Probabilistic Implicit Multitask Feature Learning Based on Weight Matrix Partition

Zhang Yu¹, Xin Zuo¹ and Jian-wei Liu¹

¹Department of Automation, College of Geophysics and Information Engineering, China University of Petroleum, Beijing (CUP), Beijing, 102249, China

*¹zhangyunepu@163.com; zxin@cup.edu.cn; liujw@cup.edu.cn

Abstract. Multitask learning (MTL) can be considered at two levels, considering the multiple features attribute that can be migrated between the most tasks simultaneously called MTL on task level, and only considering the features attribute that can be migrated between a very few tasks called MTL on feature level. A single feature is composed with an overlying structure including these above two levels, since the similarity between the multi-tasks is not unique. In the previous MTL studies, in task level, the shared feature structure of inter-task is the leading approach, and in feature level, only considering a partial of salient features is more prevalent. However, this model and structure assumptions are violated on most complex learning scenarios, and some specific feature structures is also limited. In this paper, from the Bayesian viewpoint, we propose a new method, the weight matrix is defined by stochastic variables which probabilistic distribution is overlaid, the relationship in task level are no longer represented by a unified structure, and relationship in feature level are not limited to concrete structural forms. The overlying structure is defined by two parts: multidimensional Gaussian mixture distribution according to the overall features to describe the task relationship and one-dimensional Gaussian mixture distribution to describe the feature relationship on every feature. These two parts can be elaborated by the information contained in each Gaussian sub-classes separately, then we determine the closeness of the relationship by calculating the posterior probability distribution. The advantage of our method is that we can operate the weight matrix more flexible and make full use of the probability distribution to get abundant expression in the task level and grasp the key factors in the feature learning process.

1. Introduction

The concept of multitasking learning was first proposed in [1], the key of the MTL is to exploit the multitask relationship and seek for valuable information from similar tasks to improve the learning efficiency. When the sample size in some tasks are not sufficient, single task learning will be rank deficient and is easy to appear over fitting, in order to alleviate these problems, the consideration about characteristics migration of the task is necessary, one can achieve the purpose of expanding the samples indirectly. In the early work by [2,3], they presented a model that all related tasks share a common structure and uniform feature correlation is assumed, and achieved the row sparse effect to control the number of features shared across the tasks, the result is only a few features are utilized by other tasks, and the other useful information is lost. This can cause negative migration on some tasks which have sufficient samples, and very few task sets are completely similar in feature structure.

To solve this problem, some researchers [4, 5] design classification according to the overall similarity of features, the tasks is divided into the most relevant task groups, and no information...
exchange between different groups. However, this method is still based on the hypothesis of task level that always concern the majority features, and it will meet some limitations when application. Such as recommendation systems, there are significant differences in their respective tasks and large scale features transfer are almost not exist, impact of salient features existing in some tasks will lost in the task level learning process. So some scholars are devoted to multitask feature learning and there appear some methods considering the characteristic of the feature individually. [6, 7] proposed a superposition models called robust MTL contain shared and outlier features. Then [8] apply “dirty” model to turn the outlier features into sparse structure.

For exploring the sparse characteristics of these special features, [9] established a method to cluster the features in the same dimension of each task, they point out these specific features have a remarkable promotion of learning effect. Recently, other researchers [10, 11] have proposed methods based on a partial special feature set, organized the special features into feature blocks and there is no connection between different feature blocks. But in MTL, the relationship between the features may be across tasks and across dimensions, this can be explained by “the eavesdropping mechanism” in [1], the connection between these features makes it possible to benefit from other tasks, and basically, this relation is embodied by linear relationship and is not limited only in the same dimension as feature block, because we know that in single task learning, there are coupling information among features, Therefore in MTL, this effect will not only be eliminated, but can be passed through different dimensions between tasks(see Figure 1). Considering some original related features are always divided into different sub blocks in [10], the relationship between these sub blocks shouldn’t be ignored.

Inspired by the idea, we put forward a new probabilistic method of multitask feature learning called MTPFL, the model is a superposition form which can make clustering in task level and feature level simultaneously. In task level, we divide tasks into the most relevant classes based on overall features. And in feature level, we integrate the sparse features except for the shared structure, looking for connections between them by making clustering. It is no longer limited to form blocks in feature level, moreover, we establish connections between whatever the same or different dimensions.

Multitask information migration is reflected by the weight matrix, another major advantage of our method is the flexibility of the representation of the weight matrix $W$. The usual process calculating the value of the $W$ is a continuous iterative optimal algorithm, which under the constraint of the objective function and the structure of the MTL is prescribed. In this article, we do not specify such structural constraints but construct a generative model using Gaussian mixed model (GMM) for the two levels’ relation mining in $W$, the Gaussian mixture distribution has good performance which can fit any probabilistic distribution, so the description of relationship is abundant and complete, we only need to utilize the characteristics of the dataset itself to find useful information.

