Neural Network Methods for Application in Educational Measurement

Geoffrey Converse

University of Iowa

July 15, 2021

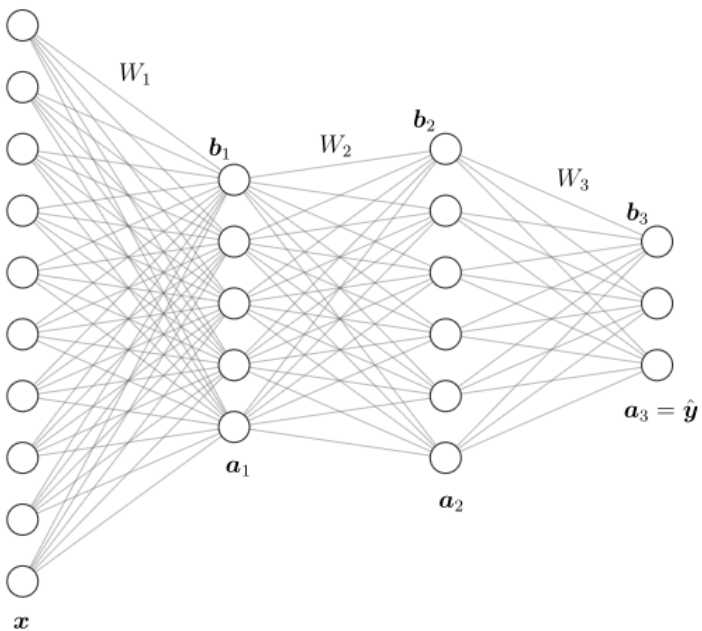
PhD Defense in Applied Mathematical and Computational Sciences

Overview

Develop machine learning algorithms to quantify student learning

Outline

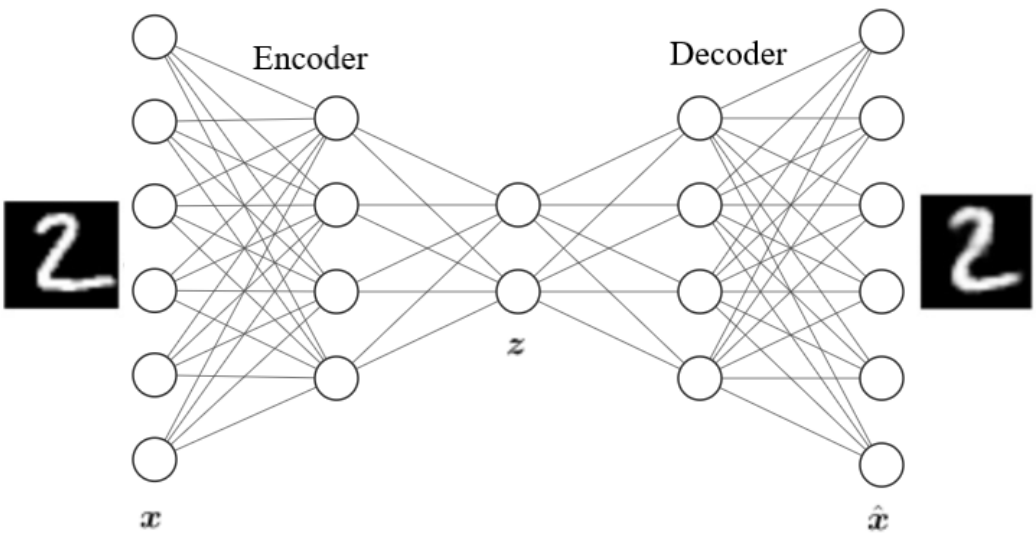
Artificial Neural Networks (ANN)



Input $\boldsymbol{x}$, approximate a true target $\boldsymbol{y}$ via a series of (learned) linear transformations and nonlinear re-scaling

$$\hat{\boldsymbol{y}} = \sigma(\, W_3\sigma(\, W_2\sigma(\, W_1\boldsymbol{x} + vectb_1) + \boldsymbol{b}_2) + \boldsymbol{b}_3)$$

Autoencoder (AE)



