Some Bayesian extensions of neural network-based graphon approximations

Creighton Heaukulani

Joint work with

Onno Kampman (Hong Kong)

EcoSta 2018, Hong Kong
June 2018

Overview

1. Review neural network graphon approximation and its gradient-based inference. When are nnets useful?

Overview

1. Review neural network graphon approximation and its gradient-based inference. When are nnets useful?

2. Consider variational inference in such a model and why.

Overview

1. Review neural network graphon approximation and its gradient-based inference. When are nnets useful?

2. Consider variational inference in such a model and why.

3. Implement an infinite stochastic blockmodel, with good reason.

Overview

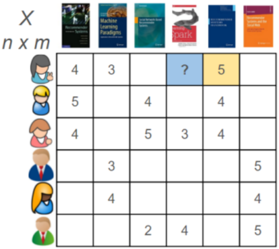
1. Review neural network graphon approximation and its gradient-based inference. When are nnets useful?

2. Consider variational inference in such a model and why.

3. Implement an infinite stochastic blockmodel, with good reason.

4. Review the pros and cons of being Bayesian here and other lessons learned along the way.

Relational data modeling



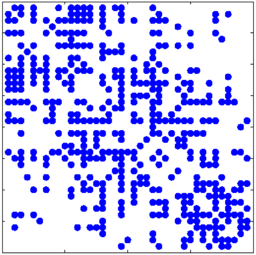
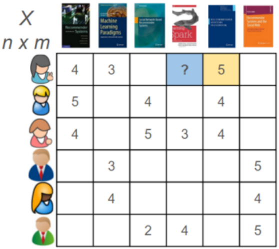
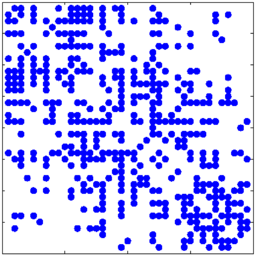


Figure from:
https://buildingrecommenders.wordpress.com/2015/11/18/overview-of-recommender-algorithms-part-2/

Relational data modeling





“Minibatch learning” with these two data structures...

- What’s the appropriate minibatch?
- Which entries are missing?

Lee et al. [2017]

Figure from:
https://buildingrecommenders.wordpress.com/2015/11/18/overview-of-recommender-algorithms-part-2/

Matrix factorization... linear models

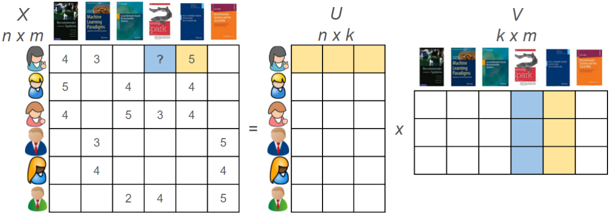
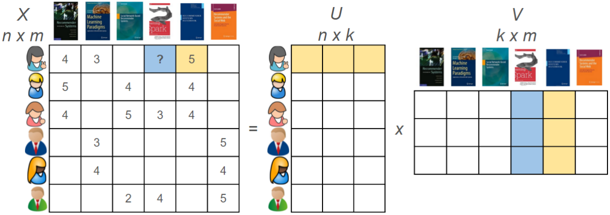


Figure from:
https://buildingrecommenders.wordpress.com/2015/11/18/overview-of-recommender-algorithms-part-2/

Matrix factorization... linear models



The (n, m)-th entry of the matrix is modeled as

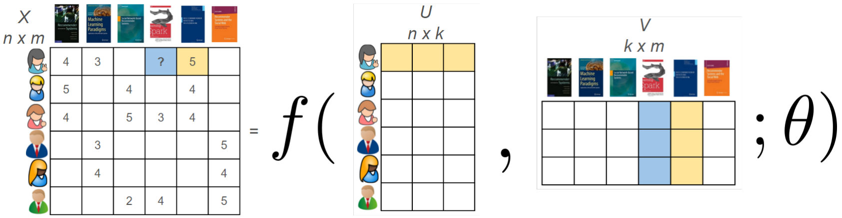
$$X_{n,m} \approx U_n^T V_m = \sum_{d=1}^{D} U_{n,d} V_{m,d}$$

Some $U_n \in \mathbb{R}^D$ and $V_m \in \mathbb{R}^D$, with D small. A linear model.

