An Introduction to Knowledge Tracing

Geoffrey Converse

University of Iowa

February 4, 2021

Outline

High-level view

- Given a sequence of responses to an (online) assessment
- Each exercise is associated with a latent concept
- How does the student's mastery of each concept progress throughout the exam?
- Applications include feedback evaluation, intelligent tutoring systems

Notation

- N students, indexed by j
- n available items, indexed by i
- K concepts, indexed by k
- L maximum length response sequence, each timestep indexed by t
  - If a student answers $< L$ questions, then pad their sequence
  - If a student answers $> L$ questions, then split into two sequences
- Interactions are presented as tuple (q_t, c_t)
  - q_t is an integer $\leq n$ that indexes an item
  - $c_t \in \{0, 1\}$ indicates correct/incorrect
  - $2n$ possible interactions – can one-hot encode (q_t, c_t)

Data Example

- Set $L = 4$, $n = 6$
- Student a answers questions $\{1, 4, 2\}$
  - $X_a = \{(PAD, PAD), (1, 0), (4, 1), (2, 1)\}$
- Student b answers questions $\{2, 5, 3, 1, 6\}$
  - $X_{b_1} = \{(PAD, PAD), (2, 0), (5, 1), (3, 1)\}$
  - $X_{b_2} = \{(PAD, PAD), (PAD, PAD), (1, 0), (6, 1)\}$
- PAD inputs are ignored
- One-hot encode each interaction in vector of length 12
  - $(1, 0) \rightarrow v_{10} = [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$
  - $(4, 1) \rightarrow v_{41} = [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]$
- Multiply with embedding matrix $A \in \mathbb{R}^{h \times 2n}: \quad \rightarrow x_{10} = Av_{10}$
  - Student a's input to neural network: $[PAD, x_{10}, x_{41}, x_{21}]$

Goal

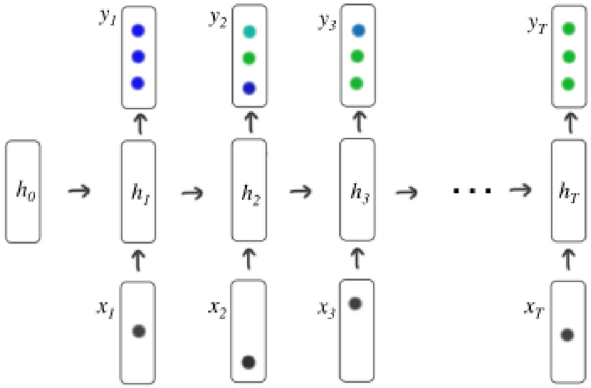
- Given a student's responses $\{(q_1, c_1), \ldots, (q_t, c_t), (q_{t+1}, ?)\}$, infer c_{t+1}
- Try to approximate

$$P(c_{t+1} = 1 | q_1, c_1, \ldots, q_t, c_t, q_{t+1})$$

- Mask future interactions while training

Recurrent Neural Networks

- Input vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_L$
- Outputs $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_L$
- Hidden states $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_L$

