Toward Implicit Sample Noise Modeling: Deviation-driven Matrix Factorization

Guang-He Lee  
Dept. of Computer Science  
National Taiwan Univ., Taiwan  
r04922045@csie.ntu.edu.tw

Shao-Wen Yang  
Intel Labs  
Intel Corporation  
shao-wen.yang@intel.com

Shou-De Lin  
Dept. of Computer Science  
National Taiwan Univ., Taiwan  
sdlin@csie.ntu.edu.tw

Abstract

The objective function of a matrix factorization model usually aims to minimize the average of a regression error contributed by each element. However, given the existence of stochastic noises, the implicit deviations of sample data from their true values are almost surely diverse, which makes each data point not equally suitable for fitting a model. In this case, simply averaging the cost among data in the objective function is not ideal. Intuitively we would like to emphasize more on the reliable instances (i.e., those contain smaller noise) while training a model. Motivated by such observation, we derive our formula from a theoretical framework for optimal weighting under heteroscedastic noise distribution. Specifically, by modeling and learning the deviation of data, we design a novel matrix factorization model. Our model has two advantages. First, it jointly learns the deviation and conducts dynamic reweighting of instances, allowing the model to converge to a better solution. Second, during learning the deviated instances are assigned lower weights, which leads to faster convergence since the model does not need to overfit the noise. The experiments are conducted in clean recommendation and noisy sensor datasets to test the effectiveness of the model in various scenarios. The results show that our model outperforms the state-of-the-art factorization and deep learning models in both accuracy and efficiency.

1 Introduction

Factorization models have been widely and successfully applied to tasks such as missing data imputation and user-item recommendation. The objective function of a factorization model usually aims to minimize an average of a regression cost contributed by each data point (Koren, Bell, and Volinsky 2009). However, in general, data can be noisy, and each data point may be contaminated by different level of noise. As long as the noise comes from certain continuous probability distribution, almost surely the sample data will deviate diversely from their expected (noiseless) values. Namely, the existence of noise will cause certain instances (i.e., those with larger noise) to be less trust-able than others.

Given certain level of trustworthiness for each data point, the instance-weighting strategy is widely adopted (Koren, Bell, and Volinsky 2009). However, such weighting scheme has its limitation since it requires prior knowledge about the trustworthiness of each data point. A related study may be anomaly/outlier detection (Xiong, Chen, and Schneider 2011), whose goal is to identify a few (salient) noisy instances from a majority of noiseless data. Our model, on the other hand, assumes the noise is implicitly embedded in every data point, while some contain more and some less. This is particularly true for sensor data where measurement errors always exist and different sensor has different degree of errors. Therefore, we argue that a learning model should consider different level of noise in each data point instead of simply detect and remove noisy data from the training set.

To address the above issues, we propose a deviation-driven modeling framework on matrix factorization (MF) to model and learn the deviation, i.e., implicit sample noise. Note that we emphasize the implicit and ubiquitous nature of sample noise, which is possibly unobservable and universally existent for data with stochastic noise. As a result, though the concept of modeling noise is closely related to robust methods (Meng and De La Torre 2013), our model should be treated as a general method.

Specifically, as the concept of deviation is similar to heteroscedasticity, we derive a deviation-driven loss function from a theoretical framework for optimal weighting of weighted linear regression under heteroscedastic noise distribution. By learning the deviation from data, we relieve one less-realistic assumption of the theory that the level of noise is known. The resulting formulation is equivalent to a Gaussian likelihood with heteroscedastic variance. The formulation exhibits two advantages. First, it jointly learns the deviation and conducts dynamic reweighting of instances, allowing the model to converge to a better solution. Second,
during learning the deviated instances are assigned lower weights, which leads to faster convergence since the model does not need to overfit the noise.