Figure from:
https://buildingrecommenders.wordpress.com/2015/11/18/overview-of-recommender-algorithms-part-2/

Neural network matrix factorization

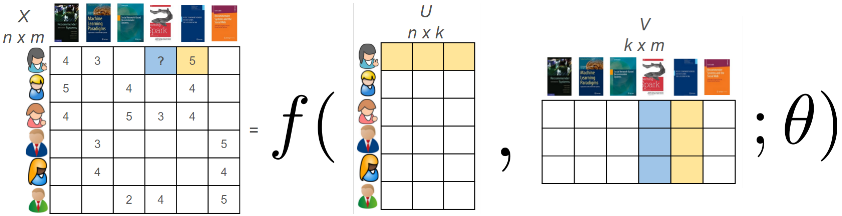
(Dziugaite and Roy [2015])



$f(\,\cdot\,;\theta)$ is a neural network with parameters θ

Neural network matrix factorization

(Dziugaite and Roy [2015])



$f(\,\cdot\,;\theta)$ is a neural network with parameters θ

The (n, m)-th entry of the matrix is modeled as

$$X_{n,m} \approx \cancel{U_n^T V_m = \sum_{d=1}^{D} U_{n,d} V_{m,d}} \;\; f(U_n, V_m; \theta)$$

Generalized to a nonlinear model.

Neural network matrix factorization

(Dziugaite and Roy [2015])

Matrix factorization	Network model
$X_{n,m} \approx f(U_n, V_m; \theta)$	$\mathbb{P}\{X_{n,m} = 1\} \approx \sigma(f(U_n, V_m; \theta))$

E.g.,

$$X_{n,m} \approx W_o \sigma(W_h \cdot [U_n, V_m] + b_h) + b_o$$

Neural network matrix factorization

(Dziugaite and Roy [2015])

Matrix factorization

$$X_{n,m} \approx f(U_n, V_m; \theta)$$

Network model

$$\mathbb{P}\{X_{n,m} = 1\} \approx \sigma(f(U_n, V_m; \theta))$$

E.g.,

$$X_{n,m} \approx W_o \sigma(W_h \cdot [U_n, V_m] + b_h) + b_o$$

Within the graphon modeling/approximation framework (Lloyd et al. [2012], Orbanz and Roy [2015]).

Neural network matrix factorization

(Dziugaite and Roy [2015])

Matrix factorization	Network model
$X_{n,m} \approx f(U_n, V_m; \theta)$	$\mathbb{P}\{X_{n,m} = 1\} \approx \sigma(f(U_n, V_m; \theta))$

E.g.,

$$X_{n,m} \approx W_o \sigma(W_h \cdot [U_n, V_m] + b_h) + b_o$$

Within the graphon modeling/approximation framework (Lloyd et al. [2012], Orbanz and Roy [2015]).

Note: Inputs of the nnet are now parameters. (A Bayesian habit?)

Neural network matrix factorization

(Dziugaite and Roy [2015])

Matrix factorization	Network model
$X_{n,m} \approx f(U_n, V_m; \theta)$	$\mathbb{P}\{X_{n,m} = 1\} \approx \sigma(f(U_n, V_m; \theta))$

Neural network matrix factorization

(Dziugaite and Roy [2015])

Matrix factorization	Network model
$X_{n,m} \approx f(U_n, V_m; \theta)$	$\mathbb{P}\{X_{n,m} = 1\} \approx \sigma(f(U_n, V_m; \theta))$

Gradient-based inference targeting, for example,

$$\begin{aligned} \text{Loss } = & \sum_{(n,m)} (X_{n,m} - f(U_n, V_m; \theta))^2 \\ & + \lambda_1 (||U||_F^2 + ||V||_F^2) && \text{regularize inputs (?)} \\ & + \lambda_2 ||\theta||_F^2 && \text{L1/L2 regularization} \end{aligned}$$

Neural network matrix factorization

(Dziugaite and Roy [2015])

Matrix factorization	Network model
$X_{n,m} \approx f(U_n, V_m; \theta)$	$\mathbb{P}\{X_{n,m} = 1\} \approx \sigma(f(U_n, V_m; \theta))$

Gradient-based inference targeting, for example,

$$\begin{aligned} \text{Loss } &= \sum_{(n,m)} (X_{n,m} - f(U_n, V_m; \theta))^2 \\ &\quad + \lambda_1 (||U||_F^2 + ||V||_F^2) && \text{regularize inputs (?)} \\ &\quad + \lambda_2 ||\theta||_F^2 && \text{L1/L2 regularization} \end{aligned}$$

Competitive performance; dominates linear baselines

When are deep learning architectures useful?

