Knowledge Tracing Machines: Factorization Machines for Knowledge Tracing

Jill-Jênn Vie Hisashi Kashima

京都大学 KYOTO UNIVERSITY

AAAI 2019

https://arxiv.org/abs/1811.03388

AI for Social Good

AI can:

- recognize images
- recognize speech
- create fakes (generation)
- play go (decision making)

as long as you have enough data.

Can it also:

- improve education
- automatic exercise generation
- prediction of student performance
- optimizing human learning

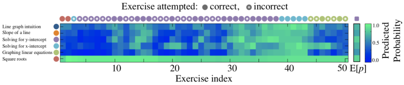
as long as you have enough data?

Student try exercises

Items	$5 - 5 = ?$	$17 - 3 = ?$	$13 - 7 = ?$
Student	correct	correct	incorrect

What can be learned from this?

Predicting student performance: knowledge tracing



Data

A population of students answering questions

- Events: "Student i answered question j correctly/incorrectly"

Side information

- Knowledge components (skills), class ID, school ID, etc.

Goal

- Learn the difficulty of questions automatically from data
- Measure the knowledge of students
- Potentially optimize their learning

Assumption

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

Limitations

- Several models for KT were developed independently
- Some models cannot handle multiple skills at the same time

In this paper

- KTM unify most models
  - Encoding to sparse features
  - Then running logistic regression or FM
- KTM can handle multiple skills
- And build upon them to achieve higher performance

Our Contributions

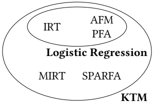
Knowledge Tracing Machines unify many existing EDM models

- It is better to estimate an item per bias, not only per skill
- Side information improves performance more than higher dim.
- Use of factorization machines in the context of educational data mining

Recurrent neural networks are powerful because they learn a more complex function that tracks the evolution of the latent state

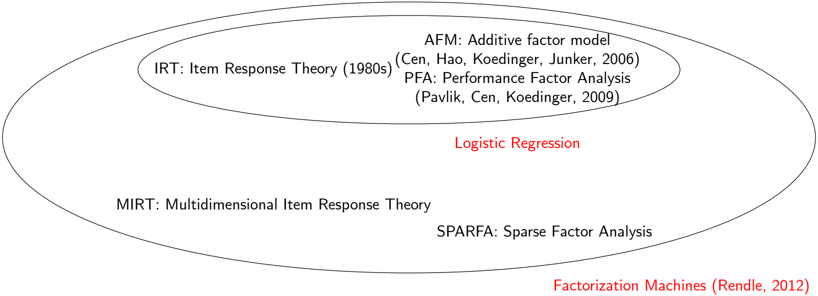
- DKT cannot handle multiple skills.
- Most existing models (like DKT) cannot handle multiple skills, but KTM do
- We can combine DKT with side information
- Actually, Wilson, Karklin, Han, and Ekanadham (2016) even managed to beat DKT with (1-dim!) IRT.

Learning outcomes of this presentation



- Existing models
  - Example: on a dummy dataset
  - Encoding into logistic regression
  - Results on big data
- Knowledge Tracing Machines
  - How to model pairwise interactions
  - Training using MCMC
  - Results on several datasets
- Future Work
  - It makes sense to consider deep neural networks
  - What does deep knowledge tracing model exactly?

Existing models



Not in the family: 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

Dummy dataset: weak generalization

Weak generalization

Filling the blanks: some students did not attempt all questions

- 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	???

`dummy.csv`

Dummy dataset: strong generalization

Strong generalization

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

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

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

`dummy.csv`

Model 1: Item Response Theory

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

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

Logistic regression

Learn $\boldsymbol{w}$ such that logit $Pr(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle$
Usually with L2 regularization: $||\boldsymbol{w}||_2^2$ penalty $\leftrightarrow$ Gaussian prior

Graphically: IRT as logistic regression

Encoding "User i answered Item j" with sparse features:

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

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

Encoding into sparse features

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.

Oh, there's a problem

	Users			Items				
	U_0	U_1	U_2	I_0	I_1	I_2	$\boldsymbol{y}_{\text{pred}}$	$\boldsymbol{y}$
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 number of attempts: AFM

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

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

Count successes and failures: PFA

$\boldsymbol{w}$ $\boxed{\theta_i}$ $\boxed{e_j}$

U_i I_j

$\boldsymbol{x}$ | 1 | 1 |

Users Items

Separate successes W_{ik} and fails F_{ik} of student i over skill k.

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

Model 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:

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

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!

