Interpretable Neural Networks for Item Response Theory Parameter Estimation

Geoffrey Converse

University of Iowa

October 1, 2019

AMCS Comprehensive Examination

Outline

Item Response Theory (IRT)

- Goal: Explain relationship between student ability and exam performance
- Assume each subject has a latent "ability" value θ

Item Response Theory (IRT)

- Goal: Explain relationship between student ability and exam performance
- Assume each subject has a latent "ability" value θ
  - θ is not directly observable
  - Often assume $\theta \sim \mathcal{N}(0, 1)$
  - Naive solution: accuracy (percent correct)

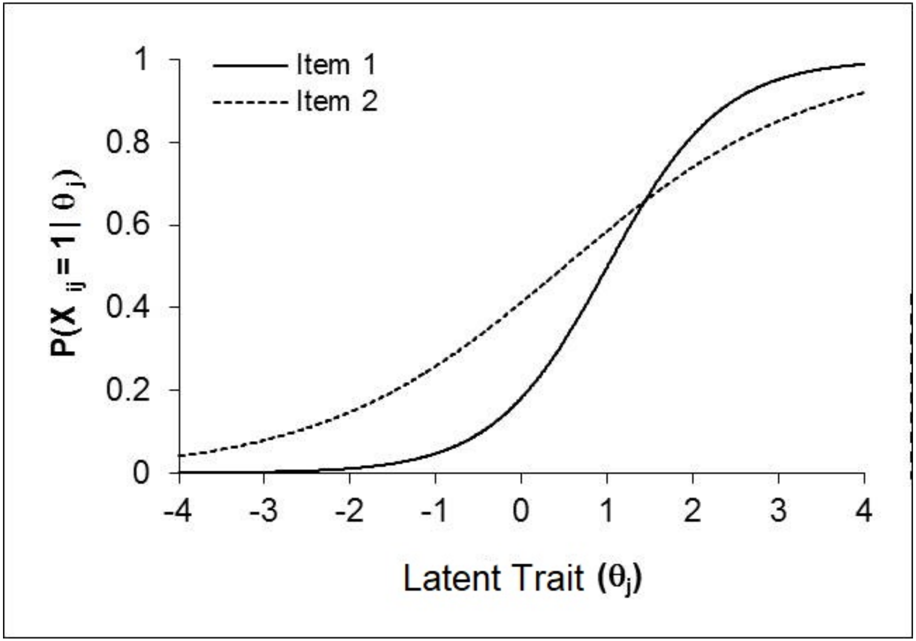
Item Response Theory (IRT)

- Goal: Explain relationship between student ability and exam performance
- Assume each subject has a latent "ability" value θ
  - θ is not directly observable
  - Often assume $\theta \sim \mathcal{N}(0, 1)$
  - Naive solution: accuracy (percent correct)
- For an assessment with n items taken by N subjects, what is the probability that student j answers item i correctly?

$$P(u_{ij} = 1 | \theta_j) = f(\theta_j; V_i)$$

  - θ_j = latent ability of subject j
  - V_i = set of parameters associated with item i

Item Characteristic Curve (ICC)



Normal Ogive Model

- Probability of a correct response follows the cumulative distribution function of a Gaussian distribution:

$$P(u_{ij} = 1|\theta_j) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{a_i(\theta_j - b_i)} e^{-z^2/2} \, dz$$

Normal Ogive Model

- Probability of a correct response follows the cumulative distribution function of a Gaussian distribution:

$$P(u_{ij} = 1|\theta_j) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{a_i(\theta_j - b_i)} e^{-z^2/2} dz$$

  - $a_i = 1/\sigma$ the discrimination parameter, with σ the standard deviation of a Gaussian distribution
  - $b_i = \mu$ the difficulty parameter, with μ the mean of a Gaussian distribution
  - If $\theta_j = b_i$, then subject j has 50% chance of correct response

2-Parameter Logistic Model (2PL)

- Probability of a correct response follows the logistic equation:
$$P(u_{ij} = 1|\theta_j) = \frac{1}{1 + e^{-a_i(\theta_j - b_i)}}$$

2-Parameter Logistic Model (2PL)

- Probability of a correct response follows the logistic equation:

$$P(u_{ij} = 1|\theta_j) = \frac{1}{1 + e^{-a_i(\theta_j - b_i)}}$$

  - Scaling a_i by a factor of 1.7 makes 2PL differ from the normal ogive by < 0.01 uniformly
  - Easier to compute than normal ogive
  - a_i = discrimination parameter
  - b_i = difficulty parameter

Assessing Multiple Skills

- Now assume that an assessment is testing K skills
  - For example, a math exam can test skills add, subtract, multiply, divide
  - Each student has a vector of skills $\Theta_j = (\theta_{j1}, .., \theta_{jK})^T$
  - Multiple skills can be assessed by a single item

Assessing Multiple Skills

- Now assume that an assessment is testing K skills
    - For example, a math exam can test skills add, subtract, multiply, divide
    - Each student has a vector of skills $\Theta_j = (\theta_{j1}, .., \theta_{jK})^T$
    - Multiple skills can be assessed by a single item
- Binary Q-matrix defines relationship between items and skills
    - $Q \in \mathbb{R}^{n \times K}$,

$$q_{ik} = \begin{cases} 1 & \text{if item } i \text{ requires skill } k \\ 0 & \text{otherwise} \end{cases}$$

Multidimensional Logistic 2-Parameter (ML2P) Model

- Probability of correct response given by:

$$P(u_{ij} = 1|\Theta_j) = \frac{1}{1 + \exp[-\sum_{k=1}^{K} a_{ik}\theta_{jk} + b_i]}$$

Multidimensional Logistic 2-Parameter (ML2P) Model

- Probability of correct response given by:

$$P(u_{ij} = 1 | \Theta_j) = \frac{1}{1 + \exp[-\sum_{k=1}^{K} a_{ik}\theta_{jk} + b_i]}$$

  - a_{ik} = discrimination parameter between item i and skill k
  - b_i = difficulty parameter

Item Parameter Estimation

- Assume student abilities θ are known
- Group students into k groups with f_j students, $j = 1, ..., k$

Item Parameter Estimation

- Assume student abilities θ are known
- Group students into k groups with f_j students, $j = 1, \ldots, k$
- $P_{ij} = P(\theta_j) = P(\theta_j, a_i, b_i)$
  - Probability that a student in group j answers item i correctly

Item Parameter Estimation

- Assume student abilities θ are known
- Group students into k groups with f_j students, $j = 1, ..., k$
- $P_{ij} = P(\theta_j) = P(\theta_j, a_i, b_i)$
  - Probability that a student in group j answers item i correctly
- Let f_j = number of students in group θ_j
- Let r_{ij} = number of students in group θ_j who answer item i correctly

Maximum Likelihood Estimation (MLE)

- $R_i = (r_{i1}, \ldots, r_{ik})$ = vector of observed responses to item i over all students
- Likelihood function of R_i:

$$P(R_i) = \prod_{j=1}^{k} P(r_{ij}) = \prod_{j=1}^{k} \binom{f_j}{r_{ij}} P_{ij}^{r_{ij}} (1 - P_{ij})^{f_j - r_{ij}}$$

Maximum Likelihood Estimation (MLE)

- $R_i = (r_{i1}, ..., r_{ik})$ = vector of observed responses to item i over all students
- Likelihood function of R_i:

$$P(R_i) = \prod_{j=1}^{k} P(r_{ij}) = \prod_{j=1}^{k} \binom{f_j}{r_{ij}} P_{ij}^{r_{ij}} (1 - P_{ij})^{f_j - r_{ij}}$$

- Maximize:

$$L = \log P(R_i) = K + \sum_{j=1}^{k} r_{ij} \log P_{ij} + (f_j - r_{ij}) \log(1 - P_{ij})$$

$$\frac{\partial L}{\partial a_i} = \sum_{j=1}^{k} (\theta_j - b_i)(r_{ij} - f_j P_{ij}) \qquad \frac{\partial L}{\partial b_i} = -a_i \sum_{j=1}^{k} (r_{ij} - f_j P_{ij})$$

Ability Parameter Estimation

- Assume:
  - Values of all item parameters are known
  - Examinees are independent $\Rightarrow$ estimate each student individually
  - All items are modeled by an ICC of the same family

