Partial VAE for Hybrid Recommender System

Chao Ma [*,1], Wenbo Gong [*,1], José Miguel Hernández-Lobato[1,2,3],
Noam Koenigstein[4], Sebastian Nowozin[2], and Cheng Zhang[2]

[1]University of the Cambridge, UK [2]Microsoft Research, Cambridge, UK
[3]Allan Turing Institute, UK [4]Microsoft R&D, Israel

Abstract

We propose a novel hybrid recommender system method that treats missing data in a principled manner and that uses amortized inference for fast predictions. We name this method, the Partial Variational Autoencoder (p-VAE). P-VAE uses a novel probabilistic generative model to handle varying numbers of user ratings in a principled way. Using the proposed amortized partial inference technique in p-VAEs, learning and inference can be efficiently performed by minimizing the so-called partial variational upper bound, without making ad-hoc assumptions on the values of missing ratings. Empirical experiments on the MovieLens dataset demonstrate the state-of-the-art performance of our method for movie recommendations.

1 Introduction

Recommender Systems Recommender systems (RS) are one of the most extensively studied, wide-spread machine learning application areas in a variety of real-world scenarios. The main purpose of a recommender system is to model the user's preferences (through ratings, etc.) on items to make recommendations on unseen ones, based on historical data of users and items. According to the type of data that is used, recommender systems can be categorized into three different types [1]: vanilla collaborative filtering (CF, based on past user-item interactions), content-based systems (based on meta information such as user profile and item content/features), and hybrid recommender systems (or content-based collaborative filtering, based on both types of information) [2]. Many of the RS approaches are still dominated by linear models [13, 20] due to their efficiency and simplicity.

Related recent works and challenges Recently, efforts have been made to generalize the linear RS models by introducing the representational power of deep learning models [3, 5, 8, 11, 12, 18, 22, 24], which can also be categorized into three categories: non-linear generalization of CF [3, 5, 8, 12, 18, 24]; non-linear item content feature representation with deep learning [11, 21, 22], and non-linear hybrid methods [2]. Despite the success of introducing deep learning into recommender systems, a central challenge remains: all of the previous methods ignore uncertainties in missing data (i.e., unrated items), and implicitly assume a constant-value imputation for those missing entries (e.g. zero imputing in [9, 12, 17, 18, 24]). This assumption is often severely violated, since the user-item interaction data are usually very large but sparsely observed, making it difficult to train CF-based methods. Additionally, few non-linear CF methods are able utilize the content information.

Contributions In this work, we address the aforementioned issue with our novel non-linear extension of CF based on the proposed Partial Variational Autoencoder (p-VAE). First of its kind, p-VAE is a principled and scalable probabilistic model capable of intrinsically handling massive missing ratings of variable sizes across users, without relying on ad-hoc assumptions on missing ratings. Additionally, our p-VAE can be easily be adapted as a hybrid RS to encode item content information.

[*]This work was done during his internship at Microsoft Research Cambridge.

[2]CF will be referring to both vanilla CF and the hybrid method for the rest of the paper.

Preprint. Work in progress.

2 Partial VAE (p-VAE) for Hybrid Recommender System

Problem formulation We consider the content-based collaborative filtering (CF) setting. Suppose we have access to the ratings of M items from N users. Let r_{nm} be the rating given by the n^{th} user to the m^{th} item, and $\mathbf{r}_n = \mathbf{r}_n^O \cup \mathbf{r}_n^U$ is the partially observed rating vector for the n^{th} user with observed entries denoted as $\mathbf{r}_n^O$ and missing ones denoted as $\mathbf{r}_n^U$. We further assume that the meta information, i.e. the user profile $\mathbf{u}_n$ and item features $\mathbf{i}_m$ may available (although our methods can also be applied both with and without such meta information). Under this setting, the goal of the RS is to recommend potentially interesting unseen items. This is done based on efficient and accurate predictions of the missing ratings $\mathbf{r}_n^U$ given the observed ratings $\mathbf{r}_n^O$ and the meta information. That is to infer $p(\mathbf{r}_n^U|\mathbf{r}_n^O, \mathbf{u}_n, \{\mathbf{i}_m\}_{1\leq m\leq M})$. For simplicity, we will omit the index n for $\mathbf{r}_n$, and drop $\mathbf{u}_n, \mathbf{i}_m$ whenever the context is clear.

