PROBABILISTIC TENSOR FACTORIZATION FOR LINK PREDICTION

by

Beyza Ermiş

B.S., Computer Engineering, Bilkent University, 2010

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2012

PROBABILISTIC TENSOR FACTORIZATION FOR LINK PREDICTION

APPROVED BY:

Assoc. Prof. Ali Taylan Cemgil .....
(Thesis Supervisor)

Assist. Prof. Arzucan Özgür .....

Assist. Prof. Evrim Acar .....

DATE OF APPROVAL: .....

ACKNOWLEDGEMENTS

I would like to thank my thesis supervisor Assoc. Prof. Ali Taylan Cemgil for believing in me and for his patience, guidance and support during my graduate study and the thesis process. Also, I thank to Assist. Prof. Evrim Acar for responding all my questions tirelessly, pointing me in the right direction with useful comments and participating in my thesis committee. Moreover, I thank to Assist. Prof. Arzucan Ozgur for participating in my thesis committee and for their insight.

I would like to thank my dear friends Fatma Guney and Safiye Celik for always being with me until the end of this formidable process, and for their support at any time. I will forever be grateful for having them together as my friends. It is also impossible to ignore the support of Murat Arar, Kubra Makara and Mirun Akyuz. I would like to thank them for their excellent friendship.

I would like to express my deepest gratitude to my dear family for supporting me throughout the years, financially, practically and with moral support. I tried so hard but I could not find any words to tell your meaning to me. Finally, I would also like to thank the members of PILAB for providing an enjoyable working environment.

ABSTRACT

PROBABILISTIC TENSOR FACTORIZATION FOR LINK PREDICTION

Link prediction is the problem of inferring the presence, absence or strength of a link between two entities, based on properties of the other observed links. In the literature, two related types of link prediction problems are considered: (i) missing and (ii) temporal. In both cases, latent variable models have been studied for link prediction tasks that consider link prediction as a noisy matrix and tensor completion problem. By using a low-rank structure of a dataset, it is possible to recover missing entries for matrices and higher-order tensors. In this thesis, we use several approaches based on probabilistic interpretation of tensor factorizations: Probabilistic Latent Tensor Factorization that can realize any arbitrary tensor factorization structure on datasets in the form of single tensor and Generalised Coupled Tensor factorization that can simultaneously fit to higher-order tensors/matrices with common latent factors. We present full Bayesian inference via variational Bayes, then we derive variational inference algorithm for Bayesian coupled tensor factorization to improve the reconstruction over Bayesian factorization of single data tensor and form update equations for these models that handle simultaneous tensor factorizations where multiple observations tensors are available. Previous studies on factorization of heterogeneous data focus on either a single loss function or a specific tensor model of interest. However, one of the main challenges in analyzing heterogeneous data is to find the right tensor model and loss function. So, we consider different tensor models and loss functions for the link prediction. Numerical experiments on synthetic and real datasets demonstrate that joint analysis of data from multiple sources via coupled factorization and variational Bayes approach improves the link prediction performance and the selection of the right loss function and tensor model is crucial for accurate prediction of unobserved links.

ÖZET

BAĞLANTI TAHMİNİ İÇİN OLASILIKSAL TENSÖR AYRIŞIMI

Bağlantı tahmini gözlemlenen bağlantıların özniteliklerine göre iki varlık arasında bir bağlantının varlığı veya yokluğu sonucuna varılması problemidir. Literatürde, (i) *gözlemlenmeyen* ve (ii) *zamansal* olmak üzere iki tip bağlantı tahmini problemi bulunmaktadır. Her iki problem için de, bağlantı tahminini bir matris ve tensor tamamlama problemi olarak değerlendiren saklı özellik tabanlı modeller üzerinde çalışılmaktadır. Bu tezde, bağlantı tahmini için tensör ayrışım modellerinin olasılıksal yorumlanmasına dayalı çeşitli yaklaşımlar kullanmaktayız. İlk olarak veri kümelerini herhangi bir tensor ayrışım modeli ile analiz edebilen Olasılıksal Saklı Tensör Ayrışımı ile tanımlanmış, daha sonra ortak tensörler içeren modellerin eşzamanlı ayrışımı ile ortak saklı faktörler çıkarabilen bir algoritmik çerçeve olan Genelleştirilmiş Bağlaşımlı Tensör Ayrışımı dahilinde tanımlanmış farklı ayrışım modelleri önermekteyiz. Tensör ayrışım metodlarında varyasyonel Bayes yoluyla tam Bayesci çıkarım sunmakta, daha sonra çıkarımı geliştirmek için bağlaşımlı tensör ayrışımı için varyasyonel Bayesci çıkarım algoritması türetmekteyiz. Ek olarak, birden fazla gözlem tensörü mevcut olduğu durumlardaki modeller için eşzamanlı tensor ayrışımını gerçekleştirebilen güncelleme denklemleri oluşturmaktayız. Heterojen verilerin ayrışımında kullanılan önceki çalışmalar ya tek bir ıraksaya veya belirli bir tensör ayrışım modeline odaklanmaktadır. Ancak, heterojen veri analizinde temel zorluklardan biri doğru tensör modelini ve ıraksayı bulmaktır. Bu nedenle, bu çalışmada farklı tensör modelleri ve ıraksayları ele almaktayız.Sentetik ve gerçek veri kümeleri üzerinde gerçekleştirdiğimiz deneyler birden fazla kaynaktan gelen verilerin bağlaşımlı tensor ayrışım yöntemi ile ortak analizinin ve varyasyonel Bayesçi yaklaşımının bağlantı tahmin performansını artırmakta olduğunu ve doğru ıraksay ve tensör model seçiminin önemini göstermektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

A_α	Hyperparameters tensor for factor α
B_α	Hyperparameters tensor for factor α
i, j, k, p, q, r	Indices of tensors and nodes of graphical models
$\mathcal{V}$	Set of indices
$\mathcal{V}_0$	Set of indices for observables X
$\mathcal{V}_\alpha$	Set of indices for latent tensor Z_α
v	Index configuration for all indices
v_0	Index configuration for observables
$v_{0,\nu}$	Index configuration for observable ν
v_α	Index configuration for factor α
$\|v\|$	Cardinality of the index configuration v
X	Observation tensor
X_v	Observation tensor indexed by v
$\hat{X}$	Model estimate for the observation tensor
$\hat{X}_v$	Model estimate for the observation tensor v
$X^{i,j,k}$	Specific element of a tensor, a scalar
$X(i, j, k)$	Specific element of a tensor, a scalar
Z	Tensor Factors to be estimated (latent tensors)
Z_α	Latent tensor indexed by α
$Z_{1:\|\alpha\|}$	Latent tensors indexed from 1 to $\|\alpha\|$
$\mathcal{B}(.)$	Lower bound of the log-likelihood
$\mathcal{L}(\theta)$	Log-likelihood of θ
$\bigtriangledown_\alpha$	Derivative of a function of $\hat{X}$ wrt Z_α
Θ	Parameter set composed of A_α, B_α
$\Delta_\alpha()$	Delta function for latent tensor α

LIST OF ACRONYMS/ABBREVIATIONS

AUC	Area Under the Curve
CP	Canonical Decomposition / Parallel Factor Analysis
CTF	Coupled Tensor Factorization
DAG	Directed Acyclic Graph
EDM	Exponential Dispersion Models
EM	Expectation Maximization
EU	Euclidean (cost)
GCTF	Generalized Coupled Tensor Factorization
GLM	Generalized Linear Models
GM	Graphical Models
GTF	Generalized Tensor Factorization
ICA	Independent Components Analysis
IS	Itakura-Saito (divergence)
KL	Kullback-Leibler (divergence)
LSI	Latent Semantic Indexing
MAP	Maximum A-Posteriori
MF	Matrix Factorization
ML	Maximum Likelihood
MUR	Multiplicative Update Rules
NMF	Non-negative Matrix Factorization
PARAFAC	Parallel Factor Analysis
PCA	Principal Component Analysis
PMF	Probabilistic Matrix Factorization
PLTF	Probabilistic Latent Tensor Factorization
ROC	Receiver Operating Characteristics
SVD	Singular Value Decomposition
TF	Tensor Factorization

1. INTRODUCTION

Links can be considered as relationships between objects in different applications such as social networks, recommender systems and web analysis [1]. Link prediction is described as a task to predict the existence of a link between an arbitrary pair of entities, based on properties of the objects and other observed links [2]. Many important applications can be cast as link prediction problems: predicting friendships among people in social networks [3], predicting potential links between users and items in recommender systems [4] or predicting users' future behaviors such as clicking advertisements for marketing [5].

In the literature, the link prediction problem fall into two categories: (i) missing, where the input is a partially observed set of links and the goal is to predict the status (presence or absence) of missing connections between pairs of entities, and (ii) temporal, where we have snapshots of the fully observed set of links up to time t as input and the goal is to predict the links at the next time step $t + 1$. While missing link prediction aims to predict the missing connections in the overall data without temporal aspect, temporal link prediction aims to predict the future structure of the links by analyzing the current structure of the links [6]. This problem has been recognized in various content [7, 8]. For instance, collaborative filtering is closely related to the problem of link prediction, where the input is a partially observed matrix of (user, item) preference scores, and the goal is to recommend new items to a user [4]. In this thesis, we consider both the problem of missing link prediction and temporal link prediction.

Many real world link prediction datasets are characterized by excessive imbalance: the number of links known to be absent is often significantly more than the number of links known to exist [5]. For example, in recommender system applications, a majority of users only rate very a few items. As a result, numerous items are only rated a few times. Such datasets, whilst large in dimension, are already very sparse [1] and potentially represent only a very incomplete picture of the reality [9]. Many researchers [10, 11] pointed out that one of the major difficulties in building statistical models for

link prediction is that the prior probability of a link is typically quite small. In this case, both model evaluation and quantifying the level of confidence in the predictions have difficulties [1].

Data fusion, therefore, is a viable candidate for addressing the challenging link prediction problem. Many studies have proposed to exploit multi-relational nature of the data and showed improved link prediction performance by incorporating related sources of information in their modeling framework. For analysis of multi-relational data, Singh and Gordon [12] as well as Long *et al.* [13] introduce collective matrix factorization. Matrix factorization-based techniques have proved useful in terms of capturing the underlying patterns in data, e.g., in recommender systems [4], and joint analysis of matrices has been widely applied in numerous disciplines including signal processing [14] and bioinformatics [15]. Recent studies extend collective matrix factorization to coupled analysis of multi-relational data in the form of matrices and higher-order tensors [16, 17] since in many disciplines, relations can be defined among more than two entities, e.g., when a user engages in an activity at a certain location, a relation can be defined over user, activity and location entities. Banerjee *et al.* [17] introduced a multi-way clustering approach for relational and multi-relational data where coupled analysis of heterogeneous data was studied using minimum Bregman information. Lin *et al.* [18] also discussed coupled matrix and tensor factorizations using KL-divergence (will be described in Chapter 2) modeling higher-order tensors by fitting a CANDECOMP/PARAFAC (CP) tensor factorization model (will also be described in Chapter 2). While these studies use alternating algorithms, Acar *et al.* [19] proposed an all-at-once optimization approach for coupled analysis.

Several approaches define a single probabilistic model over the entire links. These approaches perform probabilistic inference to make prediction about the links and to capture the correlations among the links. For instance, Taskar *et al.* [20] use relational Markov networks that model links between entities as well as their attributes. Popescul and Ungan [21] extract relational features to learn the existence of links. In addition, Getoor *et al.* [22] describe several approaches for handling link uncertainty in probabilistic relational models.

Tensor factorizations are multi-linear generalizations of matrix factorizations that analyze multi-dimensional datasets by capturing the underlying patterns [23]. Missing link prediction is also closely related to matrix and tensor completion studies. By using a low-rank structure of a dataset, it is possible to recover missing entries for matrices [24] and higher-order tensors [25, 26]. In addition, tensor factorizations have been studied to address the temporal link prediction problem, e.g., Acar *et al.* [27] combine tensor factorizations with time series analysis to predict future links, Chi and Zhu [28] use the probabilistic interpretation of tensor factorizations to derive a nonnegative tensor factorization algorithm, which can incorporate temporal trends by fixing the factor matrices.

In this thesis, we address link prediction problem by using tensor based methods. The main contributions of this thesis can be summarized as follows:

- We propose to use an approach for link prediction problem based on probabilistic interpretation of tensor factorization models, i.e PLTF framework that enables one to incorporate domain specific information to any arbitrary factorization model and provides the update rules for multiplicative gradient descent and expectation-maximization algorithms using different loss functions.
- We present a variational Bayes procedure for making inference on the PLTF framework. Exact characterization of the approximating distribution and full conditionals are observed as a product of multinomial distributions, leading to a richer approximation distribution than a naive mean field.
- We describe the computation of a variational lower bound for estimation of marginal likelihood of a tensor factorization model and construct a model selection framework for arbitrary nonnegative tensor factorization model for KL cost with the variational bound.
- We present coupled tensor factorization method as an approach to incorporating side information into collaborative prediction, where multiple data tensors and matrices are jointly decomposed, with some factor matrices shared over interrelated factorizations. We consider different tensor models, i.e., CP [29–31] and Tucker [32], and loss functions, i.e., KL-divergence, IS-divergence and Euclidean

distance, for joint analysis of heterogeneous data.

- We introduce variational Bayesian coupled tensor factorization for jointly decomposing multiple data tensors and matrices in Bayesian setting.
- Using synthetic and real datasets, we demonstrate that coupled tensor factorizations outperform low-rank approximations of a single tensor in terms of missing link prediction and the selection of the tensor model as well as the loss function is significant in terms of link prediction performance.
- We also demonstrate that variational Bayesian approach increases the prediction performance of the both single and coupled models used for both temporal and missing link prediction applications.
- We also demonstrate that it is possible to address the cold-start problem in link prediction using the proposed models.

The rest of the thesis is organized as follows: in Chapter 2, we provide the necessary background information for recalling the main concepts this thesis is based on. In Chapter 3, we introduce PLTF framework [33] used for our link prediction methods. In Chapter 4, we describe variational inference procedure for making inference on the PLTF [34] and Bayesian model selection for tensor factorization models. In Chapter 5, we discuss GCTF framework [35] for coupled factorization of several tensors and matrices; and also we introduce variational Bayesian coupled tensor factorization. In Chapter 6, we present the link prediction models and in Chapter 7, we demonstrate the related experiments of link prediction models described in the previous chapter. Finally, Chapter 8 concludes this thesis.

2. BACKGROUND

In this chapter we provide the necessary background information that would be needed to understand the methods developed in the next chapters. Those concepts include Extraction of Meaningful Information via Factorization, NMF, Multiway Data Modeling via Tensors, Tensor Factorization, Learning the Factors, Bregman Divergence and Bayesian Model Selection.

2.1. Extraction of Meaningful Information via Factorization

Factorization based data modelling has become popular together with the advances in the computational power. Matrix factorization (MF) model is one of the most fundamental factorization models in machine learning, data mining, and other areas of computational science and engineering [36]. The matrix factorization captures latent structure in the data that consists of two entities. Notationally, given a particular matrix factorization model, the objective is to estimate a set of latent factor matrices A and B

$$\text{minimize} \quad D(X \parallel \hat{X}) \quad \text{s.t.} \quad \hat{X}^{i,j} = \sum_r A^{i,r} B^{j,r} \tag{2.1}$$

where i, j are observed indices, r is latent index and $D(X \parallel \hat{X})$ is appropriate cost function.

2.1.1. Nonnegative Matrix Factorization

Recently, nonnegative matrix factorization (NMF) emerged as a useful factorization method. NMF was earlier introduced by Paatero and Tapper [37] as positive matrix factorization and subsequently popularized with a seminal paper by Lee and Seung [38]. A distinguishing feature of NMF is the requirement of nonnegativity: NMF is considered for high-dimensional and large scale data in which the representation

of each element is inherently nonnegative, and it seek low-rank factor matrices that are constrained to have only nonnegative elements [36]. There are many examples of data with nonnegative representation: a text document is represented as a vector of nonnegative numbers in a standard term-frequency encoding [39], digital images are represented by pixel intensities which can be only nonnegative in image processing and chemical concentrations or gene expression levels are nonnegatively represented in sciences.

To introduce the main idea of NMF, let us consider a matrix $X \in \mathbb{R}^{N \times M}$, in which the rows represent features and the columns represent data items. Suppose a low-rank approximation of X is given by two factor matrices W and H such that:

$$X \approx WH \tag{2.2}$$

NMF can be applied to the statistical analysis of multivariate data in the following manner [34]. Given a set of of multivariate N-dimensional data vectors, the vectors are placed in the columns of an $N \times M$ matrix X where M is the number of examples in the data set. This matrix is then approximately factorized into an $N \times R$ matrix W and an $R \times M$ matrix H. Usually R is chosen to be smaller than N or M, so that W and H are smaller than the original matrix X. This results in a compressed version of the original data matrix.

Now, for $X \in \mathbb{R}^{N \times M}$ that contain only nonnegative elements, such as text documents or images with pixel intensities, a key idea of NMF is to take advantage of the inherent nonnegativity by enforcing that low-rank factor matrices are themselves nonnegative [23]. The fact that W and H are element-wise nonnegative enables natural interpretations of the approximation model in Equation 2.2. First, the nonnegativity of basis factor W enforces that each basis component, which is a column of W, is a physically meaningful instance of original data type. If w_r contains a nonnegative element, it does not represent a text document or a digital image any more. In addition, the nonnegativity of H implies that each data item can be explained by an additive linear

combination of basis components, as opposed to an additive and subtractive combination. The additive combination naturally represents the actual interaction of real-world objects, in which a subtraction does not have a direct interpretation. Combining the two advantages, Lee and Seung [38] reported that a part-based representation can be discovered with NMF. The nonnegativity of W ensures that its column is a meaningful data type, that can be interpreted as a part. The nonnegativity of H ensures that the parts can only be combined additively without subtractions.