- Encode data into smaller dimension
- Reconstruct original input: $\mathcal{L} = ||\boldsymbol{x} - \hat{\boldsymbol{x}}||$

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

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

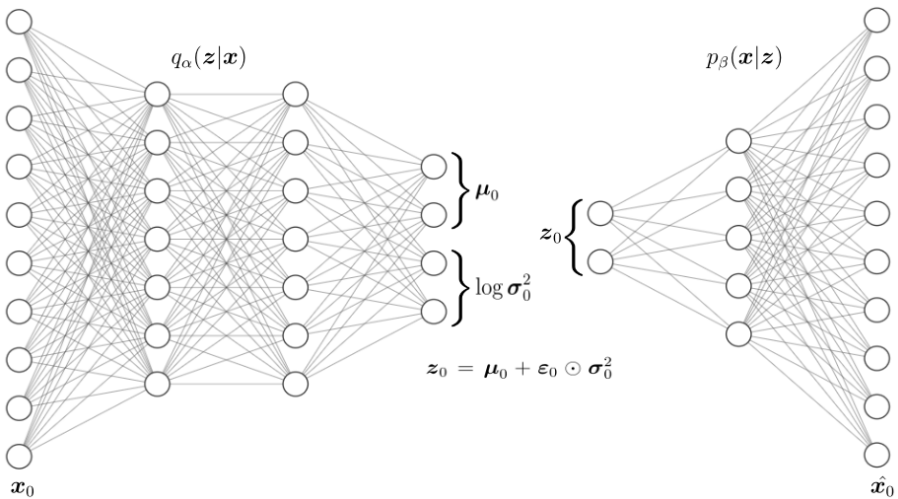
$$P(\boldsymbol{X} = \boldsymbol{x}) = \int P(\boldsymbol{X} = \boldsymbol{x}|\boldsymbol{\theta})f(\boldsymbol{\theta})\,d\boldsymbol{\theta},$$

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

VAE loss

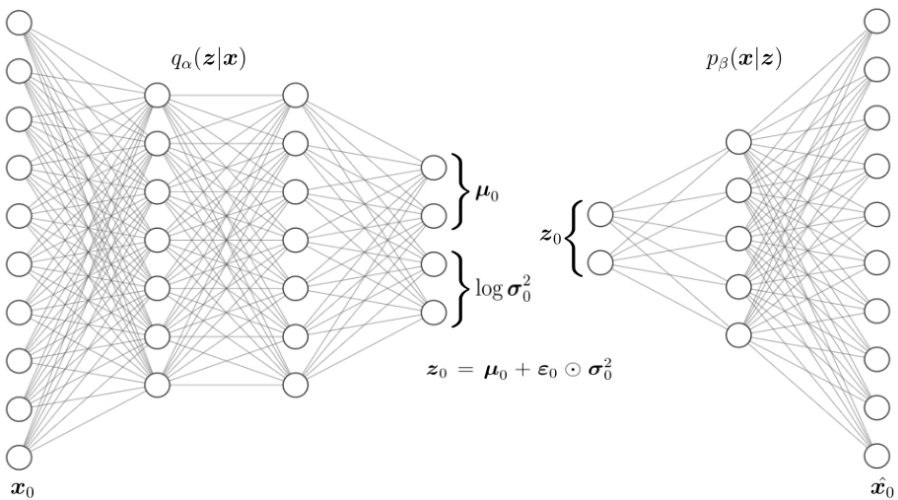
TODO: derive VAE loss

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)

mona lisa visual might be okay, but want to emphasize that encoder learns a posterior distribution

Item Response Theory (IRT)

TODO: clean this up

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

Item Response Theory (IRT)

TODO: clean this up

- 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)

TODO: clean this up

- 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

Rasch Model

- Define $\delta_i > 0$ as the difficulty of item i, and $\eta_j > 0$ the ability of subject j.
- Rasch: Probability of success depends on ratio $\frac{\delta_i}{\eta_j}$

$$P(u_{ij} = 1|\eta_j, \delta_i) = \frac{1}{1 + \delta_i/\eta_j} = \frac{\eta_j}{\eta_j + \delta_i}$$

- Logarithmic transformation: $\theta_j = \log \eta_j$ and $\beta_i = \log \delta_i$
- Rasch Model :