From VAE to partial VAE We consider extending the variational autoencoder (VAE) approach to hybrid CF, since it enables us to overcome the limited capacity of (probabilistic) linear models commonly used in CF. A regular VAE approach uses a generative model (decoder) $p(\mathbf{r}, \mathbf{z}) = p(\mathbf{r}|\mathbf{z})p(\mathbf{z})$ that generates observations $\mathbf{r}$ given latent variables $\mathbf{z}$, and an inference model (encoder) $q(\mathbf{z}|\mathbf{r})$ that infers the latent state $\mathbf{z}$ given fully observed $\mathbf{r}$. Training VAE is very efficient through optimizing a variational bound. However, in our setting of RS, there are a huge number of possible partitions $\{U, O\}$, where the size of observed ratings might vary. This makes classic approaches to train a VAE no longer directly applicable.

Previous works tries to solve this issue by manually imputing the missing $\mathbf{r}^U$ with a constant value [18]. The main drawback of this approach is that it can not differ between missing values and actually observed values. This poses learning difficulties and potential risks of poor uncertainty estimations, since rating data is typically extremely sparsely observed. Last but not the least, the parameterization of the encoder neural network is inefficient and does not make use of the sparsity of rating data.

To address this problem in a principled manner, we propose the partial VAE (p-VAE) collaborative filtering method. Similar to a VAE, P-VAE assumes a latent variable model $p(\mathbf{r}) = \int_{\mathbf{z}} p(\mathbf{r}|\mathbf{z})p(\mathbf{z})$, where $\mathbf{z}$ is the latent variable. Notice the factorized structure for $p(\mathbf{r}|\mathbf{z})$, i.e.

$$p(\mathbf{r}|\mathbf{z}) = \prod_i p_i(\mathbf{r}_i|\mathbf{z}). \tag{1}$$

This implies that given $\mathbf{z}$, the observed ratings $\mathbf{r}^O$ are conditionally independent of $\mathbf{r}^U$. Therefore, inferences about $\mathbf{r}^U$ can be reduced to $p(\mathbf{z}|\mathbf{r}^O)$. Once knowledge about $\mathbf{z}$ is obtained, we can draw correct inferences about $\mathbf{r}^U$. To approximate $p(\mathbf{z}|\mathbf{r}^O)$ we introduce an auxiliary variational inference network $q(\mathbf{z}|\mathbf{r}^O)$ and derive the partial variational upper bound,

$$\begin{aligned} D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{r}^O)\|p(\mathbf{z}|\mathbf{r}^O)) &= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{r}^O)}[\log q(\mathbf{z}|\mathbf{r}^O) - \log p(\mathbf{z}|\mathbf{r}^O)] \\ &\leq \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{r}^O)}[\log q(\mathbf{z}|\mathbf{r}^O) - \log p(\mathbf{r}^O|\mathbf{z}) - \log p(\mathbf{z})] \equiv \mathcal{L}_p. \end{aligned} \tag{2}$$

This bound, $\mathcal{L}_p$, depends only on the observation $\mathbf{r}^O$. The size of $\mathbf{r}^O$ could vary between different data points. Next, we develop the *partial inference network* for auxiliary distribution $q(\mathbf{z}|\mathbf{r}^O)$, which takes a set of partially observed ratings $\mathbf{r}^O$ whose length may vary across users or across time.

