as sequence prediction

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

Graphically: deep knowledge tracing



Graphically: there is a MIRT in my DKT



Drawback

DKT does not model individual differences.

By estimating on-the-fly the student's learning ability, we managed to get a better model.

AUC	BKT	IRT	PFA	DKT	DKT-DSC
Cognitive Tutor	0.61	0.81	0.76	0.79	0.81
Assistments 2009	0.67	0.75	0.70	0.73	0.92
Assistments 2012	0.61	0.74	0.67	0.72	0.80
Assistments 2014	0.64	0.67	0.69	0.72	0.86

Sein Minn et al. (2018). "Deep Knowledge Tracing and Dynamic Student Classification for Knowledge Tracing". Submitted at IEEE International Conference on Data Mining.

Take home message

Factorization machines are a strong baseline that take many models as special cases

Recurrent neural networks are powerful because they track the evolution of the latent state (try simpler dynamic models?)

Deep factorization machines may require more data/tuning, but neural collaborative filtering offer promising directions

Any suggestions are welcome!

`vie@jill-jenn.net`

`github.com/jilljenn/ktm`

Questions?

- Lan, Andrew S et al. (2014). "Sparse factor analysis for learning and content analytics". In: *The Journal of Machine Learning Research* 15.1, pp. 1959–2008.
- Minn, Sein et al. (2018). "Deep Knowledge Tracing and Dynamic Student Classification for Knowledge Tracing". Submitted at IEEE International Conference on Data Mining.
- Piech, Chris et al. (2015). "Deep knowledge tracing". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 505–513.
- Rendle, Steffen (2012). "Factorization Machines with libFM". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 3.3, 57:1–57:22. DOI: 10.1145/2168752.2168771.
- Vie, Jill-Jênn (2018). "Deep Factorization Machines for Knowledge Tracing". In: *The 13th Workshop on Innovative Use of NLP for Building Educational Applications*. URL: https://arxiv.org/abs/1805.00356.

Vie, Jill-Jênn et al. (2016). "Adaptive Testing Using a General Diagnostic Model". In: *European Conference on Technology Enhanced Learning*. Springer, pp. 331–339.