Knowledge Tracing: Predicting & Optimizing Human Learning

Jill-Jênn Vie Hisashi Kashima

AIP 京都大学 KYOTO UNIVERSITY

AIP-IITH workshop, March 15, 2019

Topics

- 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

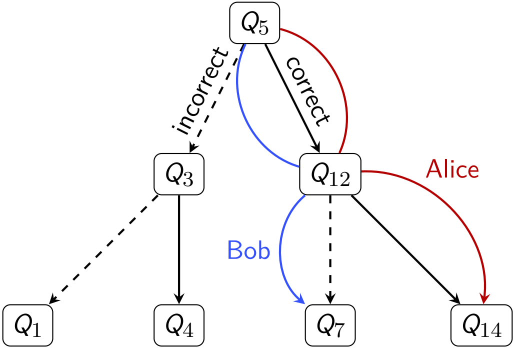
Machine teaching

Feed the best sequence of samples to train a known algorithm

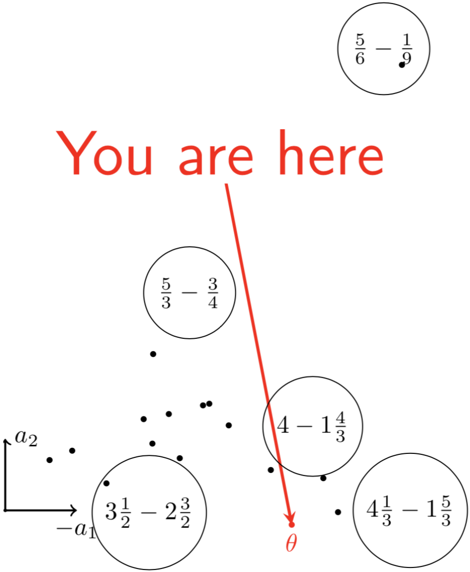
Practical intro

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

Items	$5 - 5 = ?$	$17 - 3 = ?$	$13 - 7 = ?$
New student	○	○	×

Language Learning

	PRON	VERB	PRON	NOUN	CONJ	PRON	VERB	PRON	NOUN
correct:	She	is	my	mother	and	he	is	my	father
student:	she	is		mader	and	he	is		fhader
label:	O	O	✖	✖	O	O	O	✖	✖

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 ???

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

`dummy.csv`

Our approach

- Encode data to sparse features

user	item	correct
2	2	1
2	2	0
2	2	0
2	3	0
2	3	1
1	2	???
1	1	???

data.csv

$\xrightarrow{\text{encode}}$

KTM (Items, Skills, Wins, Fails)

PFA (Skills, Wins, Fails)

IRT (Users, Items)

Users		Items			Skills			Wins			Fails		
1	2	Q_1	Q_2	Q_3	KC_1	KC_2	KC_3	KC_1	KC_2	KC_3	KC_1	KC_2	KC_3
0	1	0	1	0	1	1	0	0	0	0	0	0	0
0	1	0	1	0	1	1	0	1	1	0	0	0	0
0	1	0	1	0	1	1	0	1	1	0	1	1	0
0	1	0	0	1	0	1	1	0	2	0	0	1	0
0	1	0	0	1	0	1	1	0	2	0	0	2	1
1	0	0	1	0	1	1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0

sparse matrix X

- Run logistic regression or factorization machines
  $\Rightarrow$ recover existing models or better models

Simplest baseline: Item Response Theory (Rasch, 1960)

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

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

Really popular model, used for the PISA assessment

Can be encoded as logistic regression

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

Graphically: IRT as logistic regression

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

$$\begin{array}{rcc}
 & \text{abilities} & \text{easinesses} \\
\boldsymbol{w} & \boxed{\theta_i} & \boxed{e_j} \\
 & U_i & I_j \\
\boldsymbol{x} & 1 & 1 \\
 & \text{Users} & \text{Items}
\end{array}$$

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

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.