2. Model Formulation

In this section, we will derive our proposed probabilistic model on multitask learning. Supposing there are totally $M$ tasks, each task has a sample set $D_m = \{x_i^m, y_i^m\}_{i=1}^{N_m}$, where $m = 1, L, M$. To learn the model, we formulate the linear model on input-output pairs:

$$
y_i^m = w_0^m + x_i^{m_1}w_1^m + x_i^{m_2}w_2^m + L + x_i^{m_D}w_D^m + \epsilon_m = w_0^m + (x_i^{m})^T W^m + \epsilon_m
$$

$$= w_0^m + (x_i^{m})^T (U^m + V^m) + \epsilon_m \quad i = 1, L, N_m$$

(1)
Where $x^m_i \in \mathbb{R}^D$ is the input vector of the $m$-th task, $X^m = [x^m_1, x^m_2, \ldots, x^m_N]$ is a input matrix which row represents the observation of the $m$-th task, $X = [X^1, X^2, \ldots, X^M]$ is the matrix ensemble concatenating input matrix on all tasks, $W^m \in \mathbb{R}^M$ is the weight vector of the $m$-th task, $W \in \mathbb{R}^{D \times M}$ is the weight matrix stacked by weight vectors of the all tasks. The $Y^m \in \mathbb{R}^{N \times K}$ is the output vector of the $m$-th task and we let $Y = [y^1, y^2, \ldots, y^M]$ be vector ensemble which tilts the output vectors of all tasks and the $m$-th column in $Y$ is the response for task $m$, $e^m \sim N(0, \sigma^2)$ represent the noise. Like the previous works [6, 7, 8, 9, 10], $U$ stands for the similarity in task level while $V$ retains the features other than shared structures.

According to our hypothesis, In task level $U = [U^1, U^2, \ldots, U^M]$ are divided into several clusters according to the whole similarity of $U^m$, and we apply multidimensional Gaussian mixture distribution $U^m \sim \sum_{k=1}^K \pi_k \mathcal{N}(b_k, \gamma_k)$ as its priori probability, the mixture has $K$ components, $U^m$ follows the normal distribution under each component: $P(U^m | b_k, \gamma_k) = \frac{1}{(2\pi)^n | \gamma_k^m |^{1/2}} \exp \left( \frac{-1}{2} (U^m - b_k)^T \gamma_k^{-1} (U^m - b_k) \right)$ (2)

Where $b_k \in \mathbb{R}^D$ is the mean of the $k$-cluster and $B = [b_1, b_2, \ldots, b_K]$ is the mean vector, $\gamma_k \in \mathbb{R}^{D \times D}$ is the covariance matrix in each mixture component, $\gamma_k = \sigma_k^2 \Phi_k$ is the covariance matrix array. $\pi = (\pi_1, \ldots, \pi_K)$ is the mixture component weights vector, meanwhile, we introduce an indicator variable matrix $C \in \mathbb{R}^{N \times K}$, its row vector $C_m$ indicates the class that the $m$-task belongs to and only one of it is taken as “1” and the rest is “0”, the $C_m$ follows multinomial distribution with parameter $\pi$: $C_m \sim M(\pi_1, \ldots, \pi_K) \Rightarrow p(c_{m,k} = 1) = \pi_k$ (3)

Similar but different from $U$, we apply one-dimensional Gaussian mixture distribution $V^m \sim \sum_{g=1}^G \rho_g \mathcal{N}(q_g, \eta_g)$ to the every component $V^m_g$ in $V$ for describing the relationship between features. The $G$ is the number of feature components and it is noted that the object of this distribution $V^m_g$ is a scalar, so the mean $q_g$ and variance $\eta_g$ of the $g$-th component are scalar, we define the mean vector $Q = [q_1, \ldots, q_G]$ , the variance vector $\eta = [\eta_1, \ldots, \eta_G]$ , the weights vector $\rho = (\rho_1, \ldots, \rho_G)$ , and the indicator variable $Z \in \mathbb{R}^{(D \times M \times G)}$. The $Z$ has $D \times M$ rows and $G$ columns for it indicate all features, its element $(z^m_{dg})^g$ follows the multinomial distribution with parameter $\rho_g$ like $C$: $(z^m_{dg})^g \sim M(\rho_{1g}, \ldots, \rho_{Gg})$ and $p((z^m_{dg})^g = 1) = \rho_g$. We draw probability graph of variables in Figure 4 (see Appendix 3). Obviously, relationships in $U^m$ or $V^m$ is not independent because they are generated from a mixture of distributions, but we can write their joint distribution after introducing the indicator variable (the deduction is in Appendix 2) and acquire our generative probabilistic model as follows:

$$\gamma^m = \sum_{d=1}^D \gamma^m_d (U^m + V^m)^T + e^m \Rightarrow P(Y | U, V, C, Z, \Theta) = \prod_{m=1}^M \prod_{g=1}^G \mathcal{N}(x^m_g (U^m + V^m)^T, \sigma^2)$$

$$P(U, C | \pi, B, \gamma) = \prod_{m=1}^M \prod_{k=1}^K \pi_k^{c_{mk}} \mathcal{N}(U^m | b_k, \gamma_k)^{c_{mk}}$$

$$p(V, Z | \rho, Q, \eta) = \prod_{d=1}^D \prod_{m=1}^M \prod_{g=1}^G \rho_g^{(z^m_{dg})^g} \mathcal{N}(V | q_g, \eta_g)^{(z^m_{dg})^g}$$ (4)
The bias $w^n_0$ can be absorbed into $x^n$ by adding an extra column in the matrix $U^n$, $V^n$ and $X$. The $\Theta=\{\sigma^2, \pi, B, \gamma^2, \rho, Q, \eta^2\}$ is parameters to be estimated, because $U$, $V$, $C$, $Z$ are also unknown parameters, we will use MCEM to solve the parameters $\Theta$ in Appendix 4.