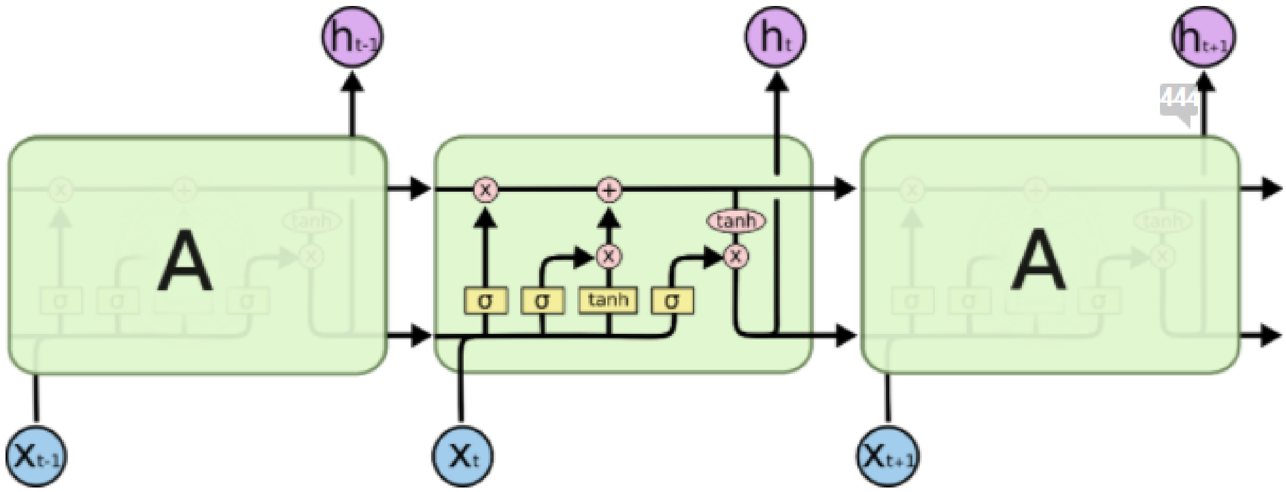


$$\boldsymbol{h}_t = \tanh(W^{hx}\boldsymbol{x}_t + W^{hh}h_{t-1} + b^h)$$

$$\boldsymbol{y}_t = \sigma(W^{yh}\boldsymbol{h}_t + b^y)$$

In knowledge tracing:
True y_1 is given in x_2

Long Short-Term Memory networks



LSTM

- Forget content: $\boldsymbol{f}_t = \sigma(W_1[\boldsymbol{x}_t, \boldsymbol{h}_{t-1}] + b_1)$
- Input content: $\boldsymbol{w}_t = \sigma(W_2[\boldsymbol{x}_t, \boldsymbol{h}_{t-1}] + b_2)$ and $\boldsymbol{a}_t = \tanh(W_3[\boldsymbol{x}_t, \boldsymbol{h}_{t-1}] + b_3)$
- Update cell state: $\boldsymbol{C}_t = (\boldsymbol{f}_t \times \boldsymbol{C}_{t-1}) + (\boldsymbol{w}_t \times \boldsymbol{a}_t)$ elementwise
- Output gate $\boldsymbol{h}_t = \sigma(W_4[\boldsymbol{x}_t, \boldsymbol{h}_{t-1}] + b_4) \times \tanh(W_5\,\boldsymbol{C}_t + b_5)$

Deep Knowledge Tracing

- Really just applied RNN / LSTM to knowledge tracing application
- Input x_t is the embedding of (q_t, c_t)
- Output y_t is the predicted probability that $c_{t+1} = 1$

Dynamic Key-Value Memory Networks

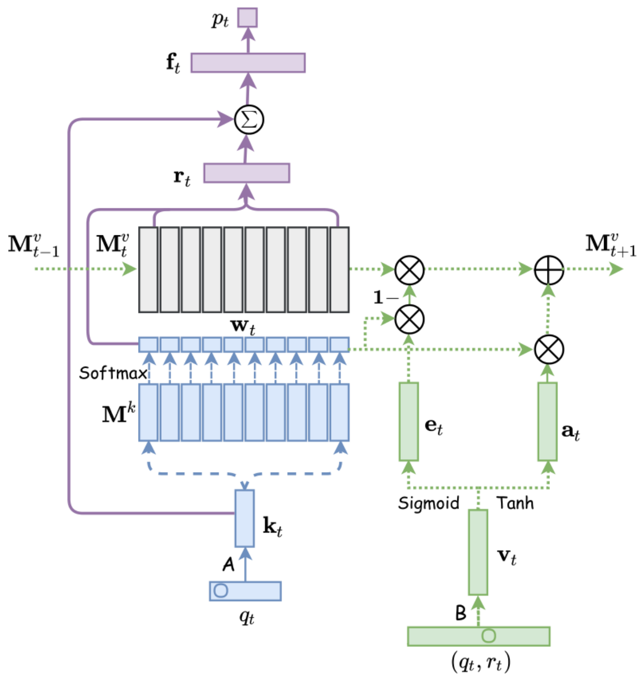


Figure 1: Architecture of DKVMN

DKVMN

- K concepts (memory slots)
- hidden dimension h
- Stored memory matrix at time t: M_t^v
- Embed question (no response) with $k_t = A \cdot q_t$
- Embed question + response with $v_t = B \cdot (q_t, c_t)$
- Trainable parameters:
  $A, B, M^k, W_1, b_1, W_2, b_2, W_e, b_e, W_a, b_a$

DKVMN

READ:

- correlation weight $w_t(i) = \mathrm{Softmax}(k_t^\top M^k(k))$
- read content $r_t = \sum_{i=1}^{K} w_t(i) M^v(i)$
- feed forward: $p_t = \sigma(\, W_2 \cdot (\, W_1[r_t, k_t] + b_1) + b_2)$

WRITE:

- Erase vector: $e_t = \sigma(\, W_e v_t + b_e)$
- Add vector: $a_t = \tanh(\, W_a v_t + b_a)$
- Update memory:

$$M_t^v(i) = \left(M_{t-1}^v(i) \cdot [1 - w_t(i) e_t]\right) + w_t(i) a_t$$

Self-Attentive Knowledge Tracing

Main mechanism: calculating "attention"

- For each interaction embedding x_t, calculate query, key, and value vectors:

$$q_t = W^Q x_t, \quad k_t = W^K x_t, \quad v_t = W^V x_t, \quad \text{matrices} \in \mathbb{R}^{h \times h}$$

- Calculate the correlation weight between interaction t and all previous exercises:

$$w_{ti} = \text{Softmax}\left(\frac{q_t^\top k_i}{\sqrt{h}}\right), \quad i \leq t$$