We then apply the deviation-driven loss to MF as Deviation-driven Matrix Factorization (DMF). In contrast to traditional sparse noise structure that assumes only limited amount of noise, a novel low-rank structure for modeling the deviation, i.e., variance in the Gaussian likelihood, is adopted to model the ubiquitous nature of deviation and combat overfitting. The illustration of DMF vs. traditional Probabilistic Matrix Factorization is shown in Figure 1.

Several experiments are conducted in both noisy sensor and clean recommendation datasets. The “clean” datasets are especially important to verify the existence of implicit noise, which is not directly observable and can be implied from the superiority of noise modeling over typical methods. In both scenario, our model demonstrates better accuracy and faster training than the state-of-the-art solutions.

2 The Deviation-driven Model

2.1 The Diverse Deviation Phenomenon

In this work, we assume each observed data instance \((x_i, y_i)\) comes from the addition of clean data instance \((x_i, y_i^\prime)\) and sample noise \(\epsilon_i\), sampled from a continuous random variable \(\epsilon_i\). Besides, the expectation of noise is assumed to be zero. Formally speaking, denoting \(\sim\) as a sampling process, we have

\[
(x_i, y_i = y_i^\prime + \epsilon_i), \epsilon_i \sim \epsilon_i, \mathcal{E}[\epsilon_i] = 0. \tag{1}
\]

For example, in Figure 2 we conduct a random experiment to generate sample data points with \(y_i^\prime = x\) and \(\epsilon_i \sim \mathcal{N}(0, 0.01)\) as circle markers, and correspondingly expected data points as square marker. Even with homogeneous noise, the sample data still deviated from their mean diversely. Theoretically, given any two continuous noises \(\epsilon_i\) and \(\epsilon_j\) with probability density functions \(p(x)\) and \(q(x)\), it is almost surely that the two noises are diverse, as \(Pr(\epsilon_i \neq \epsilon_j) = \int_{-\infty}^{\infty} p(x)(1 - \int_{-\infty}^{\infty} q(y) dy) dx = \int_{-\infty}^{\infty} p(x) dx = 1\).

Accordingly, as each data point brings different level of confidence for describing the original distribution, to conduct a fair treatment for data, it is important to consider the diverse deviation for individual data point.

To address the above issue, an intuitive method is to conduct a weighting scheme for each data point. We begin our derivation from a theoretical framework on a linear model, weighted linear regression (WLR), which solves the following optimization task:

\[
w = \arg \min_w \frac{1}{N} \sum_{i=1}^{N} \frac{(y_i - w^T x_i)^2}{W_i}, \tag{2}
\]

where \(W_i\) is the weighting for instance \((x_i, y_i)\) and \(w\) is the linear model to optimize. Given that data are generated by a linear model \(w^\prime\) as \(y_i = y_i^\prime + \epsilon_i = (w^\prime)^T x_i + \epsilon_i\), it is reasonable to minimize \(E[(y - w^\prime)^2]\) as \(\text{Var}(w)\) with respect to \(W_i\), since \(E[y_i] = y_i^\prime\) and thus \(\text{Var}[w] = w^\prime^2\). Accordingly, from the generalized Gauss-Markov theorem (Shalizi forthcoming), the optimal \(W_i\) is \(\text{Var}(\epsilon_i)\), the variance of noise.

\[
W_i = \text{Var}(\epsilon_i) = E[\epsilon_i - E[\epsilon_i]]^2 = E[\epsilon_i^2] = \epsilon_i^2. \tag{3}
\]

In Figure 2, we compare linear regression (LR) with DLR. The superior performance of DLR is clear for fitting the clean (expected) data. The simple experiment demonstrates the promising efficacy of incorporating deviation, which motivates modeling deviation in more realistic scenario.