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

2-Parameter Logistic Model (2PL)

- Probability of a correct response follows the logistic equation:

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

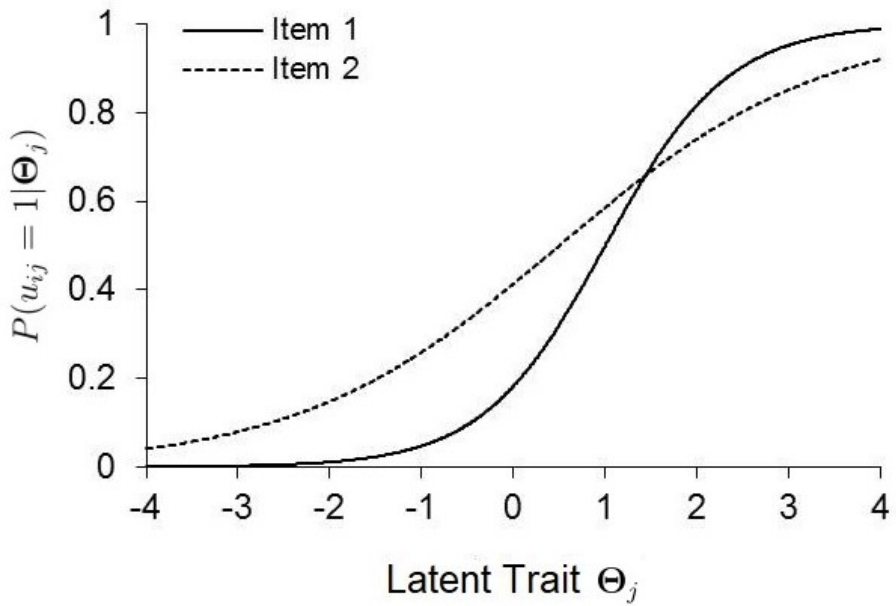
2-Parameter Logistic Model (2PL)

- Probability of a correct response follows the logistic equation:

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

  - a_i = discrimination parameter
  - b_i = difficulty parameter

Item Characteristic Curve (ICC)



Multidimensional IRT

- 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

Multidimensional IRT

- 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

Joint Maximum Likelihood Estimation (JMLE)

TODO: clean this up and summarize JMLE in one slide

- Need to estimate student and item parameters simultaneously

Joint Maximum Likelihood Estimation (JMLE)

TODO: clean this up and summarize JMLE in one slide

- Need to estimate student and item parameters 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)

TODO: clean this up and summarize JMLE in one slide

- Need to estimate student and item parameters 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:

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

Marginal Maximum Likelihood (MMLE)

TODO: clean and summarize MMLE in one slide

- 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)

TODO: clean and summarize MMLE in one slide

- 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)