2.2. Multiway Data Modeling via Tensors

Tensors, appear as a natural generalization of the notion of scalars, vectors, and matrices, provide a mathematical and algorithmic framework for analyzing multi-scale, multi-dimensional data and extracting meaningful information from them. Indeed we could collapse all multiway datasets to matrices but important structural information might get lost. Instead, we use a method for factorization of these multiway datasets that respects the multi-linear structure of data which includes more than two semantically meaningful dimensions [40]. However, since there are many natural ways to factorize a multiway dataset, there exist related models with distinct names, such as CANDECOMP/PARAFAC (CP) and TUCKER. We review some of the most common tensor factorization models that will be used in the next chapters (see the tutorial reviews in [40, 41] for a comprehensive list of tensor factorization models).

2.2.1. Tensor Factorization

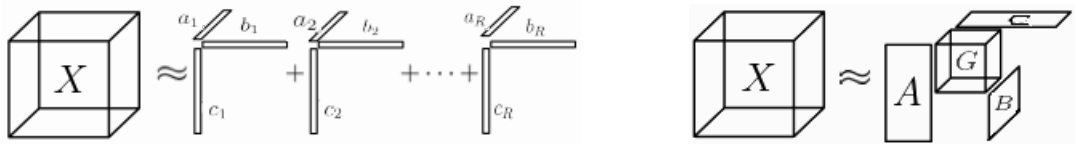
Tensor decompositions originated with Hitchcock in 1927 [31], and the idea of a multi-way model is attributed to Cattell in 1944 [42]. Later it is popularized in the field of psychometrics in 1966 by Tucker [32] and in 1970 by Carroll, Chang and Harshman [29, 43]. In addition to psychometrics, over time many applications have emerged in various fields such as chemometrics for analysis of fluorescence emission spectra, signal processing for audio applications, and biomedical for analysis of EEG data.

The factorization models emerged over the years have close relationship with each other. Going back to Hitchcock [31] in 1927, he proposed expressing a tensor as sum of finite number of rank-one tensors (simply outer product of vectors) as

$$\hat{X} = \sum_r v_{r,1} \circ v_{r,2} ... \circ v_{r,n} \tag{2.3}$$

an example for $n = 3$, i.e. a 3-way tensor, it can be expressed as

$$\hat{X}^{i,j,k} = \sum_r A^{i,r} B^{j,r} C^{k,r} \tag{2.4}$$



(a) CP factorization (b) Tucker factorization

Figure 2.1. Two widely used low rank tensor factorizations: CP factorization and Tucker factorization.

This special decomposition is discovered and named by many researchers independently such as CANDECOMP (canonical decomposition) [43] and PARAFAC (parallel factors) [29] where Kiers simply named them as CP [44]. In 1963, Tucker introduced a factorization which resembled high order PCA or SVD for the tensors [32]. It summarizes given tensor X using a core tensor G considered to be a compressed version of X. These two factorization methods are illustrated in Figure 2.1. Then, for each mode (simply the dimensions) there is a factor matrix interacting with the rest of the factors as follows

$$\hat{X}^{i,j,k} = \sum_{p,q,r} G^{p,q,r} A^{i,p} B^{j,q} C^{k,r} \tag{2.5}$$

2.2.2. Learning the Factors

Error minimization between the observation X and the model output $\hat{X}$ is one of the significant methods used for computation of the factors. After computation, this error is distributed back proportionally to the factors and they are adjusted accordingly in an iterative update schema [34]. We use various cost functions denoted by $D(X \parallel \hat{X})$ to quantify the quality of the approximation. The iterative algorithm, then, optimizes the factors in the direction of the minimum error

$$\hat{X}^* = \underset{\hat{X}}{\operatorname{argmin}} D(X \parallel \hat{X}) \tag{2.6}$$

Squared Euclidean cost is the most common choice of available cost functions

$$D(X \parallel \hat{X}) = \parallel X - \hat{X} \parallel^2 = \sum_{i,j} (X^{i,j} - \hat{X}^{i,j})^2 \tag{2.7}$$

while another is the Kullback-Leibler divergence defined as

$$D(X \parallel \hat{X}) = \sum_{i,j} X^{i,j} \log \frac{X^{i,j}}{\hat{X}^{i,j}} - X^{i,j} + \hat{X}^{i,j} \tag{2.8}$$

In addition, KL becomes relative entropy when X and $\hat{X}$ are normalized probability distributions. Finally, the Itakura-Saito divergence is defined as

$$D(X \parallel \hat{X}) = \sum_{i,j} \frac{X^{i,j}}{\hat{X}^{i,j}} - \log \frac{X^{i,j}}{\hat{X}^{i,j}} - 1 \tag{2.9}$$

2.2.3. Bregman Divergence as Generalization of Cost Functions

Seperate optimization effort and time is required to obtain inference algorithms for the factors with various cost functions [34]. For instance, the authors obtained two different versions of update equations for Euclidean and KL cost functions by a separate development [38] for NMF. On the other hand, Bregman divergence provides

us to express large class of cost functions in the same expression [45]. Assuming ϕ is a convex function, the Bregman divergence $D_\phi(X \parallel \hat{X})$ for matrix arguments is defined as

$$D_\phi(X \parallel \hat{X}) = \sum_{i,j} \phi(X^{i,j}) - \phi(\hat{X}^{i,j}) - \frac{\partial \phi(X)}{\partial \hat{X}^{i,j}}(X^{i,j} - \hat{X}^{i,j}) \tag{2.10}$$

The Bregman divergence is a nonnegative quantity as $D_\phi(X \parallel \hat{X}) \geq 0$. It is zero if and only if $X = \hat{X}$. Major class of the cost functions can be generated by the Bregman divergence by applying appropriate functions $\phi(.)$. For example, squared Euclidean distance is obtained by the function $\phi(x) = \frac{1}{2}x^2$ while the KL divergence and the IS divergence are generated by the functions $\phi(x) = x \log x$ and $\phi(x) = -logx$ respectively [45].

2.2.4. Bayesian Model Selection

We encounter with a model selection problem that deals with choosing the model order, i.e. the cardinality of the latent index r for matrix factorization given in Equation 2.1 as $\hat{X}^{i,j} = \sum_r A^{i,r} B^{j,r}$. On the other hand, selection of the right generative model and determination of the cardinality of the latent indices through many options are difficult tasks, so model selection is more complex for tensor factorization problem. As an example, given an observation $X^{i,j,k}$ with three indices one can propose a CP generative model as $\hat{X}^{i,j,k} = \sum_r A^{i,r} B^{j,r} C^{k,r}$, or a TUCKER model $\hat{X}^{i,j,k} = \sum_{p,q,r} A^{i,p} B^{j,q} C^{k,r} G^{p,q,r}$.

Bayesian model selection handles the determination of the correct number of the latent factors and their structural relations as well as cardinality of latent indices. We associate a factorization model with a random variable m interacting with the observed data x simply as $p(m|x) \propto p(x|m)p(m)$. Then, we choose the model having the highest posterior probability such that $m* = \text{argmax}_m p(m|x)$. Assuming the model priors $p(m)$ are equal the quantity $p(x|m)$ becomes important since comparing $p(m|x)$ is the

same as comparing $p(x|m)$. The quantity $p(x|m)$ is called marginal likelihood [46] and it is the average over the parameter space as:

$$p(x|m) = \int d\theta p(x|\theta, m)p(\theta|m) \tag{2.11}$$

Then comparing two models m_1 and m_2 for the observation x we use the ratio of the marginal likelihoods:

$$\frac{p(x|m_1)}{p(x|m_2)} = \frac{\int d\theta_1 p(x|\theta_1, m_1)p(\theta_1|m_1)}{\int d\theta_2 p(x|\theta_2, m_2)p(\theta_2|m_2)} \tag{2.12}$$

where this ratio is known as Bayes Factors [46]. Computation of the integral for the marginal likelihood is itself a difficult task that requires averaging on parameter space, so several approximation methods such as sampling or deterministic approximations are used for this task. Bounding the log marginal likelihood with variational inference [46, 47] is one of the approximation methods, where an approximating distribution q is introduced into the log marginal likelihood equation as:

$$p(x|m_1) \geq \mathcal{B} = \int d\theta q(\theta) \log \frac{p(x, \theta|m)}{q(\theta)} \tag{2.13}$$

In Chapter 4 we study a nonnegative tensor factorization model selection framework with KL error by lower bounding the marginal likelihood via a factorized variational Bayes approximation. The bound equations are generic in nature such that they are capable of computing the bound for any arbitrary tensor factorization model with and without missing values.

3. PROBABILISTIC LATENT TENSOR FACTORIZATION

In this chapter we first review a probabilistic framework for multiway analysis of high dimensional datasets [33]. By exploiting a link between graphical models and tensor factorization models, any arbitrary tensor factorization structure with various cost functions such as Kullback-Leibler (KL), Euclidean (EU) or Itakura-Saito (IS) can be realized with this framework. Then, we describe the details of the maximum likelihood (ML) estimation based on expectation maximization (EM) solution, i.e. fixed point update equations for latent factors.

3.1. Latent Tensor Factorization (TF) Model

We define a tensor Λ as a multiway array with an index set $\mathcal{V} = i_1, i_2, ..., i_N$ where each index $i_n = 1...|i_n|$ for $n = 1...N$. Here, $|i_n|$ denotes the cardinality of the index i_n. An element of the tensor Λ is a scalar that we denote by $\Lambda(i_1, i_2, ..., i_N)$ or $\Lambda_{i_1,i_2,...,i_N}$ or as a shorthand notation by $\Lambda(v)$. Here, v will be a particular configuration from the product space of all indices in $\mathcal{V}$. For our purposes, it will be convenient to define a collection of tensors, $Z_{1:N} = Z_\alpha$ for $\alpha = 1...N$, sharing a set of indices $\mathcal{V}$. Here, each tensor Z_α has a corresponding index set $\mathcal{V}_\alpha$ such that $\cup_{\alpha=1}^{N} \mathcal{V}_\alpha = \mathcal{V}$.

Then, v_α denotes a particular configuration of the indices for Z_α while $\bar{v}_\alpha$ denotes a configuration of the compliment $\bar{\mathcal{V}}_\alpha = \mathcal{V}/\mathcal{V}_\alpha$.

A tensor *contraction* or *marginalization* is simply adding the elements of a tensor over a given index set, i.e., for two tensors Λ and $\hat{X}$ with index sets $\mathcal{V}$ and $\mathcal{V}_0$ we write $\hat{X}(v_0) = \sum_{\bar{v}_0} \Lambda(v)$ or $\hat{X}(v_0) = \sum_{\bar{v}_0} \Lambda(v_0, \bar{v}_0)$. To clarify our notation, we present the following matrix factorization example:

$$X(i,j) \approx \hat{X}(i,j) = \sum_k Z_1(i,k) Z_2(k,j). \tag{3.1}$$

which is a tensor contraction operation. Although never done in practical computation, formally we can define $\Lambda(i,j,k) = Z_1(i,k)Z_2(k,j)$ and sum over the index k to find the result. In our formalism, we define $\mathcal{V} = i,j,k$, where $\mathcal{V}_0 = i,j$, $\mathcal{V}_1 = i,k$ and $\mathcal{V}_2 = k,j$. Hence $\bar{\mathcal{V}}_0 = k$ and we write $\hat{X}(v_0) = \sum_{\bar{v}_0} Z_1(v_1)Z_2(v_2)$.

A *tensor factorization* (TF) model is the product of a collection of tensors $Z_{1:N} = Z_\alpha$ for $\alpha = 1...N$ each defined on the corresponding index set V_α, collapsed over a set of indices $\mathcal{V}_0$. Given a particular TF model, the latent TF problem is to estimate a set of latent tensors $Z_{1:N}$

$$\text{minimize} \quad D(X \parallel \hat{X}) \quad \text{s.t.} \hat{X}(v_0) = \sum_{\bar{v}_0} \prod_{\alpha} Z_\alpha(v_\alpha) \tag{3.2}$$

where X is an observed tensor and $\hat{X}$ is the 'prediction'. Here, both objects are defined over the same index set $\mathcal{V}_0$ and are compared elementwise. The function $D(. \parallel .) \geq 0$ is a cost function. For example, the *TUCKER3* factorization aims to find Z_α for $\alpha = 1...4$ that solves the following optimization problem:

$$\text{minimize} \quad D(X \parallel \hat{X}) \quad \text{s.t.} \hat{X}^{i,j,k} = \sum_{p,q,r} Z_1^{i,p} Z_2^{j,q} Z_3^{k,r} Z_4^{p,q,r} \tag{3.3}$$

The probabilistic Latent Tensor Factorization framework (PLTF) enables one to incorporate domain specific information to any arbitrary factorization model and provides the update rules for multiplicative gradient descent and expectation-maximization algorithms.

The PLTF framework is defined as a natural extension of the matrix factorization model that is given in Equation 3.1:

$$X(v_0) \approx \hat{X}(v_0) = \sum_{\bar{v}_0} \prod_{\alpha} Z_\alpha(v_\alpha), \tag{3.4}$$

where $\alpha = 1, ...K$ denotes the factor index. In this framework, the goal is to compute

an approximate factorization of a given higher-order tensor, i.e., a multiway array, X in terms of a product of individual factors Z_α. Here, we define V as the set of all indices in a model, V_0 as the set of visible indices, V_α as the set of indices in Z_α, and $\bar{V}_\alpha = V - V_\alpha$ as the set of all indices not in Z_α. We use small letters as v_α to refer to a particular setting of indices in V_α. Since the product $\prod_\alpha Z_\alpha(v_\alpha)$ is collapsed over a set of indices, the factorization is latent.

In Section 3.2, we review a probabilistic model where the minimization problem is turned to an equivalent maximum likelihood estimation problem, i.e., solving the TF problem in Equation 3.2 will be equivalent to maximization of $\log p(X|Z_{1:N})$ with respect to Z_α.

3.2. Probability Model

The usual approach to estimate the factors Z_α is trying to find the optimal $Z^*_{1:K} = \text{argmin}_{Z_{1:K}}\, d(X||\hat{X})$, where $d(.)$ is a divergence typically taken as Euclidean, Kullback-Leibler or Itakura-Saito divergences. Since the analytical solution for this problem is intractable, one should refer to iterative or approximate inference methods.

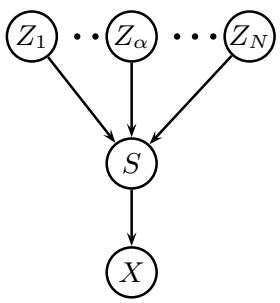


Figure 3.1. The generative model of the PLTF framework as a Bayesian network. The directed acyclic graph describes the dependency structure of the variables: the full joint distribution can be written as $p(X, S, Z_{1:K}) = p(X|S)p(S|Z_{1:K}) \prod_\alpha p(Z_\alpha)$.

The graphical model for the PLTF framework is depicted in Figure 3.1 and the

overall probabilistic model is defined as follows:

$$\Lambda(v) = \prod_{\alpha}^{N} Z_{\alpha}(v_{\alpha}) \qquad \text{(intensity)}$$

$$S(v) \sim \mathcal{PO}(S; \Lambda(v)) \qquad \text{(KL-cost)}$$

$$S(v) \sim \mathcal{N}(S; \Lambda(v), 1) \qquad \text{(EU-cost)}$$

$$S(v) \sim \mathcal{N}(S; 0, \Lambda(v)) \qquad \text{(IS-cost)}$$

$$X(v_0) = \sum_{\bar{v}_0} S(v) \qquad \text{(observation)}$$

$$\hat{X}(v_0) = \sum_{\bar{v}_0} \Lambda(v) \qquad \text{(parameter)}$$

$$M(v_0) = \begin{cases} 0 & X(v_0) \text{ is missing} \\ 1 & \text{otherwise.} \end{cases} \qquad \text{(mask array)}$$

Here, $\Lambda(v)$ the product of the factors is intensity or latent intensity field, $S(v)$ is latent source, and $X(v_0)$ is augmented data. There is a probability distribution associated with $S(v)$. Note that due to reproductivity property of Poisson and Gaussian distributions [48] the observation $X(v_0)$ has the same type of distribution as $S(v)$. Moreover, missing data is handled smoothly as in the likelihood [49, 50].

$$p(X, S|Z) = \prod_{v} \left(p(X(v_0)|S(v))p(S(v)|\Lambda(v))\right)^{M(v_0)} \tag{3.5}$$

3.3. $PLTF_{KL}$ Fixed Point Update Equation

The log likelihood $\mathcal{L}_{KL}$ is given as:

$$\mathcal{L}_{KL} = \sum_{v} M(v_0) \left(S(v) \log \Lambda(v) - \Lambda(v) - \log S(v)!\right) \tag{3.6}$$

subject to the constraint $X(v_0) = \sum_{\bar{v}_0} S(v)$ whenever $M(v_0) = 1$. We can easily optimise $\mathcal{L}_{KL}$ for Z_{α} by an EM algorithm. In the E-step we calculate the posterior

expectation $\langle S(v)|X(v_0)\rangle$ by identifying the posterior $p(S|X,Z)$ as a multinomial distribution [50]. In the M-step we solve the optimization problem $\partial\mathcal{L}_{KL}/\partial Z_\alpha(v_\alpha) = 0$ to get the fixed point update:

$$\langle S(v)|X(v_0)\rangle = \frac{X(v_0)}{\hat{X}(v_0)}\Lambda(v) \tag{3.7}$$

$$Z_\alpha(v_\alpha) = \frac{\sum_{\bar{v}_0} M(v_0)\langle S(v)|X(v_0)\rangle}{\sum_{\bar{v}_0} M(v_0)\frac{\partial\Lambda(v)}{\partial Z_\alpha(v_\alpha)}} \tag{3.8}$$

with the following equalities:

$$\frac{\partial\Lambda(v)}{\partial Z_\alpha(v_\alpha)} = \partial_\alpha\Lambda(v) = \prod_{\alpha\prime\neq\alpha} Z_{\alpha\prime(v_{\alpha\prime})} \qquad \Lambda(v) = Z_\alpha(v_\alpha)\partial_\alpha\Lambda(v) \tag{3.9}$$

After substituting Equation 3.7 and Equation 3.8, and noting that $Z_\alpha(v_\alpha)$ being independent of the sum $\sum_{\bar{v}_\alpha}$ we obtain the following multiplicative fixed point iteration for Z_α:

$$Z_\alpha(v_\alpha) \leftarrow Z_\alpha(v_\alpha)\frac{\sum_{\bar{v}_\alpha} M(v_0)\frac{X(v_0)}{\hat{X(v_0)}}\partial_\alpha\Lambda(v)}{\sum_{\bar{v}_\alpha} M(v_0)\partial_\alpha\Lambda(v)} \tag{3.10}$$

We define the tensor valued function $\Delta_\alpha(A) : \mathbb{R}^{|A|} \to \mathbb{R}^{|Z_\alpha|}$ (associated with Z_α) as

$$\Delta_\alpha(A) \equiv \left[\sum_{v\notin\mathcal{V}_\alpha}\left(A(v)\prod_{\alpha\prime\neq\alpha} Z_{\alpha\prime}(v_{\alpha\prime})\right)\right] \tag{3.11}$$

$\Delta_\alpha(A)$ is an object the same size of Z_α. We also use the notation $\Delta_{Z_\alpha}(A)$ especially when Z_α are assigned distinct letters. $\Delta_\alpha(A)(v_\alpha)$ refers to a particular element of $\Delta_\alpha(A)$. Using this new definition, we rewrite Equation 3.10 more compactly as

$$Z_\alpha \leftarrow Z_\alpha \circ \Delta_\alpha(M \circ X/\hat{X})/\Delta_\alpha(M) \tag{3.12}$$

where $\circ$ and / stand for elementwise multiplication (Hadamard product) and division respectively. Hence the key observation is that the Δ_α function is just computing a tensor product and collapses this product over indices not appearing in Z_α, which is algebraically equivalent to computing a marginal sum.