3. Experiments

In this section, we apply the algorithm to both synthetic and real datasets to compare with other state-of-the-art multi-task learning as follows: 1) Ridge regression method[12]; 2) convex feature learning[3] ($L_{2,1}$); 3) Clustered MTL[5] (CMTL); 4) Method for Trace Norm Minimization constraining weight matrix[6] (Low rank) ; 5) Dirty method with some sparse features[8] (Dirty); 6) robust multi-task learning[13] (rMTL); 7) robust multi-task feature learning[7] (rMTFL); 8) convex multitask learning with flexible task clusters[9] (Flex-Clus); 9) Task-Feature Co-Clusters in Multi-Task Learning[10] (CoCMTL). The key evaluating criteria are the root mean squared error (RMSE), the normalization mean squared error (NMSE), and averaged mean squared error (AMSE).

The synthetic data: this section contains seven scenarios, in each scenario, in order to make sure it’s a standard normalized form, the input $x$ is generated from normal distribution $\mathcal{N}(0,1)$. We randomly generate 2000 examples and split it into 10 tasks, each task has 200 examples and each samples has 30 features, the number of training, validation, test samples is 30, 70 and 100, a single sample is represented as $y_i^n=(x_i^n)^T w^n_0 + \mathcal{N}(0,10)$, where the last term is noise, according to the seven multi-tasking learning methods, we set seven types of $W$ in virtual scenes to simulate the relationships:

**Scene1:** In this scene, all tasks are independent, $W^n_0 \sim \mathcal{N}(\mu^n_0, I), \mu^n_0 \sim \mathcal{U}(0,5), m=1, L, 10, \mathcal{U}$ is the uniform distribution and $\mathcal{N}$ is Gaussian distribution, it is the same symbol in following scenes.

**Scene2** (uniform feature structure): Tasks are related like $L_{2,1}[3]$, we guarantee the similarity of the inter-tasks by generating $W \sim \mathcal{N}(\mu, I)$ from the same distribution, where $\mu \sim \mathcal{U}(0,5)$.

**Scene3** (task clustering): We make $W$ obey several Gaussian probability clusters, and set three distributions to represent three clusters, i.e., $W_i \sim \mathcal{N}(\mu'_i, I), \mu'_i \sim \mathcal{U}(0,5), m=1, L, 10, i=1, 2, 3$, each cluster contains 4, 3 and 3 tasks respectively, the difference between clusters are obviously.

**Scene4** (Flex-Clus): We simulate the special case of outlier features, for example, the recommendation system mentioned in flexible task clusters[9], we generate $W$ from $\mathcal{N}(\mu, I)$, where $\mu \sim \mathcal{U}(0,5)$, then we replace the last two rows features ($W_{ed}$, $i=1, L, 10$ and $d=1, L, 30$ with $10 + \mathcal{N}(0,10)$.

**Scene5** (feature block): For simulating the scene in [10], we divide $W$ into $P$ and $Q$. $P$ describes the similarity of the all tasks, which is generated from $\mathcal{N}(\mu, I)$, where $\mu \sim \mathcal{U}(0,1)$. On this foundation, 30 features are scattered into several rows: $q_{1,5}$, $q_{6,15}$ and $q_{16,30}$. Then we sample 3, 4, 3 columns from $\mathcal{N}(\nu', I)$ separately to form blocks $q^n_0$, where $\nu' \sim \mathcal{U}(0,10)$ and $m=1, 2, 3$. $Q$ is the block diagonalization of $q^n_0$, according to this assumption, the features in different blocks do not overlap and tasks are divided into different clusters by these feature blocks. Finally we get $W = P + Q$.

**Scene6**: The form of $W$ and $P$ is the same as scene5, but we want to simulate when there exist some relationships between these feature blocks. So we generate a column that elements have some relationship from a mixture distribution $q' \sim \sum_{j=1}^{10} \mathcal{N}(\nu', I)$ where $i=1, K, 30$ and $\nu' \sim \mathcal{U}(0,10)$, then we scatter and copy $q^n_0$ as scene5 to form $Q$. 

4
**Scene7:** The difference from above is: the task level $P$ is arisen from a Gaussian mixture distribution $p^m = \sum_{i=1}^{4} \mathcal{N}(\mu^i, I)$ and $\mu^i \sim \mathcal{U}(0,1)$, there have a certain relationship with each other but their closeness is not inhomogeneous. On the basis of scene6, more general the situation is that features are not always form blocks and there are always some features left out, for simulating similar features in all feature spaces, we consider a more complex feature relationship, all features have a certain connection more or less, which means $(q^m)_d \sim \sum_{j=1}^{10} \mathcal{N}(\nu^j, I)$, $\nu^j \sim \mathcal{U}(0,10)$. We use ten components to describe it, then rearrange $(q^m)_d$ to $D$ rows and $M$ columns to get $Q$, finally we got $W = P + Q$.

There is no relation between tasks in scene1, so the experimental results (in Appendix 1) show Ridge achieve the best performance and most multitask learning methods has not enough worked, because the common structure is completely absent and it cause negative migration, but we can see the latter four methods focus on feature are next to Ridge, they learned some information due to only a few features can be migrated, so the performances are still better than methods focus on task level.