TODO: clean and summarize MMLE in one slide

- 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

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

Difficulties of IRT Parameter Estimation

TODO: summarize the problems with high-dim theta

Combining IRT and ANN

TODO: clean this up

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

Combining IRT and ANN

TODO: clean this up

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

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

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

Model Description

- No hidden layers in the decoder

Model Description

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

Model Description

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

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

Model Description

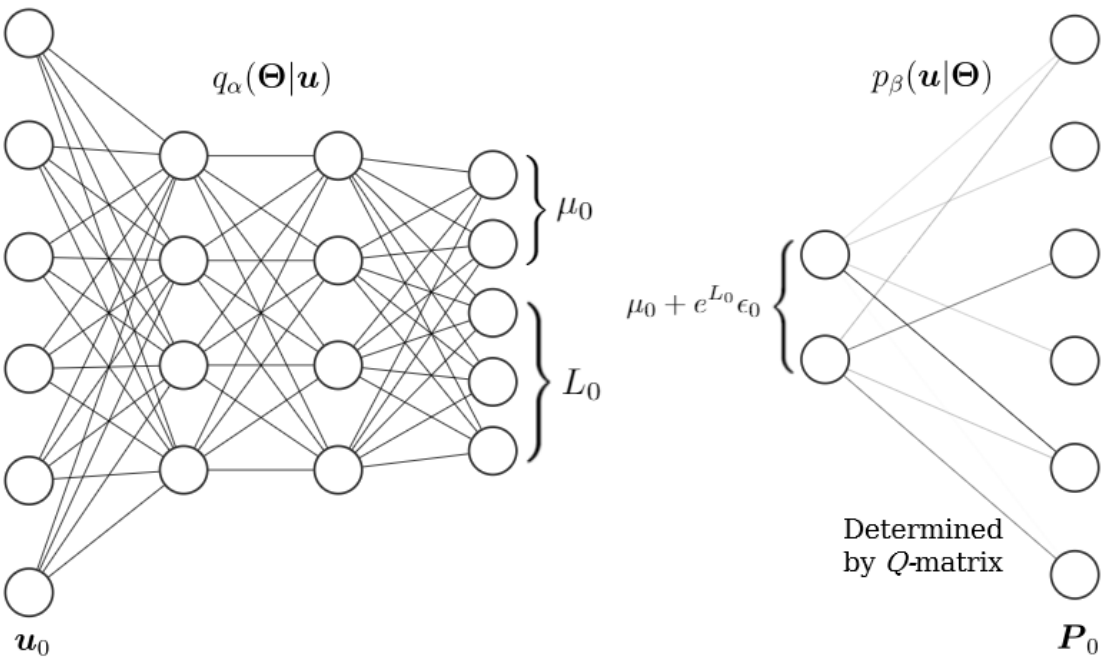
- No hidden layers in the decoder
- Restrict nonzero weights in the decoder according to Q-matrix
- Sigmoidal activation function in output layer
- Require decoder weights to be nonnegative
- Fit VAE latent space to $\mathcal{N}(0, I)$

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

TODO: this image is for correlated latent traits



VAE vs AE Comparison

TODO: clean up and be better

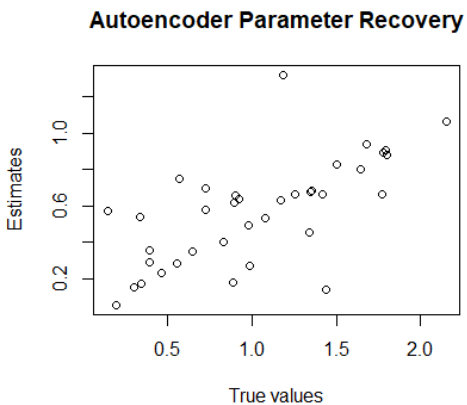
- Guo, Cutumisu, and Cui proposed using AE in skill estimation
- Directly compare neural networks in ML2P application
  - Parameter recovery
  - Skill estimation
- How useful is prior $p(\boldsymbol{\Theta})$?
- What is the effect of the KL-Divergence term in the loss function?