3.4. $PLTF_{EU}$ Fixed Point Update Equation

The derivation follows closely Section 3.4 where we merely replace the Poisson likelihood with that of a Gauissian. The complete data log-likelihood becomes

$$\mathcal{L}_{EU} = \sum_v M(v_0) \left(-\frac{1}{2}\log(2) - \frac{1}{2}(S(v) - \Lambda(v))^2 \right) \tag{3.13}$$

subject to the constraint $X(v_0) = \sum_{\bar{v}_0} S(v)$ whenever $M(v_0) = 1$. The sufficient statistics of the Gaussian posterior $p(S|Z, X)$ are available in closed form as

$$\langle S(v)|X(v_0)\rangle = \Lambda(v) - \frac{1}{K}(X(v_0) - \hat{X}(v_0)) \tag{3.14}$$

where K is the cardinality of unobserved configurations, i.e. the number of all possible configurations of $\bar{v}_0$ and hence $K = |\bar{v}_0|$. Then, the solution of the M step after plugging Equation 3.13 in $\frac{\partial \mathcal{L}_{EU}}{\partial Z_\alpha(v_\alpha)}$ and by setting it to zero

$$\frac{\partial \mathcal{L}_{EU}}{\partial Z_\alpha(v_\alpha)} = \sum_{\bar{v}_\alpha} M(v_0) \Big((X(v_0) - \hat{X}(v_0)) \partial_\alpha \Lambda(v) \Big) \quad = \Delta_\alpha(M \circ X) - \Delta_\alpha(M \circ \hat{X}) = 0 \tag{3.15}$$

The solution of this fixed point equation leads to iterative schemata: multiplicative updates (MUR).

3.4.1. $PLTF_{EU}$ Multiplicative Update Rules (MUR)

This method is indeed gradient ascent similar to [38] by setting $\eta(v_\alpha) = Z_\alpha(v_\alpha)/\Delta_\alpha(M \circ \hat{X})(v_\alpha)$ as

$$Z_\alpha(v_\alpha) \leftarrow Z_\alpha(v_\alpha) + \eta(v_\alpha)\frac{\partial \mathcal{L}_{EU}}{\partial Z_\alpha(v_\alpha)} \tag{3.16}$$

Then the update rule becomes simply

$$Z_\alpha \leftarrow Z_\alpha \circ \Delta_\alpha(M \circ X)/\Delta_\alpha(M \circ \hat{X}) \tag{3.17}$$

In this study, we use nonnegative variants of the two most widely-used low-rank tensor factorization models, i.e., Tucker model given in Equation 2.5 and the more restricted CP model given in Equation 2.4, as baseline methods. These models can be defined in the PLTF notation as follows. Given a three-way tensor X, its CP model is defined as:

$$X(i,j,k) \approx \hat{X}(i,j,k) = \sum_r Z_1(i,r)Z_2(j,r)Z_3(k,r) \tag{3.18}$$

where the index sets $V = \{i,j,k,r\}$, $V_0 = \{i,j,k\}$, $V_1 = \{i,r\}$, $V_2 = \{j,r\}$ and $V_3 = \{k,r\}$. A Tucker model of X is defined in the PLTF notation as follows:

$$X(i,j,k) \approx \hat{X}(i,j,k) = \sum_{p,q,r} Z_1(i,p)Z_2(j,q)Z_3(k,r)Z_4(p,q,r) \tag{3.19}$$

where the index sets $V = \{i,j,k,p,q,r\}$, $V_0 = \{i,j,k\}$, $V_1 = \{i,p\}$, $V_2 = \{j,q\}$, $V_3 = \{k,r\}$ and $V_4 = \{p,q,r\}$.

The update equation for non-negative generalized tensor factorization can be used

for both models given in Equation 3.18 and Equation 3.19 and is expressed as:

$$Z_\alpha \leftarrow Z_\alpha \circ \frac{\Delta_\alpha(M \circ \hat{X}^{-p} \circ X)}{\Delta_\alpha(M \circ \hat{X}^{1-p})} \qquad s.t. \;\; Z_\alpha(v_\alpha) > 0. \tag{3.20}$$

where M is a $0-1$ mask array with $M(v_0) = 1$ ($M(v_0) = 0$) if $X(v_0)$ is observed (missing). Here p determines the cost function, i.e., $p = \{0, 1, 2\}$ correspond to the β-divergence [23] that unifies Euclidean, Kullback-Leibler, and Itakura-Saito cost functions, respectively.

This update rule can be used iteratively for all non-negative Z_α and converges to a local minimum provided we start from some non-negative initial values. For updating Z_α, we need to compute the Δ function twice for arguments $A = M_\nu \circ \hat{X}_\nu^{-p} \circ X_\nu$ and $A = M_\nu \circ \hat{X}_\nu^{1-p}$. It is easy to verify that update equations for the KL-NMF (non-negative matrix factorization) problem (for $p = 1$) are obtained as a special case of Equation 3.20.

As an example, we show the multiplicative update rule for CP model in Equation 3.18 is generated by $PLTF_{KL}$. The model estimate and the fixed point equation for Z_1 are as

$$Z_1(i,r) \leftarrow Z_1(i,r) \frac{\sum_{j,k}(M(i,j,k)X(i,j,k)/\hat{X}(i,j,k))Z_2(j,r)Z_3(k,r)}{\sum_{j,k} M(i,j,k)Z_2(j,r)Z_3(k,r)} \tag{3.21}$$

As a further example, this rule specializes for the update of Z_4 factor in the Tucker model in Equation 3.19 to

$$Z_4(p,q,r) \leftarrow Z_4(p,q,r) \frac{\sum_{i,j,k} Z_1(i,p)Z_2(j,q)Z_3(k,r)M(i,j,k)\hat{X}(i,j,k)/X(i,j,k)}{\sum_{i,j,k} Z_1(i,p)Z_2(j,q)Z_3(k,r)M(i,j,k)} \tag{3.22}$$

Other factor updates are similar. Note that these updates also respect the sparsity pattern of the data X as specified by the mask M and can be efficiently implemented on large-but-sparse data (See Chapter A for more detail).

In this chapter, we have reviewed a probabilistic framework [33] for multiway analysis of high dimensional datasets. We use this framework for analysis of real datasets for both missing and temporal link prediction problems and show the results in Chapter 7.

4. VARIATIONAL INFERENCE AND MODEL SELECTION FOR PROBABILISTIC TENSOR FACTORIZATION

This chapter constructs a model selection framework for arbitrary nonnegative tensor factorization model for KL cost via a variational bound on the marginal likelihood [46, 51]. In this chapter, we explicitly focus on using the KL divergence and nonnegative factorizations while the treatment in this chapter can be extended for other error measures and divergences noting that we already outline the general equations for model selection. Our probabilistic treatment generalizes the statistical treatment of NMF models described in [50, 52].

This chapter is organized as follows. Section 4.1 introduces Bayesian model selection and model selection with Variational methods. It also describes variation methods for $PLTF_{KL}$ models. Section 4.2 is about computing a lower bound for marginal likelihood. Finally, Section 4.3 deals with the implementation issues followed by various experiments.

4.1. Model Selection for $PLTF_{KL}$ Models

For matrix factorization models the model selection problem becomes choosing the model order, i.e. the cardinality of the latent index, whereas for tensor factorization models selecting the right generative model among many alternatives can be a difficult task. The difficulty is due to the fact that it is not clear how to choose i) the cardinality of the latent indices, ii) the actual structure of the factorization. For example, given an observation $X^{i,j,k}$ with three indices one can propose a CP generative model as $\hat{X}^{i,j,k} = \sum_r Z_1^{i,r} Z_2^{j,r} Z_3^{k,r}$, or a TUCKER model $\hat{X}^{i,j,k} = \sum_{p,q,r} Z_1^{i,p} Z_2^{j,q} Z_3^{k,r} Z_4^{p,q,r}$ or some arbitrary model as $\hat{X}^{i,j,k} = \sum_{p,q} Z_1^{i,p} Z_2^{j,p} Z_3^{p,q}$.

For a Bayesian point of view, a model is associated with a random variable

Θ and interacts with the observed data X simply as $p(\Theta|X) \propto p(X|\Theta)p(\Theta)$. The quantity $p(X|\Theta)$ is called *marginal likelihood* [46] and it is average over the space of the parameters, in our case, S and Z as [50].

$$p(X|\Theta) = \int_Z dZ \sum_S p(X|S, Z, \Theta)p(S, Z|\Theta) \tag{4.1}$$

On the other hand, computation of this integral is itself a difficult task that requires averaging on several models and parameters. There are several approximation methods such as sampling or deterministic approximations such as Gaussian approximation. Another approximation method is to bound the log marginal likelihood by using *variational inference* [46, 47, 50] where an approximating distribution q is introduced into the log marginal likelihood equation:

$$\log p(X|\Theta) \geq \int_Z dZ \sum_S q(S|Z) \log \frac{p(X, S, Z|\Theta)}{q(S, Z)} \tag{4.2}$$

where the bound attains its maximum and becomes equal to the log marginal likelihood whenever $q(S, Z)$ is set as $p(S, Z|X, \Theta)$, that is the exact posterior distribution. However, the posterior is usually intractable, and rather, inducing the approximating distribution becomes easier. Here, the approximating distribution q is chosen such that it assumes no coupling between the hidden variables such that it factorizes into independent distributions as $q(S, Z) = q(S)q(Z)$. As exact computation is intractable, we will resort to standard variational Bayes approximations [46, 47]. The interesting result is that we get a belief propagation algorithm for marginal intensity fields rather than marginal probabilities.

4.1.1. Fixed Point Update Equation for $PLTF_{KL}$

Here, we introduce the generative Probabilistic Latent Tensor Factorization KL model ($PLTF_{KL}$)

$$Z_\alpha(v_\alpha) \sim \mathcal{G}(Z_\alpha(v_\alpha); A_\alpha(v_\alpha), B_\alpha(v_\alpha)/A_\alpha(v_\alpha)) \tag{4.3}$$

with the following fixed point iterative update equation for the component Z_α obtained via EM as:

$$Z_\alpha(v_\alpha) \leftarrow \frac{(A_\alpha(v_\alpha)-1) + Z_\alpha(v_\alpha)\sum_{\bar{v}_\alpha} M(v_0)\frac{X(v_0)}{\hat{X}(v_0)}\prod_{\alpha\prime\neq\alpha} Z_{\alpha\prime}(v_{\alpha\prime})}{\frac{A_\alpha(v_\alpha)}{B_\alpha(v_\alpha)} + \sum_{\bar{v}_\alpha} M(v_0)\prod_{\alpha\prime\neq\alpha} Z_{\alpha\prime}(v_{\alpha\prime})} \tag{4.4}$$

where $\hat{X}(v_0)$ is the model estimate defined as earlier $\hat{X}(v_0) = \sum_{\bar{v}_0}\prod_\alpha Z_\alpha(v_\alpha)$. We note that the gamma hyperparameters $A_\alpha(v_\alpha)$ and $B_\alpha(v_\alpha)/A_\alpha(v_\alpha)$ are chosen for computational convenience for sparseness representation such that the distribution has a mean $B_\alpha(v_\alpha)$ and standard deviation $B_\alpha(v_\alpha)/\sqrt{A_\alpha(v_\alpha)}$ and for small $A_\alpha(v_\alpha)$ most of the parameters are forced to be around 0 favoring for a sparse representation [50]. So, Equation 4.4 can be approximated as:

$$Z_\alpha(v_\alpha) \leftarrow \frac{\sum_{\bar{v}_\alpha} M(v_0)\frac{X(v_0)}{\hat{X}(v_0)}\prod_{\alpha\prime\neq\alpha} Z_{\alpha\prime}(v_{\alpha\prime})}{\sum_{\bar{v}_\alpha} M(v_0)\prod_{\alpha\prime\neq\alpha} Z_{\alpha\prime}(v_{\alpha\prime})} \tag{4.5}$$

4.1.1.1. Tensor forms via Δ function. We make use of Δ function to make the notation shorter and implementation friendly. A tensor valued $\Delta^Z_\alpha(Q)$ function associated with component Z_α is defined as follows:

$$\Delta^Z_\alpha(Q) = \left[\sum_{\bar{v}_\alpha}\left(Q(v_0)\prod_{\alpha\prime\neq\alpha} Z_{\alpha\prime}(v_{\alpha\prime})\right)\right] \tag{4.6}$$

Recall that $\Delta^Z_\alpha(Q)$ is an object the same size of Z_α while $\Delta^Z_\alpha(Q)(v_\alpha)$ refers to a particular element of $\Delta^Z_\alpha(Q)$. Now, Equation 4.5 can be written into a form that by use

of $\Delta_\alpha^Z(.)$ as:

$$Z_\alpha \leftarrow Z_\alpha \circ \Delta_\alpha(M \circ X/\hat{X})/\Delta_\alpha(M) \tag{4.7}$$

where as usual $\circ$ and / stand for elementwise multiplication(Hadamard product) and division respectively. We use update Equation 4.7 in the following chapters for PLTF-EM method to compare with the PLTF-VB method.

4.1.2. Variational Update Equations for $PLTF_{KL}$

Here, we formulate the fixed point update equation for the update of the factor Z_α as an expectation of the approximated posterior distribution [34]. Approximation for posterior distribution $q(Z)$ is identified as the gamma distribution with the following parameters:

$$Z_\alpha(v_\alpha) \sim \mathcal{G}(Z_\alpha(v_\alpha); C_\alpha(v_\alpha), D_\alpha(v_\alpha)) \tag{4.8}$$

where the shape and scale parameters are:

$$C_\alpha(v_\alpha) = A_\alpha(v_\alpha) + \sum_{\bar{v}_\alpha} \frac{X(v_0)}{\hat{X}_L(v_0)} \prod_\alpha L_\alpha(v_\alpha) \tag{4.9}$$

$$D_\alpha(v_\alpha) = \left(\frac{A_\alpha(v_\alpha)}{B_\alpha(v_\alpha)} + \sum_{\bar{v}_\alpha} \prod_{\alpha\prime \neq \alpha} \langle Z_{\alpha\prime}(v_{\alpha\prime}) \rangle \right)^{-1} \tag{4.10}$$

Hence the expectation of the factor Z_α is identified as the mean of the gamma distribution and given in the iterative fixed point update equation obtained via variational Bayes:

$$\langle Z_\alpha(v_\alpha) \rangle = C_\alpha(v_\alpha) D_\alpha(v_\alpha) \tag{4.11}$$

$$= \frac{A_\alpha(v_\alpha) + L_\alpha(v_\alpha) \sum_{\bar{v}_\alpha} \frac{X(v_0)}{\hat{X}_L(v_0)} \prod_{\alpha\prime \neq \alpha} L_{\alpha\prime}(v_{\alpha\prime})}{\frac{A_\alpha(v_\alpha)}{B_\alpha(v_\alpha)} + \sum_{\bar{v}_\alpha} \prod_{\alpha\prime \neq \alpha} E_{\alpha\prime}(v_{\alpha\prime})} \tag{4.12}$$

$E_\alpha(v_\alpha)$ and $L_\alpha(v_\alpha)$ (L due to 'Log') are two forms of expectations of $Z_\alpha(v_\alpha)$ while $\hat{X}_E(v_0)$ and $\hat{X}_L(v_0)$ are model outputs generated by the components $E_\alpha(v_\alpha)$ and $L_\alpha(v_\alpha)$. While $\hat{X}_E$ is not being used in Equation 4.12 we define it here, in addition to $\hat{X}_L$, (and use it later on) since $\hat{X}_E$ has the same shape as $\hat{X}_L$. Indeed $\hat{X}_E$ and $\hat{X}_L$ can be regarded as different 'views' of $\hat{X}$ since they have the same shape (dimensions) as $\hat{X}$ and their computations are done via the same matrix primitives as $\hat{X}$. Here:

$$E_\alpha(v_\alpha) = \langle Z_\alpha(v_\alpha) \rangle = C_\alpha(v_\alpha) D_\alpha(v_\alpha) \tag{4.13}$$

$$L_\alpha(v_\alpha) = \exp\left(\langle \log Z_\alpha(v_\alpha) \rangle\right) = \exp\left(\psi\left(C_\alpha(v_\alpha)\right)\right) D_\alpha(v_\alpha) \tag{4.14}$$

$$\hat{X}_E(v_0) = \sum_{\bar{v}_0} \prod_\alpha E_\alpha(v_\alpha) \tag{4.15}$$

$$\hat{X}_L(v_0) = \sum_{\bar{v}_0} \prod_\alpha L_\alpha(v_\alpha) \tag{4.16}$$

Note that the Variational Bayesian (VB) version of the update equation given in Equation 4.12 closely resembles the EM version given in Equation 4.4. Indeed when the observed values are large, digamma function becomes $\lim_{x\to\infty} \psi(x)/\log(x) = 1$, and this, in turn, gives $L_\alpha(v_\alpha) \simeq E_\alpha(v_\alpha)$ and $\hat{X}_L(v_0) \simeq \hat{X}_E(v_0)$.