2.2 Deviation-driven Squared Error

To extend DLR for MF on a matrix \(R\) with size \(N_1 \times N_2\), we can correspond the label \(y_i\) with matrix element \(R_{ij}\) and the label prediction \(w^T x_i\) with matrix prediction \(\bar{R}_{ij}\). However, the weight \(W_i = \epsilon_i^2\) is unknown in practice. Thus, certain strategies to learn the deviation are needed. Here we propose to use a positive estimation \(\hat{\sigma}_{ij}^2\) to model the deviation \(\epsilon_i^2\), together with a deviation regularizer \(\Psi\) on the value of \(\hat{\sigma}_{ij}^2\) to prevent it from growing unlimitedly, since a larger \(\hat{\sigma}_{ij}^2\) leads to smaller cost for data point \(R_{ij}\). The final deviation-driven squared error objective function is shown in (4).

\[
\min \frac{1}{N_1 N_2} \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} \frac{(R_{ij} - \bar{R}_{ij})^2}{\hat{\sigma}_{ij}^2} + \Psi(\hat{\sigma}_{ij}^2). \tag{4}
\]

Since there are two variables to estimate, \(\bar{R}_{ij}\) and \(\hat{\sigma}_{ij}^2\), we will use (main) prediction to describe \(\bar{R}_{ij}\), and deviation estimation to describe \(\hat{\sigma}_{ij}^2\) throughout the paper.

If adopting a natural log deviation regularizer \(\Psi = \ln\), it is not hard to see that the formula (4) is equivalent to the maximization of Gaussian likelihood as,

\[
\max \prod_{i=1}^{N_1} \prod_{j=1}^{N_2} \frac{1}{\sqrt{2\pi} \hat{\sigma}_{ij}} e^{-\frac{(R_{ij} - \bar{R}_{ij})^2}{2\hat{\sigma}_{ij}^2}}. \tag{5}
\]
The regularization term $\Psi(\hat{\sigma}^2_{ij}) = \ln(\hat{\sigma}^2_{ij})$ in (4) corresponds to the $\hat{\sigma}^2_{ij}$ in the denominator in (5).

The representation provides a physical meaning of deviation as the variance, which is the theoretically optimal weighting in the generalized Gauss-Markov theorem [Shalizi forthcoming], in a Gaussian likelihood. Moreover, contrasting most conventional modeling for Gaussian likelihood with shared variance [Agarwal and Chen 2010; Mnih and Salakhutdinov 2007; Salakhutdinov and Mnih 2008], the deviation-driving modeling represents Gaussian likelihood in a per-instance variance manner.

2.3 Deviation-driven Matrix Factorization

Matrix Factorization Given a matrix $R$, MF reconstruct it as matrix $\hat{R}$ by low rank matrices $U$ and $V$ with latent dimension $D$. A widely used variant (Chen et al. 2011) is to add some bias and scalar terms to the factorization as follows:

$$\hat{R}_{ij} = U^T_i V_j + u_i + v_j + \mu. \quad (6)$$

$u_i$ and $v_j$ are biases, which are scalars to be learned, for user $i$ and item $j$, respectively. $\mu$ is the average value for observed elements of $R$ in the training data. In the remainder of the paper, $u_i$, $v_j$, and $\mu$ are omitted for clarity purpose.

Deviation-driven Modeling Here we propose the idea of Deviation-driven Matrix Factorization (DMF). Conventionally, the data can be modeled by a Gaussian likelihood with mean constructed by MF. To adapt MF to a deviation-driven model, we have to further model the variance in the Gaussian likelihood. The key concern lies in the choice of variance structure. Given the ubiquitous nature of noise, a sparse structure is unsuitable and it is necessary to build a prediction model for every element on the matrix. In order to regularize the capability of such prediction, a low-rank structure is desirable. Consequently, we can adapt another MF to construct the variance $\sigma^2_{ij}$ by deviation latent factors, $P_i$ and $Q_j$.