Ability Parameter Estimation

- Assume:
  - Values of all item parameters are known
  - Examinees are independent ⇒ estimate each student individually
  - All items are modeled by an ICC of the same family
- $U_j = (u_{1j}, ..., u_{nj}|\theta_j)$ = binary vector of student j's responses

$$P(U_j|\theta_j) = \prod_{i=1}^{n} P_{ij}^{u_{ij}}(1 - P_{ij})^{1-u_{ij}}$$

Ability Parameter Estimation

- Assume:
  - Values of all item parameters are known
  - Examinees are independent $\Rightarrow$ estimate each student individually
  - All items are modeled by an ICC of the same family
- $U_j = (u_{1j}, \ldots, u_{nj}|\theta_j)$ = binary vector of student j's responses

$$P(U_j|\theta_j) = \prod_{i=1}^{n} P_{ij}^{u_{ij}} (1 - P_{ij})^{1-u_{ij}}$$

- Maximize log-likelihood $L = \log P(U_j|\theta_j)$:

$$\frac{\partial L}{\partial \theta_j} = a_i \sum_{i=1}^{n} (u_{ij} - P_{ij})$$

Joint Maximum Likelihood Estimation (JMLE)

- Many applications don't have ability or item parameters available
- Need to estimate both simultaneously

Joint Maximum Likelihood Estimation (JMLE)

- Many applications don't have ability or item parameters available
- Need to estimate both simultaneously
- N students and n items, build $n \times N$ response matrix $[u_{ij}]$ and ability vector $\Theta = (\theta_1, ..., \theta_N)^T$
- Probability of item responses:

$$P(U|\Theta) = \prod_{j=1}^{N} \prod_{i=1}^{n} P_{ij}^{u_{ij}} (1 - P_{ij})^{1-u_{ij}}$$

Joint Maximum Likelihood Estimation (JMLE)

- Many applications don't have ability or item parameters available
- Need to estimate both simultaneously
- N students and n items, build $n \times N$ response matrix $[u_{ij}]$ and ability vector $\Theta = (\theta_1, ..., \theta_N)^T$
- Probability of item responses:

$$P(U|\Theta) = \prod_{j=1}^{N} \prod_{i=1}^{n} P_{ij}^{u_{ij}} (1 - P_{ij})^{1-u_{ij}}$$

- Maximize log-likelihood:

$$L = \log P(U|\Theta) = \sum_{j=1}^{N} \sum_{i=1}^{n} u_{ij} \log P_{ij} + (1 - u_{ij}) \log(1 - P_{ij})$$

Joint Maximum Likelihood Estimation (JMLE)

- Maximizing log-likelihood gives $2n + N$ equations.
- Using Newton's Method: $A_{t+1} = A_t - B_t^{-1} F_t$
  - $A = (\hat{a}_1, \hat{b}_1, ..., \hat{a}_n, \hat{b}_n, \hat{\theta}_1, ..., \hat{\theta}_N)^T$ vector of estimates
  - $B = (2n + N) \times (2n + N)$ matrix of 2nd order partials
  - $F =$ vector of 1st order partials

Joint Maximum Likelihood Estimation (JMLE)

- Maximizing log-likelihood gives $2n + N$ equations.
- Using Newton's Method: $A_{t+1} = A_t - B_t^{-1} F_t$
    - $A = (\hat{a}_1, \hat{b}_1, ..., \hat{a}_n, \hat{b}_n, \hat{\theta}_1, ..., \hat{\theta}_N)^T$ vector of estimates
    - $B = (2n + N) \times (2n + N)$ matrix of 2nd order partials
    - $F =$ vector of 1st order partials
- Assumptions:
    - Each examinee is independent
    - Items are independent
    - Examinees and items are independent

Joint Maximum Likelihood Estimation (JMLE)

- Maximizing log-likelihood gives $2n + N$ equations.
- Using Newton's Method: $A_{t+1} = A_t - B_t^{-1} F_t$
  - $A = (\hat{a}_1, \hat{b}_1, ..., \hat{a}_n, \hat{b}_n, \hat{\theta}_1, ..., \hat{\theta}_N)^T$ vector of estimates
  - $B = (2n + N) \times (2n + N)$ matrix of 2nd order partials
  - $F =$ vector of 1st order partials
- Assumptions:
  - Each examinee is independent
  - Items are independent
  - Examinees and items are independent
- Simplification: Assume most cross-derivatives are zero

Joint Maximum Likelihood Estimation (JMLE)

$$B = \begin{bmatrix} \frac{\partial^2 L}{\partial a_1^2} & \frac{\partial^2 L}{\partial a_1 b_1} & & & & & & \\ \frac{\partial^2 L}{\partial a_1 b_1} & \frac{\partial^2 L}{\partial b_1^2} & & & & & & \\ & & \ddots & & & & & \\ & & & \frac{\partial^2 L}{\partial a_n^2} & \frac{\partial^2 L}{\partial a_n \zeta_n} & & & \\ & & & \frac{\partial^2 L}{\partial a_n b_n} & \frac{\partial^2 L}{\partial b_n^2} & & & \\ & & & & & \ddots & & \\ & & & & & & \frac{\partial L^2}{\partial \theta_1^2} & & \\ & & & & & & & \ddots & \\ & & & & & & & & \frac{\partial L^2}{\partial \theta_N^2} \end{bmatrix}$$

Joint Maximum Likelihood Estimation (JMLE)

- Need good initial ability estimates
- Possibly unbounded a_i and θ_j
- Solution can diverge
    - Large discrimination parameter estimates can lead to large ability estimates

Marginal Maximum Likelihood (MMLE)

- Assume that θ follows some distribution $g(\theta)$
- Maximize the mariginal likelihood for each student

$$P(U_j) = \int P(U_j|\theta)\,g(\theta)\,d\theta$$

Marginal Maximum Likelihood (MMLE)

- Assume that θ follows some distribution $g(\theta)$
- Maximize the mariginal likelihood for each student

$$P(U_j) = \int P(U_j|\theta)\,g(\theta)\,d\theta$$

- Marginal likelihood function

$$L = \prod_{j=1}^{N} P(U_j) = \prod_{j=1}^{N} \int P(U_j|\theta)\,g(\theta)\,d\theta$$

Marginal Maximum Likelihood (MMLE)

- Assume that θ follows some distribution $g(\theta)$
- Maximize the mariginal likelihood for each student

$$P(U_j) = \int P(U_j|\theta)\, g(\theta)\, d\theta$$

- Marginal likelihood function

$$L = \prod_{j=1}^{N} P(U_j) = \prod_{j=1}^{N} \int P(U_j|\theta)\, g(\theta)\, d\theta$$

- Posterior probability

$$P(\theta_j|U_j) = \frac{P(U_j|\theta_j)\, g(\theta_j)}{P(U_j)} = \frac{P(U_j|\theta_j)\, g(\theta_j)}{\int P(U_j|\theta)\, g(\theta)\, d\theta}$$

Marginal Maximum Likelihood (MMLE)

$$\frac{\partial \log L}{\partial x_i} = \sum_{j=1}^{N} \frac{1}{P(U_j)} \int \frac{\partial}{\partial x_i} [P(U_j|\theta)] g(\theta) \, d\theta$$

Marginal Maximum Likelihood (MMLE)

$$
\begin{aligned}
\frac{\partial \log L}{\partial x_i} &= \sum_{j=1}^{N} \frac{1}{P(U_j)} \int \frac{\partial}{\partial x_i} [P(U_j|\theta)] g(\theta)\, d\theta \\
&= \sum_{j=1}^{N} \frac{1}{P(U_j)} \int \frac{\partial}{\partial x_i} [\log P(U_j|\theta)] P(U_j|\theta) g(\theta)\, d\theta
\end{aligned}
$$

Marginal Maximum Likelihood (MMLE)

$$
\begin{aligned}
\frac{\partial \log L}{\partial x_i} &= \sum_{j=1}^{N} \frac{1}{P(U_j)} \int \frac{\partial}{\partial x_i} [P(U_j|\theta)] g(\theta)\, d\theta \\
&= \sum_{j=1}^{N} \frac{1}{P(U_j)} \int \frac{\partial}{\partial x_i} [\log P(U_j|\theta)] P(U_j|\theta) g(\theta)\, d\theta \\
&= \sum_{j=1}^{N} \int \frac{\partial}{\partial x_i} [\log P(U_j|\theta)] P(\theta|U_j)\, d\theta
\end{aligned}
$$