	Skills			Wins			Fails				
	S_0	S_1	S_2	S_0	S_1	S_2	S_0	S_1	S_2	$\boldsymbol{y}_{\text{pred}}$	$\boldsymbol{y}$
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

Test on a large dataset: Assistments 2009

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

model	dim	AUC	improvement
PFA: skills, wins, fails	0	0.685	+0.07
AFM: skills, attempts	0	0.616	

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

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

model	dim	AUC	improvement
KTM: items, skills, wins, fails	0	0.746	+0.06
IRT: users, items	0	0.691	+0.06
PFA: skills, wins, fails	0	0.685	+0.07
AFM: skills, attempts	0	0.616	

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"

IRT

$$y = \theta_i + e_j$$

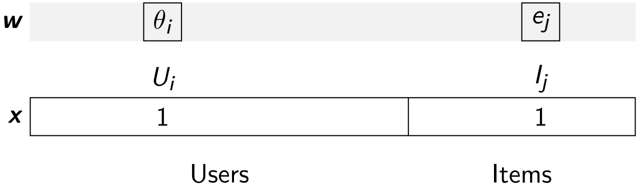
Multidimensional IRT (similar to collaborative filtering)

$$y = \theta_i + e_j + \langle \boldsymbol{v}_{\text{user } \boldsymbol{i}}, \boldsymbol{v}_{\text{item } \boldsymbol{j}} \rangle$$

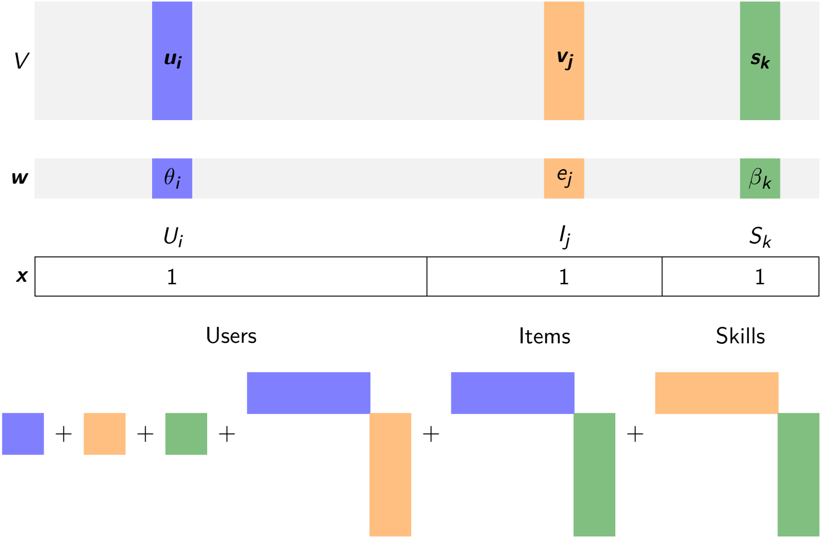
With side information

$$y = \theta_i + e_j + \langle \boldsymbol{v}_{\text{user } \boldsymbol{i}}, \boldsymbol{v}_{\text{item } \boldsymbol{j}} \rangle + \langle \boldsymbol{v}_{\text{user } \boldsymbol{i}}, \boldsymbol{v}_{\text{mobile}} \rangle + \langle \boldsymbol{v}_{\text{item } \boldsymbol{j}}, \boldsymbol{v}_{\text{mobile}} \rangle$$

Graphically: logistic regression



Graphically: factorization machines



Formally: factorization machines

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

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

Multidimensional item response theory: $\operatorname{logit} p(\boldsymbol{x}) = \langle \boldsymbol{u_i}, \boldsymbol{v_j} \rangle + e_j$ is a particular case.