The result in scene2 shows that like $L_{2,1}$ or Low rank based on common structure have almost the better same performance because they are just different in the quantity of selected features, but the setting in scene2 have the same data structure and we have adopted the unit variance which means there are no probability relations between the features in every task.

In scene3 with the number of task clusters increasing, the prediction accuracy of the traditional methods basis on shared structure decline, CMTL performs well than the other overlapping models, because the $P$ in those models based on the shared structure without considering the task clustering.

In the scene4, there appear two significant difference lines of features, which means there exist nonlinear relation between the feature vectors, because the significant change in the direction of features and most methods do not have the structural constraint, the prediction accuracy of uniform structured algorithm like $L_{2,1}$ norm, CMTL occur descend at different degrees. The model assumption of rMTL and rMTFL can conform to this scene because they focus on the isolate task or feature, besides the last three feature learning methods.

In these latter several scenes, accuracy of methods that only considering characteristics of the task descend, and the methods which considering characteristics of the features gradually became prominent. The rMTL and rMTFL does not deal with these forms of feature blocks. It is worth noting that although rMTL contain a low rank constraint on the matrix, but the other part of the overlapping is the shared structure constraint like $L_{2,1}$ norm, this part still produces negative migration, especially in the last three case it occur task clusters or more specific features, so when we compare rMTL with Low rank method, the latter is better. For scene5, Flex-clus perform well, but with the increase of the feature correlation in scene6 and scene7, CoCMTL and MTPFL give better results. The thermal map of the learned weight matrix is shown in figure 2 and result of the evaluating indicator is shown in table 1a and table 1b. We can see that MTPFL have advantage in feature learning, especially when these feature blocks are related, at the same time we can draw a conclusion that classification at the task level is necessary.

**The school data:** the dataset is consist of 15362 students from 139 schools, the exam score prediction of each schools can be seen as a task and each student was taken as a samples which contains 27 attributes, such as the year of admission, gender, ethnic and etc. Two attributes are in a representation of percentage number while the others are encoded into “0”, “1", we also add an attribute to offset the bias $W_0$. Notice that the sample sizes in each task are different and the number of samples in some tasks is even less than the number of feature attributes, the result of single task regression is inaccurate, so the data set is suitable for MTL. We select 10%, 20%, 30% of the samples
in each task as training set respectively, validation set is selected as 45%, and test sets are correspond to 45%, 35%, 25%.

The number of iterations is ten thousand times, after iteration, we can see from Figure 3 (In Appendix 1) although the training sets have different percentages, most irrelevant feature of $U$ are induced into several clusters in task level, this is consistent with our hypothesis that task groups are more reasonable, and clusters are changing owing to the sample information varies with the capacity of different training sets. Meanwhile features of $U$ became stable sparse because they are taken into consideration in $V$, some notable distinct lines show that these features are more strongly related to most tasks, which has an obvious auxiliary role for tasks with small sample capacity.

Matrix $V$ mainly contains some unique features whose influence changing with tasks. It's very clear in Figure 3 that there are a lot of discrete features that belong to the same cluster. It is worth noting that, for example, the features in the eighth and ninth line are very dense, comparing the same positions of $U$, we can see that it is also significant in $V$, this is because according to our hypothesis, except for their influence in task level, they still have a part of their own characteristics, which is consistent with the property of the dataset because they are important features. The result shows that features between different tasks can be interrelated when the structure of $V$ become gradually stable. And the comparison with other methods is in appendix 1 (Table 2).

**Computer survey data:** the dataset is from a real statistic data which also be used in many literature like [3], it describes an analysis of buying preferences for 20 brands of computers in the means of score “0-10”, each brand contains 180 investigated customer samples that can be considered as separate tasks, attributes are 13 items configuration about computer like cpu, ram, cache, price, etc. Since these attributes are the same for each brand computer, that is, all samples (computer attributes) of the tasks (customer) are the same. Among the output set, we eliminate the samples which have too many invalid scoring items, eventually there are 172 samples left in the dataset. As we know, there are some correlation between these features, such as the quality of CPU directly affects the overall configuration, and better hardware are required to match it quickly, thus it affects the relationship on other features in the learning process. We verify it by experimental accuracy and MTPFL achieved a better performance, the results are shown in Appendix 1 (Table 3).

4. Conclusion

In this article, we propose a new generative model for multitask feature learning, the weight matrix is divided into two parts, and modeling the task and feature relationship simultaneously, according to the relationship between the task relation and feature modeling, and generative method can describe the relationship between these two parts. The selection of cluster number and the accuracy of sampling are still the research directions in the next work.