Marginal Maximum Likelihood (MMLE)

$$
\begin{aligned}
\frac{\partial \log L}{\partial x_i} &= \sum_{j=1}^{N} \frac{1}{P(U_j)} \int \frac{\partial}{\partial x_i} [P(U_j|\theta)] g(\theta)\, d\theta \\
&= \sum_{j=1}^{N} \frac{1}{P(U_j)} \int \frac{\partial}{\partial x_i} [\log P(U_j|\theta)] P(U_j|\theta) g(\theta)\, d\theta \\
&= \sum_{j=1}^{N} \int \frac{\partial}{\partial x_i} [\log P(U_j|\theta)] P(\theta|U_j)\, d\theta \\
\frac{\partial \log L}{\partial a_i} &= \sum_{j=1}^{N} \int (\theta - b_i)(u_{ij} - P_{ij}) P(\theta|U_j)\, d\theta \\
\frac{\partial \log L}{\partial b_i} &= -a_i \sum_{j=1}^{N} \int (u_{ij} - P_{ij}) P(\theta|U_j)\, d\theta
\end{aligned}
$$

Quadrature with Nodes at Ability Levels X_k

- Approximate integral by some quadrature rule with q nodes for ability levels X_k, $1 \leq k \leq q$

Quadrature with Nodes at Ability Levels X_k

- Approximate integral by some quadrature rule with q nodes for ability levels X_k, $1 \leq k \leq q$

$$\frac{\partial \log L}{\partial a_i} \approx \sum_{j=1}^{N} \sum_{k=1}^{q} (X_k - b_i)(u_{ij} - P_{ik}) P(X_k | U_j)$$

$$\frac{\partial \log L}{\partial b_i} \approx -a_i \sum_{j=1}^{N} \sum_{k=1}^{q} (u_{ij} - P_{ik}) P(X_k | U_j)$$

Quadrature with Nodes at Ability Levels X_k

- Approximate integral by some quadrature rule with q nodes for ability levels X_k, $1 \leq k \leq q$

$$\frac{\partial \log L}{\partial a_i} \approx \sum_{j=1}^{N} \sum_{k=1}^{q} (X_k - b_i)(u_{ij} - P_{ik}) P(X_k | U_j)$$

$$\approx \sum_{k=1}^{q} (X_k - b_i)(\bar{r}_{ik} - P_{ik} \bar{f}_k)$$

$$\frac{\partial \log L}{\partial b_i} \approx -a_i \sum_{j=1}^{N} \sum_{k=1}^{q} (u_{ij} - P_{ik}) P(X_k | U_j)$$

$$\approx -a_i \sum_{k=1}^{q} (\bar{r}_{ik} - P_{ik} \bar{f}_k)$$

- $\bar{f}_k$ = number of expected examinees at ability level k
- $\bar{r}_{ik}$ = number of expected correct responses to item i at ability level k

MMLE via Expectation-Maximization (EM) Algorithm

For each item:

- E-step:

MMLE via Expectation-Maximization (EM) Algorithm

For each item:

- E-step:
  - Use quadrature to estimate posterior probability $P(\theta_j | U_j) \approx P(X_k | U_j)$ for each student

MMLE via Expectation-Maximization (EM) Algorithm

For each item:

- E-step:
  - Use quadrature to estimate posterior probability $P(\theta_j | U_j) \approx P(X_k | U_j)$ for each student
  - Find *expected* number of examinees at each level $\bar{f}_k = \sum_{j=1}^{N} P(X_k | U_j)$
  - Find *expected* number of correct responses $\bar{r}_{ik} = \sum_{j=1}^{N} u_{ij} P(X_k | U_j)$

MMLE via Expectation-Maximization (EM) Algorithm

For each item:

- E-step:
  - Use quadrature to estimate posterior probability $P(\theta_j | U_j) \approx P(X_k | U_j)$ for each student
  - Find *expected* number of examinees at each level $\bar{f}_k = \sum_{j=1}^{N} P(X_k | U_j)$
  - Find *expected* number of correct responses $\bar{r}_{ik} = \sum_{j=1}^{N} u_{ij} P(X_k | U_j)$
- M-step:

MMLE via Expectation-Maximization (EM) Algorithm

For each item:

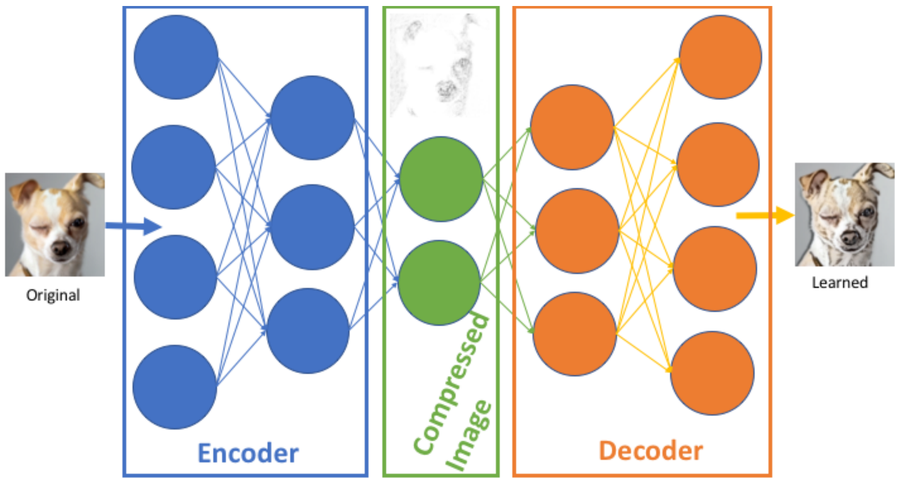
- E-step:
  - Use quadrature to estimate posterior probability $P(\theta_j | U_j) \approx P(X_k | U_j)$ for each student
  - Find *expected* number of examinees at each level $\bar{f}_k = \sum_{j=1}^{N} P(X_k | U_j)$
  - Find *expected* number of correct responses $\bar{r}_{ik} = \sum_{j=1}^{N} u_{ij} P(X_k | U_j)$
- M-step:
  - Find item parameters a_i and b_i which maximize marginal log likelihood: solve

$$\frac{\partial \log L}{\partial a_i} = 0$$

$$\frac{\partial \log L}{\partial b_i} = 0$$

Autoencoder (AE)

- Encode data into smaller dimension
- Reconstruct original input



Variational Autoencoder (VAE)

- Commonly used as a generative neural network
- Learn a low-dimensional latent representation θ which is capable of generating original data x from some distribution

$$f(\boldsymbol{\theta}|\boldsymbol{x}) = \frac{P(\boldsymbol{X} = \boldsymbol{x}|\theta)f(\boldsymbol{\theta})}{P(\boldsymbol{X} = \boldsymbol{x})}$$

$$P(\boldsymbol{X} = \boldsymbol{x}) = \int P(\boldsymbol{X} = \boldsymbol{x}|\boldsymbol{\theta})f(\boldsymbol{\theta})\,d\boldsymbol{\theta},$$

Variational Autoencoder (VAE)

- Commonly used as a generative neural network
- Learn a low-dimensional latent representation θ which is capable of generating original data x from some distribution

$$f(\boldsymbol{\theta}|\boldsymbol{x}) = \frac{P(\boldsymbol{X} = \boldsymbol{x}|\theta)f(\boldsymbol{\theta})}{P(\boldsymbol{X} = \boldsymbol{x})}$$

$$P(\boldsymbol{X} = \boldsymbol{x}) = \int P(\boldsymbol{X} = \boldsymbol{x}|\boldsymbol{\theta})f(\boldsymbol{\theta})\,d\boldsymbol{\theta},$$

- Approximate $f(\boldsymbol{\theta}|\boldsymbol{x})$ with some $q(\boldsymbol{\theta}|\boldsymbol{x}) \Rightarrow$ minimize KL Divergence