Since variance has a non-negative constraint, we can impose a non-negative prior on deviation latent factors, thus the inner product of non-negative vectors is also non-negative. In addition, to avoid a zero value of $\sigma^2_{ij}$ that leads to infinite objective value, we add a small positive value $\Delta \sigma^2$ as (7):

$$\sigma^2_{ij} = P_i^T Q_j + \Delta \sigma^2. \quad (7)$$

To impose a non-negative prior, we select exponential distribution. The usage of exponential prior leads to sparse solution (Gopalan, Hofman, and Blei 2013), which is favorable for its efficiency in computation and storage. Accordingly, the prior distribution of deviation latent factor results in [8]:

$$p(P_i|\lambda_P) = \mathcal{EXP}(P_i|\lambda_P I), \ p(Q_j|\lambda_Q) = \mathcal{EXP}(Q_j|\lambda_Q I). \quad (8)$$

With Gaussian prior on mean latent factors, the mean prior and likelihood of DMF can be represented as the follows, with $l_{ij}$ being a binary indicator of the availability of $R_{ij}$ in training data.

$$p(U_i|\sigma_U^2) = \mathcal{N}(U_i|0, \sigma_U^2 I), \ p(V_j|\sigma_V^2) = \mathcal{N}(V_j|0, \sigma_V^2 I), \quad (9)$$

$$p(R_{ij}|U_i, V_j, P_i, Q_j) = \mathcal{N}(R_{ij}|\hat{R}_{ij}, \sigma^2_{ij})|^{l_{ij}}. \quad (10)$$

The log posterior distribution of DMF is given as the following form,

$$- \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} I_{ij}(R_{ij} - U_i^T V_j)^2/2(P_i^T Q_j + \Delta \sigma^2) - \sum_{i=1}^{N_1} (U_i^T U_i/2\sigma_U^2 + \lambda_P P_i^T I)$$

$$- \sum_{j=1}^{N_2} V_j^T V_j/2\sigma_V^2 + \lambda_Q Q_j^T I - D/2N_1\ln(\sigma_U^2)$$

$$- D/2N_2\ln(\sigma_V^2) + DN_1\ln(\lambda_P) + DN_2\ln(\lambda_Q) - 1/2 \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} I_{ij} \ln(P_i^T Q_j + \Delta \sigma^2) + \text{const},$$

subject to $P_{ik}, Q_{jk} \geq 0, \ i \in [1, N_1], j \in [1, N_2]. \quad (11)$

The non-negative constraint is given under exponential distribution. Note that the minimization for the prediction error $(R_{ij} - U_i^T V_j)^2$ is controlled by the deviation $\sigma^2_{ij}$, which is also learned through factorization $P_i^T Q_j$.

2.4 Learning DMF

To learn DMF, one plausible strategy is to resort to stochastic gradient ascent (SG) for maximizing the log posterior distribution (MAP) since the objective function is differentiable. However, with the non-negative constraint on deviation latent factors, learning $P$ and $Q$ is not trivial. The constraint results in non-negative MF with the deviation latent factors. Fortunately, there have been several methods proposed for such constraint (Lin 2007), while we exploit a simple yet effective method called the projected gradient method. That is, every time a single step SG is done, all negative results in deviation latent factors are replaced with 0.

It is not hard to realize that the SG learning of DMF is linear to the number of observed data. That is, $O(\# \text{of epochs} \times \# \text{of observed data} \times D)$. Moreover, the space complexity is $O((N_1 + N_2) \times D)$. Both time and space complexity are the same as conventional MF model. In our experiment, the deviation latent factors achieve at least 33% sparsity through exponential prior. Finally, DMF can be considered as a joint learning machine to learn both mean and deviation, whose advantage will be discussed in the next section.

3 Discussion

In this section, we provide some insights of deviation-driven learning to distinguish its roles in the learning process.