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

	Skills	Wins	Fails
	easiness of skill	bonus per success	bonus per failure
$\boldsymbol{w}$	$\boxed{\beta_i}$	$\boxed{\gamma_j}$	$\boxed{\delta_j}$
	S_k	W_{ik}	F_{ik}
$\boldsymbol{x}$	1	1	1

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

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

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

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

model	dim	AUC	improvement
KTM: items, skills, wins, fails	0	0.746	+0.06
IRT: users, items	0	0.691	
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

$$y = \theta_i + e_j$$

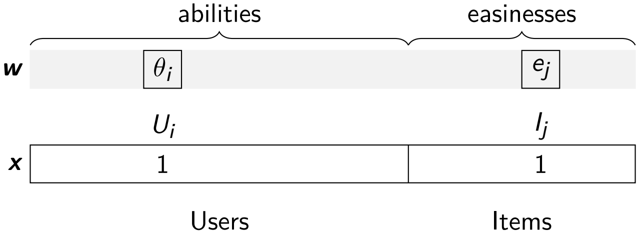
Multidimensional IRT (similar to collaborative filtering)

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

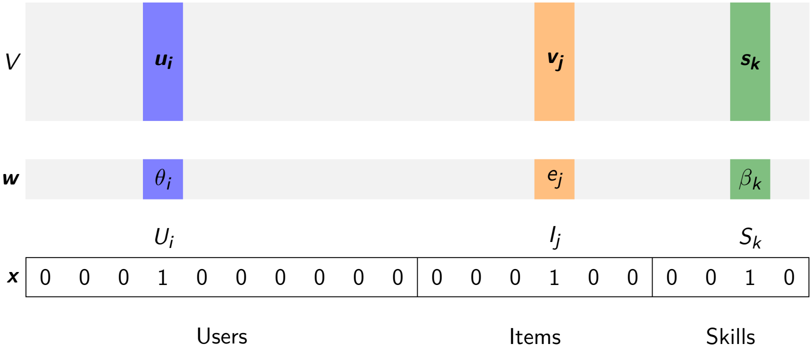
With side information

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

Graphically: logistic regression

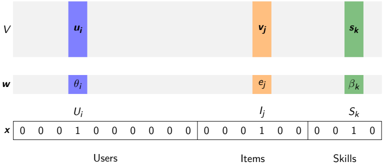


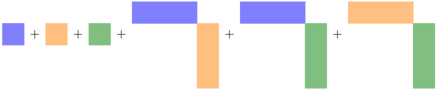
Graphically: factorization machines



Formally: factorization machines

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





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

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

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

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

```
for each iteration do
    Sample hyperp. (λ_i, μ_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).

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

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

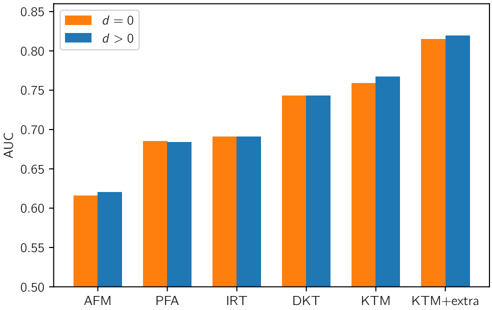
On a MOOC of Computer Science, 562201 attempts of 1730 students over 234 items of 29 categories

Existing work on Assistments

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

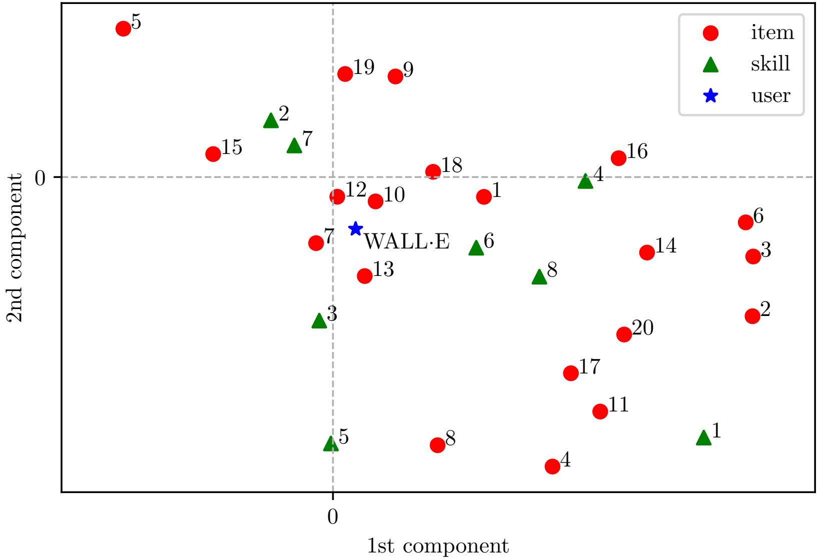