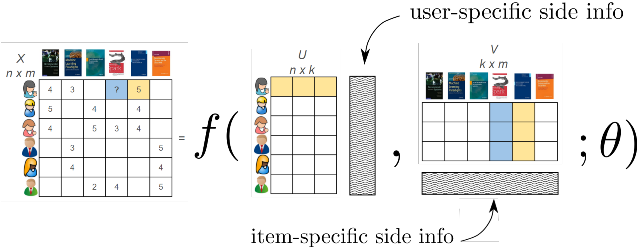
... for this matrix factorization problem anyway ...

When are deep learning architectures useful?

... for this matrix factorization problem anyway ...

Pros:

- Black-box for incorporating side information

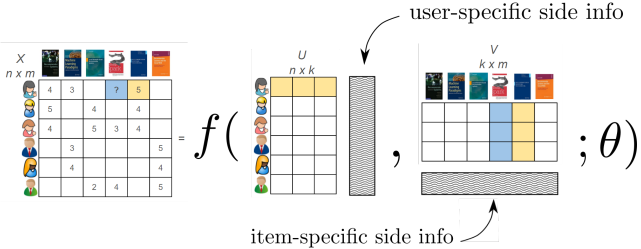


When are deep learning architectures useful?

... for this matrix factorization problem anyway ...

Pros:

- Black-box for incorporating side information



- Gradient-based learning tools (e.g., Tensorflow/Torch/etc.)

When are deep learning architectures useful?

... for this matrix factorization problem anyway ...

Cons:

- Lack of interpretability
- (What does that really mean? Why is this a problem?)

When are deep learning architectures useful?

... for this matrix factorization problem anyway ...

Cons:

- Lack of interpretability
- (What does that really mean? Why is this a problem?)

Motivates things like a “stochastic blockmodel”...

- In some (most?) cases, consumers don’t necessarily need to interpret the inferred nnet...
- Will often settle for some interpretable (inferred) components
  - like convincing clusterings of the users.

A stochastic blockmodeling extension

- Let $Z_n \in \{1, \ldots, K\}$ denote to which of K clusters/components user n is assigned.

A stochastic blockmodeling extension

- Let $Z_n \in \{1, \ldots, K\}$ denote to which of K clusters/components user n is assigned.
- Let $U_k \in \mathbb{R}^D$ be the features for cluster k.

A stochastic blockmodeling extension

- Let $Z_n \in \{1, \ldots, K\}$ denote to which of K clusters/components user n is assigned.
- Let $U_k \in \mathbb{R}^D$ be the features for cluster k.
- Construct entries like:

Matrix factorization	Network modeling
$X_{n,m} \approx f(U_{Z_n}, V_m; \theta)$	$\mathbb{P}\{X_{i,j} = 1\} \approx \sigma(f(U_{Z_i}, U_{Z_j}; \theta))$

A stochastic blockmodeling extension

- Let $Z_n \in \{1, \ldots, K\}$ denote to which of K clusters/components user n is assigned.
- Let $U_k \in \mathbb{R}^D$ be the features for cluster k.
- Construct entries like:

Matrix factorization	Network modeling
$X_{n,m} \approx f(U_{Z_n}, V_m; \theta)$	$\mathbb{P}\{X_{i,j} = 1\} \approx \sigma(f(U_{Z_i}, U_{Z_j}; \theta))$

- So, reduced N sets of parameters to just K
- ... like clustering the users (rows of the matrix)

A stochastic blockmodeling extension

- Without knowledge of Z_n $\Rightarrow$ infer from data.

A stochastic blockmodeling extension

- Without knowledge of $Z_n \Rightarrow$ infer from data.
- Requires (IMO) a Bayesian approach... Variational inference.

A stochastic blockmodeling extension

- Without knowledge of Z_n $\Rightarrow$ infer from data.
- Requires (IMO) a Bayesian approach... Variational inference.
- Straightforward application: "Variational inference for Dirichlet process mixtures" Blei and Jordan [2006]

A stochastic blockmodeling extension

- Without knowledge of $Z_n \Rightarrow$ infer from data.
- Requires (IMO) a Bayesian approach... Variational inference.
- Straightforward application: "Variational inference for Dirichlet process mixtures" Blei and Jordan [2006]
- Informally, prediction looks like

$$\mathbb{P}\{X^*_{i,j} = 1\} \approx \mathbb{E}_{q(Z)}[\sigma(f(U_{Z_i}, U_{Z_j}; \theta))]$$

$q(Z) \approx p(Z \mid X)$ an approximation to the posterior.