References

[1] Caruana R. Learning many related tasks at the same time with backpropagation// International Conference on Neural Information Processing Systems. MIT Press, 1994:657-664
[2] Evgeniou T, Pontil M. Regularized multi--task learning// Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2004:109-117
[3] Argyriou, A.; Evgeniou, T.; and Pontil, M. 2008. Convex multi-task feature learning. Machine Learning 73(3):243–272
[4] Xue Y, Liao X, Carin L, et al. Multi-Task Learning for Classification with Dirichlet Process Priors . Journal of Machine Learning Research, 2007, 8(1):35-63
[5] Jacob L, Bach F, Vert J P. Clustered Multi-Task Learning: A Convex Formulation. Advances in Neural Information Processing Systems, 2008:745-752
[6] Ji S, Ye J. An accelerated gradient method for trace norm minimization// International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June. DBLP, 2009:457-464
[7] Gong P, Ye J, Zhang C. Robust Multi-Task Feature Learning// Acm Sigkdd International Conference on Knowledge Discovery & Data Mining. KDD, 2012:895
[8] Jalali A, Sanghavi S, Ruan C, et al. A dirty model for multi-task learning[C]//Advances in neural information processing systems. 2010: 964-972
Zhong W, Kwok J. Convex Multitask Learning with Flexible Task Clusters. 2012
Xu L, Huang A, Chen E. Exploiting Task-Feature Co-Clusters in Multi-Task Learning. 2015
Murugesan K, Carbonell J, Yang Y. Co-Clustering for Multitask Learning. 2017
Arthur E. Hoerl, Robert W. Kennard. Ridge Regression: Biased Estimation for Nonorthogonal Problems. Technometrics, 1970, 12(1):55-67
Chen J, Zhou J, Ye J. Integrating low-rank and group-sparse structures for robust multi-task learning// ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2011:42-50

Appendix 1:

![Figure 2. The thermal map of the learned weight matrix under these methods, structure learned by MTPFL is the closest to the Ground Truth.]

### Table 1a. Result on synthetic data. (The former five methods)

| ratio | L21 | Low rank | CMTL | dirty |
|-------|-----|----------|------|-------|
| Syn1  | 2.5742[1] | 2.9862[7] | 2.9199[6] | 3.1353[9] | 3.2055[10] |
| Syn2  | 2.3553[10] | 2.1562[1] | 2.1591[2] | 2.1867[7] | 2.2421[9] |
| Syn3  | 3.3253[10] | 3.1674[9] | 2.8329[4] | 2.5299[1] | 3.1601[8] |
| RMSE  | Syn4  | 2.9485[10] | 2.6042[6] | 2.5299[3] | 2.7394[8] | 2.8950[9] |
| Syn5  | 2.7134[7] | 2.7284[8] | 2.3795[3] | 2.6002[5] | 2.9139[10] |
| Syn6  | 2.5118[10] | 2.2305[4] | 2.0466[2] | 2.3304[8] | 2.3525[9] |
| Syn7  | 3.3176[9] | 3.2387[6] | 2.9828[2] | 3.2842[8] | 3.2494[7] |

### Table 1b. Result on synthetic data. (The latter five methods)

| scene | rMTL | rMTFL | Flex-clus | CoCMTL | MTPFL |
|-------|------|-------|-----------|--------|-------|
| Syn1  | 3.0742[8] | 2.9101[5] | 2.6039[3] | 2.7929[4] | 2.6016[2] |
| Syn2  | 2.1936[8] | 2.1851[6] | 2.1698[4] | 2.1742[5] | 2.1606[3] |
| Syn3  | 2.8290[3] | 3.1203[7] | 3.0425[6] | 2.9745[5] | 2.7551[2] |
| RMSE  | Syn4  | 2.5337[4] | 2.5440[5] | 2.4213[2] | 2.6795[7] | 2.3837[1] |
| Syn5  | 2.4311[4] | 2.7327[9] | 2.6345[7] | 2.2751[2] | 2.0738[1] |
| Syn6  | 2.1547[3] | 2.2482[6] | 2.2490[7] | 2.2405[5] | 2.0112[1] |
| Syn7  | 3.1225[4] | 3.3606[10] | 3.1453[5] | 3.0967[3] | 2.5124[1] |

### Table 2a. Result on school data. (The former five methods)

| ratio | L21 | Low rank | CMTL | dirty |
|-------|-----|----------|------|-------|
| 10%   | 11.4316[9] | 11.2647[6] | 11.2339[4] | 11.3226[7] | 11.4412[10] |
| 20%   | 11.0421[8] | 11.0133[7] | 10.7506[5] | 10.7733[6] | 11.0759[9] |
| 30%   | 10.6908[10] | 10.5944[7] | 10.5834[5] | 10.5936[6] | 10.6535[8] |
| 10%   | 1.0935[8] | 0.9662[6] | 0.9633[4] | 0.9738[5] | 1.1226[9] |
| NMSE  | 20%   | 1.0143[9] | 0.8641[6] | 0.8601[5] | 0.9309[7] | 0.9645[8] |
AIACT IOP Publishing

IOP Conf. Series: Journal of Physics: Conf. Series 1061 (2018) 012008
doi:10.1088/1742-6596/1061/1/012008

30% 0.9402[10] 0.8751[7] 0.8260[4] 0.8469[6] 0.8935[9]
10% 0.2866[8] 0.2553[5] 0.2523[4] 0.2805[7] 0.2932[9]
AMSE 20% 0.2723[9] 0.2451[7] 0.2398[5] 0.2437[6] 0.2525[8]
30% 0.2456[10] 0.2289[7] 0.2198[5] 0.2213[6] 0.2319[8]