Kullback-Leibler Divergence

- *Entropy* measures average information gained from 1 sample:

$$H(P) = \mathbb{E}_P[-\log P(x)] = -\sum_i P(x_i) \log P(x_i)$$

Kullback-Leibler Divergence

- *Entropy* measures average information gained from 1 sample:

$$H(P) = \mathbb{E}_P[-\log P(x)] = -\sum_i P(x_i) \log P(x_i)$$

- *Cross entropy* measures average information needed when using approximate distribution $Q(x)$ instead of true distribution $P(x)$:

$$H(P, Q) = \mathbb{E}_P[-\log Q(x)] = -\sum_i P(x_i) \log Q(x_i)$$

Kullback-Leibler Divergence

- *Entropy* measures average information gained from 1 sample:

$$H(P) = \mathbb{E}_P[-\log P(x)] = -\sum_i P(x_i) \log P(x_i)$$

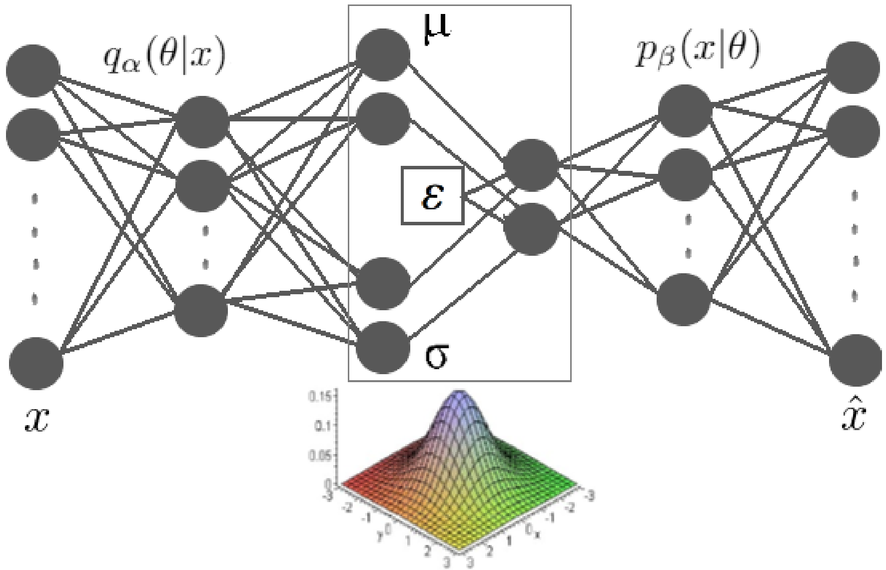
- *Cross entropy* measures average information needed when using approximate distribution $Q(x)$ instead of true distribution $P(x)$:

$$H(P, Q) = \mathbb{E}_P[-\log Q(x)] = -\sum_i P(x_i) \log Q(x_i)$$

- *Kullback-Leibler Divergence* measures difference between two probability distributions $P(x)$ and $Q(x)$:

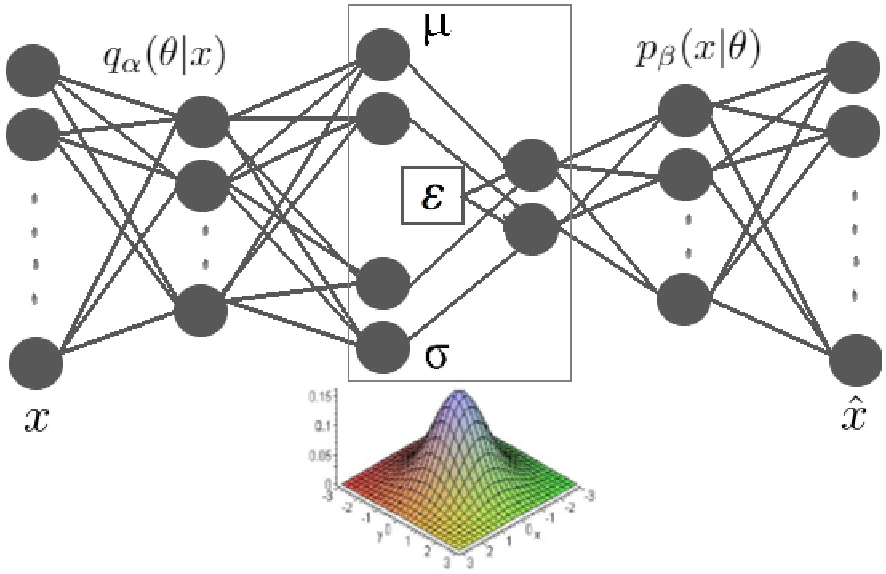
$$\mathcal{D}_{KL}\left[P(x) || Q(x)\right] = H(P, Q) - H(P) = \sum_i P(x_i) \log \left(\frac{P(x_i)}{Q(x_i)} \right)$$

Variational Autoencoder (VAE)



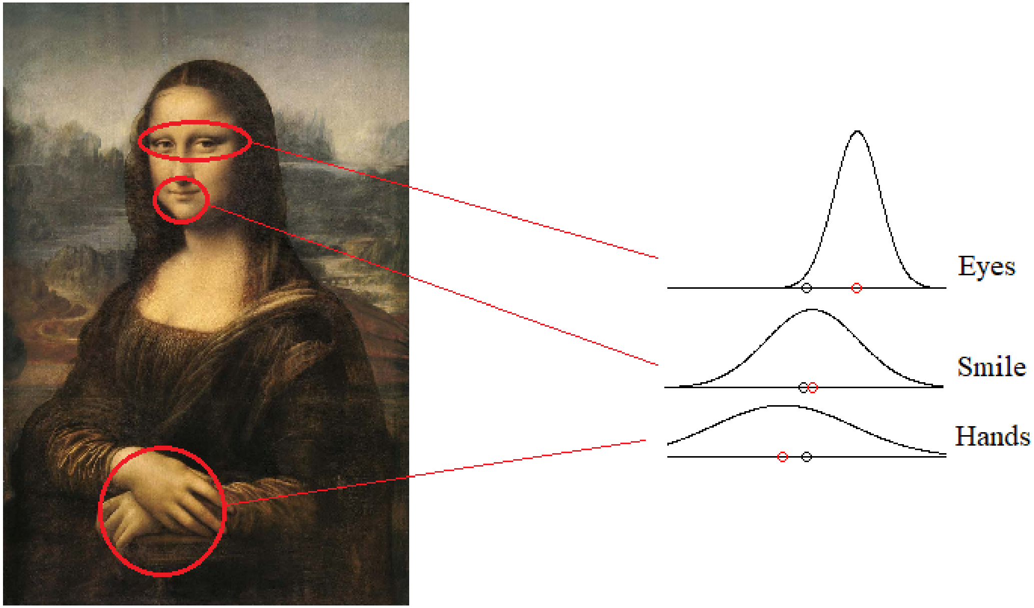
- Fit encoded space to a normal distribution
  - Loss funcion: $L(x) = L_0(x) + KL[q_\alpha(\theta|x)||\mathcal{N}(0, I)]$

Variational Autoencoder (VAE)



- Fit encoded space to a normal distribution
  - Loss funcion: $L(x) = L_0(x) + KL[q_\alpha(\theta|x)||\mathcal{N}(0, I)]$
- Sample $\varepsilon \sim \mathcal{N}(0, 1)$, set $z = \mu + \sigma\varepsilon$

Variational Autoencoder (VAE)



Combining IRT and ANN

- Key similarities:
  - IRT and VAE assume normally distributed latent space

Combining IRT and ANN

- Key similarities:
  - IRT and VAE assume normally distributed latent space
  - ML2P model and sigmoidal activation function:

$$P(u_{ij} = 1|\Theta_j) = \frac{1}{1 + \exp[-\sum_{k=1}^{K} a_{ik}\theta_{jk} + b_i]}$$

$$\sigma(z) = \sigma(\vec{w}^T\vec{a} + b) = \frac{1}{1 + \exp[-\sum_{k=1} w_k a_k - b]}$$

ML2P-VAE Model Description

- No hidden layers in the decoder

ML2P-VAE Model Description

- No hidden layers in the decoder
- Restrict nonzero weights in the decoder according to Q-matrix

ML2P-VAE Model Description

- No hidden layers in the decoder
- Restrict nonzero weights in the decoder according to Q-matrix
- Sigmoidal activation function in output layer

ML2P-VAE Model Description

- No hidden layers in the decoder
- Restrict nonzero weights in the decoder according to Q-matrix
- Sigmoidal activation function in output layer
- Require decoder weights to be nonnegative