4.2. Variational Bound and Sufficient Statistics

The marginal likelihood of the observed data under a tensor factorization model $p(X)$ is often necessary for certain problems such as model selection. We lower bound the marginal likelihood for any arbitrary $PLTF_{KL}$ model based on variational Bayes; while clearly other Bayesian model selection such as MCMC [50, 53] can also be used. To bound the marginal log-likelihood, an approximating distribution $q(S, Z)$ over the hidden structure S and Z is introduced as:

$$\mathcal{L}(\Theta) = \log p(X|\Theta) \geq \int_Z dZ \sum_S q(S,Z) \log \frac{p(X,S,Z|\Theta)}{q(S,Z)} \tag{4.17}$$

$$= \langle \log p(X,S,Z|\Theta) \rangle_{q(S,Z)} + H\left[q(S,Z)\right] = \mathcal{B}_{VB}[q] \tag{4.18}$$

The bound is tight whenever q equals to the posterior as $q(S,Z) = p(S,Z|X,\Theta)$ but computing the posterior $p(S,Z|X,\Theta)$ is intractable. At this point variational Bayes suggests approximating q. The simplest selection for q from the family of approximating distribution is the one which poses no coupling for the members of the hidden structure S, Z. That is, we take a factorized approximation $q(S,Z) = q(S)q(Z)$ such that:

$$q(S,Z) = \left(\prod_{v_0} q\left(S(v_0,:)\right)\right)\left(\prod_{\alpha}\prod_{v_0} q\left(Z_\alpha(v_\alpha)\right)\right) \tag{4.19}$$

where : symbol in $S(v_0,:)$ is used to indicate the slice of the array. That is $S(v_0,:)$ is the slice of the latent tensor S as the observed variables in configurations v_0 are being fixed. Then, we have:

$$q^{(n+1)}_{S(v_0,:)} \propto \exp\left(\langle \log p(X,S,Z|\Theta)\rangle_{q^{(n)}/q_{S(v_0,:)}}\right) \tag{4.20}$$

$$q^{(n+1)}_{Z_\alpha(v_\alpha)} \propto \exp\left(\langle \log p(X,S,Z|\Theta)\rangle_{q^{(n+1)}/q_{Z_\alpha(v_\alpha)}}\right) \tag{4.21}$$

where the superscript (n) indicates the iteration index. This iteration monotonically improves the individual factors of the q distribution, that is, $\mathcal{B}[q^{(n)}] \leq \mathcal{B}[q^{(n+1)}]$ for $n = 1, 2, ...$ given an initialization $q^{(0)}$.

First, we start with formulating the approximating distribution $q(S)$. When we expand the log and drop $\log P(Z|\Theta)$ and all other irrelevant S terms $q_{S(v_0,:)}$ we end up with:

$$q_{S(v_0,:)} \propto \exp\left(\langle \log p(X|S) + \log p(S|Z)\rangle_{q/q_{S(v_0,:)}}\right) \tag{4.22}$$

$$\propto \exp\left(\sum_{\bar{v}_0}\left(S(v)\langle \log\prod_{\alpha} Z_\alpha(v_\alpha)\rangle - \log\Gamma(S(v)+1)\right) + \log\delta\left(X(v_0) - \sum_{\bar{v}_0} S(v)\right)\right) \tag{4.23}$$

$$\propto \exp\left(\sum_{\bar{v}_0}\left(S(v)\sum_{\alpha}\log\langle Z_\alpha(v_\alpha)\rangle - \log\Gamma(S(v)+1)\right)\right) + \delta\left(X(v_0) - \sum_{\bar{v}_0} S(v)\right) \tag{4.24}$$

Exactly, the slice $S(v_0, :)$ is distributed with the multinomial distribution as $X(v_0)$ is the total number of observations. Here, s is a vector of a priori independent Poisson random variables s_i. λ is intensity vector conditioned on the sum $x = \Sigma_i s_i$ and x is multinomial distributed with cell probabilities $p = \lambda/\Sigma_i \lambda_i$. The joint posterior density of s is denoted by $\mathcal{M}(s; x, p)$. Finally we obtain the approximating distribution as:

$$q_{S(v_0,:)} \sim \mathcal{M}(S(v_0, :), X(v_0), P(v_0, :)) \tag{4.25}$$

Then, the cell probabilities and sufficient statistics for $q_{S(v_0,:)}$ are:

$$P(v) = \frac{\exp(\Sigma_\alpha \langle \log Z_\alpha(v_\alpha) \rangle)}{\Sigma_{\bar{v}_0} \exp(\Sigma_\alpha \langle \log Z_\alpha(v_\alpha) \rangle)} \tag{4.26}$$

$$\langle S(v) \rangle = X(v_0) P(v) \tag{4.27}$$

The cell probabilities $P(v)$ can be further transformed into compact form as:

$$P(v) = \frac{\exp(\Sigma_\alpha \langle \log Z_\alpha(v_\alpha) \rangle)}{\Sigma_{\bar{v}_0} \exp(\Sigma_\alpha \langle \log Z_\alpha(v_\alpha) \rangle)} \tag{4.28}$$

$$= \frac{\prod_\alpha \exp(\langle \log Z_\alpha(v_\alpha) \rangle)}{\Sigma_{\bar{v}_0} \prod_\alpha \exp(\langle \log Z_\alpha(v_\alpha) \rangle)} \tag{4.29}$$

$$= \frac{\prod_\alpha L_\alpha(v_\alpha)}{\Sigma_{\bar{v}_0} \prod_\alpha L_\alpha(v_\alpha)} \tag{4.30}$$

Now, we turn to formulating $q(Z)$. The distribution $q_{Z_\alpha(v_\alpha)}$ is obtained similarly. After we expand the log and drop irrelevant terms, it becomes proportional to:

$$q_{Z_\alpha(v_\alpha)} \propto \exp\left(\langle \log p(S|Z) + \log p(Z|\Theta) \rangle_{q/q_{Z_\alpha(v_\alpha)}} \right) \tag{4.31}$$

$$\propto \log Z_\alpha(v_\alpha) \left(A_\alpha(v_\alpha) - 1 + \sum_{\bar{v}_\alpha} \langle S(v) \rangle \right)$$

$$- Z_\alpha(v_\alpha) \left(\frac{A_\alpha(v_\alpha)}{B_\alpha(v_\alpha} + \sum_{\bar{v}_\alpha} \prod_{\alpha\prime \neq \alpha} \langle Z_{\alpha\prime}(v_{\alpha\prime}) \rangle \right) \tag{4.32}$$

which is the distribution

$$q_{Z_\alpha(v_\alpha)} \sim \mathcal{G}(C_\alpha(v_\alpha), D_\alpha(v_\alpha)) \tag{4.33}$$

where the shape and scale parameters for $q_{Z_\alpha(v_\alpha)}$ are given in Equation 4.9 and Equation 4.10.

Finally, we reach sufficient statistics are obtained by the definition of the gamma distribution as given in Equation 4.13 and Equation 4.14.

4.2.1. Handling Missing Data

Here, slight modifications are needed in the VB-based update equation. We start with the modification on the full joint. Priors are not part of the observation model so they are not affected. The first two terms of $\langle \log p(X, S, Z|\Theta) \rangle_{q(S,Z)}$ become:

$$\begin{aligned}
&\sum_{v_0} M(v_0) \left\langle \log \delta \left(X(v_0) - \sum_{\bar{v}_0} S(v) \right) \right\rangle \\
&+ \sum_{v_0} M(v_0) \left(\langle S(v) \rangle \left\langle \log \prod_{\alpha} Z_\alpha(v_\alpha) \right\rangle - \prod_{\alpha} \langle Z_\alpha(v_\alpha) \rangle - \langle \log \Gamma(S(v)+1) \rangle \right) ...
\end{aligned} \tag{4.34}$$

and this results in the following:

$$\begin{aligned}
q_{Z_\alpha}(v_\alpha) &\propto \log\langle Z_\alpha(v_\alpha) \rangle \left(A_\alpha(v_\alpha) - 1 + \sum_{\bar{v}_\alpha} M(v_0) \langle S(v) \rangle \right) \\
&\quad - \langle Z_\alpha(v_\alpha) \rangle \left(\frac{A_\alpha(v_\alpha)}{B_\alpha(v_\alpha)} + \sum_{\bar{v}_\alpha} M(v_0) \prod_{\alpha\prime \neq \alpha} \langle Z_{\alpha\prime}(v_{\alpha\prime}) \rangle \right) \qquad (4.35) \\
&\propto \mathcal{G}(C_\alpha(v_\alpha), D_\alpha(v_\alpha)) \qquad (4.36)
\end{aligned}$$

This modifies the gamma parameters for $q(Z)$ given in Equation 4.9 and Equa-

tion 4.10 to include the mask $M(v_0)$ as follows:

$$C_\alpha(v_\alpha) = A_\alpha(v_\alpha) + \sum_{\bar{v}_\alpha} M(v_0)\langle S(v)\rangle \tag{4.37}$$

$$D_\alpha(v_\alpha) = \left(\frac{A_\alpha(v_\alpha)}{B_\alpha(v_\alpha)} + \sum_{\bar{v}_\alpha} M(v_0) \prod_{\alpha\prime\neq\alpha} \langle Z_{\alpha\prime}(v_{\alpha\prime})\rangle\right)^{-1} \tag{4.38}$$

The other terms are not affected since mask matrix is already in the definition of $C_\alpha(v_\alpha)$ and $D_\alpha(v_\alpha)$. $\hat{X}_E$ and $\hat{X}_L$ are already defined in terms of $C_\alpha(v_\alpha)$ and $D_\alpha(v_\alpha)$. Moreover, $A_\alpha(v_\alpha)$ and $B_\alpha(v_\alpha)$ are priors and not part of the observation model.

Now, for $C_\alpha(v_\alpha)$, we need to find out $\Sigma_{\bar{v}_\alpha}\langle S(v)\rangle$, which can be written as:

$$\sum_{\bar{v}_\alpha}\langle S(v)\rangle = \sum_{\bar{v}_\alpha} X(v_0)p(v) = \sum_{\bar{v}_\alpha} \frac{X(v_0)}{\hat{X}_L(v_0)} \prod_{\alpha} L_\alpha(v_\alpha) \tag{4.39}$$

$$= L_\alpha(v_\alpha) \sum_{\bar{v}_\alpha} \frac{X(v_0)}{\hat{X}_L(v_0)} \prod_{\alpha\prime\neq\alpha} L_{\alpha\prime}(v_{\alpha\prime}) \tag{4.40}$$

After consideration of the missing data for our approach, C_α and D_α can be written using the $\Delta^E_\alpha(.)$ and $\Delta^L_\alpha(.)$ as:

$$C_\alpha = A_\alpha + L_\alpha \circ \Delta^L_\alpha(M \circ X/\hat{X}_L) \tag{4.41}$$

$$D_\alpha = \left(\frac{A_\alpha}{B_\alpha} + \Delta^E_\alpha(M)\right)^{-1} \tag{4.42}$$

that, in turn, since $\langle Z_\alpha\rangle$ is $C_\alpha \circ D_\alpha$, E_α and L_α the sufficient statistics for $q(Z_\alpha)$ become:

$$\langle Z_\alpha\rangle = E_\alpha \leftarrow \frac{A_\alpha + L_\alpha \circ \Delta^L_\alpha(M \circ X/\hat{X}_L)}{\frac{A_\alpha}{B_\alpha} + \Delta^E_\alpha(M)} \tag{4.43}$$

$$\exp\langle\log(Z_\alpha)\rangle = L_\alpha \leftarrow \exp(\psi(C_\alpha)) \circ D_\alpha \tag{4.44}$$

After straightforward substitutions, we obtain the variational probabilistic latent tensor factorization algorithm, that can compactly be expressed as in Figure 4.1.

Algorithm 1 Variational Inference for PLTF (PLTF-VB)

Input: X (observation), M (mask array), A and B (priors)

Output: E (expected value of factors), and $\mathcal{B}$ (bound)

Here $N = |\alpha|$

for $\alpha = 1...N$ **do**

$L_\alpha \sim \mathcal{G}(A_\alpha, B_\alpha/A_\alpha)$

$E_\alpha \sim \mathcal{G}(A_\alpha, B_\alpha/A_\alpha)$

end for

Main loop

for $epoch = 1...MAXITER$ **do**

Compute $\hat{\mathbf{X}}_L$ and $\hat{\mathbf{X}}_E$

$\hat{\mathbf{X}}_L(v_0) = \sum_{\bar{v}_0} \prod_\alpha L_\alpha(v_\alpha)$

Computation for $\hat{\mathbf{X}}_E$ is similar and is omitted

for $\alpha = 1...N$ **do**

$C_\alpha = A_\alpha + L_\alpha \circ \Delta_\alpha^L(M \circ X/\hat{\mathbf{X}}_L)$

$D_\alpha = 1/((A_\alpha/B_\alpha) + \Delta_\alpha^E(M))$

$E_\alpha = C_\alpha \circ D_\alpha$

end for

for $\alpha = 1...N$ **do**

$L_\alpha = \exp(\psi(C_\alpha)) \circ D_\alpha$

end for

end for

Figure 4.1. Variational inference for PLTF (PLTF-VB).

4.3. Experiments

In this section, we demonstrate the use of the proposed variational Bayesian PLTF (PLTF-VB) for model selection. First, we study model selection on synthetic datasets and show that the proposed approach can accurately determine the number of components in a CP model. We also show the performance of PLTF-VB for model selection on real datasets, i.e., the UCLAF and Digg datasets will be described in

detail in Chapter 6. For variational bound computation, we use the equation given in Equation 4.18. Then, we show the effect of hyperparameter adaptation on prediction performance by comparing the performances with several number of different values for hyperparameters. We once again use both synthetic and real datasets, i.e., DBLP dataset will be described in Chapter 6.

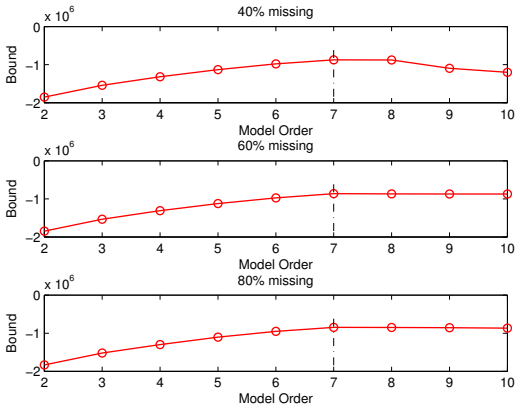
4.3.1. Model Selection

Using both synthetic and real dataset, we assess the performance of our approach in a model selection context where the goal is to determine the cardinality of the latent index r of the CP model $X^{i,j,k} = \sum_r A^{i,r}B^{j,r}C^{k,r}$. We denote the cardinality of an index i as $|i|$. $|r|$ is set to be from 2 to 10 (ignoring 1) incremented by 1 at each run. In the experiments, the iteration number is set to 2000, the shape parameter A and the scale parameter B of the gamma priors are set to be 0.5 and 10. As an initialization, a number of random initializations, i.e., 10, are used and the best performing one is picked.

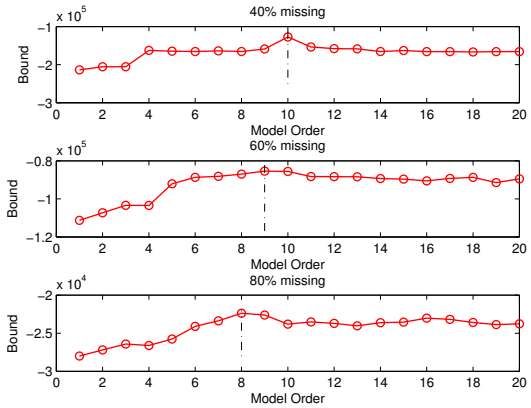
Third-order tensors of different sizes following a CP model are generated. The cardinality of the observed indices $i \times j \times k$ (i.e., the size of the data) are set to $50 \times 50 \times 50$ and $500 \times 500 \times 500$. The cardinality of the latent index r is set to 7 as the true model order. Each run is repeated 10 times and average bound score is plotted in the figures as model order is on the x-axis while bound score given in Equation 4.17 on the y-axis.