Amortized Inference with partial observations Inspired by the *Point Net (PN)* approach for point cloud classification [16, 23], we specify the approximate distribution $q(\mathbf{z}|\mathbf{r}^O)$ by a *permutation invariant set function encoding*, given by:

$$\mathbf{c}(\mathbf{r}^O) := g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_{|O|})), \tag{3}$$

where $|O|$ is the number of the observed ratings, $\mathbf{s}_m$ carries the information of the rating and item identity. For example, $\mathbf{s}_m = [\mathbf{e}_m, r_m]$ or $\mathbf{s}_m = \mathbf{e}_m * r_m$. Here, $\mathbf{e}_m$ is the ID vector of the m^{th} item. There are many ways to define $\mathbf{e}_m$ under different settings, such as by using the meta information, or optimizing $\mathbf{e}_m$ from scratch during learning when the meta information is not available. In our experiments, we treat $\mathbf{e}_m$ as the concatenation of the fixed meta information that comes with the item data and the user profile, and a learnable set of identity embeddings. We use $\mathbf{s}_m = \mathbf{e}_m * r_m$ in this work, and show that this setting generalizes naive Zero Imputation (ZI) VAE [14] (see Appendix A)

We use a neural network $h(\cdot)$ to map input $\mathbf{s}$ from $\mathbb{R}^{D+1}$ to $\mathbb{R}^K$, where D is the dimension of each $\mathbf{e}_m$, and r_m is a scalar, K is the latent space size. The Key to the PN structure is the permutation invariant aggregation operation $g(\cdot)$, such as max-pooling or summation. In this way, the mapping $\mathbf{c}(\mathbf{r}^O)$ is invariant to permutations of elements of $\mathbf{r}^O$ and $\mathbf{r}^O$ can have arbitrary length. Finally, the fixed-size code $\mathbf{c}(\mathbf{r}^O)$ is fed into an ordinary amortized inference network, that transforms the code into the statistics of a multivariate Gaussian distribution to approximate $p(\mathbf{z}|\mathbf{r}^O)$. In practice, since the dimension of item feature $\mathbf{i}_m$ often satisfies $D \ll M$, this parameterization of encoder is very efficient compared with typical VAE approaches, which requires a huge $M \times K$ weight matrix.

Note that our proposed method generalizes many existing methods such as [19, 18, 14, 12, 9]. The relationship between our method and those related work are presented in Appendix B.

3 Experiments

Method	Test RMSE
Matrix Factorization Methods	
PMF†	0.883
LLORMA-Global [10]	0.865
BiasMF∗	0.845
NNMF [3]	0.843
User based generative models	
U-RBM∗	0.881
U-Autoencoder	0.870
U-ZI	0.872
U-p-VAE	**0.856**
U-p-VAE(meta)	**0.856**
Item based generative models	
I-RBM∗	0.854
I-Autoencoder	0.847
I-ZI	0.851
I-p-VAE	**0.841**

Table 1: Test RMSE on MovieLens 1M. †: Taken from [3]. ∗: Taken from [18].

MovieLens: experiment setting We apply our p-VAE recommender system to the MovieLens dataset [4]. We choose the Movielens 1M, which contains 1,000,206 rating records of 3,952 movies by 6,040 users. Movielens consists of large and sparsely observed data (around 5% observed ratings). We apply 90%/10% training-test ratio to split the dataset. We compare the baselines from previous work, including Probabilistic matrix factorization[13], Restricted Boltsman Machines[17], AutoRec[18], Biased Matrix factorization[7], Neural Non-negative Matrix Factorization[3], LLORMA[10]. We provide one additional baseline: Zerp imputing VAE (ZI). ZI-VAE is the degenerated case of p-VAE that uses ad-hoc zero imputing for unobserved ratings (also used in other AE-based baselines), which is as an VAE extension of [9].