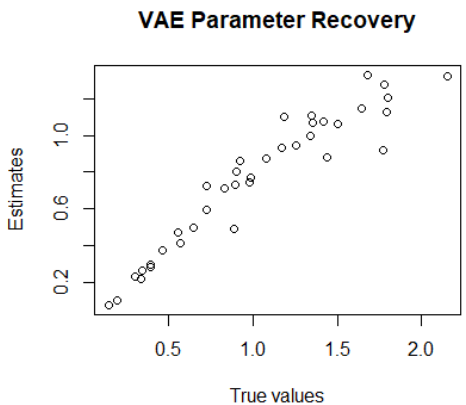
VAE vs AE Comparison

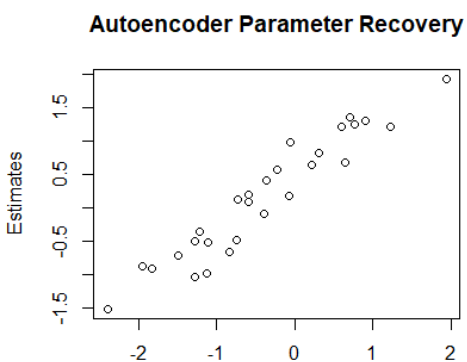
TODO: clean up and be better

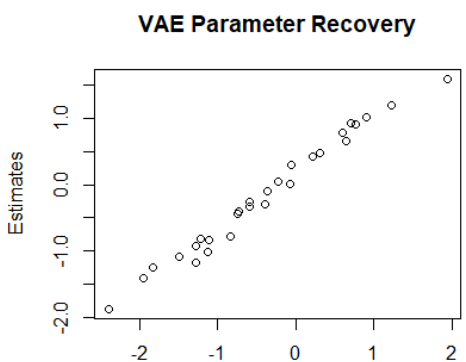
- Guo, Cutumisu, and Cui proposed using AE in skill estimation
- Directly compare neural networks in ML2P application
  - Parameter recovery
  - Skill estimation
- How useful is prior $p(\boldsymbol{\Theta})$?
- What is the effect of the KL-Divergence term in the loss function?
- Results presented at Artificial Intelligence in Education (AIED) 2019

VAE vs AE Comparison

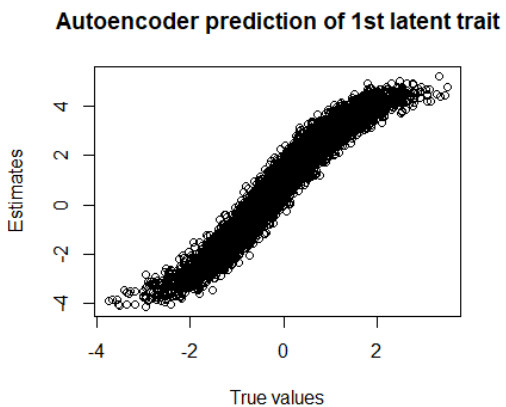


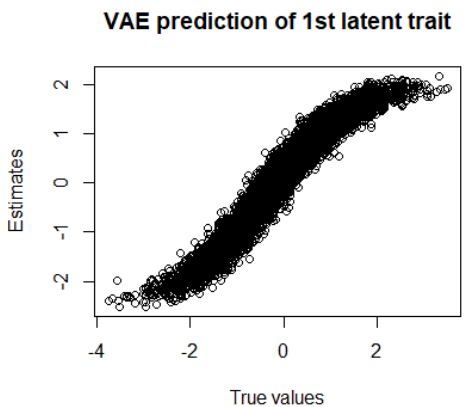






VAE vs AE Comparison





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

Correlated Latent Traits in IRT

- In real applications, independent skills are not realistic: $\boldsymbol{\Theta} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, not $\mathcal{N}(0, I)$.
  - Example: students who are good at adding are also good at subtracting

Correlated Latent Traits in IRT

- In real applications, independent skills are not realistic: $\boldsymbol{\Theta} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, not $\mathcal{N}(0, I)$.
  - Example: students who are good at adding are also good at subtracting
- 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 $c_{ij} = c_{ji}$

Correlated Latent Code in VAE

- In most VAE applications, it is convenient to assume latent code $\boldsymbol{z}$ is *independent*
  - Forces each dimension of $\boldsymbol{z}$ to measure different features
  - $\boldsymbol{z}$ is *abstract*, with **no real-world understanding**
- For ML2P-VAE, we know that latent code $\boldsymbol{z}$ approximates latent traits $\boldsymbol{\Theta}$
  - We may have **domain knowledge** of the distribution of $\boldsymbol{\Theta}$

KL Divergence for correlated latent code

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

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

KL Divergence for correlated latent code

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

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

- 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

Requirements for Correlated VAE

- 1. 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 $z = \mu_0 + G\varepsilon$

Requirements for Correlated VAE

- 1. 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 $z = \mu_0 + G\varepsilon$
- 2. KL Divergence calculation uses μ_0, Σ_0, and requires $\det \Sigma_0 > 0$

Correlated 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

Correlated 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

Correlated 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

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

Correlated 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

Correlated VAE Implementation

TODO: clean up proof **Theorem**: Let L_0 be any lower triangular matrix. Then $\Sigma_0 = e^{L_0} \cdot \left(e^{L_0}\right)^\top$ is symmetric and positive definite.

Proof.