Table 2b. Result on school data. (the latter five methods)

| ratio | rMTL   | rMTFL  | Flex-clus | CoCMTL  | MTPFL  |
|-------|--------|--------|-----------|----------|---------|
| 10%   | 11.2515[5] | 11.3735[8] | 11.1491[2] | 11.2221[3] | 11.0653[1] |
| RMSE  | 20%    | 10.7361[4] | 11.0771[10] | 10.7037[3] | 10.7013[2] | 10.6369[1] |
| 30%   | 10.5755[4] | 10.6731[9] | 10.4154[2] | 10.4971[3] | 10.3615[1] |
| 10%   | 1.0694[7] | 1.2982[10] | 0.8983[3] | 0.8357[1] | 0.8532[2] |
| NMSE  | 20%    | 0.8393[4] | 1.0389[10] | 0.8121[3] | 0.8092[2] | 0.7961[1] |
| 30%   | 0.8413[5] | 0.8901[8] | 0.7970[3] | 0.7849[2] | 0.7703[1] |
| 10%   | 0.2606[6] | 0.3394[10] | 0.2282[1] | 0.2389[2] | 0.2447[3] |
| AMSE  | 20%    | 0.2363[4] | 0.2813[10] | 0.2172[2] | 0.2297[3] | 0.2123[1] |
| 30%   | 0.2161[4] | 0.2325[9] | 0.2057[2] | 0.2080[3] | 0.1948[1] |

Figure 3. Comparison of $U$, $V$ and $W$’s thermal map with different training sample percent in School data, the rows stand for tasks and columns stand for features. And close colors represent approximate relationship and they are maximum probability from the same probability cluster.

Table 3. Result on computer survey data.

| Algorithm | NMSE   | AMSE   | RMSE  |
|-----------|--------|--------|-------|
| Ridge     | 2.4754 | 1.5162 | 2.1371|
| L2,1      | 2.4703 | 1.6324 | 2.1192|
| Low rank  | 2.1868 | 1.3651 | 2.0346|
| CMTL      | 2.2427 | 1.5567 | 2.0314|
| dirty     | 2.4341 | 1.6247 | 2.1257|
| rMTL      | 2.3679 | 1.4734 | 2.0425|
| rMTFL     | 2.3625 | 1.4029 | 2.0685|
| Flex-clus | 1.9949 | 1.2459 | 2.0284|
| CoCMTL    | 1.8943 | 1.3392 | 2.0252|
| MTPFL     | 1.8683 | 1.2081 | 2.0142|

Appendix 2:
The distribution of the task level vector $U^m$ after introducing hidden variables $C$ is:
\[ p(U^m | \pi, B, \gamma^2) = \sum_{k=1}^{K} \pi_k \mathcal{N}(U^m | b_k, \gamma_k^2) \]

\[ p(c_{m,k} = 1) = \pi_k \Rightarrow P(C_m) = \prod_{k=1}^{K} \pi_k^{c_{m,k}} \]

\[ p(U^m | \pi, B, \gamma, c_{m,k} = 1) = \mathcal{N}(U^m | b_k, \gamma_k^2) \Rightarrow p(U^m | \pi, B, \gamma^2, C_m) = \prod_{k=1}^{K} \mathcal{N}(U^m | b_k, \gamma_k^2)^{c_{m,k}} \]

\[ p(U, C | \pi, B, \gamma^2) = \prod_{m=1}^{M} p(C_m) p(U^m | \pi, B, \gamma^2, C_m) = \prod_{m=1}^{M} \prod_{k=1}^{K} \pi_k^{c_{m,k}} \mathcal{N}(U^m | b_k, \gamma_k^2)^{c_{m,k}} \]

The distribution of the feature level point \( V^m_d \) after introducing hidden variables \( Z \) is:

\[ p(V^m_d | \rho, Q, \eta^2) = \sum_{g=1}^{G} \rho_g \mathcal{N}(V^m_d | q_g, \eta_g^2) \]

\[ P((z^m_d)_g = 1) = \rho_g \Rightarrow P(Z^m) = \prod_{g=1}^{G} \rho_g^{(z^m_d)_g} \]

\[ p(V^m_d | \rho, Q, \eta^2, (z^m_d)_g = 1) = \mathcal{N}(V^m_d | q_g, \eta_g^2) \Rightarrow p(V^m_d | \rho, Q, \eta^2, Z^m) = \prod_{g=1}^{G} \mathcal{N}(V^m_d | q_g, \eta_g^2)^{(z^m_d)_g} \]

\[ p(V, Z | \rho, Q, \eta^2) = \prod_{d=1}^{D} \prod_{m=1}^{M} p(Z^m_d) p(V^m_d | \rho, Q, \eta^2, Z^m) = \prod_{d=1}^{D} \prod_{m=1}^{M} \prod_{g=1}^{G} \rho_g^{(z^m_d)_g} \mathcal{N}(V | q_g, \eta_g^2)^{(z^m_d)_g} \]

\[ \text{Appendix 3:} \]

\[ \text{Figure 4. Graphical model for probabilistic weight matrix partition.} \]

\[ \text{Appendix 4:} \]

In this section we will adopt the Monte Carlo EM (MCEM) method to estimate the parameters \( \Theta(\cdot) \) and derive parameters update equations. First we write the logarithm conditional posterior probability \( \log p(Y | X, \Theta) \) after augmenting the data with \( U, V, C \) and \( Z \) as:

\[ \log p(Y | X, \Theta) = \sum_{C} \sum_{Z} \int_{d}^{d} \log p(Y, U, V, C, Z | X, \Theta) d_x d_y \]

The initialization are needed before running the algorithm, the two mixture components proportion \( \pi \) and \( \rho \) are randomly allocated, \( \gamma^2 \) and \( \eta^2 \) are randomly assigned covariance ensembles, \( B \) and \( Q \) come from a uniformly distribution, and we randomly select rows in \( Z \) and \( C \) with “1”. After the initial value of \( \Theta \) has been determined, we sample from the distribution of \( U^{(0)} \) and \( V^{(0)} \), and in order to prevent occurring empty Gaussian components in the following iterations, we use the \( K \)-means method
to recalculate $\pi^{(0)}, B^{(0)}, \rho^{(0)}, Q^{(0)}, C^{(0)}$ as real initial values $\Theta^{(0)}$, then we determine $Z^{(0)}$ and $C^{(0)}$ according to result of the K-means. Then we formally enter the MCEM algorithm.