We group these baselines into three categories: matrix factorization methods, user-based generative models, and item based generative models. In user based models, the ratings from each user are treated as a training example. On the contrary, in item based models, the ratings of the same item are treated as a training example. Both U-p-VAE and I-p-VAE has one hidden layer with 100 and 500 dimensional hidden units, respectively. In include two versions of U-p-VAE, with or without meta features of users and items. In particular, for U-p-VAE, we further improve our results by running Hamiltonian Monte Carlo (HMC) [15] on latent space z with 20 leapfrog steps and 10 samples, following [6]. Our methods are trained for 400 epochs using Adam with a learning rate of 0.001, with batch size 100.

Results Table 1 compares p-VAE with baselines in terms of RMSE, where we have directly quoted their results since the experimental setting in these works is the same as ours. The exception is autoencoders where we report results from our implementation, since it is a special case of our method. Overall, in both user-based categories and item-based categories, p-VAE outperforms all baselines. Note that, our best performing method, the item based p-VAE (I-p-VAE) also outperforms the all matrix factorization methods. The results of ZI-VAE are only slightly better (or worse in item based version) than previous non-probabilistic Autoencoder baselines, which explains that the main performance gain of p-VAE comes from the better uncertainty estimation of missing data provided by amortized partial inference component. This confirms our conjecture in the previous section. The meta information does not help p-VAE to achieve better results due to that the proposed infernece method is already accurate enough without the help of the simple meta data.

4 Conclusion

We proposed the Partial VAE (p-VAE) as a novel and highly accurate generative CF model for recommander system. The p-VAE addresses a number of issues in regular CF settings with autoencoder approaches. In particular, the p-VAE provides a principled approach for handling large amounts of missing values in user-item rating data. For future work, we will continue to improve our model and investigate its performance via more sophisticated evaluation methods on more data sets.

References

[1] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge & Data Engineering*, (6):734–749, 2005.

[2] Yifan Chen and Maarten de Rijke. A collective variational autoencoder for top-n recommendation with side information. In *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems*, pages 3–9. ACM, 2018.

[3] Gintare Karolina Dziugaite and Daniel M Roy. Neural network matrix factorization. *arXiv preprint arXiv:1511.06443*, 2015.

[4] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):19, 2016.

[5] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, pages 173–182. International World Wide Web Conferences Steering Committee, 2017.

[6] Matthew D Hoffman. Learning deep latent gaussian models with markov chain monte carlo. In *International Conference on Machine Learning*, pages 1510–1519, 2017.

[7] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 263–272. Ieee, 2008.

[8] Giannis Karamanolakis, Kevin Raji Cherian, Ananth Ravi Narayan, Jie Yuan, Da Tang, and Tony Jebara. Item recommendation with variational autoencoders and heterogeneous priors. In *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems*, pages 10–14. ACM, 2018.

[9] Oleksii Kuchaiev and Boris Ginsburg. Training deep autoencoders for collaborative filtering. *arXiv preprint arXiv:1708.01715*, 2017.

[10] Joonseok Lee, Seungyeon Kim, Guy Lebanon, and Yoram Singer. Local low-rank matrix approximation. In *International Conference on Machine Learning*, pages 82–90, 2013.

[11] Xiaopeng Li and James She. Collaborative variational autoencoder for recommender systems. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 305–314. ACM, 2017.

[12] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. Variational autoencoders for collaborative filtering. *arXiv preprint arXiv:1802.05814*, 2018.

[13] Andriy Mnih and Ruslan R Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2008.

[14] Alfredo Nazabal, Pablo M Olmos, Zoubin Ghahramani, and Isabel Valera. Handling incomplete heterogeneous data using vaes. *arXiv preprint arXiv:1807.03653*, 2018.

[15] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11):2, 2011.

[16] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[17] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798. ACM, 2007.

[18] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*, pages 111–112. ACM, 2015.

[19] David H Stern, Ralf Herbrich, and Thore Graepel. Matchbox: large scale online bayesian recommendations. In *Proceedings of the 18th international conference on World wide web*, pages 111–120. ACM, 2009.