In the first experiment we use the dataset with the size of $50 \times 50 \times 50$ to test model selection process when 40%, 60% and 80% of data is unobserved respectively. What we expect to see on top right of Figure 4.2 is simply that at around true model order of $|r| = 7$ the bound to be the highest to demonstrate that PLTF-VB can find the true model order correctly in all cases of the presence of missing data.

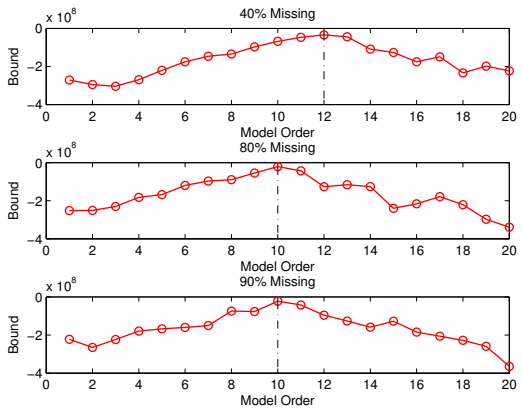
In the second experiment, to illustrate the model order selection performance under missing data case of our model with real data, we use UCLAF dataset. As



(a) Missing data case with synthetic data



(b) Missing data case with UCLAF



(c) Missing data case with Digg

Figure 4.2. Model order selection using variational bound for CP generated data.

for the experiment settings, the gamma hyperparameters are set to 0.5 for scale and 10 for shape for all the components and $|r|$ is set to be from 1 to 20. We can see the performance of our model with 40%, 60% and 80% missing data at top right of Figure 4.2.

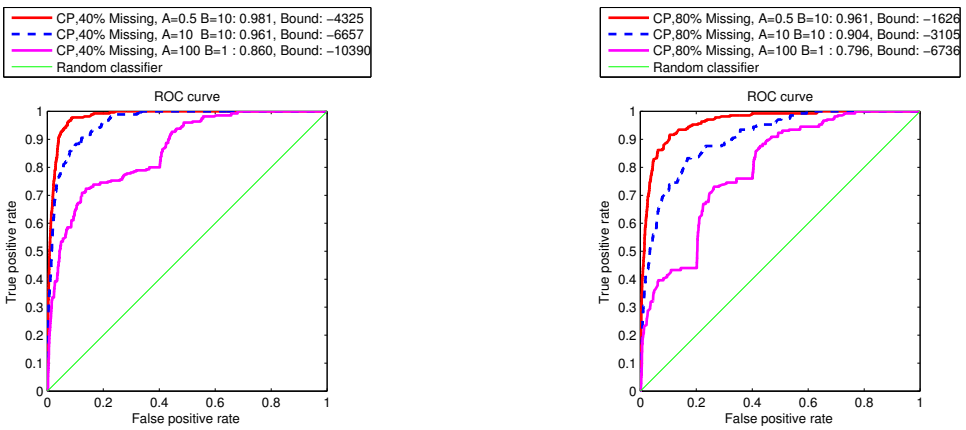
In the third experiment, we use Digg dataset. As for the experiment settings, the gamma hyperparameters are set to 0.5 for scale and 10 for shape for all the components and $|r|$ is set to be from 1 to 20. Bottom of Figure 4.2 shows the performance of our model with 40%, 80% and 90% missing data.

We observe that model order selection experiments give consistent results on both synthetic and real data. We also can see from experiments with real data, when the

amount of missing data increases the best model order decreases.

4.3.2. Hyperparameter Selection

In this section, we also extend our analysis to empirical Bayes scenarios where hyperparameters are also learned from data. We observe that hyperparameter adaptation is crucial for obtaining good prediction performance. In our simulations, results for PLTF-VB without hyperparameter adaptation were occasionally poorer than the PLTF-EM estimates. We set both shape A and scale B hyperparameters same for all components $Z_{1:3}$. We tried different values for hyperparameters to obtain the best prediction results under missing data case. In this section, we evaluate the results in terms of the Area Under Receiver Operating Characteristic Curve (AUC) that will be defined in Chapter 7. Figure 4.3 show the comparison of three different hyperparame-



(a) 40% missing (b) 80% missing

Figure 4.3. Effect of hyperparameter selection on UCLAF dataset with CP model when R=2.

ter settings; $A = 0.5, B = 10$, $A = 10, B = 10$ and $A = 100, B = 1$ on UCLAF datasets in terms of link prediction performance. As we can see, we obtain best result when initialising the shape hyperparameter $A = 0.5$ and scale hyperparameter $B = 10$ for all settings of missing data. So, we use these values of hyperparameter A and B for the following experiments in Section 7. In addition, we obtain that when we set $A < 1$ and $B > 10$, we get better results.

5. COUPLED MODELS

Recent technological advances, such as the Internet, multi-media devices or social networks provide abundance of relational data. For instance, in retail recommender systems, in addition to retail data showing *who has bought which items*, we may also have access to customers' social networks, i.e., *who is friends with whom*. In such complex problems, jointly analyzing data from multiple sources has great potential to increase our ability for capturing the underlying structure in data.

In this chapter, first, we review Generalized Coupled Tensor factorization (GCTF) framework [35] for coupled factorization of multiple tensors and matrices. Then, we introduce a variational Bayes procedure for making inference on coupled tensor factorization models that provides more powerful modelling and allows more sophisticated inference.

5.1. Generalized Coupled Tensor factorization

GCTF framework is a generalisation of the Probabilistic Latent Tensor factorization (PLTF) [33], which is described in Chapter 3, to coupled factorization. PLTF tries to solve the following approximation problem given in Equation 3.4 as

$$X(v_0) \approx \hat{X}(v_0) = \sum_{\bar{v}_0} \prod_{\alpha} Z_{\alpha}(v_{\alpha})$$

The Generalised Coupled Tensor factorization model takes the PLTF model one step further where, in this case, we have multiple observed tensors X_{ν} that are supposed to be factorised simultaneously:

$$X_{\nu}(v_{0,\nu}) \approx \hat{X}_{\nu}(v_{0,\nu}) = \sum_{\bar{v}_{0,\nu}} \prod_{\alpha} Z_{\alpha}(v_{\alpha})^{R^{\nu,\alpha}} \tag{5.1}$$

where $\nu = 1, ...|\nu|$ and R is a *coupling matrix* that is defined as follows:

$$R^{\nu,\alpha} = \begin{cases} 1 & \text{if } X_\nu \text{ and } Z_\alpha \text{ connected} \\ 0 & \text{otherwise} \end{cases} . \tag{5.2}$$

Note that, as distinct from the PLTF model, there are multiple visible index sets ($V_{0,\nu}$) in the GCTF model, each specifying the attributes of the observed tensor X_ν.

5.1.1. Inference

The inference, i.e., estimation of the shared latent factors Z_α, can be achieved via iterative optimisation (see [35]). For nonnegative data and factors, one can obtain the following compact fixed point equation where each Z_α is updated in an alternating fashion fixing the other factors $Z_{\alpha\prime}$ for $\alpha\prime \neq \alpha$

$$Z_\alpha \leftarrow Z_\alpha \circ \frac{\sum_\nu R^{\nu,\alpha} \Delta_{\alpha,\nu}(M_\nu \circ \hat{X}_\nu^{-p} \circ X_\nu)}{\sum_\nu R^{\nu,\alpha} \Delta_{\alpha,\nu}(M_\nu \circ \hat{X}_\nu^{1-p})}. \tag{5.3}$$

where $\circ$ is the Hadamard product and M_ν is a $0-1$ mask array with $M_\nu(v_{0,\nu}) = 1$ ($M_\nu(v_{0,\nu}) = 0$) if $X_\nu(v_{0,\nu})$ is observed (missing). Here p, as in Equation 3.20, determines the cost function, i.e., $p = \{0, 1, 2\}$ corresponds to Euclidean, Kullback-Leibler, and Itakura-Saito cost functions, respectively. In this iteration, the key quantity is the $\Delta_{\alpha,\nu}$ function that is defined as follows:

$$\Delta_{\alpha,\nu}(A) = \left[\sum_{v_{0,\nu} \cap \bar{v}_\alpha} A(v_{0,\nu}) \sum_{\bar{v}_0 \cap \bar{v}_\alpha} \prod_{\alpha' \neq \alpha} Z_{\alpha'}(v_{\alpha'})^{R^{\nu,\alpha'}} \right] \tag{5.4}$$

Update rules for different values of p are summarised in Table 5.1.

Table 5.1. Update rules for different p values.

p	Cost Function	Multiplicative Update Rule
0	Euclidean	$Z_\alpha \leftarrow Z_\alpha \circ \frac{\sum_\nu R^{\nu,\alpha}\Delta_{\alpha,\nu}(M_\nu \circ X_\nu)}{\sum_\nu R^{\nu,\alpha}\Delta_{\alpha,\nu}(M_\nu \circ \hat{X}_\nu)}$
1	Kullback-Leibler	$Z_\alpha \leftarrow Z_\alpha \circ \frac{\sum_\nu R^{\nu,\alpha}\Delta_{\alpha,\nu}(M_\nu \circ \hat{X}_\nu^{-1} \circ X_\nu)}{\sum_\nu R^{\nu,\alpha}\Delta_{\alpha,\nu}(M_\nu)}$
2	Itakura-Saito	$Z_\alpha \leftarrow Z_\alpha \circ \frac{\sum_\nu R^{\nu,\alpha}\Delta_{\alpha,\nu}(M_\nu \circ \hat{X}_\nu^{-2} \circ X_\nu)}{\sum_\nu R^{\nu,\alpha}\Delta_{\alpha,\nu}(M_\nu \circ \hat{X}_\nu^{-1})}$

5.2. Variational Update Equation for Coupled Tensor Factorization for KL Cost

In this section, we present a variational Bayesian method to make inference on the coupled tensor factorization models and to derive update equations for these models that handles the simultaneous tensor factorizations where multiple observations tensors are available. We present variational Bayesian coupled tensor factorization as an approach to exploiting side information, i.e., each decomposition is coupled by sharing some factor matrices. We use this method to improve the performance of PLTF-VB algorithm defined in Figure 4.1 by incorporating knowledge in the additional matrices.

In the same way as Section 5.1, we address the problem when multiple observed tensors X_ν for $\nu = 1...|\nu|$ are factorised simultaneously. Each observed tensor X_ν now has a corresponding index set $V_{0,\nu}$ and a particular configuration will be denoted by $v_{0,\nu} \equiv u_\nu$. And, we also define the $|\nu| \times |\alpha|$ coupling matrix R. Finally, we define another particular configuration for index set of S_ν will be denoted by $\bigcup_{R_{\nu,\alpha}=1} v_\alpha \equiv r_\nu$.

In Chapter 4, we obtain the approximating distributions as:

$$q_{S(v_0,:)} \sim \mathcal{M}(S(v_0,:), X(v_0), P(v_0,:))$$

where the cell probabilities and sufficient statistics for $q_{S(v_0,:)}$ are:

$$P(v) = \frac{\prod_\alpha L_\alpha(v_\alpha)}{\Sigma_{\bar{v}_0} \prod_\alpha L_\alpha(v_\alpha)}$$
$$\langle S(v) \rangle = X(v_0) P(v)$$

For the coupled factorization, we get the following expression of the cell probabilities $P_\nu(r_\nu)$ here as:

$$P_\nu(r_\nu) = \frac{\exp(\Sigma_\alpha \langle \log Z_\alpha(v_\alpha) \rangle)}{\Sigma_{\bar{u}_\nu} \exp(\Sigma_\alpha \langle \log Z_\alpha(v_\alpha) \rangle)} \tag{5.5}$$
$$= \frac{\prod_\alpha \exp(\langle \log Z_\alpha(v_\alpha) \rangle)}{\Sigma_{\bar{u}_\nu} \prod_\alpha \exp(\langle \log Z_\alpha(v_\alpha) \rangle)} \tag{5.6}$$
$$= \frac{\prod_\alpha L_\alpha(v_\alpha)}{\Sigma_{\bar{u}_\nu} \prod_\alpha L_\alpha(v_\alpha)} \tag{5.7}$$
$$= \frac{\prod_\alpha L_\alpha(v_\alpha)}{(\hat{X}_L)_v(u_\nu)} \tag{5.8}$$

Then the sufficient statistics $\langle S_\nu(r_\nu) \rangle$ turns to

$$\langle S_v(r_\nu) \rangle = X_\nu(u_\nu) P_v(r_\nu) = \frac{X_v(u_\nu)}{(\hat{X}_L)_v(u_\nu)} \prod_\alpha L_\alpha(v_\alpha) \tag{5.9}$$

Now we turn to formulating $q(Z)$. The distribution $q_{Z_\alpha(v_\alpha)}$ is obtained similarly as after we expand the log and drop irrelevant terms it becomes proportional to

$$q_{Z_\alpha(v_\alpha)} \propto \exp\left(\langle \log p(S|Z) + \log p(Z|\Theta) \rangle_{q/q_{Z_\alpha(v_\alpha)}} \right) \tag{5.10}$$
$$\propto \log Z_\alpha(v_\alpha) \left(A_\alpha(v_\alpha) - 1 + \sum_\nu R^{\nu,\alpha} \sum_{\bar{v}_\alpha} \langle S_\nu(r_\nu) \rangle^{R^{\nu,\alpha}} \right)$$
$$- Z_\alpha(v_\alpha) \left(\frac{A_\alpha(v_\alpha)}{B_\alpha(v_\alpha} + \sum_\nu R^{\nu,\alpha} \sum_{\bar{v}_\alpha} \prod_{\alpha\prime \neq \alpha} \langle Z_{\alpha\prime}(v_{\alpha\prime}) \rangle^{R^{\nu,\alpha}} \right) \tag{5.11}$$

which is the distribution

$$q_{Z_\alpha(v_\alpha)} \sim \mathcal{G}(C_\alpha(v_\alpha), D_\alpha(v_\alpha))$$

where the shape and scale parameters for $q_{Z_\alpha(v_\alpha)}$ are

$$C_\alpha(v_\alpha) = A_\alpha(v_\alpha) + \sum_\nu R^{\nu,\alpha} \sum_{\bar{v}_\alpha} \langle S_\nu(r_\nu) \rangle^{R^{\nu,\alpha}} \tag{5.12}$$

$$= A_\alpha(v_\alpha) + \sum_\nu R^{\nu,\alpha} \sum_{\bar{v}_\alpha} \frac{X_\nu(u_\nu)}{(\hat{X}_L)_\nu(u_\nu)} \prod_\alpha L_\alpha(v_\alpha)^{R^{\nu,\alpha}} \tag{5.13}$$

$$D_\alpha(v_\alpha) = \left(\frac{A_\alpha(v_\alpha)}{B_\alpha(v_\alpha)} + \sum_\nu R^{\nu,\alpha} \sum_{\bar{v}_\alpha} \prod_{\alpha\prime \neq \alpha} \langle Z_{\alpha\prime}(v_{\alpha\prime}) \rangle^{R^{\nu,\alpha}} \right)^{-1} \tag{5.14}$$

After consideration of the missing data for our approach, C_α and D_α can be written using the $\Delta^E_\alpha(.)$ and $\Delta^L_\alpha(.)$ as:

$$C_\alpha = A_\alpha + \sum_\nu R^{\nu,\alpha} L_\alpha \circ \Delta^L_\alpha(M_\nu \circ X_\nu / (\hat{X}_L)_\nu) \tag{5.15}$$

$$D_\alpha = \left(\frac{A_\alpha}{B_\alpha} + \sum_\nu R^{\nu,\alpha} \Delta^E_\alpha(M_\nu) \right)^{-1} \tag{5.16}$$

that, in turn, since $\langle Z_\alpha \rangle$ is $C_\alpha \circ D_\alpha$, E_α and L_α the sufficient statistics for $q(Z_\alpha)$ become:

$$\langle Z_\alpha \rangle = E_\alpha \leftarrow \frac{A_\alpha + \sum_\nu R^{\nu,\alpha} L_\alpha \circ \Delta^L_\alpha(M_\nu \circ X_\nu / (\hat{X}_L)_\nu)}{\frac{A_\alpha}{B_\alpha} + \sum_\nu R^{\nu,\alpha} \Delta^E_\alpha(M_\nu)} \tag{5.17}$$

$$\exp\langle \log(Z_\alpha) \rangle = L_\alpha \leftarrow \exp(\psi(C_\alpha)) \circ D_\alpha \tag{5.18}$$

After straightforward substitutions, we obtain the variational generalized coupled tensor factorization algorithm, that can compactly be expressed as in Figure 5.1.

Algorithm 2 Variational Inference for coupled tensor factorization (CTF-VB)

Input: X_ν (observation), M (mask array), A and B (priors)

Output: E (expected value of factors), and $\mathcal{B}$ (bound)

Here $M = |\nu|$

for $\nu = 1...M$ **do**

Here $N = |\alpha|$

for $\alpha = 1...N$ **do**

$L_\alpha \sim \mathcal{G}(A_\alpha, B_\alpha/A_\alpha)$

$E_\alpha \sim \mathcal{G}(A_\alpha, B_\alpha/A_\alpha)$

end for

Main loop

for $epoch = 1...MAXITER$ **do**

for $\nu = 1...M$ **do**

Compute $(\hat{\mathbf{X}}_L)_\nu$ and $(\hat{\mathbf{X}}_E)_\nu$

$(\hat{\mathbf{X}}_L)_\nu(u_\nu) = \sum_{\bar{u}_\nu} \prod_\alpha L_\alpha(v_\alpha)^{R^{\nu,\alpha}}$

Computation for $(\hat{\mathbf{X}}_E)_\nu$ is similar and is omitted

for $\alpha = 1...N$ **do**

$C_\alpha = A_\alpha + \sum_\nu R^{\nu,\alpha} L_\alpha \circ \Delta^L_\alpha(M_\nu \circ X_\nu/(\hat{\mathbf{X}}_L)_\nu)$

$D_\alpha = 1/((A_\alpha/B_\alpha) + \sum_\nu R^{\nu,\alpha} \Delta^E_\alpha(M_\nu))$

$E_\alpha = C_\alpha \circ D_\alpha$

end for

end for

end for

for $\alpha = 1...N$ **do**

$L_\alpha = \exp(\psi(C_\alpha)) \circ D_\alpha$

end for

end for

Figure 5.1. Variational inference for CTF (CTF-VB) for KL cost.