E-step: in the E-step, we calculate the approximate expectation of the log-likelihood:

$$Q(\Theta \mid \Theta(t)) = E \{ \log p(Y, U, V, C, Z \mid X, \Theta(t)) \mid Y, X, \Theta(t) \} \approx \sum_{n=1}^{N} \log p(Y, z^{(n,t)}, \rho^{(n,t)}, \psi^{(n,t)}) \mid X, \Theta(t)) \tag{4}$$

On successive iterations, at $t$-th step of the iteration, to simulate $U, V, C, Z$, we implement repeatedly Gibbs sampling from conditional mathematical expectation $z^{(n,t)}, \rho^{(n,t)}, \psi^{(n,t)}$ of $U, V, C,$ and $Z$:

$$z^{(n,t)} = E[U \mid Y, V, C, Z, X, \hat{\Theta}(t)] = \Pr(U \mid Y, V, C, Z, X, \hat{\Theta}(t)) \tag{5}$$

Note that the $W$ is the sum of $U$ and $V$, we construct generative models for the two components $U$ and $V$ respectively, the posterior distribution of $U$ and $V$ can be derived from the complete log-likelihood:

$$L = \log p(Y, U, V, C, Z \mid X, \Theta) = \log(Y \mid X, U, V, C, Z, \Theta) + \log p(U, C \mid V, Z, \Theta) + \log p(V, Z \mid \Theta)$$

$$= \log(Y \mid X, U, V, \sigma^2) + \log p(U, C \mid \pi, b, \gamma^2) + \log p(V, Z \mid \rho, d, \eta^2)$$

$$= L_{\text{lognorm}} - \frac{1}{2\sigma^2} \sum_{n=1}^{N} \left( \chi_n^2 - \sum_{j=1}^{d} \chi_{n,j}^0 (U_{n,j}^0 + V_{n,j}^0)^2 \right) + \frac{1}{2\sigma^2} \sum_{n=1}^{M} \sum_{d,t=1}^{D} \left( z_{n,j}^{(d,t)} \right)^2 \left( \log \rho_{d} - \frac{(V_{n,d}^0 - q_{d})^2}{2\eta^2_d} \right)$$

$$+ \frac{(U_{n,1}^0 - 2U_{n,0}^0)^2}{2C_{1,0}^0} + \frac{(V_{n,1}^0 - 2V_{n,0}^0)^2}{2Z_{1,0}^0} \tag{6}$$

The $K(\cdot)$ and $F(\cdot)$ contain mathematical variables that are independent from $V$ and $U$ respectively. The process solving posterior distribution about $U$ need to separate the related parameters from likelihood functions (6) as follows:

$$L = K(\cdot) - \frac{1}{2\sigma^2} \left( \sum_{n=1}^{N} \left( U_{n,1}^0 - 2U_{n,0}^0 \right)^2 \right) - \frac{1}{2\sigma^2} \left( \sum_{n=1}^{N} \left( V_{n,1}^0 - 2V_{n,0}^0 \right)^2 \right)$$

$$= K(\cdot) - \frac{1}{2\sigma^2} \left( \sum_{n=1}^{N} \left( U_{n,1}^0 - 2U_{n,0}^0 \right)^2 \right) - \frac{1}{2\sigma^2} \left( \sum_{n=1}^{N} \left( V_{n,1}^0 - 2V_{n,0}^0 \right)^2 \right)$$

$$\frac{1}{2(\sigma^2 + C_{n,0}^0)} \left[ \left( \frac{1}{\sigma^2} \right)^2 \left( \frac{1}{C_{n,0}^0} \right)^2 \right]$$

$$= K(\cdot) - \frac{1}{2\sigma^2} \left( \sum_{n=1}^{N} \left( U_{n,1}^0 - 2U_{n,0}^0 \right)^2 \right) - \frac{1}{2\sigma^2} \left( \sum_{n=1}^{N} \left( V_{n,1}^0 - 2V_{n,0}^0 \right)^2 \right)$$

$$\frac{1}{\left( \frac{1}{\sigma^2} + \frac{1}{\sigma^2} \right)} \left[ \left( \frac{1}{\sigma^2} \right)^2 \left( \frac{1}{\sigma^2} \right)^2 \right]$$

$$= K(\cdot) - \frac{1}{2\sigma^2} \left( \sum_{n=1}^{N} \left( U_{n,1}^0 - 2U_{n,0}^0 \right)^2 \right) - \frac{1}{2\sigma^2} \left( \sum_{n=1}^{N} \left( V_{n,1}^0 - 2V_{n,0}^0 \right)^2 \right)$$