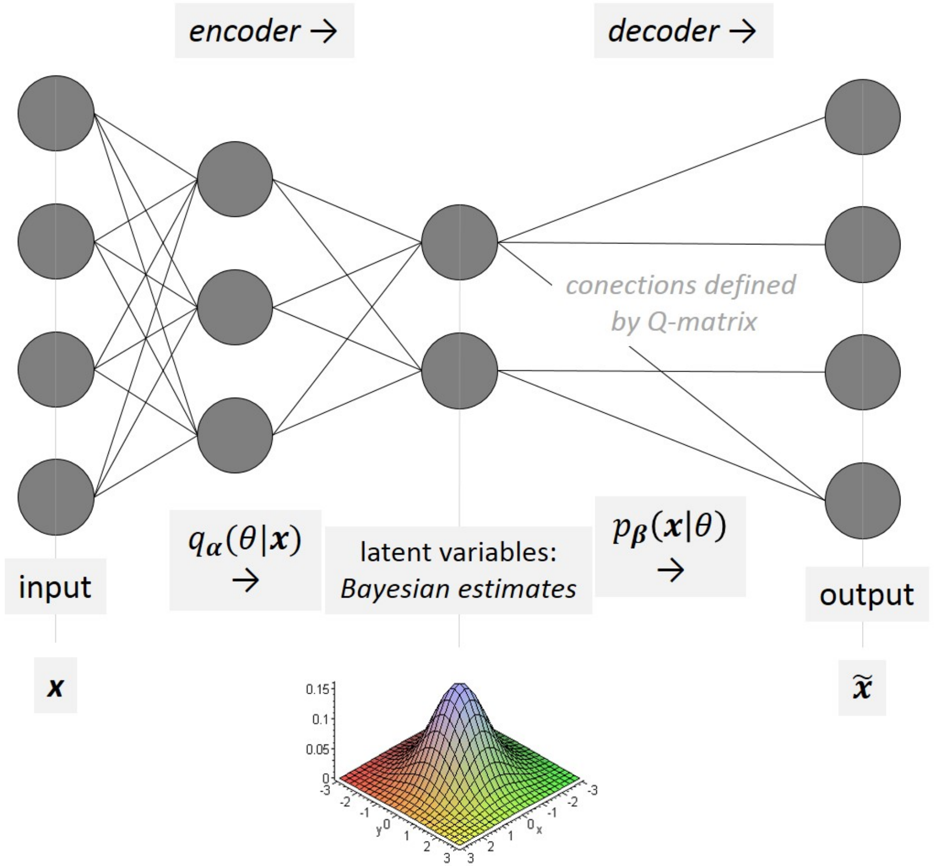
ML2P-VAE Model Description

- No hidden layers in the decoder
- Restrict nonzero weights in the decoder according to Q-matrix
- Sigmoidal activation function in output layer
- Require decoder weights to be nonnegative
- Fit VAE latent space to $\mathcal{N}(0, I)$

ML2P-VAE Model Description

- No hidden layers in the decoder
- Restrict nonzero weights in the decoder according to Q-matrix
- Sigmoidal activation function in output layer
- Require decoder weights to be nonnegative
- Fit VAE latent space to $\mathcal{N}(0, I)$
- Decoder interpreted as the ML2P model
  - Activation of nodes in learned distribution $\Rightarrow$ latent skills
  - Weights in decoder $\Rightarrow$ discrimination parameters
  - Bias of output nodes $\Rightarrow$ difficulty parameters

ML2P-VAE


encoder →
decoder →
conections defined
by Q-matrix
input
x
$q_\alpha(\theta|x)$ →
latent variables:
Bayesian estimates
$p_\beta(x|\theta)$ →
output
$\tilde{x}$
0.15
0.1
0.05
0

ML2P-VAE Testing

- Testing
  - Simulated 28 item assessment with 3 latent skills and pre-determined Q-matrix
    - Discrimination and difficulty parameters randomly chosen
  - N subjects with latent skills drawn from $N(0, I)$

ML2P-VAE Testing

- Testing
  - Simulated 28 item assessment with 3 latent skills and pre-determined Q-matrix
    - Discrimination and difficulty parameters randomly chosen
  - N subjects with latent skills drawn from $N(0, I)$
  - For each student j:
    - For each item, calculate probability of success P_{ij} from ML2P model
    - Sample from these probabilities to generate response set $U_j = (u_{j,1}, \ldots u_{j,28})$

ML2P-VAE Testing

- Testing
  - Simulated 28 item assessment with 3 latent skills and pre-determined Q-matrix
    - Discrimination and difficulty parameters randomly chosen
  - N subjects with latent skills drawn from $N(0, I)$
  - For each student j:
    - For each item, calculate probability of success P_{ij} from ML2P model
    - Sample from these probabilities to generate response set $U_j = (u_{j,1}, \ldots u_{j,28})$
- Results presented at International Joint Conference on Neural Networks (IJCNN) 2019

ML2P-VAE Results

Relative Error

Size	a_1	a_2	a_3	b
500	0.779	0.699	0.759	1.188
5,000	0.539	0.281	0.585	1.673
10,000	0.284	0.159	0.264	1.894

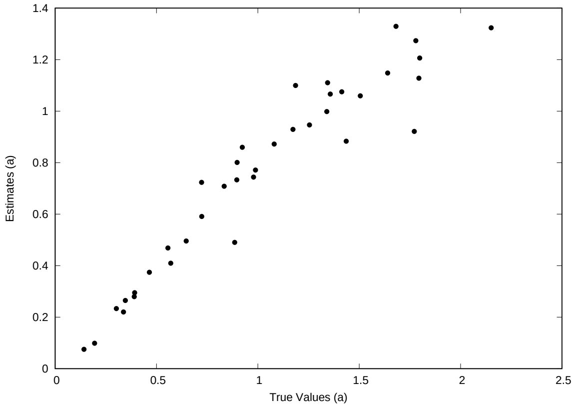
Root Mean Square Error

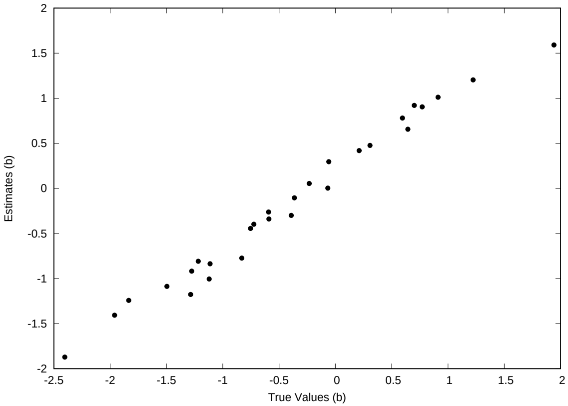
Size	a_1	a_2	a_3	b
500	0.976	0.931	0.850	1.038
5,000	0.587	0.823	0.414	1.494
10,000	0.322	0.346	0.264	1.670

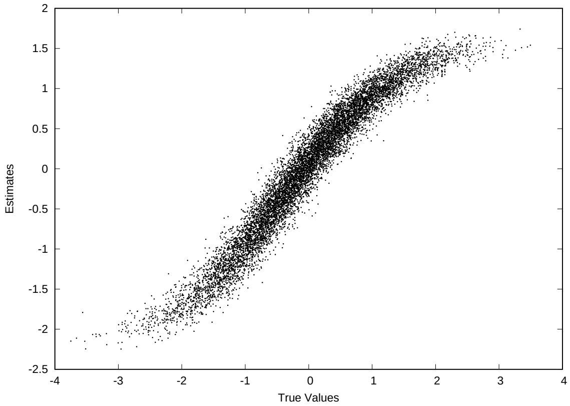
Correlation

Size	a_1	a_2	a_3	b
500	0.457	0.547	0.381	0.987
5000	0.779	0.710	0.990	0.982
10000	0.924	0.920	0.986	0.990