6. LINK PREDICTION

In this chapter, we propose several tensor factorization models and different loss functions for the solution of both missing and temporal link prediction problems. First, we deal temporal link prediction task by using arbitrary tensor factorization models. Then, we use coupled tensor factorization models for the solution of missing link prediction tasks. Finally, we use the variational Bayesian approach described in Chapter 4 for both of the problems.

6.1. Link Prediction with PLTF

In this section, by using the PLTF framework, we propose a solution for temporal link prediction task with arbitrary tensor model and different loss functions. In our approach, we capture temporal trends within a tensor where time is represented as a separate dimension.

As we are not restricted in using a standard model topology such as CP or Tucker, we are able to design application specific models easily. So, we describe the datasets first; then we define the particular factorization model that is convenient for the application.

6.1.1. DBLP

In the DBLP data [6], the entity sets are authors and conferences, and each time $t = 1, ..., T$ corresponded to one year of data. It contains publications from 1959 through the end of 2008. Since, most of the publications ($\sim 94\%$ of the whole data) are in later than 1990, we consider publications between 1991 and 2008.

In order to evaluate the precision of our link prediction algorithm we need to split the observed data into a training set, which is used as input for the tensor analysis, and a test set, which is compared against the estimated results. So, we divide the data

into eight training/test sets, such that each training set contains $T = 10$ years and the corresponding test set contains the following 11^{th} year. We only keep those authors that have at least 20 publications in the training data, and each test set contains only the authors and conferences available in the corresponding training set. Then, the data is organized as a third-order tensor X of size $I \times J \times T$. We let $X(i, j, t)$ denote the total number of papers by author i at conference j in year t. However, we aim to predict whether the i_{th} author is going to publish at the j_{th} conference during the test year. Therefore, each nonzero entry in the test set is treated as 1, i.e., a positive link, regardless of the actual number of publications; otherwise, it is 0 indicating that there is no link between the corresponding author-conference pair. As a result, we have a three-way observation tensor X with elements 0 and 1, where 0 denotes a known absent link and 1 denotes a known present link.

In this section, we present a tensor factorization model for temporal link prediction on DBLP dataset. In our model, we use Euclidean distance and KL divergence as cost functions and we apply the hierarchical factorization approach to a CP-style tensor decomposition model. This gives us the following model:

$$\hat{X}(i, j, t) = \sum_{r} A(i, r)B(j, r)C(t, r) \tag{6.1}$$

$$C(t, r) = \sum_{m} D(t, m)E(m, r) \tag{6.2}$$

In our model, we hierarchically factorize the factor matrix C in order to encapsulate the harmonic structure. The basic idea behind factorizing the factor matrix C is to capture the linear trend in the data and form a harmonic basis for the prediction of test year. After replacing C with the decomposed version, we get the following model:

$$\hat{X}_1(i, j, t) = \sum_{m,r} A(i, r)B(j, r)D(t, m)E(m, r) \tag{6.3}$$

After obtaining the latent factors, we employ exponential smoothing to extrapo-

late future points in time based on the latent temporal trends retrieved by the tensor factorization. Exponential smoothing is a useful method in statistics that can be applied to time series data, either to produce smoothed data for presentation, or to make forecasts [54]. In exponential smoothing, time series data are a sequence of observations. In addition, the raw data sequence is often represented by x_t and the output of the exponential smoothing algorithm is commonly written as s_t. This output can be regarded as our best estimate of x in the next time sequence. The expression of exponential smoothing is given as below, when we assume that the sequence of observations begins at time $t = 0$:

$$s_1 = x_0 \tag{6.4}$$

$$s_{t+1} = \alpha x_t + (1-\alpha)s_t \tag{6.5}$$

$$= \alpha x_t + \alpha(1-\alpha)x_{t-1} + \alpha(1-\alpha)^2 x_{t-2} + ... + (1-\alpha)^t x_0 \tag{6.6}$$

where α is the smoothing factor ($0 < \alpha < 1$), which assigns exponentially decreasing weights over time.

In our method, we use the exponential smoothing to estimate the $t+1$ frontal slice of tensor X. First, we predict vector $D(t+1,:)$, which can be derived from the temporal trend captured in the columns of factor matrix D. The exponential smoothing for each entry of vector $D(t+1,:)$ can be shown as:

$$D_{t+1,:} = \alpha D_{t,:} + \alpha(1-\alpha)D_{t-1,:} + \alpha(1-\alpha)^2 D_{t-2,:} + ... + (1-\alpha)^t D_{1,:} \tag{6.7}$$

Then, by using the vector $D_{t+1,:}$, we can calculate the missing slice as:

$$\hat{X}(i,j,t+1) = \sum_{m,r} A(i,r)B(j,r)D(t+1,m)E(m,r) \tag{6.8}$$

6.1.2. Network Traffic Data

In this section, we address the problem of recovering missing entries of a tensor which is very related to the missing link prediction problem by using both CP 3.18 and Tucker 3.19 tensor factorization models, and also by using the same model (Equation 6.3) described in Section 6.1.1. One application domain where this problem is frequently encountered is computer network traffic analysis.

For this link prediction task, we use the Géant data [55] in [25]. This data includes the network traffic matrices exchanged between 23 routers over a period of 4 months collected using 15-minute intervals. The traffic matrices consist of the amount of network data exchanged between source (computers) and destination (routers) pairs. Since the content of the traffic matrices evolve over time, this dataset can be used to form a third-order tensor with source routers, destination routers and time modes where each entry indicates the amount of traffic sent from a source to a destination during a particular time interval. In this dataset, missing data arises due to the expense of the data collection process.

In our study, we stored the Géant data as a third-order tensor X of size $I \times J \times T$ where I and J is equal to 23 and T is equal to 2756. We let $X(i, j, t)$ denote the the amount of traffic sent from a source i to a destination j at time t.

6.2. Link Prediction with Coupled Tensor Factorization

In this section, by using the GCTF framework, we propose a solution for link prediction task with different coupled models and loss functions. In the same way as Section 6.2, we are not restricted in using a standard model topology, we are able to design application specific models easily. While the coupled models are in principle applicable generally, the choice of a particular factorization is strongly guided by the needs of the application so we first describe the datasets then give the suitable factorization model.

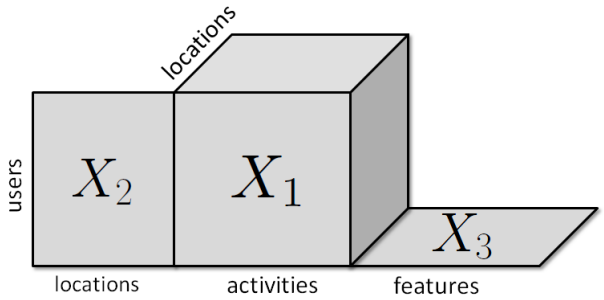


Figure 6.1. A third-order tensor coupled with two matrices in two different modes (UCLAF dataset).

6.2.1. UCLAF Dataset

UCLAF dataset [56] is extracted from the GPS data that include information of three types of entities: user, location and activity (See Figure 6.1 for an illustration of the data). The relations between the user-location-activity triplets are used to construct a three-way tensor X_1. In tensor X_1, an entry $X_1(i, j, k)$ indicates the frequency of a user i visiting location j and doing activity k there; otherwise, it is 0. Since we address the link prediction problem in this study, we define the user-location-activity tensor X_1 as:

$$X_1(i,j,k) = \begin{cases} 1 & \text{if user } i \text{ visits location } j \text{ and performs activity } k \text{ there} \\ 0 & \text{otherwise} \end{cases} .$$

To construct the dataset, raw GPS points were clustered into 168 meaningful locations and the user comments attached to the GPS data were manually parsed into activity annotations for the 168 locations. Consequently, the data consists of 164 users, 168 locations and 5 different types of activities, i.e., 'Food and Drink', 'Shopping', 'Movies and Shows', 'Sports and Exercise', and 'Tourism and Amusement'.

Additionally, the collected data includes additional side information: the location features from the POI (points of interest) database and the user-location preferences from the GPS trajectory data, represented as the matrix X_2 and X_3 respectively. In our model the user-location preferences matrix has entries $X_2(i, m)$ of size $I \times J$, where

I is the number of users and J is the number of locations. We use a separate index m for the location index in X_2 instead of j. In order to relax the model as the entries in X_1 and X_2 are measuring distinct quantities: $X_2(i, m)$ represents the frequency of a user i visiting location m and stayed there over a time threshold while X_1 only indicates an activity by a specific user i at location j. The relation between the location entries j and m in X_1 and X_2 are coupled via a common factor over the users. Finally, we represent the location-feature values with matrix X_3 of size $J \times N$, where J is the number of locations, that has the same location type in X_1, and N is the number of features. In particular, an entry $X_3(j, n)$ represents the number of different POIs given a location j. Using the location features, we could gain information about location similarities based on their feature values.

In this dataset, 18 users have no location and activity information. Therefore, we have used the data from the remaining 146 users. Moreover, to decrease the effect of outliers for both EUC and KL cost functions, location-feature matrix is preprocessed as follows:

$$X_3(j, n) = \begin{cases} 1 + \log(X_3(j, n)) & X_3(j, n) > 0 \\ 0 & \text{otherwise} \end{cases}.$$

In our experiments, number of users is $I = 146$, number of locations $J = 168$, number of activities $K = 5$ and number of location features $N = 14$.

We have a three-way observation tensor X_1 with elements 0 and 1, where 0 denotes a known absent link and 1 denotes a known present link, and two auxiliary matrices X_2 and X_3 that provide side information. Our aim is to restore the missing links in X_1. This is a difficult link prediction problem since X_1 contains less than 1% of all possible links or an entire slice of X_1 may be missing. Using low-rank factorization of a tensor to estimate missing entries will be ineffective, in particular, in the case of structured missing data such as missing slices.

We form two coupled models in order to fill in the missing links in tensor X_1.

For both models, we use KL divergence and Euclidean as cost functions in our non-negative decomposition problems. In the first model, we applied the coupled approach to a CP-style tensor decomposition model by analysing the tensor X_1 jointly with the additional matrices X_2 and X_3 in order to solve the sparsity problem in X_1 effectively. This gives us the following model:

$$\hat{X}_1(i,j,k) = \sum_r A(i,r)B(j,r)C(k,r) \tag{6.9}$$

$$\hat{X}_2(i,m) = \sum_r A(i,r)D(m,r) \tag{6.10}$$

$$\hat{X}_3(j,n) = \sum_r B(j,r)E(n,r) \tag{6.11}$$

Here, we have three observed tensors, that share common factors; therefore, we have a coupled tensor factorization problem. The coupling matrix R with $|\alpha| = 5$, $|\nu| = 3$ for this model is defined as follows:

$$R = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix} \qquad \text{with} \quad \begin{aligned} \hat{X}_1 &= \textstyle\sum A^1 B^1 C^1 D^0 E^0 \\ \hat{X}_2 &= \textstyle\sum A^1 B^0 C^0 D^1 E^0 \\ \hat{X}_3 &= \textstyle\sum A^0 B^1 C^0 D^0 E^1 \end{aligned} . \tag{6.12}$$

Note that, X_1 and X_2 share the common factor matrix A with entries $A(i,r)$; we can interpret each row of $A(i,:)$ as user i's latent position in a $|r|$ dimensional *preferences* space. The factor matrix B with entries $B(j,r)$ represents the latent position of the location j in the same preferences space. A user i at location j tends to make the activity k where the weight $A(i,r)B(j,r)$ is large for at least one r, i.e., there is a match between the users preference and what the location 'has to offer'. The location specific factor B is also influenced by the location-feature matrix X_3 .

Here we show the computation for A, i.e. for Z_1, which is the common factor of

X_1 and X_2:

$$\Delta^{\varepsilon}_{A,1}(Q) = \left[\sum_{j,k} Q^{i,j,k}(B^{j,r}C^{k,r})^{\varepsilon}\right] = Q_1(BC)^{\varepsilon} \tag{6.13}$$

$$\Delta^{\varepsilon}_{A,2}(Q) = \left[\sum_{m} Q^{i,m}(D^{m,r})^{\varepsilon}\right] = Q_2 D^{\varepsilon} \tag{6.14}$$

$$A \leftarrow A \circ \frac{Q_1(BC) + Q_2 D}{\hat{X}_1^{-p}(BC) + \hat{X}_2^{-p} D} \tag{6.15}$$

and the computation for B, i.e. for Z_2, which is the common factor of X_1 and X_3:

$$\Delta^{\varepsilon}_{B,1}(Q) = \left[\sum_{i,k} Q^{i,j,k}(A^{i,r}C^{k,r})^{\varepsilon}\right] = Q_1(AC)^{\varepsilon} \tag{6.16}$$

$$\Delta^{\varepsilon}_{B,2}(Q) = \left[\sum_{n} Q^{j,n}(E^{n,r})^{\varepsilon}\right] = Q_2 E^{\varepsilon} \tag{6.17}$$

$$B \leftarrow B \circ \frac{Q_1(AC) + Q_3 E}{\hat{X}_1^{-p}(AC) + \hat{X}_3^{-p} E} \tag{6.18}$$

Following the same line of thought, we applied the coupled approach using a Tucker decomposition to form our second model. Consequently, we get the following:

$$\hat{X}_1(i,j,k) = \sum_{p,q,r} A(i,p)B(j,q)C(k,r)D(p,q,r) \tag{6.19}$$

$$\hat{X}_2(i,m) = \sum_{p} A(i,p)E(m,p) \tag{6.20}$$

$$\hat{X}_3(j,n) = \sum_{r} B(j,q)F(n,q) \tag{6.21}$$

In this case, we have the following R matrix with $|\alpha| = 6$, $|\nu| = 3$

$$R = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{with} \quad \begin{aligned} \hat{X}_1 &= \textstyle\sum A^1 B^1 C^1 D^1 E^0 F^0 \\ \hat{X}_2 &= \textstyle\sum A^1 B^0 C^0 D^0 E^1 F^0 \\ \hat{X}_3 &= \textstyle\sum A^0 B^1 C^0 D^0 E^0 F^1 \end{aligned} . \tag{6.22}$$

In this model, once again, the factor A is shared by X_1 and X_2, while the factor B is shared by X_1 and X_3. In contrast to the coupled CP model sketched in Equation 6.9, this model assumes that a user i at location j tends to make the activity k with the weight $\sum_{p,q} A(i,p)B(j,q)D(p,q,r)$. Here, a latent preference space interpretation is less intuitive but the model has more freedom to represent the dependence.

6.2.2. Digg Dataset

We also address another link prediction task to demonstrate the utility of coupled tensor factorization models. Digg is a social news resource that allows users to submit, digg and comment on news stories. Also, users can create social networks by designating other users as friends' activities [18]. Lin *et al.* have collected data from a large set of user actions from Digg The dataset is a subset of data scrapped from Digg by Munmun De Choudnury during January 2009 [57]. It includes stories, users and their actions (submit, digg, comment and reply) with respect to the stories, as well as the explicit friendship (contact) relation among these users. It also includes the topics of the stories and extract keywords from the stories' titles.

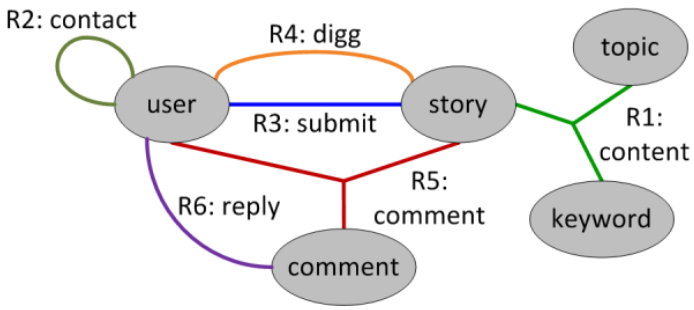


Figure 6.2. Entities and relations included in Digg dataset[1].

In this dataset, there are five types of entities: user, story, comment, keyword and topic and six relationships among them. The relations are summarized in Figure 6.2 and Table 6.1 [18]. In addition, the total number of tuples in each tensor sequence per relation is listed in Table 6.1. Except for the contact relation, all relations have

[1]Figure 6.2 is taken from [18].

timestamps. So, they assume the contact relation is static and consider the other relations are dynamic. For dynamic relations, they extract tuples with timestamps ranging from August 1 to August 27, 2008. To study the data evolution, they segment the data duration into nine time slots (i.e. every three days), and construct a sequence of data tensors for each dynamic relation. However, in our work, we will not use the segmented data. We integrate the nine segment together and evaluate missing link prediction tasks on this integrated data. The prediction results are compared with the actual diggs and comments as ground truth.

Table 6.1. Summary of the relations in Digg dataset[2].