[20] Xiaoyuan Su and Taghi M Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009, 2009.

[21] Hao Wang, Xingjian Shi, and Dit-Yan Yeung. Relational stacked denoising autoencoder for tag recommendation. In *AAAI*, pages 3052–3058, 2015.

[22] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1235–1244. ACM, 2015.

[23] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in Neural Information Processing Systems*, pages 3391–3401, 2017.

[24] Yin Zheng, Bangsheng Tang, Wenkui Ding, and Hanning Zhou. A neural autoregressive approach to collaborative filtering. In *International Conference on Machine Learning*, pages 764–773, 2016.

Appendix

A Zero imputing as a Point Net

Here we present how the zero imputing (ZI) and PointNet (PN) approaches relate.

Zero imputation with inference net In ZI, the natural parameter of λ (e.g., Gaussian parameters in variational autoencoders) is approximated using the following neural network:

$$f(\mathbf{x}) := \sum_{l=1}^{L} w_l^{(1)} \sigma(\mathbf{w}_l^{(0)} \mathbf{x}^T), \tag{4}$$

where L is the number of hidden units, $\mathbf{x}$ is the input image with x_i be the value of the i^{th} pixel. To deal with partially observed data $\mathbf{x} = \mathbf{x}_o \cup \mathbf{x}_u$, ZI simply sets all $\mathbf{x}_u$ to zero, and use the full inference model $f(\mathbf{x})$ to perform approximate inference.

PointNet parameterization The PN approach approximates the natural parameter λ by a permutation invariant set function

$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)),$$

where $\mathbf{s}_i = [x_i, \mathbf{e}_i]$, $\mathbf{e}_i$ is the I dimensional embedding/ID/location vector of the i^{th} pixel, $g(\cdot)$ is a symmetric operation such as max-pooling and summation, and $h(\cdot)$ is a nonlinear feature mapping from $\mathbb{R}^{I+1}$ to $\mathbb{R}^K$ (we will always refer h as *feature maps*). In the current version of the partial-VAE implementation, where Gaussian approximation is used, we set $K = 2H$ with H being the dimension of latent variables. We set g to be the element-wise summation operator, i.e. a mapping from $\mathbb{R}^{KO}$ to $\mathbb{R}^K$ defined by:

$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)) = \sum_{i \in O} h(\mathbf{s}_i).$$

This parameterization corresponds to products of multiple Exp-Fam factors $\prod_{i \in O} \exp\{-\langle h(\mathbf{s}_i), \Phi \rangle\}$.

From PN to ZI To derive the PN correspondence of the above ZI network we define the following PN functions:

$$h(\mathbf{s}_i) := \mathbf{e}_i * x_i$$
$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)) := \sum_{k=1}^{I} \theta_k \sigma(\sum_{i \in O} h_k(\mathbf{s}_i)),$$

where $h_k(\cdot)$ is the k^{th} output feature of $h(\cdot)$. The above PN parameterization is also permutation invariant; setting $L = I$, $\theta_l = w_l^{(1)}$,$(\mathbf{w}_l^{(0)})_i = (\mathbf{e}_i)_l$ the resulting PN model is equivalent to the ZI neural network.

Generalizing ZI from PN perspective In the ZI approach, the missing values are replaced with zeros. However, this ad-hoc approach does not distinguish missing values from actual observed zero values. In practice, being able to distinguish between these two is crucial for improving uncertainty estimation during partial inference. One the other hand, we have found that PN-based partial VAE experiences difficulties in training. To alleviate both issues, we proposed a generalization of the ZI approach that follows a PN perspective. One of the advantages of PN is setting the *feature maps* of the unobserved variables to zero instead of the related weights. As discussed before, these two approaches are equivalent to each other only if the factors are linear. More generally, we can parameterize the PN by:

$$h^{(1)}(\mathbf{s}_i) := \mathbf{e}_i * x_i$$
$$h^{(2)}(h_i^{(1)}) := NN_1(h_i^{(1)})$$
$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)) := NN_2(\sigma(\sum_{i \in O} h_k^{(2)}(h_i^{(1)}))),$$

where NN_1 is a mapping from $\mathbb{R}^I$ to $\mathbb{R}^K$ defined by a neural network, and NN_2 is a mapping from $\mathbb{R}^K$ to $\mathbb{R}^{2H}$ defined by another neural network.