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

Training using MCMC

Algorithm 1 MCMC implementation of FMs

Prior on every V

for each iteration **do**

  Sample hyperparameters from posterior using MCMC

  Sample weights w

  Sample vectors V

  Sample predictions y

end for

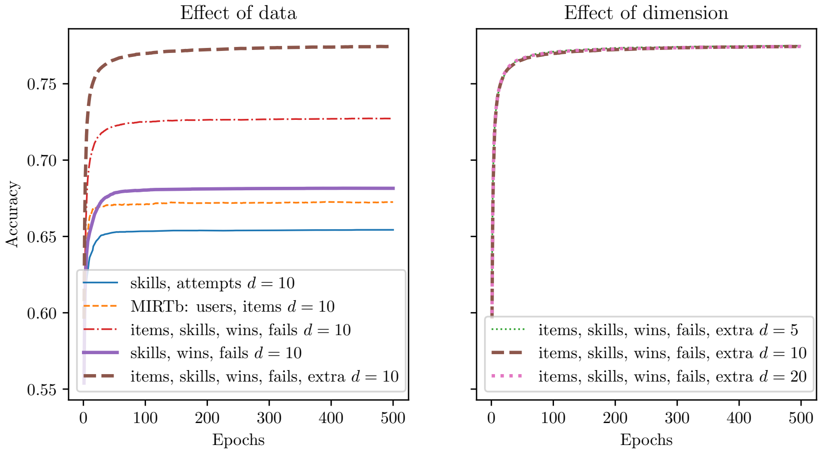
Implementation libFM with pyWFM wrapper.

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

Datasets

Name	Users	Items	Skills	Skills/i	Entries	Sparsity	Attempts/u
fraction	536	20	8	2.800	10720	0.000	1.000
timss	757	23	13	1.652	17411	0.000	1.000
ecpe	2922	28	3	1.321	81816	0.000	1.000
assistments	4217	26688	123	0.796	346860	0.997	1.014
berkeley	1730	234	29	1.000	562201	0.269	1.901
castor	58939	17	2	1.471	1001963	0.000	1.000

Results on the Assistments dataset



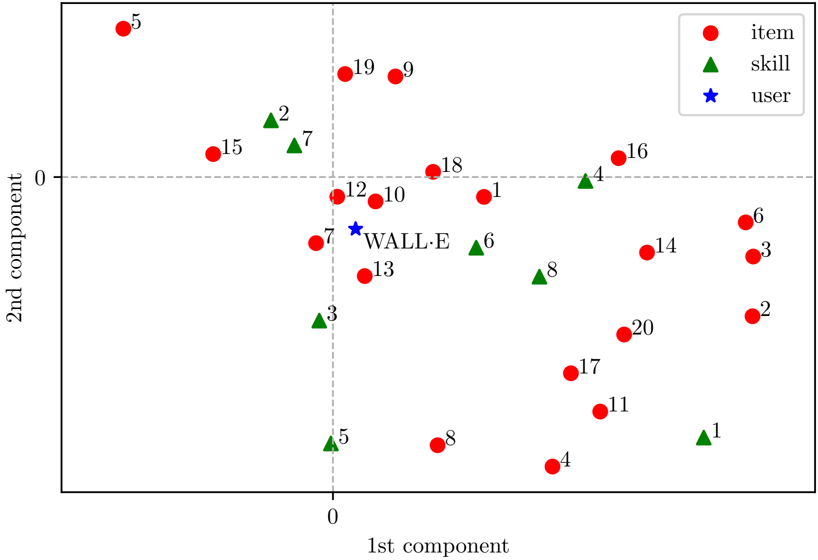
Accuracy results on the Assistments dataset

model	dim	AUC	improvement
KTM: items, skills, wins, fails	10	0.752	+0.01
KTM: items, skills, wins, fails	0	0.746	
DKT (Wilson et al., 2016)	100	0.743	+0.05
IRT: users, items	0	0.691	+0.06
PFA: skills, wins, fails	0	0.685	+0.07
AFM: skills, attempts	0	0.616	

AUC results on all datasets

AUC	AFM	PFA	IRT	MIRTb20	KTM(iswf0)	KTM(iswf20)	KTM(iswfe5)
assistments	0.6163	0.6849	0.6908	0.6907	0.7589	0.7502	**0.8186**
berkeley	0.675	0.6839	0.7532	0.7519	0.7753	**0.7780**	–
ecpe	–	–	**0.6811**	**0.6810**	–	–	–
fraction	–	–	0.6662	**0.6672**	–	–	–
timss	–	–	**0.6946**	0.6932	–	–	–
castor	–	–	**0.7603**	0.7599	–	–	–

Bonus: interpreting the learned embeddings



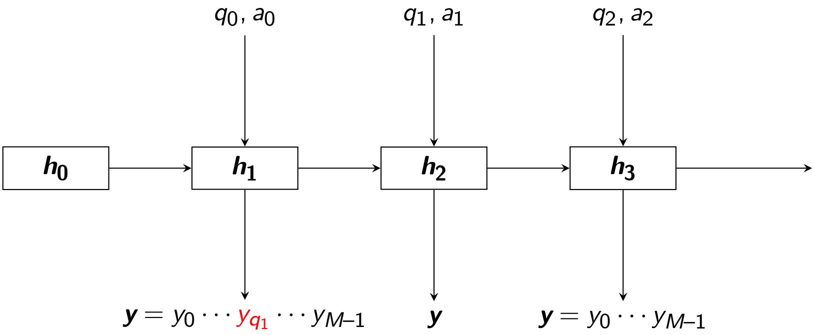
What 'bout recurrent neural networks?