$$\frac{1}{\left( \frac{1}{\sigma^2} + \frac{1}{\sigma^2} \right)} \left[ \left( \frac{1}{\sigma^2} \right)^2 \left( \frac{1}{\sigma^2} \right)^2 \right]$$
Note that $U$ defines the weight matrices, the sampling process of $U$ is vector by vectors, according to our hypothesis, the posterior distribution $p(U^m | C, V, y^m, x^m, \Theta)$ ~ $\mathcal{N}(\mu_v, \Sigma_v)$ still follows Gaussian distribution, from (7) the mean and variance of $U^m$ are:

$$\mu_v = [(X^m)^T X^m + \sigma_v^2 C_m (\gamma^2)^{-1} I_{D,m}]^{-1} \times \{\sigma_v^2 C_m B [C_m (\gamma^2)]^{-1} + (X^m)^T Y^m - (X^m)^T X^m V^m\}$$

$$\sum_v = \sigma_v^2 [(X^m)^T X^m + \sigma_v^2 C_m (\gamma^2)^{-1} I_{D,m}]^{-1}$$

(8)

The solving process about posterior distribution $p(V^m | Z,Y, X, \Theta)$ ~ $\mathcal{N}(\mu_v, \Sigma_v)$ is similar as $U$, but $V$ is relation to elements in the matrix, we need change structural of $X,Y,U,V$ to $X_{blk}, Y_{blk}, U_{blk}, V_{blk}$:

$$X_{blk} = \begin{bmatrix} X^1 & X^1 & 0 & \cdots & 0 \\ Y^1 & Y^2 & \cdots & \cdots & Y^M \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ X^M & \cdots & \cdots & \cdots & X^M \end{bmatrix}$$

$$Y_{blk} = \begin{bmatrix} y^1 & y^1 & \cdots & \cdots & y^1 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ y^M & \cdots & \cdots & \cdots & y^M \end{bmatrix}$$

$$U_{vec} = ((U^1)^T, (U^2)^T, (U^M)^T)$$

$$V_{vec} = ((V^1)^T, (V^2)^T, (V^M)^T)$$

And the mean and variance of $V^m$ is:

$$\mu_v = ((X_{blk})^T X_{blk} + \sigma_v^2 Z_m \eta_m^{-1})^{-1} \times \{\sigma_v^2 Z_m \eta_m^{-1} + (X_{blk})^T Y_{blk} - (X_{blk})^T X_{blk} U_{vec}\}$$

$$\sum_v = \sigma_v^2 ((X_{blk})^T X_{blk} + \sigma_v^2 Z_m \eta_m^{-1})^{-1}$$

(9)

$C$ and $Z$ are the probability that the data is generated by every Gaussian component. So the scales of $\psi, \zeta$ are the same as $C, Z$, the probability that $U^m$ is generated by the $k$-th Gaussian components is:

$$\psi^m_k = E[c_{m,k} | Y, \hat{\Theta}(t)] = \Pr[c_{m,k} = 1 | U^m, \hat{\Theta}(t)] = \frac{\pi_k \mathcal{N}(U^m | b_k, \gamma_k^2)}{\sum_k \pi_k \mathcal{N}(U^m | b_k, \gamma_k^2)}$$

(11)

Similarly, the probability that $Vvec$ is generated by the $g$-th Gaussian components is:

$$(\zeta^m_g) = E[(z^m_g) | Y, \hat{\Theta}(t)] = \Pr[(z^m_g) = 1 | V^m, \hat{\Theta}(t)] = \frac{\rho_g \mathcal{N}(V^m | q_g, \eta_g^2)}{\sum_g \rho_g \mathcal{N}(V^m | q_g, \eta_g^2)}$$

(12)

Up to now, we have derived the conditional posterior distribution of four hidden auxiliary matrices $U, V, C$ and $Z$, and then we can implement repeatedly Gibbs sampling.

**M-step:** on the basis of obtained the updated values of unobserved matrices in E-step. In the M-step, to maximize the $Q(\Theta|\hat{\Theta}(t))$ and get updating parameters $\hat{\Theta}(t+1)$, we take the derivatives of the likelihood function and set equal to zero, and recalculate the weight, variance and mean of each clusters. Meanwhile we define the probability of number of components in the clusters:

$$N_k = \sum_{m=1}^M \psi^m_k, \quad N_g = \sum_{m=1}^M \sum_{d=1}^D \zeta^m_g$$

(13)

Then the updating formulas of the parameters $\Theta = \{\sigma^2, \pi, B, \gamma^2, \rho, Q, \eta^2\}$ are:

$$\sigma^2 = \frac{\sum_{m=1}^M \sum_{d=1}^D (Y^m - X^m(U^m + V^m))^2}{\sum_{m=1}^M \psi^m}$$

$$b_k = \frac{1}{N_k} \sum_{m=1}^M \psi^m_k U^m$$

$$\gamma_k^2 = \frac{1}{N_k} \sum_{m=1}^M \psi^m_k (U^m - b_k)(U^m - b_k)^T$$

$$\rho_g = \frac{\sum_{m=1}^M \zeta^m_g \sum_{d=1}^D \zeta^m_d V^m}{N_g}$$

$$q_g = \frac{1}{N_g} \sum_{m=1}^M \sum_{d=1}^D \zeta^m_g V^m$$

(14)