3.1 Deviation as Learning Weight Scheduling

In literature, several methods have been proposed for the per-coordinate learning rate scheduling on first order gradient descent learning [Duchi, Hazan, and Singer 2011]
Table 1: Statistics of recommender system (RS) and Internet of Things (IoT) Datasets

|                  | MovieLens1M | MovieLens10M | Netflix | IoT        | London | Berkeley |
|------------------|-------------|--------------|---------|------------|--------|----------|
| # of users       | 6,040       | 69,878       | 480,189 |            |        |          |
| # of movies      | 3,706       | 10,677       | 17,770  |            |        |          |
| # of locations   | -           | -            | -       | 15         | 58     |          |
| # of sensors     | -           | -            | -       | 19         | 4      |          |
| # of timestamps  | -           | -            | -       | 7,479      | 46,613 |          |
| # of data        | 1M          | 10M          | 100M    | 476,638    | 9,158,515 |          |
| sparsity         | 95.53%      | 98.66%       | 98.82%  | 77.64%     | 15.31% |          |

4.1 DMF for Recommendation

Experiments are conducted on three standard movie recommendation datasets: MovieLens1M, MovieLens10M, and Netflix dataset. Each dataset can be represented by a matrix \( R \): the two dimensions denote user and movie, respectively. The statistics is shown in the left of Table 1.

The experiments are conducted using the protocol proposed in the state-of-the-art solutions (Lee et al. 2013; Sedhain et al. 2015). Specifically, for each dataset, we produce a random 90% and 10% split of data for training and testing, and repeat 5 times to generate 5 different splits. Among the training data, 10% of them are used as validation set for parameter tuning. For testing data without user or movie observed in training data, a default rating 3 is predicted by convention (Sedhain et al. 2015). We run our experiments in each split and report the average root mean squared error (RMSE) on testing data.

We compare DMF with following collaborative filtering (CF) methods:

- **Restricted Boltzmann Machines (RBM)** (Chen et al. 2011). A widely used generative model for CF.
- **Biased Matrix Factorization (Biased MF)**. A simple yet successful method for recommendation as introduced in (9), the non-deviation-driven counterpart of DMF.
- **Robust Collaborative Filtering (RCF)** (Mehta, Hofmann, and Nejdl 2007). A robust weighted MF model applying Huber M-estimator (Huber 2011) to iteratively estimate the weighting during optimization. \( L_2 \) regularization is added for better generalization performance.
**Robust Bayesian Matrix Factorization (RBMF)** \cite{lakshminarayanan2011robust}. A robust MF adopting a rank-1 structure for noise and \textsuperscript{1}Student-t prior on mean latent factors.

**Local Low-Rank Matrix Approximation (LLORMA)** \cite{lee2013local}. The state-of-the-art matrix approximation model assuming the target matrix is locally low-rank.

**AutoRec** \cite{sedhain2015autorec}. The state-of-the-art deep learning recommendation model based on Auto-Encoder.

We also conduct experiments for the state-of-the-art anomaly detection method, Direct Robust MF (DRMF) \cite{xiong2010directrobust}, which imposes an upper-bound on the cardinality of detected noise set to guarantee the sparsity of noise. Since DRMF is designed for anomaly detection in full matrix, Singular Value Decomposition is used for factorization. We make a generalization for CF by adopting Biased MF for factorization. We make a generalization for CF by adopting Biased MF for factorization. \textsuperscript{1}10\%, 5\%, 1\% of the training data as the upper-bound are tested in MovieLens1M and MovieLens10M data, but in all the setting the training and testing performance of DRMF decrease at almost every iteration, thus is not suitable for comparison.

AdaGrad learning rate scheduling \cite{duchi2011adaptive} is used for training DMF. Parameters are fine-tuned through validation. Detail parameters are omitted due to space limitation. The complete parameter setting and implementation can be found at \footnote{\url{http://db.csail.mit.edu/labdata/labdata.html}}\textsuperscript{1}.

The experiment results are presented in Table \textsuperscript{2}. DMF outperforms all the state-of-the-art in MovieLens10M and Netflix by a large margin. Only the exception comes from the smaller dataset (i.e., MovieLens1M) where our model might have overfitted the training data due to insufficient data.