- Multiply w_{ti} by correponding value vectors:

$$A_{ti} = w_{ti} v_i$$

$$h_t = \sum_{i \leq t} A_{ti}$$

SAKT

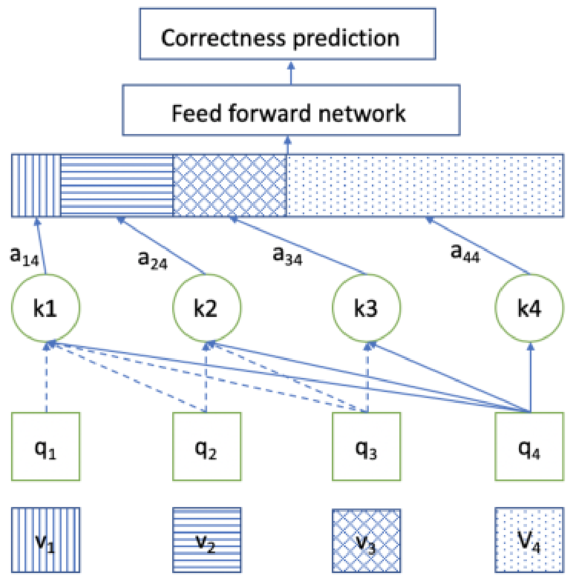
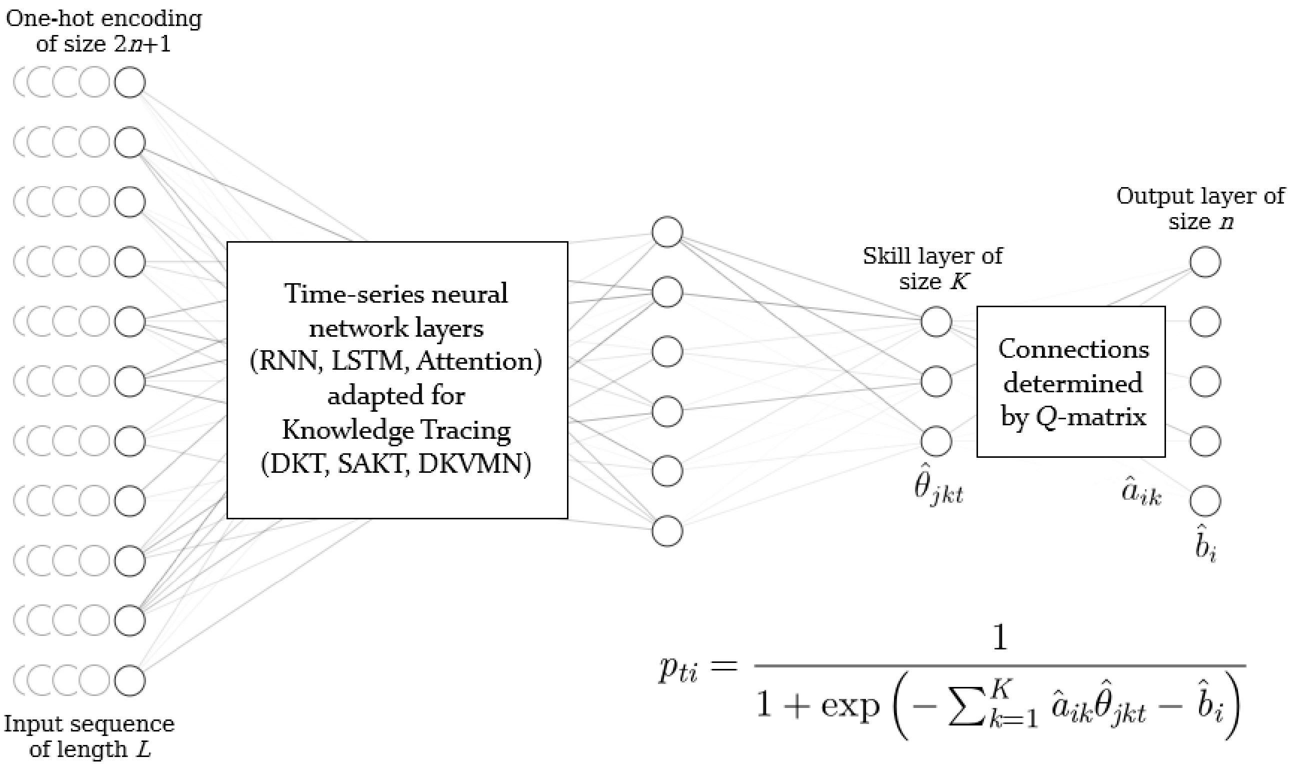


Figure 2: SAKT architecture

h_t is sent through feed forward network to make prediction about next interaction

IRT-inspired knowledge tracing



$$p_{ti} = \frac{1}{1 + \exp\left(-\sum_{k=1}^{K} \hat{a}_{ik}\hat{\theta}_{jkt} - \hat{b}_i\right)}$$

Figure 3: Proposed model incorporating IRT with KT

References

[1] Corbett and Anderson. "Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge." User Modeling and User-Adapted Interaction, 1995. Volume 4, pages 253-278.

[2] Piech, Bassen, Huang, Ganguli, Shami, Guibas, Shol-Dickstein. "Deep Knowledge Tracing." Advances in Neural Information Processing Systems, 2015.

[3] Zhang, Shi, King, Yeung. "Dynamic Key-Value Memory Networks for Knowledge Tracing." International World Wide Web Conference, 2017. Pages 765-774.

[4] Pandey and Karypis. "A Self-Attentive model for Knowledge Tracing."