Jill-Jênn Vie and Hisashi Kashima (2019). "Knowledge Tracing Machines: Factorization Machines for Knowledge Tracing". In: *33th AAAI Conference on Artificial Intelligence*. URL: http://arxiv.org/abs/1811.03388

AUC results on the Assistments dataset



model	dim	AUC	improvement
KTM: items, skills, wins, fails, extra	5	0.819	
KTM: items, skills, wins, fails, extra	0	0.815	+0.05
KTM: items, skills, wins, fails	10	0.767	
KTM: items, skills, wins, fails	0	0.759	+0.02
DKT (Wilson et al., 2016)	100	0.743	+0.05
IRT: users, items	0	0.691	
PFA: skills, wins, fails	0	0.685	+0.07
AFM: skills, attempts	0	0.616	

Bonus: interpreting the learned embeddings



What 'bout recurrent neural networks?

Deep Knowledge Tracing: knowledge tracing as sequence prediction

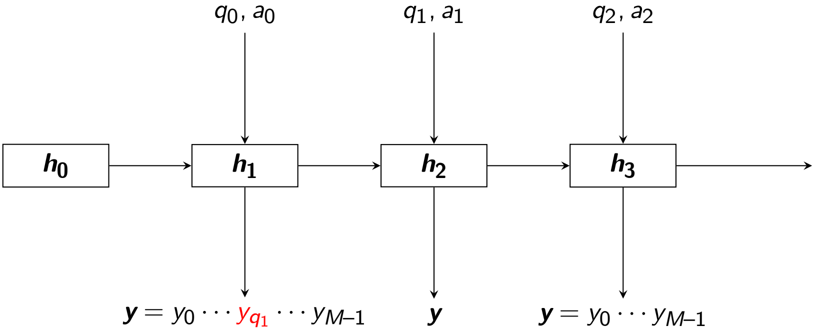
- Each student on skill q_t has performance a_t
- How to predict outcomes $\boldsymbol{y}$ on every skill k?
- Spoiler: by measuring the evolution of a latent state $\boldsymbol{h_t}$

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

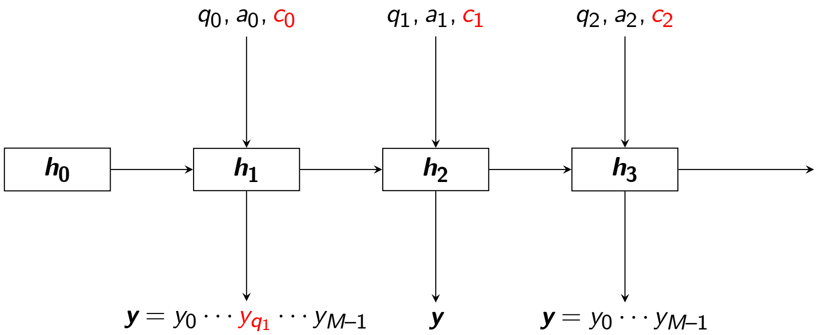
Our approach: encoder-decoder

$$\begin{cases} \boldsymbol{h_t} = Encoder(\boldsymbol{h_{t-1}}, \boldsymbol{x_t^{in}}) \\ p_t = Decoder(\boldsymbol{h_t}, \boldsymbol{x_t^{out}}) \end{cases} \quad t = 1, \ldots, T$$

Graphically: deep knowledge tracing

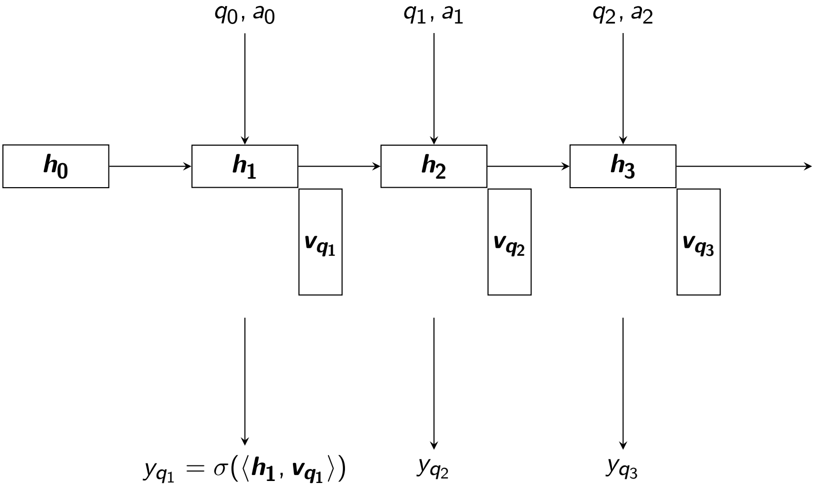


Deep knowledge tracing with dynamic student classification



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*, pp. 1182–1187. URL: https://arxiv.org/abs/1809.08713

DKT seen as encoder-decoder



Results on Fraction dataset

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

Model	Encoder	Decoder	x_t^{out}	ACC	AUC
Ours	GRU $d = 2$	bias	iswf	**0.880**	**0.944**
KTM	counter	bias	iswf	0.853	0.918
PFA	counter	bias	swf	0.854	0.917
Ours	$\varnothing$	bias	iswf	0.849	0.917
Ours	GRU $d = 50$	$\varnothing$		0.814	0.880
DKT	GRU $d = 2$	$d = 2$	s	0.772	0.844
Ours	GRU $d = 2$	$\varnothing$		0.751	0.800