Since DMF, Biased MF, RCF, and RBMF share similar search space for main prediction, it is worthy to compare the these MF models. First, DMF outperforms Biased MF, its non-deviation-driven counterpart, significantly in all datasets. Since the three datasets are not recognized as noisy in literature, the phenomenon may imply potential implicit noise on the data as we hypothesized. It demonstrates that jointly learning the deviation and rating yields more accurate prediction. For RCF, using an M-estimator to estimate the weight is even worse than vanilla Biased MF. Compared with RBMF, DMF is superior in larger datasets, demonstrating the efficacy of low-rank (e.g., 100) variance structure versus the rank-1 variance structure in RBMF.

Finally, compared with nonlinear AutoRec model, the superior performance of linear DMF is observed. We argue the phenomenon comes from the sparsity of real world data that a nonlinear model is prone to overfit. Similar scenario appears in knowledge base completion that a linear model is reported to dominate non-linear models \cite{yang2014large}.

### 4.2 Missing Sensor Data Imputation on IoT data

The task of missing sensor data imputation is to infer missing testing data given observed training data. The experiments are conducted on two real world IoT datasets, an outdoor air quality monitoring dataset from London and an indoor sensor dataset from Intel Berkeley Research lab, which is open to public\footnote{\url{http://db.csail.mit.edu/labdata/labdata.html}}. The IoT data are naturally noisy due to sensor measurement errors. Each dataset can be represented by a 3-mode tensor $T$: the three dimensions denote location, timestamp, and sensor type, respectively. Note that there are multiple types of sensors (e.g., humidity, light and heat sensors) in each location. The statistics are shown in the right of Table \textsuperscript{1}. Since multi-modal sensors may have different ranges of values, data are normalized for each type of sensor by standard score.

We make a generalization of DMF to Deviation-driven Tensor Factorization (DTF) based on PARAFAC tensor factorization \cite{xiong2010directrobust} for modeling mean and variance, respectively. The time complexity of DTF is trivially 1.5 times of DMF. All experiments are repeated 100 times, and the average result is reported. Here we exploit mean squared error (MSE) as our evaluation metric. DTF is compared with the following classic sensor data imputation and factorization models:

- **Spatial k-Nearest Neighbors (kNN)**. Average of $K$ spatially nearest data points as prediction.
- **Inversed Distance Weighting (IDW)**. Using the inverse of distance for data points as the weighting for interpolation.
- **Probabilistic Tensor Factorization (PTF)** \cite{xiong2010directrobust}.

![Figure 3: Prediction error over various percentage of training data in London and Berkeley IoT datasets. The range of x-axis may vary according to the original sparsity of dataset. For example, we only have 22.36\% of data available in London data, so we conduct experiments using \{5\%, 10\%, 15\%, 20\%\} of training data.

|                  | MovieLens1M | MovieLens10M | Netflix  |
|------------------|-------------|--------------|---------|
| RBM              | 0.881       | 0.823        | 0.845   |
| Biased MF        | 0.845       | 0.803        | 0.844   |
| RCF              | 0.867       | 0.809        | 0.855   |
| RBMF             | 0.837       | 0.795        | 0.824   |
| LLORMA           | 0.833       | 0.782        | 0.834   |
| AutoRec          | 0.831       | 0.782        | 0.823   |
| DMF              | 0.841       | 0.775        | 0.806   |
Table 3: Running time analysis (in seconds) in MovieLens1M datasets

|                  | Biased MF | AutoRec | DMF |
|------------------|-----------|---------|-----|
| Total running    | 207       | 647     | 88  |
| Running time/epoch| 2.1       | 3.3     | 4.4 |

2010). PTF is equivalent to PARAFAC tensor factorization with $L_2$ norm regularization, the non-deviation-driven counterpart of DTF.

- **Tucker** (Kolda and Bader 2009). Tucker Decomposition with $L_2$ regularization added for better performance.