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

Correlated VAE Implementation

TODO: clean up proof **Theorem**: Let L_0 be any lower triangular matrix. Then $\Sigma_0 = e^{L_0} \cdot \left(e^{L_0}\right)^\top$ 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

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

Correlated VAE Implementation

TODO: clean up proof **Theorem**: Let L_0 be any lower triangular matrix. Then $\Sigma_0 = e^{L_0} \cdot \left(e^{L_0}\right)^\top$ 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:

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

Correlated VAE Implementation

TODO: clean up proof **Theorem**: Let L_0 be any lower triangular matrix. Then $\Sigma_0 = e^{L_0} \cdot \left(e^{L_0}\right)^\top$ 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 $y \in \mathbb{R}^k$,

Correlated VAE Implementation

TODO: clean up proof **Theorem**: Let L_0 be any lower triangular matrix. Then $\Sigma_0 = e^{L_0} \cdot \left(e^{L_0}\right)^\top$ 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

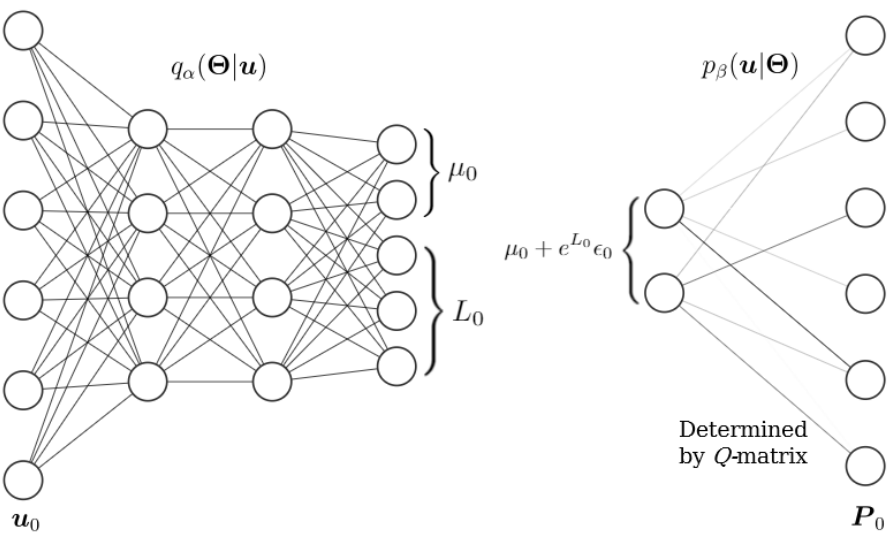
$$G_0 := e^{L_0} = \sum_{n=0}^{\infty} \frac{L_0^n}{n!} = I + L_0 + \frac{1}{2} L_0^2 + \cdots$$

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 $y \in \mathbb{R}^k$,

VAE architecture for correlated latent traits



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

Datasets

a table might be sufficient

Correlated ML2P-VAE Results

Sim 6 or Sim 4

Correlated ML2P-VAE Results

ECPE

Correlated ML2P-VAE Results

Sim 20

Correlated ML2P-VAE Results

big table

Scalability of ML2P-VAE

Plots of accuracy/time elapsed vs num training samples

Package in R

TODO: give better explanation of why this is worth mentioning

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

Package in R

TODO: give better explanation of why this is worth mentioning

- 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

ML2Pvae package

another frame for R package? not sure if I need two frames here

Bayesian Knowledge Tracing

TODO: background/motivation of KT

RNN / LSTM

RNN, LSTM

Attention-based networks

Transformer / Attention

Deep Knowledge Tracing

DKT

SAKT

SAKT

Other methods

might want to mention DKVMN or PFA

Do deep models actually "trace" knowledge?

motivation

IRT-inspired Knowledge Tracing

method description

IRT-inspired Knowledge Tracing

image of architecture

Datasets

datasets

Results

Table and theta trace plot

Recovery of IRT parameters

disc and theta recovery plots

Learning a Q-matrix

cor heatmap and clustering

Extending ML2P-VAE to other IRT models

3PL and Samejima

Other application areas

BDI and personality questionnaires

Utilizing more domain knowledge

use Q matrix in attn calculation missing responses with embedding of interactions

Summary

Summary

Thank you!

References

TODO: choose citations in the right way

[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.

Neural Network Methods for Application in Educational Measurement

Geoffrey Converse

University of Iowa

July 15, 2021