Deep Knowledge Tracing: model the problem as sequence prediction

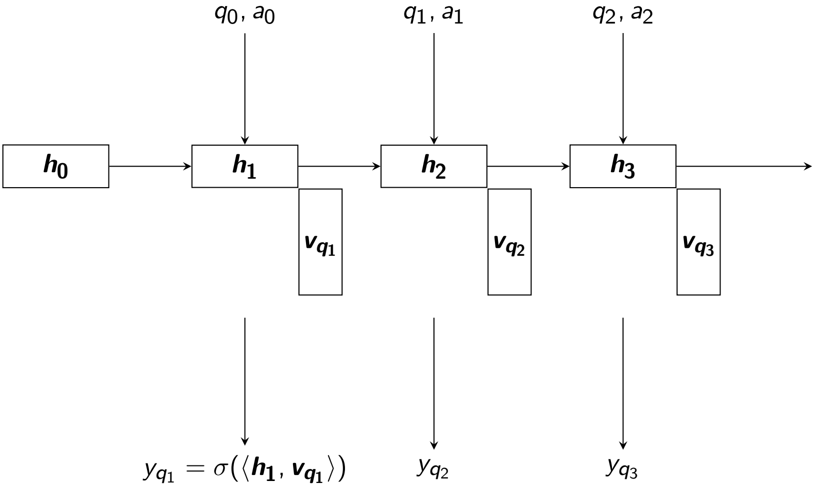
- Each student on skill q_t has performance a_t
- How to predict outcomes $\boldsymbol{y}$ on every skill k?
- Spoiler: by measuring the evolution of a latent state $\boldsymbol{h_t}$

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

Graphically: deep knowledge tracing



Graphically: there is a MIRT in my DKT



Improvement over Deep Knowledge Tracing

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

AUC	BKT	IRT	PFA	DKT	DKT-DSC
Assistments 2009	0.67	0.75	0.70	0.73	0.91
Assistments 2012	0.61	0.74	0.67	0.72	0.87
Assistments 2014	0.64	0.67	0.69	0.72	0.87
Cognitive Tutor	0.61	0.81	0.76	0.79	0.81

Sein Minn, Yi Yu, Michel Desmarais, Feida Zhu, and Jill-Jênn Vie (2018). "Deep Knowledge Tracing and Dynamic Student Classification for Knowledge Tracing". In: *Proceedings of the 18th IEEE International Conference on Data Mining*, to appear. URL: https://arxiv.org/abs/1809.08713

Results

model	dim	AUC	improvement
DKT-DSC + *KTM* (<You>, 2019?)	200	???	
DKT-DSC (Minn et al., 2018)	200	0.910	+0.16
KTM: items, skills, wins, fails	10	0.752	+0.01
KTM: items, skills, wins, fails	0	0.746	
DKT (Wilson et al., 2016)	100	0.743	+0.05
IRT: users, items	0	0.691	+0.06
PFA: skills, wins, fails	0	0.685	+0.07
AFM: skills, attempts	0	0.616	

Future work

- Side info in DKT
- Adaptive testing
- Higher order
- Response time, spaced repetition
- Ordinal regression

Take home message

Factorization machines are a strong baseline that unifies many existing EDM models

- It is better to estimate an item per bias, not only per skill
- Side information improves performance more than higher d

Recurrent neural networks are powerful because they track the evolution of the latent state

- Most existing models (like DKT) cannot handle multiple skills, but KTM do
- We should combine DKT with side information

Any suggestions are welcome!

Read our article:

Knowledge Tracing Machines

https://arxiv.org/abs/1811.03388

Try the code:

https://github.com/jilljenn/ktm

Feel free to chat:

vie@jill-jenn.net

Do you have any questions?

Minn, Sein, Yi Yu, Michel Desmarais, Feida Zhu, and Jill-Jênn Vie (2018). "Deep Knowledge Tracing and Dynamic Student Classification for Knowledge Tracing". In: *Proceedings of the 18th IEEE International Conference on Data Mining*, to appear. URL: https://arxiv.org/abs/1809.08713.

Piech, Chris, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein (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.

Wilson, Kevin H., Yan Karklin, Bojian Han, and Chaitanya Ekanadham (2016). "Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation". In: *Proceedings of the 9th International Conference on Educational Data Mining (EDM)*, pp. 539–544.