ML2P-VAE Results







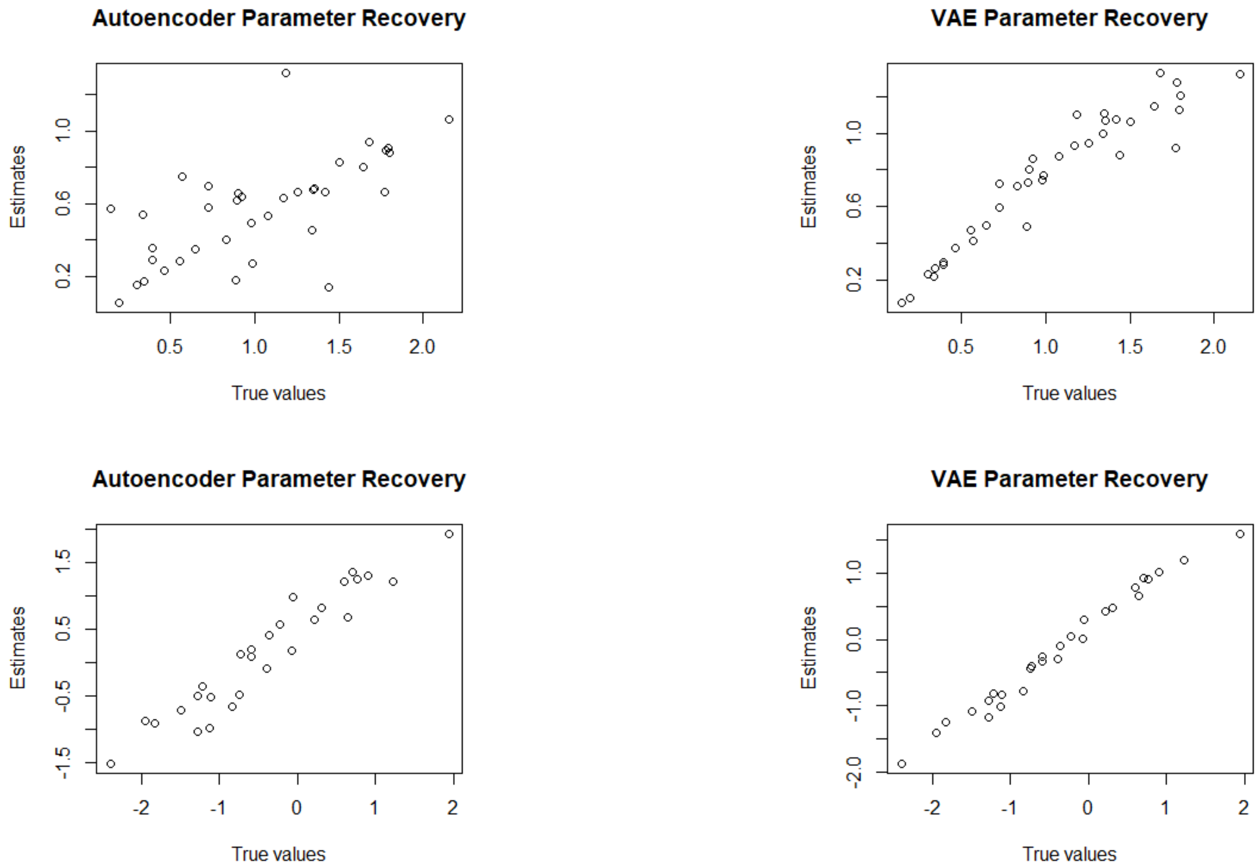
VAE vs AE Comparison

- Guo, Cutumisu, and Cui proposed using AE in skill estimation
- Directly compare neural networks in ML2P application
  - Parameter recovery
  - Skill estimation

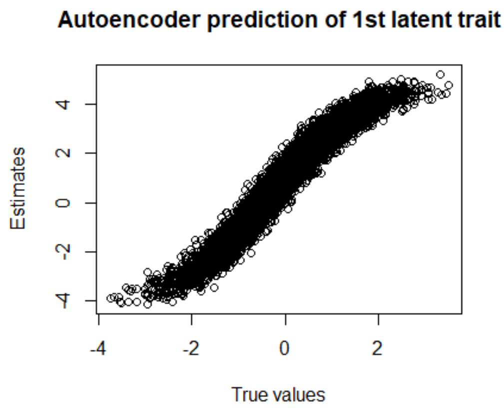
VAE vs AE Comparison

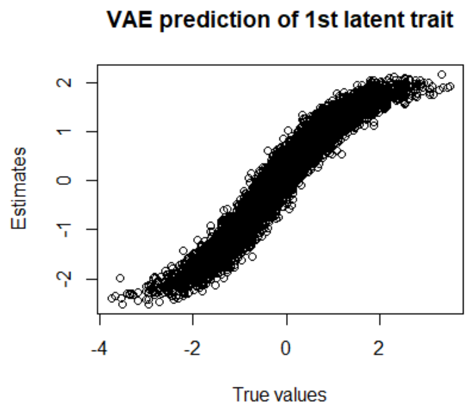
- Guo, Cutumisu, and Cui proposed using AE in skill estimation
- Directly compare neural networks in ML2P application
  - Parameter recovery
  - Skill estimation
- Results presented at Artificial Intelligence in Education (AIED) 2019

VAE vs AE Comparison



VAE vs AE Comparison





- Similar skill estimate correlation, but on different scale
- VAE much more accurate parameter recovery

New Application: Sports Analytics

- Other fields attempt to capture unobservable latent skills
- Player Evaluation
  - Moneyball
  - Baseball stats: WAR, ISO, etc.

New Application: Sports Analytics

- Other fields attempt to capture unobservable latent skills
- Player Evaluation
  - Moneyball
  - Baseball stats: WAR, ISO, etc.

Baseball Player Evaluation

- Goal: Develop **new** measures for unobservable skills

Baseball Player Evaluation

- Goal: Develop **new** measures for unobservable skills
- Relate measured statistics to underlying skills that MLB players need
- Similar model architecture as ML2P-VAE

Baseball Player Evaluation

- Goal: Develop **new** measures for unobservable skills
- Relate measured statistics to underlying skills that MLB players need
- Similar model architecture as ML2P-VAE
- Predicting latent skills by 13 measured baseball statistics :
  - 1B, 2B, HR, R, RBI, BB, IBB, K, GDP, SB, CS, SAC.

Baseball Player Evaluation

- Goal: Develop **new** measures for unobservable skills
- Relate measured statistics to underlying skills that MLB players need
- Similar model architecture as ML2P-VAE
- Predicting latent skills by 13 measured baseball statistics :
  - 1B, 2B, HR, R, RBI, BB, IBB, K, GDP, SB, CS, SAC.
- Four (independent) latent skills to predict:
  - Contact
  - Baserunning
  - Power
  - Pitch Intuition

Baseball Skill Q-Matrix

Contact	Baserunning	Power	Pitch Intuition	
1	0	0	0	Singles
1	0	1	0	Doubles
0	0	1	0	Homeruns
0	1	0	0	Runs
0	0	1	0	Runs Batted In
0	0	0	1	Walks
0	0	1	0	Intentional Walks
1	0	0	1	Strikeouts
1	0	0	0	Sacrifice
0	1	0	0	Grounded into Double Play
0	1	0	0	Stolen Bases
0	1	0	0	Caught Stealing