Results on Berkeley dataset

562201 attempts of 1730 students over 234 CS-related items of 29 categories.

Model	Encoder	Decoder	x_t^{out}	ACC	AUC
Ours	GRU $d = 50$	bias	iswf	**0.707**	**0.778**
KTM	counter	bias	iswf	**0.704**	**0.775**
Ours	$\varnothing$	bias	iswf	0.700	0.770
DKT	GRU $d = 50$	$d = 50$	s	0.684	0.751
Ours	GRU $d = 100$	$\varnothing$		0.682	0.750
PFA	counter	bias	swf	0.630	0.683
DKT	GRU $d = 2$	$d = 2$	s	0.637	0.656

Jill-Jênn Vie and Hisashi Kashima (n.d.). "Encode & Decode: Generalizing Deep Knowledge Tracing and Multidimensional Item Response Theory". under review. URL: http://jiji.cat/bigdata/edm2019_submission.pdf

Take home message

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

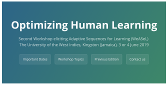
- Increase success rate of the student (Clement et al., JEDM 2015)
- Identify something that the student does not know (Teng et al., ICDM 2018, Seznec et al., AISTATS 2019)
- See more on https://humanlearn.io

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

Corbett, Albert T and John R Anderson (1994). “Knowledge tracing: Modeling the acquisition of procedural knowledge”. In: *User modeling and user-adapted interaction* 4.4, pp. 253–278.

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*, pp. 1182–1187. 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.

Rasch, Georg (1960). *Studies in mathematical psychology: I. Probabilistic models for some intelligence and attainment tests.* Nielsen & Lydiche.

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 and Hisashi Kashima (n.d.). "Encode & Decode: Generalizing Deep Knowledge Tracing and Multidimensional Item Response Theory". under review. URL: http://jiji.cat/bigdata/edm2019_submission.pdf.

– (2019). "Knowledge Tracing Machines: Factorization Machines for Knowledge Tracing". In: *33th AAAI Conference on Artificial Intelligence*. URL: http://arxiv.org/abs/1811.03388.