Stick-breaking, mean-field variational inference

Stick-breaking construction:

Let $V_i \sim \text{beta}(1, c)$, $i = 1, 2, \ldots$ and

$$\pi_k = V_k \prod_{\ell=1}^{k-1} (1 - V_\ell), \quad k = 1, 2, \ldots,$$

$$Z_n \mid \pi \sim \text{Discrete}(\pi), \quad n \leq N.$$

Log likelihood is, for example,

$$\sum_{(i,j)} \log p(X_{i,j} \mid f(U_{Z_i}, U_{Z_j}; \theta)) + \log p(Z \mid V) + \log p(V)$$

Stick-breaking, mean-field variational inference

Let q denote a "variational approximation" to the posterior:

$$\begin{aligned} q(V_k) &= \text{beta}(V_k; a_k, b_k), \\ q(Z_n) &= \text{Discrete}(Z_n; \eta_n). \end{aligned}$$

Maximize the following lower bound on the log marginal likelihood

$$\begin{aligned} \log p(X) \geq & \mathbb{E}_{q(Z,V)}\Big[\sum_{(i,j)} \log p(X_{i,j} \mid f(U_{Z_i}, U_{Z_j}; \theta))\Big] \\ & - \mathrm{KL}[q(Z,V) || p(Z,V)] \end{aligned}$$

KL the Kullback–Leibler divergence.

Stick-breaking, mean-field variational inference

Algorithm:

- Initialize q.
- Iterate:
  - Update

$$q(Z_n = k) \propto \exp\Big\{\mathbb{E}_q[\log V_k] + \sum_{\ell \geq k+1} \mathbb{E}_q[\log(1 - V_\ell)] + \mathbb{E}_q\Big[\sum_{(i,j)} \log p(X_{i,j} \mid Z, \{Z_n = k\})\Big]\Big\},$$

  - Take a gradient step

$$\Theta \leftarrow \Theta + \eta \nabla_\Theta \Big\{\mathbb{E}_q\Big[\sum_{(i,j)} \log p(X_{i,j} \mid f(U_{Z_i}, U_{Z_j}; \theta))\Big] - \mathrm{KL}[q(Z, V) || p(Z, V)]\Big\}$$

  some schedule η and all parameters Θ

Stick-breaking, mean-field variational inference

- Easily integrates with gradient-based learning (i.e., use Tensorflow/Torch/etc.)

Stick-breaking, mean-field variational inference

- Easily integrates with gradient-based learning (i.e., use Tensorflow/Torch/etc.)
- Computing gradients requires stochastic approximation
  - Stochastic reparameterizations Salimans and Knowles [2013], Kingma and Welling [2014]
  - Score function estimators with control variates Ranganath et al. [2014], Paisley et al. [2012]

Stick-breaking, mean-field variational inference

- Easily integrates with gradient-based learning (i.e., use Tensorflow/Torch/etc.)

- Computing gradients requires stochastic approximation
  - Stochastic reparameterizations Salimans and Knowles [2013], Kingma and Welling [2014]

  - Score function estimators with control variates Ranganath et al. [2014], Paisley et al. [2012]

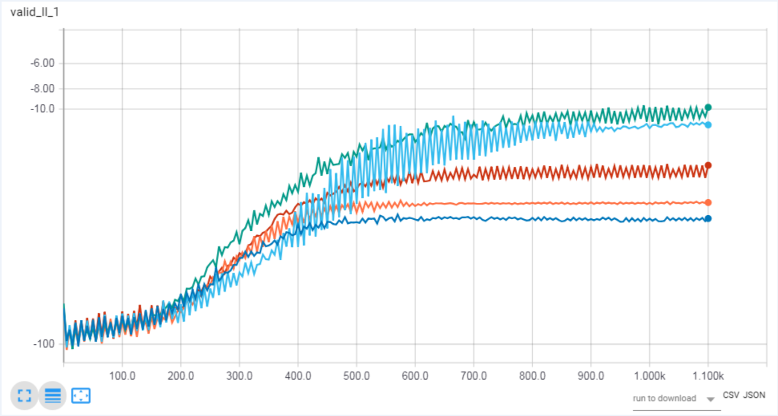
  - Often easy with packages such as Tensorflow Contrib's "distributions"

We can learn the usual cool structure

Some inferred NIPS Coauthorship clusters:

LeCun_Y	Giles_C	Jordan_M	Ferguson_D
Bengio_Y	LeCun_Y	Ghahramani_Z	Jaakola_T
Bottou_L	Liu_S	Bishop_C	Doucet_A
Dayan_P	Zemel_R	Amari_S	Bartlett_P
Frey_B_J	Mueller_P	Chapelle_O	Bartlett_M
Koller_D	Ng_A_Y	Burges_C	Guyon_I
Bishop_C	Opper_M	Edelman_S	Kearns_M
Jackel_L	Pearlmutter_B	Hinton	Burges_C
Graf_H	Rumelhart_D	Buhmann_J	Hinton_G
Doya_K	Poggio_T	Johnson_D	Jung_T

But it's not without pain points...