Relation	Tensor	# Tuples
R1 (content)	dynamic (story, keyword, topic)	151.779
R2 (contact)	static (user, user)	56.440
R3 (submit)	dynamic (user, story)	44.005
R4 (digg)	dynamic (user, story)	1.157.529
R5 (comment)	dynamic (user, story, comment)	241.800
R6 (reply)	dynamic (user, comment)	94.551

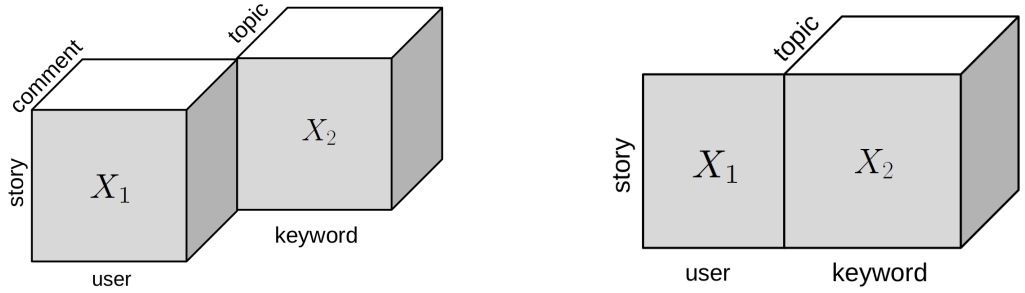
Based on the Digg scenario, we design two prediction tasks on Digg dataset:
(i) comment prediction - what stories a user will comment on, and
(ii) digg prediction - what stories a user will digg.

For the first prediction task, we have a three-way observation tensor X_1 with elements 0 and 1 and an auxiliary tensor X_2 that provide side information. Our aim is to restore the missing links in X_1 (See Figure 6.3a for an illustration of the modeled data). This time, X_1 contains less than 0.07% of all possible links. To increase the correctness of prediction, we design another model and use one more auxiliary matrix X_3, for the comment prediction.

For the second prediction task, likewise, we have an observation matrix X_1 with

[2]Table 6.1 is taken from [18].

elements 0 and 1 and an auxiliary tensor X_2 that provide side information. Our aim is to restore the missing links in X_1 (See Figure 6.3b for an illustration of the modeled data). This is also a difficult link prediction problem since X_1 contains less than 0.008% of all possible links.



(a) A third-order tensor coupled another third-order tensor in one mode.

(b) A third-order tensor coupled with a matrix in one mode.

Figure 6.3. Digg Dataset, Comment Prediction Model and Digg Prediction Model.

6.2.2.1. Comment Prediction. For comment prediction, the relation between the user-story-comment (R5) is used to construct the tensor X_1 of size $I \times J \times K$ where I is the number of users, J is the number of stories and K is the number of comments. In X_1, an entry $X_1(i, j, k)$ indicates that a user i comment on news stories j with comment k; otherwise, it is 0. Since we address the link prediction problem in this study, we define the user-story matrix X_1 as:

$$X_1(i,j) = \begin{cases} 1 & \text{if user } i \text{ comment on news stories } j \text{ with comment } k \\ 0 & \text{otherwise} \end{cases} .$$

Additionally, the data includes the topics of the stories and extracted keywords from the stories' titles. We represented this data as the three-way tensor X_2. In our model the story-keyword-topic tensor has entries $X_2(j, m, n)$ of size $J \times M \times N$, where J is the number of stories, K is the number of keywords and M is the number of topics.

We form two coupled models in order to fill in the missing links in matrix X_1. For both models, we use Euclidean distance, KL divergence and IS divergence as cost functions in our nonnegative decomposition problems. In the first model, we applied

the coupled approach to a CP-style tensor decomposition model by analysing the tensor X_1 jointly with the additional tensor X_2 in order to solve the sparsity problem in X_1 effectively. This gives us the following model:

$$\hat{X}_1(i,j,k) = \sum_r A(i,r)B(j,r)C(k,r) \tag{6.23}$$

$$\hat{X}_2(j,m,n) = \sum_r B(j,r)D(m,r)E(n,r) \tag{6.24}$$

Here, we have two observed tensors, that share common factors; therefore, we have a coupled tensor factorization problem. The coupling matrix R with $|\alpha| = 5$, $|\nu| = 2$ for this model is defined as follows:

$$R = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix} \qquad \text{with} \quad \begin{matrix} \hat{X}_1 = \sum A^1 B^1 C^1 D^0 E^0 \\ \hat{X}_2 = \sum A^0 B^1 C^0 D^1 E^1 \end{matrix} . \tag{6.25}$$

Note that, X_1 and X_2 share the common factor matrix B with entries $B(j,r)$; we can interpret each row of $B(j,:)$ as story j's latent position in a $|r|$ dimensional *preferences* space. The factor matrix A with entries $A(i,r)$ represents the latent position of the user i in the same preferences space. A user i tends to comment the story j with comment k where the weight $A(i,r)B(j,r)C(k,r)$ is large for at least one r, i.e., there is a match between the users preference and what the story 'has to offer'.

Then, we applied the coupled approach using a Tucker decomposition to form our second model. Consequently, we get the following:

$$\hat{X}_1(i,j,k) = \sum_{p,q,r} A(i,p)B(j,q)C(k,r)D(p,q,r) \tag{6.26}$$

$$\hat{X}_2(j,m,n) = \sum_q B(j,q)E(m,q)F(n,q) \tag{6.27}$$

In this case, we have the following R matrix with $|\alpha| = 6$, $|\nu| = 2$

$$R = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix} \qquad \text{with} \quad \begin{matrix} \hat{X}_1 = \sum A^1 B^1 C^1 D^1 E^0 F^0 \\ \hat{X}_2 = \sum A^0 B^1 C^0 D^0 E^1 F^1 \end{matrix} . \tag{6.28}$$

where the factor B is shared by X_1 and X_2. In contrast to the coupled CP model sketched in Equation 6.23, this model assumes that a user i tends to comment the story j with comment k, with the weight $\sum_{p,q} A(i,p)B(j,q)C(k,r)D(p,q,r)$.

For comment prediction, we evaluate the prediction performance over different relational contexts. We observe that different combinations of the relations affect the prediction performance. We incorporate the digg relation R4 with R1 and R5: the relation between the user-story-comment triplets (R5) is used to construct the tensor X_1, the relation between story-keyword-topic triplets (R1) is used to construct the tensor X_2 and the relation between user-story (R4) is used to construct the matrix X_3.

In our model the user-story-comment tensor has entries $X_1(i,j,k)$. However, in our model we use a separate index t for the story index in X_3 instead of j. The rationale behind this choice is to relax the model as the entries in X_1 and X_2 are measuring distinct quantities: $X_1(i,j,k)$ represents whether the user i make comment on story j with comment k, while X_3 only indicates a vote (i.e. digg) by a specific user i on story t. The relation between the story entries j and t in X_1 and X_3 are coupled via a common factor over the users.

Finally, we form two coupled models in order to fill in the missing links in tensor X_1. For both models, we use Euclidean distance, KL divergence and IS divergence as cost functions in our nonnegative decomposition problems. In the first model, we applied the coupled approach to a CP-style tensor decomposition model by analysing the tensor X_1 jointly with the additional tensor X_2 and X_3 in order to solve the sparsity

problem in X_1 effectively. This gives us the following model:

$$\hat{X}_1(i,j,k) = \sum_r A(i,r)B(j,r)C(k,r) \tag{6.29}$$

$$\hat{X}_2(j,m,n) = \sum_r B(j,r)D(m,r)E(n,r) \tag{6.30}$$

$$\hat{X}_3(i,t) = \sum_r A(i,r)F(t,r) \tag{6.31}$$

Here, we have three observed tensors, that share common factors; therefore, we have a coupled tensor factorization problem. The coupling matrix R with $|\alpha| = 6$, $|\nu| = 3$ for this model is defined as follows:

$$R = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{with} \quad \begin{aligned} \hat{X}_1 &= \textstyle\sum A^1 B^1 C^1 D^0 E^0 F^0 \\ \hat{X}_2 &= \textstyle\sum A^0 B^1 C^0 D^1 E^1 F^0 \\ \hat{X}_3 &= \textstyle\sum A^1 B^0 C^0 D^0 E^0 F^1 \end{aligned} . \tag{6.32}$$

Note that, X_1, X_3 share the common factor matrix A with entries $A(i,r)$; we can interpret each row of $A(i,:)$ as user i's latent position in a $|r|$ dimensional *preferences* space.

Likewise, we applied the coupled approach using a Tucker decomposition to form our second model. Consequently, we get the following:

$$\hat{X}_1(i,j,k) = \sum_{p,q,r} A(i,p)B(j,q)C(k,r)D(p,q,r) \tag{6.33}$$

$$\hat{X}_2(j,m,n) = \sum_q B(j,q)E(m,q)F(n,q) \tag{6.34}$$

$$\hat{X}_3(t,j) = \sum_p A(i,p)G(t,p) \tag{6.35}$$

6.2.2.2. Digg Prediction. Similarly, for digg prediction, the relation between the user-story (R4) is used to construct the matrix X_1 of size $I \times J$ where I is the number of users and J is the number of stories. In matrix X_1, an entry $X_1(i,j)$ indicates that a user i vote (i.e. digg) on news stories j; otherwise, it is 0. Since we address the link prediction problem in this study, we define the user-story matrix X_1 as:

$$X_1(i,j) = \begin{cases} 1 & \text{if user } i \text{ vote (i.e. digg) on news stories } j \\ 0 & \text{otherwise} \end{cases} .$$

Additionally, the data includes the topics of the stories and extracted keywords from the stories' titles. We represented this data as the three-way tensor X_2. In our model the story-keyword-topic tensor has entries $X_2(j,k,m)$ of size $J \times K \times M$, where J is the number of stories, K is the number of keywords and M is the number of topics.

Once again, we form two coupled models in order to fill in the missing links in matrix X_1. For both models, we use Euclidean distance, KL divergence and IS divergence as cost functions in our nonnegative decomposition problems. In the first model, we applied the coupled approach based on CP-style tensor decomposition model by analysing the matrix X_1 jointly with the additional tensor X_2 in order to solve the sparsity problem in X_1 effectively. This gives us the following model:

$$\hat{X}_1(i,j) = \sum_r A(i,r)B(j,r) \tag{6.36}$$

$$\hat{X}_2(j,k,m) = \sum_r B(j,r)C(k,r)D(m,r) \tag{6.37}$$

The coupling matrix R with $|\alpha| = 4$, $|\nu| = 2$ for this model is defined as follows:

$$R = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \qquad \text{with} \quad \begin{matrix} \hat{X}_1 = \sum A^1 B^1 C^0 D^0 \\ \hat{X}_2 = \sum A^0 B^1 C^1 D^1 \end{matrix} . \tag{6.38}$$

Here, X_1 and X_2 share the common factor matrix B with entries $B(j,r)$; we can interpret each row of $B(j,:)$ as story j's latent position in a $|r|$ dimensional *preferences* space. The factor matrix A with entries $A(i,r)$ represents the latent position of the user i in the same preferences space. A user i tends to vote the story j where the weight $A(i,r)B(j,r)$ is large for at least one r, i.e., there is a match between the users preference and what the story 'has to offer'.

Then we use Tucker decomposition for the coupled approach and get:

$$\hat{X}_1(i,j) = \sum_p A(i,p)B(j,p) \tag{6.39}$$

$$\hat{X}_2(j,k,m) = \sum_{p,q,r} B(j,p)C(k,q)D(m,r)E(p,q,r) \tag{6.40}$$

6.3. Link Prediction with Variational Inference

In this section, we use the tensor factorization models proposed in Section 6.1 and Section 6.2 to demonstrate the usage of the proposed variational Bayesian PLTF (PLTF-VB) algorithm and variational Bayesian CTF (CTF-VB) algorithm for both missing link prediction and temporal link prediction problems.

More specifically, by using the models proposed above with variational approach, we show that a variational Bayesian method involves regularization effect even when the prior is non-informative, which is intrinsically different from the maximum a posteriori (MAP) approach. Previous studies on non-identifiable models showed that, when combined with full Bayesian estimation, regularization effect is significantly stronger than the MAP method [58, 59]. We also show that variational Bayesian method avoids overfitting (or in some cases this might cause underfitting) even when non-informative priors are provided by means of the regularization effect.

First, on the DBLP dataset, we study the temporal link prediction problem. We compare the performance of the proposed variational Bayesian PLTF (PLTF-VB)

with the standard PLTF (PLTF-EM) in terms of temporal link prediction. For this prediction task, we use the model given in Equation 6.3.

Then, on the single tensor X_1 of UCLAF dataset [56], we compare the performance of the proposed variational Bayesian PLTF (PLTF-VB) with the classical PLTF (PLTF-EM) in terms of missing link prediction recovery. We use standart CP tensor decomposition model given in Equation 3.18 and Tucker tensor decomposition model given in Equation 3.19 for this prediction task.

Finally, on the Digg dataset [18], we study missing link prediction problem by using proposed variational Bayesian CTF (CTF-VB) and GCTF for both comment prediction and digg prediction tasks. We use CP model and Tucker model for comment prediction task whereas we use CP model and Tucker model for digg prediction task. We also use the models given in Equation 6.29 and Equation 6.33 for comment prediction task which includes relation (R4) as side information.

We use the algorithms that implements variational fixed point update equation given in Figure 4.1 and Figure 5.1 to demonstrate the performances of the models given throughout this section. We use the equation given in Equation 4.18 for variational bound computation.

This chapter verifies that by using the tensor factorization frameworks defined in previous chapters, we can easily propose a solution for link prediction task with different models and loss functions. We are not restricted in using a standard model topology, we are able to design application specific models which is guided by the needs of the application.

In Chapter 7, with comprehensive experiments we will assess the performance of various alternative tensor models, given throughout this section, and loss functions in terms of link prediction. In addition, we will compare the performances of EM and VB approaches.

7. EXPERIMENTS

In this section, we conduct experiments on several datasets in order to evaluate the performance of the tensor factorization models proposed in Chapter 6.

First, we design experiments to evaluate the performance of our model given in Section 6.1 in terms of capturing the temporal profiles in the time factor matrix. Our aim is to demonstrate that, this model can handle the temporal link prediction problem on a single tensor by using PLTF framework. In addition, we want to compare the performances of different cost functions on this application to indicate the importance of using the right loss function for accurate prediction.

Then, we compare the performances of low-rank approximation of a single tensor and coupled tensor factorizations in terms of missing link prediction problem by using different loss functions. We use the models given in Section 6.2 . We expect to see that coupled tensor factorizations outperform low-rank approximations of a single tensor especially when the fraction of unobserved elements is high. In addition, we want to show the effect of loss function selection to prediction of missing links.

Finally, we use both single and coupled tensor factorization models that are used in previous experiments with variational Bayesian approach to compare our variational algorithms with the classical ones. As we explained in Section 6.3, variational Bayesian method involves self-regularization effect, so it avoids overfitting. Consequently, we expect that variational Bayesian approach will increase the prediction performance of the models used for both temporal and missing link prediction applications.

Table 7.1 summarizes the experimental settings of this section. This table lists the datasets used throughout this section and the corresponding experiments' sections. It also shows the tensor factorization methods and algorithms used for the each dataset.

All experiments were performed using MATLAB 2010b on 2.4GHz Core i5 520M

Table 7.1. Overview of the Experimental Setting.

Dataset	Factorization Method	Link Prediction Type	Section
Network Traffic (Géant)	Single TF (PLTF)	Temporal	Section 7.1.2
DBLP	Single TF (PLTF)	Temporal	Section 7.1.3
	Single TF (PLTF-VB)	Temporal	Section 7.3.1
UCLAF	Coupled TF (GCTF)	Missing	Section 7.2.1
	Coupled TF (CTF-VB)	Missing	Section7.3.2
Digg	Coupled TF (GCTF)	Missing	Section 7.2.2
	Coupled TF (CTF-VB)	Missing	Section 7.3.3

processor and 4GB RAM. Timings were performed using MATLAB's tic and toc functions.

Evaluation Metrics. In our experiments, as evaluation metrics, we use Area Under the Receiver Operating Characteristic Curve (AUC), P@K (the precision of the top K results) for link prediction results and Relative Error (RE) for tensor completion results.

AUC. ROC curve, is a graphical plot which illustrates the performance of a binary classifier system as its discrimination threshold is varied [60].In a ROC curve the true positive rate (Sensitivity) is plotted in function of the false positive rate (100-Specificity) for different cut-off points of a parameter. Each point on the ROC curve represents a sensitivity/specificity pair corresponding to a particular decision threshold. Therefore, the closer the ROC curve is to the upper left corner, the higher the overall accuracy of the test. The area under the ROC curve (AUC) is a measure of how well a parameter can distinguish between two groups. When using normalized units, AUC is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming 'positive' ranks higher than 'negative').

Link prediction datasets are characterized by extreme imbalance, i.e., the number

of links known to be present is often significantly less than the number of edges known to be absent. This issue motivates the use of AUC as a performance measure since AUC is viewed as a robust measure in the presence of imbalance [61].

P@K. Precision at k (P@K) measures the precision at a fixed number of retrieved items (i.e., top K) of the ordered list $r\prime$ and the unordered list r [62].Assume $TopK$ and $TopK\prime$ are the retrieved items of r and $r\prime$ respectively, then the P@K is defined as $P@K = \frac{|TopK \cap TopK\prime|}{K}$

We use P@K to measure the performance of prediction. As might be expected, the accuracy of link prediction also varies according to the precision measure chosen. Due to its robustness P@K is a frequently used measure in the domain of information retrieval (IR) and machine learning (ML) [54]. For example, we compute the precision based on the top 10 stories retrieved for each user on Digg dataset. The overall P@10 for the set of users is computed by taking the mean of P@10 per user. We use P@1000 that computes how many correct links (true positives) are in the first top 1000 scores on DBLP dataset to understand the behavior of different set-ups better.

RE. As an evaluation metric for tensor completion performance, we use the *Relative Error* metric, which is usually employed to express accuracy of an estimate. Formally, the relative error between an exact value X and an estimated value $\hat{X}$ is given by $\frac{\|X-\hat{X}\|}{\|X\|}$.