The experiment are conducted through various percentages of data as training data. The setting for each dataset may vary according to its original sparsity. Results are shown in Figure 3. The result of Tucker is not shown in Berkeley dataset because it is significantly worse than the others. It can be noticed that except the most sparse setting in each dataset, our method outperforms all competitors significantly. A reasonable explanation is that when data are already sparse, the downgrading of certain potentially noisy data can lead to insufficient amount of information for training. For winning cases, we also conduct unpaired t-test and confirmed that all $P$ values are smaller than 0.0001. In summary, this section demonstrates the extendibility of DMF to tensor factorization model and the efficacy of deviation-driven learning on noisy data.

4.3 Running Time and Convergence

We also compare the running time with the strongest competitor, AutoRec, and the counterpart of DMF, Biased MF, in MovieLens1M. All above models are implemented in C++. AdaGrad optimizer (Duchi, Hazan, and Singer 2011) is used for DMF and Biased MF, and resilient propagation optimizer (Riedmiller and Braun 1993) is used for AutoRec, as suggested in the original paper (Sedhain et al. 2015). The running time per epoch for each model is shown in Table 3. Though the running time per epoch of DMF is the longest, it takes much fewer epochs to converge and in turn spent the least amount of total running time.

We also conduct learning curve comparison of DMF with Biased MF and AutoRec. The results are shown in Figure 4. Comparing Biased MF with DMF, it confirms our hypothesis that by downplaying the importance of data with larger deviation (i.e., noisy data), our model can converge faster than Biased MF. In summary, DMF achieves its best result in the shortest running time and number of epochs.

5 Related Works

There are some related works regarding weighted MF, robust MF, and the structure of noise modeling.

In traditional weighting method (Koren, Bell, and Volinsky 2009), the confidence level of data point is assumed to be explicitly given, or learned separately with the model. (Hu, Koren, and Volinsky 2008) and (Liang et al. 2016) are designed for an implicit feedback scenario by a weighting mechanism, while our method addresses an explicit feedback scenario. (Schnabel et al. 2016) aims at weighting the distribution of data to get unbiased estimation, but we argue that real world data, like MovieLens1M and Netflix, are biased and thus should use biased data for evaluation.

Robust MF is a family of MF models addressing noise modeling, but lots of works are designed for full matrix factorization rather than CF. For example, (Meng and De La Torre 2013) and (Cao et al. 2015) aim at recovering clean image. For robust CF, RCF (Mehta, Hofmann, and Nejdl 2007) proposes to use Huber M-estimator (Huber 2011) to estimate the weighting, compared with our learning strategy. RBMF (Lakshminarayanan, Bouchard, and Archambeau 2011) is the most related one, which also models heteroscedastic variance; however, RBMF adopts a rank-1 variance structure, rather than our low-rank structure, in order to constitute Student-t prior on mean latent factor. In addition, the formulation of RBMF is intractable, requiring variational inference method to derive a tractable lower-bound, while DMF can be trained directly by SG. In our empirical study, RCF and RBMF are inferior to DMF in large datasets.

In literature, a great portion of noise-aware models adopt a sparse structure for noise (Xiong, Chen, and Schneider 2011), contrary to our low-rank structure. The discrepancy stems from the underlying assumption. A sparse noise structure deals with only a few (salient) noises, while we assume noise is everywhere. Moreover, low-rank structure is an efficient methodology to avoid overfitting the noises.

6 Conclusions

This paper proposes a deviation-driven objective function that considers diverse weights for each instance while learning a model. The main challenge lies in the fact that the deviation, or noise level, of each instance is unknown, which leads to the major contribution of this paper to propose a joint learning framework that can dynamically estimate the deviation and utilize it to adjust the cost function during training. Note that the recommendation datasets we used are all not known as noisy, which implies that even when data are clean, modeling the deviation as proposed can still yield superior performance, maybe due to the fact that data are actually implicitly noisy or not equally important in model training. Future works include exploitation of the idea to other learning tasks such as classification and clustering.