Baseline Evaluation Stats

- Contact Rate

$$CR = \frac{AB - K}{AB}$$

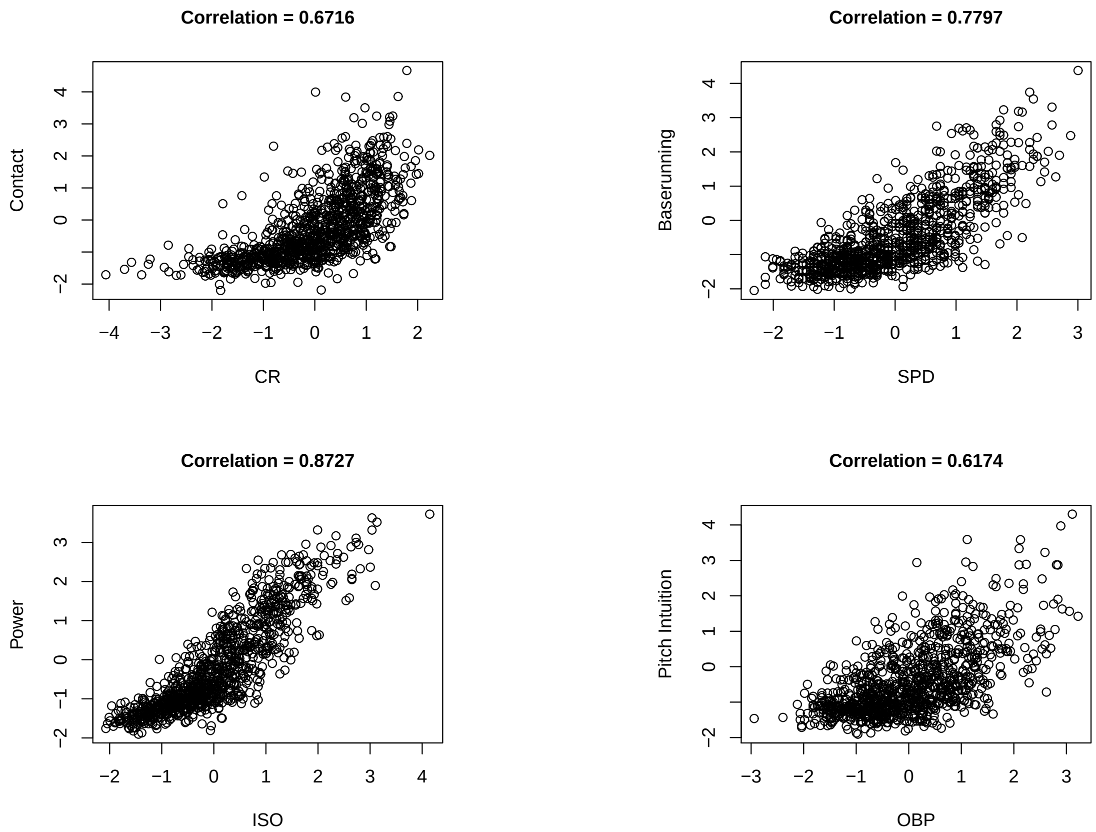
- Isolated Power

$$ISO = \frac{(2B) + (2 \cdot 3B) + (3 \cdot HR)}{AB}$$

- Speed Statistic: a linear combination of stolen base percentage, attempts, triples, double plays, runs, and position
- On-Base Percentage

$$OBP = \frac{H + BB + HBP}{AB + BB + HBP + SAC}$$

Results


Correlation = 0.6716
Contact
CR
Correlation = 0.7797
Baserunning
SPD
Correlation = 0.8727
Power
ISO
Correlation = 0.6174
Pitch Intuition
OBP

Results

Year	Player
1981	Tim Foli
1977	Bert Campaneris
1974	Len Randle
1994	Felix Ferman
1979	Craig Reynolds

Table 1: Contact

Year	Player
1981-1982	Rickey Henderson
1974	Lou Brock
1985	Vince Coleman
1981	Tim Raines
1994	Kenny Lofton

Table 3: Baserunning

Year	Player
2001-2004	Barry Bonds
1970	Willie McCovey
2001	Sammy Sosa
2009	Albert Pujols
1998	Mark McGwire

Table 2: Power

Year	Player
2002, 2004	Barry Bonds
1952	Elmer Valo
1951, 1954	Ted Williams
1951	Johnny Pesky
1975	Joe Morgan

Table 4: Pitch Intuition

Full Covariance Matrix in IRT

- In real applications, independent skills are not realistic
    - Example: differentiation rules

Full Covariance Matrix in IRT

- In real applications, independent skills are not realistic
  - Example: differentiation rules
- Covariance matrix is symmetric, positive definite matrix

$$\Sigma = \begin{bmatrix} \sigma_1^2 & c_{12} & \cdots & c_{1k} \\ c_{21} & \sigma_2^2 & \cdots & c_{2k} \\ \vdots & & \ddots & \vdots \\ c_{k1} & \cdots & c_{k(k-1)} & \sigma_k^2 \end{bmatrix}$$

- With variances σ_i^2 and covariances $c_{ij} = c_{ji}$

Full Covariance Matrix in VAE

- KL Divergence between two k-dimensional multivariate normal distributions:

$$\mathcal{D}_{KL}\left[\mathcal{N}(\mu_0, \Sigma_0) || \mathcal{N}(\mu_1, \Sigma_1)\right] =$$
$$\frac{1}{2}\left(\mathrm{tr}(\Sigma_1^{-1}\Sigma_0) + (\mu_1 - \mu_0)^T\Sigma_1^{-1}(\mu_1 - \mu_0) - k + \ln\left(\frac{\det \Sigma_1}{\det \Sigma_0}\right)\right)$$

Full Covariance Matrix in VAE

- KL Divergence between two k-dimensional multivariate normal distributions:

$$\mathcal{D}_{KL}\left[\mathcal{N}(\mu_0, \Sigma_0) || \mathcal{N}(\mu_1, \Sigma_1)\right] =$$
$$\frac{1}{2}\left(\mathrm{tr}(\Sigma_1^{-1}\Sigma_0) + (\mu_1 - \mu_0)^T \Sigma_1^{-1}(\mu_1 - \mu_0) - k + \ln\left(\frac{\det \Sigma_1}{\det \Sigma_0}\right)\right)$$

- When fitting a VAE, $\mathcal{N}(\mu_1, \Sigma_1)$ is known, so μ_1 and Σ_1 are constant
- μ_0 and Σ_0 obtained from feeding one sample through the encoder

Full Covariance VAE Implementation

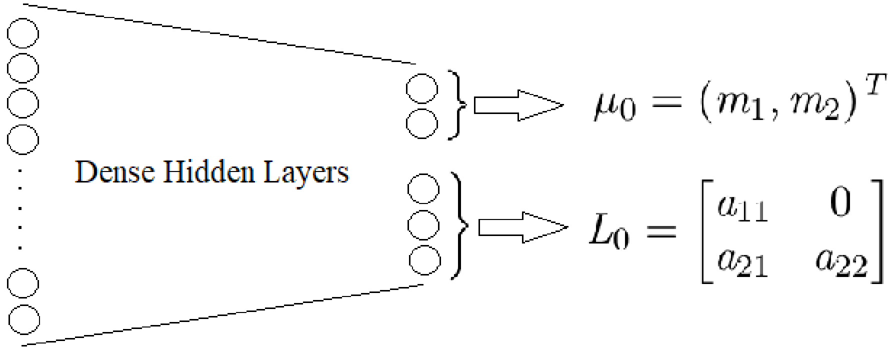
- To sample from a multivariate normal distribution $\mathcal{N}(\mu_0, \Sigma_0)$:
  - Find a matrix G such that $GG^T = \Sigma_0$
  - Sample $\varepsilon = (\varepsilon_1, ..., \varepsilon_k)^T$ with each $\varepsilon_i \sim \mathcal{N}(0, 1)$
  - Generate sample $z = \mu_0 + G\varepsilon$

Full Covariance VAE Implementation

- To sample from a multivariate normal distribution $\mathcal{N}(\mu_0, \Sigma_0)$:
  - Find a matrix G such that $GG^T = \Sigma_0$
  - Sample $\varepsilon = (\varepsilon_1, ..., \varepsilon_k)^T$ with each $\varepsilon_i \sim \mathcal{N}(0, 1)$
  - Generate sample $z = \mu_0 + G\varepsilon$
- KL Divergence calculation uses μ_0, Σ_0, and requires $\det \Sigma_0 > 0$

Full Covariance VAE Implementation

- To sample from a multivariate normal distribution $\mathcal{N}(\mu_0, \Sigma_0)$:
  - Find a matrix G such that $GG^T = \Sigma_0$
  - Sample $\varepsilon = (\varepsilon_1, ..., \varepsilon_k)^T$ with each $\varepsilon_i \sim \mathcal{N}(0, 1)$
  - Generate sample $z = \mu_0 + G\varepsilon$
- KL Divergence calculation uses μ_0, Σ_0, and requires $\det \Sigma_0 > 0$



Encoder structure for VAE learning $\mathcal{N}(0, I)$

Full Covariance VAE Implementation

- Architecture: Encoder outputs $k + k(k+1)/2$ nodes
  - k nodes for μ_0, and $k(k+1)/2$ nodes for L_0 lower triangular

Full Covariance VAE Implementation

- Architecture: Encoder outputs $k + k(k+1)/2$ nodes
  - k nodes for μ_0, and $k(k+1)/2$ nodes for L_0 lower triangular
- Sampling: Calculate $G_0 = e^{L_0}$
  - Note G_0 is lower triangular, nonsingular
  - Send sample $z = \mu_0 + G_0\varepsilon$ through decoder

Full Covariance VAE Implementation

- Architecture: Encoder outputs $k + k(k+1)/2$ nodes
  - k nodes for μ_0, and $k(k+1)/2$ nodes for L_0 lower triangular
- Sampling: Calculate $G_0 = e^{L_0}$
  - Note G_0 is lower triangular, nonsingular
  - Send sample $z = \mu_0 + G_0\varepsilon$ through decoder
- KL Divergence: Calculate $\Sigma_0 = G_0 G_0^T$
  - Note that

$$\begin{aligned}\det \Sigma_0 &= \det(e^{L_0}(e^{L_0})^T) = \det e^{L_0} \cdot \det(e^{L_0})^T \\ &= e^{\operatorname{tr} L_0} \cdot e^{\operatorname{tr} {L_0}^T} = \left(e^{\operatorname{tr} L_0}\right)^2 \\ &> 0\end{aligned}$$

Full Covariance VAE Implementation

- Architecture: Encoder outputs $k + k(k+1)/2$ nodes
  - k nodes for μ_0, and $k(k+1)/2$ nodes for L_0 lower triangular
- Sampling: Calculate $G_0 = e^{L_0}$
  - Note G_0 is lower triangular, nonsingular
  - Send sample $z = \mu_0 + G_0\varepsilon$ through decoder
- KL Divergence: Calculate $\Sigma_0 = G_0 G_0^T$
  - Note that

$$\begin{aligned}\det \Sigma_0 &= \det(e^{L_0}(e^{L_0})^T) = \det e^{L_0} \cdot \det(e^{L_0})^T \\ &= e^{\operatorname{tr} L_0} \cdot e^{\operatorname{tr} L_0{}^T} = \left(e^{\operatorname{tr} L_0}\right)^2 \\ &> 0\end{aligned}$$

  - Claim: Σ_0 is symmetric positive definite

Full Covariance VAE Implementation

Claim: Σ_0 is symmetric and positive definite.