B Connections to related work

In this section, we briefly present the connections of our proposed model to some of the related works. We show that ours is a more general and deeper model.

From AutoRec to Partial VAE AutoRec [18] (and Deep AutoRec [9])is a auto-encoder based collaborative filtering method. In particular, AutoRec can be used for modeling user's behavior or item's behavior, which are referred as U-AutoRec or I-AutoRec respectively. In the following, we will assume the U-AutoRec as the only difference between these two is the input observations. The encoder network will take all ratings (observed and unobserved) as input, where the unobserved data are marked by 0. This encoder will then map the user observations into a latent vector z which can be seen as the latent representation of the user. Then, this user representation is transformed by the decoder network to impute the missing ratings as well as reconstructing the observed ones. This encoding-decoding procedure is deterministic and trained by minimizing the RMSE between the reconstruction and observed ratings.
The encoder of partial VAE is a non-linear generalization of the AutoRec. According to Appendix A, Partial VAE can be seen as ZI through multiplying observed values by item embedding e_i, which is represented as $h^{(1)}(s_i)$. In AutoRec, $h^{(1)}(s_i)$ will be summed over all observed i, whereas partial VAE will first map each factor $h^{(i)}(s_i)$ through a non-linear mapping $NN_1(\cdot)$, followed by a pooling function $g(\cdot)$ (e.g. summation).
Another distinction is the training objective. In AutoRec, the latent representation z is a free-parameter which are obtained by minimizing the RMSE. On the other hand, partial VAE adopts a Bayesian treatment which places a prior distribution over the random variable z. Training objective is maximizing partial variational lower bound. Thus, the partial VAE can be regarded as the nonlinear, variational generalization to AutoRec.

Connections to HI-VAEs Heterogeneous-Incomplete VAE (HI-VAE)[14] is a recent concurrent work that tries to modify the original VAE to handle heterogenous (mixed continuous and discrete) or incomplete (with missing data at random). Under missing data, HI-VAE also propose to use partial variational bound for learning and inference. However, their inference method tries to handle missing data with zeros. As discussed previously, the main drawback of this approach is that it can not differ between missing values and actually observed values. This will also result in the degeneration of approximate posterior, which means that the VAE model tends to reconstruct unobserved data with the single best solution that ignores the data uncertainty. Essentially, as shown in Appendix A, this approach can be viewed as a special case of PoinNet parameterization of encoder with a specific choice of aggregation function. In a sense, HI-VAE is the variational version of AutoRec.

Relations to Matchbox [19] Matchbox is an effective model that models the user-item interaction matrix by assuming that the ratings can be factorized into the inner product of user trait vector and item trait vector. Each trait vector is then modelled by the linear transformation on user/item meta features. Matchbox hence generalized the SVD matrix factorization approach, and a fully Bayesian treatment is adopted to the model, where approximate inference is implemented by variational message passing and expectation propagation. Matchbox further generalized this approach to handle different user feedback models and model dynamical drift over time. From the fundamental model point of view, Matchbox is essentially a Bayesian (w.r.t weights) linear variational autoencoder, while our partial VAE is non-Bayesian w.r.t decoder-encoder weights. The missing data problem during inference phase is avoided since non-amortized variational inference is used, therefore approximate messages must be re-optimized whenever each of the rating configurations changes, while in partial VAE, these information are amortized into a single model that can be applied to all missing patterns and user-item configurations.