Things we've found

- With variational inference, hidden layers unnecessary.
  (Movielens 100K, comparing with Dziugaite and Roy [2015].)

Things we've found

- With variational inference, hidden layers unnecessary. (Movielens 100K, comparing with Dziugaite and Roy [2015].)

- Regularizing nnet weights important, inputs/features not so.

$$\mathbb{P}\{X_{n,m} = 1\} \approx \sigma(f(U_n, V_m; \theta))$$

Things we've found

- With variational inference, hidden layers unnecessary.
  (Movielens 100K, comparing with Dziugaite and Roy [2015].)

- Regularizing nnet weights important, inputs/features not so.

$$\mathbb{P}\{X_{n,m} = 1\} \approx \sigma(f(U_n, V_m; \theta))$$

  - What does this mean for Bayesian inference? (!!)

Things we've found

- With variational inference, hidden layers unnecessary. (Movielens 100K, comparing with Dziugaite and Roy [2015].)

- Regularizing nnet weights important, inputs/features not so.

$$\mathbb{P}\{X_{n,m} = 1\} \approx \sigma(f(U_n, V_m; \theta))$$

  - What does this mean for Bayesian inference? (!!)

- Some evidence layers useful when U contains side information.
  - E.g., Movie genre in Movielens 100K
  - Author word counts across papers in NIPS dataset

Some conclusions

- Lots to be desired in current deep learning research

Some conclusions

- Lots to be desired in current deep learning research
  - ...But it has its conveniences...

Some conclusions

- Lots to be desired in current deep learning research
  - ...But it has its conveniences...
  - ...And produces some interesting suggestions for the Bayesian perspective.

Some conclusions

- Lots to be desired in current deep learning research
  - ...But it has its conveniences...
  - ...And produces some interesting suggestions for the Bayesian perspective.
- With Bayesian inference on your deep nnet...

Some conclusions

- Lots to be desired in current deep learning research
  - ...But it has its conveniences...
  - ...And produces some interesting suggestions for the Bayesian perspective.

- With Bayesian inference on your deep nnet...
  - You may find all that structure isn't necessary...

Some conclusions

- Lots to be desired in current deep learning research
  - ...But it has its conveniences...
  - ...And produces some interesting suggestions for the Bayesian perspective.

- With Bayesian inference on your deep nnet...
  - You may find all that structure isn't necessary...
  - ...until you add data (not parameters).

Some conclusions

- Lots to be desired in current deep learning research
  - ...But it has its conveniences...
  - ...And produces some interesting suggestions for the Bayesian perspective.
- With Bayesian inference on your deep nnet...
  - You may find all that structure isn't necessary...
  - ...until you add data (not parameters).
- I wish we focused more on (scalable) MCMC inference with deep learning architectures.

D. M Blei and M. I. Jordan. Variational inference for Dirichlet process mixtures. Bayesian Analysis, 1(1): 121–143, 2006.

G. K. Dziugaite and D. M. Roy. Neural network matrix factorization. arXiv preprint arXiv:1511.06443, 2015.

D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In ICLR, 2014.

J. Lee, C. Heaukulani, Z. Ghahramani, L. F. James, and S. Choi. Bayesian inference on random simple graphs with power law degree distributions. In Proceedings of the 34th International Conference on Machine Learning, 2017.

J. Lloyd, P. Orbanz, Z. Ghahramani, and D. M. Roy. Random function priors for exchangeable arrays with applications to graphs and relational data. In Advances in Neural Information Processing Systems 25, 2012.

P. Orbanz and D. M. Roy. Bayesian models of graphs, arrays and other exchangeable random structures. IEEE transactions on pattern analysis and machine intelligence, 37(2):437–461, 2015.

J. Paisley, D. M. Blei, and M. I. Jordan. Variational Bayesian inference with stochastic search. In ICML, 2012.

R. Ranganath, S. Gerrish, and D. M. Blei. Black box variational inference. In AISTATS, 2014.

T. Salimans and D. A. Knowles. Fixed-form variational posterior approximation through stochastic linear regression. Bayesian Analysis, 8(4):837–882, 2013.