Proof.

For each sample x_0, the encoder returns $L_0 \in \mathbb{R}^{k \times k}$ lower triangular.

$\square$

Full Covariance VAE Implementation

Claim: Σ_0 is symmetric and positive definite.

Proof.

For each sample x_0, the encoder returns $L_0 \in \mathbb{R}^{k \times k}$ lower triangular. Consider the matrix exponential

$$G_0 := e^{L_0} = \sum_{n=0}^{\infty} \frac{L_0^n}{n!} = I + L_0 + \frac{1}{2} L_0^2 + \cdots$$

$\square$

Full Covariance VAE Implementation

Claim: Σ_0 is symmetric and positive definite.

Proof.

For each sample x_0, the encoder returns $L_0 \in \mathbb{R}^{k \times k}$ lower triangular. Consider the matrix exponential

$$G_0 := e^{L_0} = \sum_{n=0}^{\infty} \frac{L_0^n}{n!} = I + L_0 + \frac{1}{2} L_0^2 + \cdots$$

G_0 is lower triangular, since addition and multiplication preserve lower triangular. G_0 is also nonsingular:

$$\det G_0 = \det e^{L_0} = e^{\operatorname{tr} L_0} \neq 0$$

□

Full Covariance VAE Implementation

Claim: Σ_0 is symmetric and positive definite.

Proof.

For each sample x_0, the encoder returns $L_0 \in \mathbb{R}^{k \times k}$ lower triangular. Consider the matrix exponential

$$G_0 := e^{L_0} = \sum_{n=0}^{\infty} \frac{L_0^n}{n!} = I + L_0 + \frac{1}{2} L_0^2 + \cdots$$

G_0 is lower triangular, since addition and multiplication preserve lower triangular. G_0 is also nonsingular:

$$\det G_0 = \det e^{L_0} = e^{\operatorname{tr} L_0} \neq 0$$

Set $\Sigma_0 := G_0 G_0^T$. Now for any nonzero $y \in \mathbb{R}^k$,

□

Full Covariance VAE Implementation

Claim: Σ_0 is symmetric and positive definite.

Proof.

For each sample x_0, the encoder returns $L_0 \in \mathbb{R}^{k \times k}$ lower triangular. Consider the matrix exponential

$$G_0 := e^{L_0} = \sum_{n=0}^{\infty} \frac{L_0^n}{n!} = I + L_0 + \frac{1}{2} L_0^2 + \cdots$$

G_0 is lower triangular, since addition and multiplication preserve lower triangular. G_0 is also nonsingular:

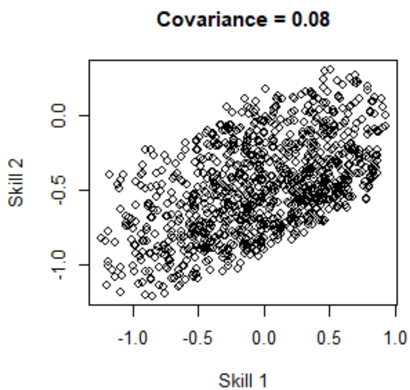
$$\det G_0 = \det e^{L_0} = e^{\operatorname{tr} L_0} \neq 0$$

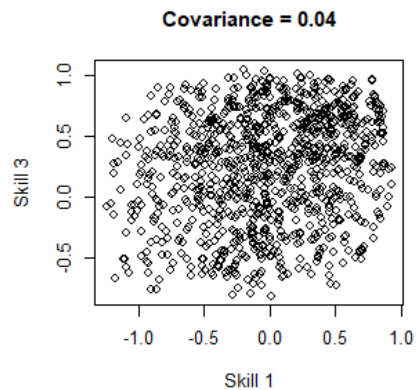
Set $\Sigma_0 := G_0 G_0^T$. Now for any nonzero $y \in \mathbb{R}^k$,

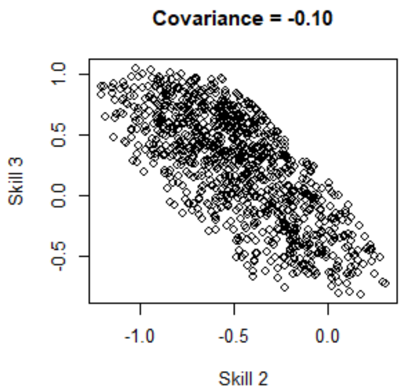
$$\langle \Sigma_0 y, y \rangle = y^T \Sigma_0 y = y^T G_0 G_0^T y = \langle G_0^T y, G_0^T y \rangle = || G_0^T y ||_2^2 > 0$$

□

Full Covariance VAE Experimentation







$$\Sigma = \begin{bmatrix} .36 & .12 & .06 \\ .12 & .29 & -.13 \\ .06 & -.13 & .26 \end{bmatrix}$$

$$\hat{\Sigma} = \begin{bmatrix} .25 & .08 & .04 \\ .08 & .10 & -.10 \\ .04 & -.10 & .20 \end{bmatrix}$$

$$\mu = (0, -.5, .3)^T$$

$$\hat{\mu} = (-.01, -.47, .23)^T$$

Package in R

- Create software package in R with ML2P-VAE method
- Submit to CRAN for resarchers without tensorflow experience to use

Package in R

- Create software package in R with ML2P-VAE method
- Submit to CRAN for resarchers without tensorflow experience to use
- Package functions:
  - Construct ML2P-VAE model to desired architecture
  - Option for independent latent traits, or full covariance matrix
  - Sufficient documentation and working examples

Future Work

- Continue working on current projects
- Analyze convergence for ML2P-VAE
- Experiment with real data

References

[1] Wainer and Thissen, D. "Test Scoring". Erlbaum Associates, Publishers, 2001.

[2] da Silva, Liu, Huggins-Manley, Bazan. "Incorporating the Q-matrix into Multidimensional Item Response Models." Journal of Educational and Psychological Measurement, 2018.

[3] Baker and Kim. "Item Response Theory: Parameter Estimation Techniques". CRC Press, 2004.

[4] Bock and Aitken. "Marginal Maximum Likelihood Estimation of Item Parameters: Application of an EM Algorithm". Psychometrika, 1981.

[5] Nielsen, Michael. "Neural Networks and Deep Learning". Determination Press, 2015.

[6] Curi, Converse, Hajewski, Oliveira. "Interpretable Variational Autoencoders for Cognitive Models." In Proceddings of the International Joint Conference on Neural Networks (IJCNN), 2019.

[7] Q. Guo, M. Cutumisu, and Y. Cui. "A Neural Network Approach to Estimate Student Skill Mastery in Cognitive Diagnostic Assessments". In: 10th International Conference on Educational Data Mining. 2017.

[8] Converse, Curi, Oliveira. "Autoencoders for Educational Assessment." In Proceedings of the Conference on Artifical Intelligence in Education (AIED), 2019.

[9] Converse, Curi, Oliveira, and Arnold. "Variational Autoencoders for Baseball Player Evaluation." In Proceedings of the Fuzzy Systems and Data Mining Conference (FSDM), 2019.

Interpretable Neural Networks for Item Response Theory Parameter Estimation

Geoffrey Converse

University of Iowa

October 1, 2019