The following results show the *average link prediction performance of 10 independent runs* in terms of AUC and P@K.

7.1. Performance of PLTF

In this section, we assess the performance of the tensor factorization models on a single tensor in terms of both missing and temporal link prediction problem. We

present results of experiments involving link prediction for multiple time steps. First, we use synthetic data which includes several different periodic patterns to explore the ramifications of temporal predictions. Then, we design experiments to evaluate the performance of our models on Network Traffic Data (for missing link prediction) and DBLP dataset (for temporal link prediction).

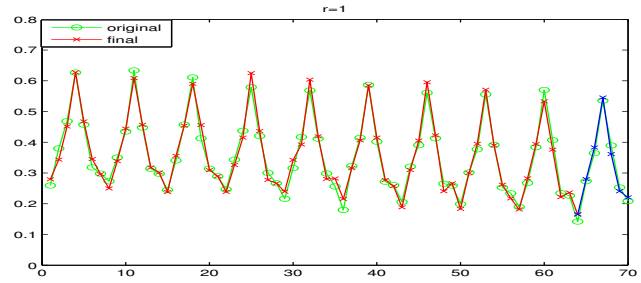
7.1.1. Synthetic Dataset

7.1.1.1. Experimental Setting. We design experiments to evaluate the performance of our temporal link prediction method defined in Section 6.1 in terms of capturing the temporal profiles in the time factor matrix. Different components may have different trends, for example, they may have increasing, decreasing, or steady profiles. Our aim is to use the periodic information, predicting even further out in time.

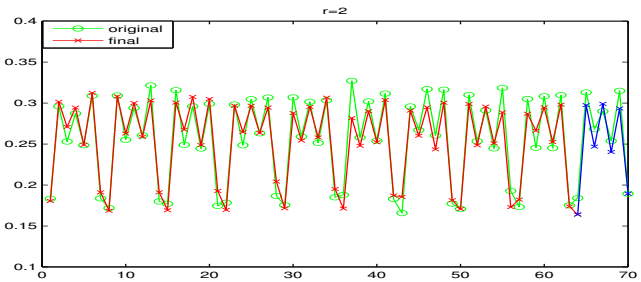
We generate simulated data that shows connections between two sets of entities over time. In our simulated example, we assume that each time t corresponds to one day and that the temporal profiles correspond roughly to a seven-day period ($L = 7$). We assume that our data can be modeled by $R = 3$ components. We assume we have P periods of training data, so that we have $T = LP$ time observations to train on. In our tests, we use $L = 7$ and $P = 9$. We also assume that we have 1 period of testing data. We add white Gaussian noise to the data after creating noise-free versions of the training and testing tensors.

7.1.1.2. Results. The aim of this study is to predict the entries in testing data. So, we replace all nonzeros in the test tensor with ones (i.e., positive links) and leave the rest as zeros (i.e., no link). Since the data is noisy, the fit of the model to the data is not perfect. Figure 7.1 shows the three different temporal patterns in the data. The dotted line is the original data, the first portion of which is used to generate the training data. The crossed line is the pattern that is extracted via the model defined in Equation 6.3. Note that it is generally a good fit to the true data shown as dotted. Finally, the last segment ($t = 64 - 70$) of the crossed line is the prediction using the pattern extracted

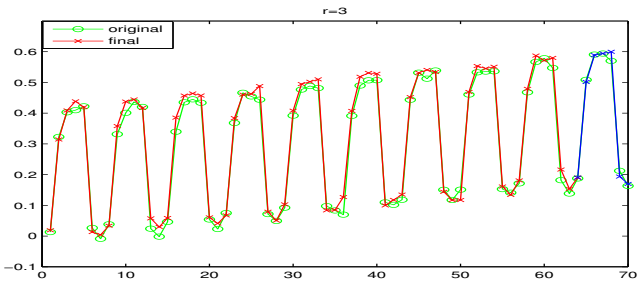
by our model.



(a) r = 1



(b) r = 2



(c) r = 3

Figure 7.1. Temporal patterns: the dotted line is the *true* data, the crossed line is the temporal pattern that is computed by model given in Equation 6.3, the last segment $(t = 64 - 70)$ of the crossed line is the prediction of the test period.

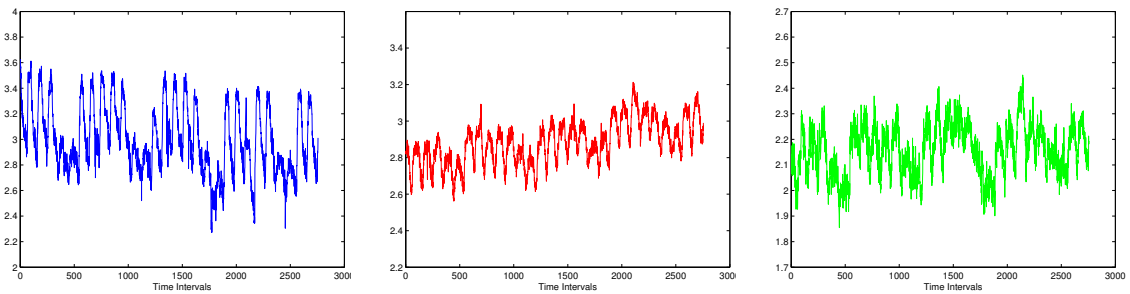
This experiment shows that the method given in Section 6.1 performs well even when a large percentage of the strongest signals (i.e., link information) in the training data is missing. Such robustness is crucial for applications where noise or missing data is common, for example, analysis of computer network traffic.

7.1.2. Network Traffic Data

7.1.2.1. Experimental Setting. We design experiments to evaluate the performance of our models in terms of recovering missing entries of a tensor. By setting different amounts of data to missing in source-destination-time tensor X, we compare (i)CP model, (ii) Tucker model and (iii) Model given in Equation 6.3 on X using both KL-divergence and the Euclidean as cost functions.

In all experiments, we use randomly missing entries. We set number of components R to 2 and number of linear trends T to 3.

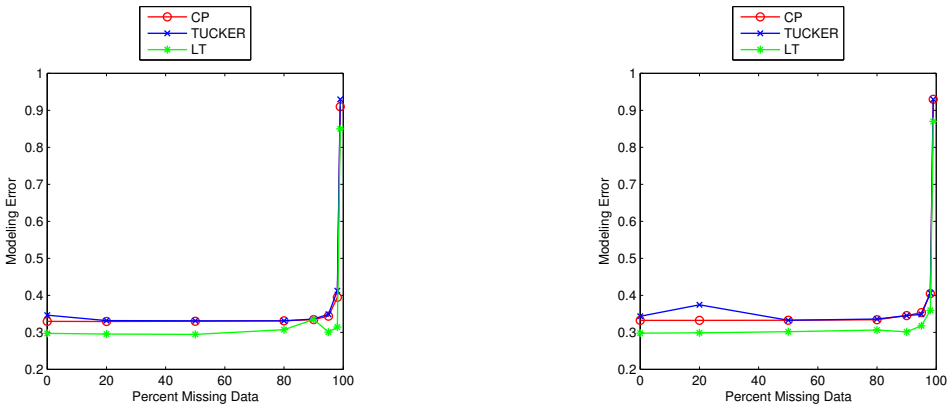
7.1.2.2. Results. First, we show that model given in Equation 6.3 with $R = 2$ and $M = 3$ fits the data well but not perfectly and there is some unexplained variation left in the residuals. We can see the three temporal patterns in Figure 7.2.



(a) m = 1 (b) m = 2 (c) m = 3

Figure 7.2. Temporal patterns captured by model in Equation 6.3.

Then, in order to demonstrate the power of our model in Equation 6.3, we compared the missing entry recovery performances of standard CP and Tucker models with model in Equation 6.3 using EUC and KL cost functions at different amounts, i.e., $\{20, 50, 80, 90, 98, 99\}$, of randomly unobserved elements. For both of the cost functions, model in Equation 6.3 outperforms the standard models. Figure 7.3 shows the comparison of CP, Tucker and model in Equation 6.3 with different cost functions for different amounts of missing data amounts for Géant data. As we can see, relative error is lower for model in Equation 6.3 than the standard models.



(a) EUC (b) KL

Figure 7.3. Tensor completion score of CP, Tucker and Model in Equation 6.3 for different amounts of missing data amounts for Géant data.

In order to assess the performance of our models in terms of recovering missing data, randomly chosen entries of X are set to missing. Then, the extracted factor matrices are used to reconstruct the data and fill in the missing entries. These recovered values are compared with the actual values based on the relative error (RE). We observe that the average ME is around 0.29 for model in Equation 6.3 and 0.33 for CP and TUCKER models when there is little missing data and increases very slowly as we increase the amount of missing data. The average ME is only slightly higher, approximately 0.31 for model in Equation 6.3 and 0.40 for CP and TUCKER models, even when 95% of the entries are missing. However, the average ME increases sharply when the amount of missing data is 99%.

7.1.3. DBLP Dataset

7.1.3.1. Experimental Setting. We design experiments to evaluate the performance of our model given in Equation 6.3 in terms of temporal link prediction problem on DBLP dataset. We compare the performances of Euclidean distance, KL divergence and IS divergence by using this model.

7.1.3.2. Results. First, we demonstrate the effect of the cost function modeling the data. Figure 7.4 illustrates the performance of Euclidean distance, KL divergence and

IS divergence for proposed model and it clearly shows that KL cost function seems to perform better than EUC and IS. Then, we compare our method with the existing

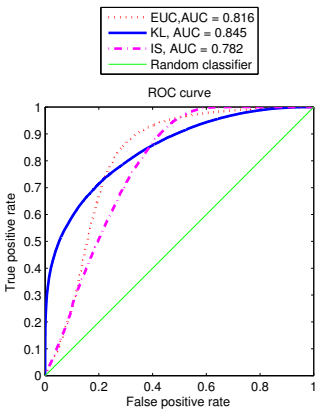


Figure 7.4. Average prediction result of new links in the test sets.

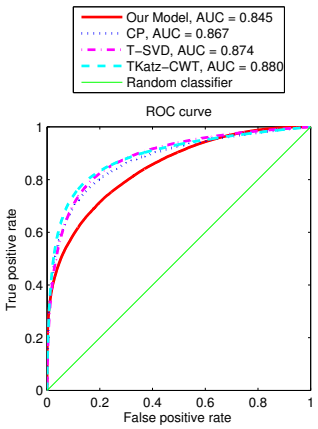


Figure 7.5. Average prediction result of several algorithms on DBLP data.

methods that are used for temporal link prediction problem described in [6]. Figure 7.5 shows that, the other methods outperforms our method in terms of AUC metric. In all the experiments, we take the value of smoothing factor α as 0.5. However, we will show in Section 7.3.1, our model in Equation 6.3 outperforms these methods when we use variational inference. In addition, we compute how many correct links (true positives) are in the first top 1000 scores for every test set and cost function. We will show these results in Table 7.4 as well as the variational Bayes's results. Finally, we compare the average timings of the algorithms. PLTF-EM takes ∼1550 sec. and PLTF-VB takes ∼2099 sec. across all training sets in average while CP takes ∼1304 sec [6]. The CP and PLTF-EM methods are quite faster than PLTF-VB.

7.2. Performance of Coupled Models

In this section, we assess the performance of the coupled models proposed in Section 6.2.1 and Section 6.2.2 in terms of missing link prediction.

7.2.1. UCLAF Dataset

First, we demonstrate that coupled tensor factorizations outperform low-rank approximations of a single tensor in terms of missing link prediction. Then we compare different tensor models and loss functions and show that selection of the tensor model and loss function is significant in terms of link prediction performance, especially when the data is sparse. Furthermore, we study the case with completely missing slices, which corresponds to the cold-start problem in our link prediction setting and demonstrate that it is still possible to predict missing links using the proposed coupled models whereas low-rank approximations of a single tensor would fail to do so.

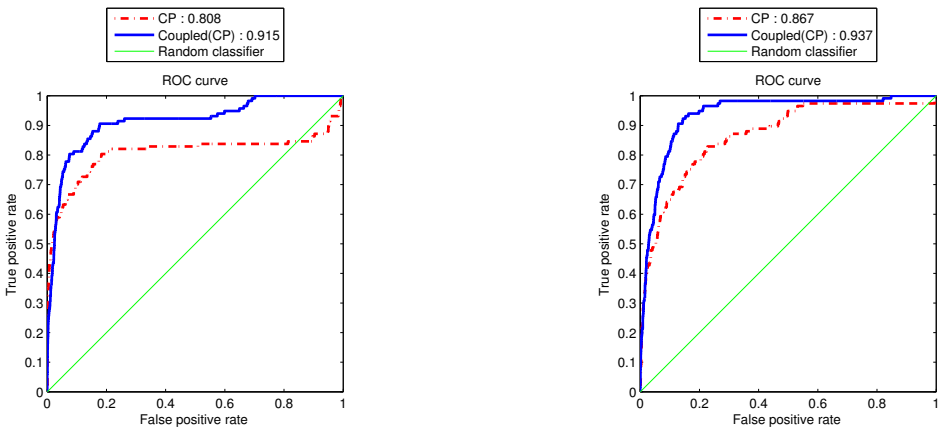
7.2.1.1. Experimental Setting. We design experiments to evaluate the performance of our models given in Section 6.2.1. By setting different amounts of data to missing in user-location-activity tensor X_1, we compare the following models using both KL-divergence and the Euclidean as cost functions:

- *Low-rank approximations of a single tensor:* (i) CP and (ii) Tucker factorization of user-location-activity tensor X_1,
- *Coupled tensor factorizations:* (i) CP factorization of X_1 coupled with factorization of user-location matrix X_2 and location-feature matrix X_3 (ii) Tucker factorization of X_1 coupled with factorization of X_2 and X_3.

We use two patterns of missing data: (i) randomly missing entries and (ii) randomly missing slices. In all experiments, number of components, i.e., number of columns in each factor matrix, Z_i, is set to 2.

7.2.1.2. Results. In order to demonstrate the power of coupled analysis, we compared the link prediction performance of standard CP and Tucker models with coupled ones using EUC and KL cost functions at different amounts, i.e., $\{40, 60, 80, 90, 95\}$, of randomly unobserved elements. For all cases, coupled models outperform the standard models clearly. Figure 7.6 shows the comparison of CP and coupled CP models with

different cost functions when 80% of the data is missing. As we can see, the coupled models that try to use as much additional information as possible to help alleviate the data sparsity issue perform better than the standard models; in particular, when the percentage of missing data is high. When the fraction of missing data was more than 80%, the standard models could not find a solution.



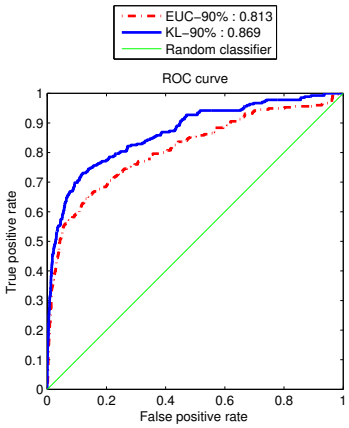
(a) EUC with 80% missing (b) KL with 80% missing

Figure 7.6. Comparison of CP and Coupled(CP) models.

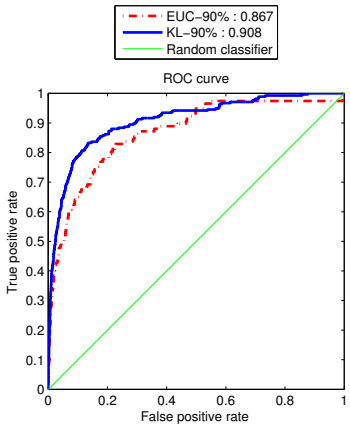
In order to demonstrate the effect of the cost function modeling the data, we have also carried out experiments on both coupled CP and Tucker models at different missing data fractions. For all cases, the KL cost function seems to perform better than EUC, especially when the fraction of missing entries is high. Figure 7.7 illustrates the performance of Euclidean distance and Kullback-Leibler divergence for both coupled CP and Tucker models when 90% of the data is unobserved.

Finally, Figure 7.8 shows the comparison of coupled CP and Tucker models in order to illustrate the tensor model modeling the data best. We can see that Tucker model outperforms the CP model; because Tucker model is more flexible due to the full core tensor which is helpful for us to explore the structural information embedded in the data.

We also study the missing slice problem, which is particularly important in link prediction because we may often have new users starting to use an application, e.g., a location-activity recommender system. Since they are new users, they will have no

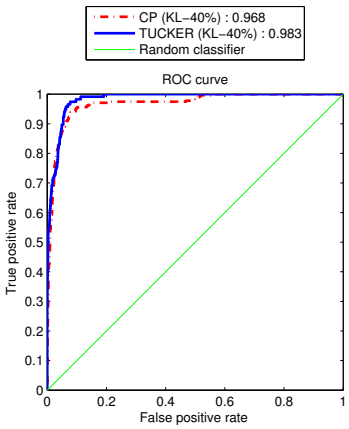


(a) Coupled (CP) Model

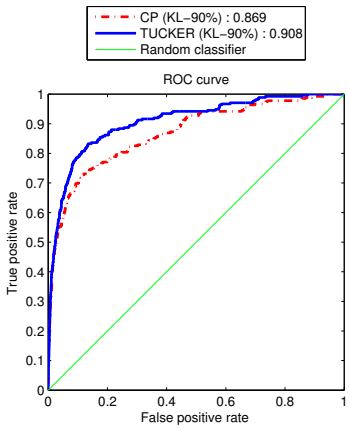


(b) Coupled (Tucker) Model

Figure 7.7. Comparison of EUC distance and KL divergence with 90% missing data.



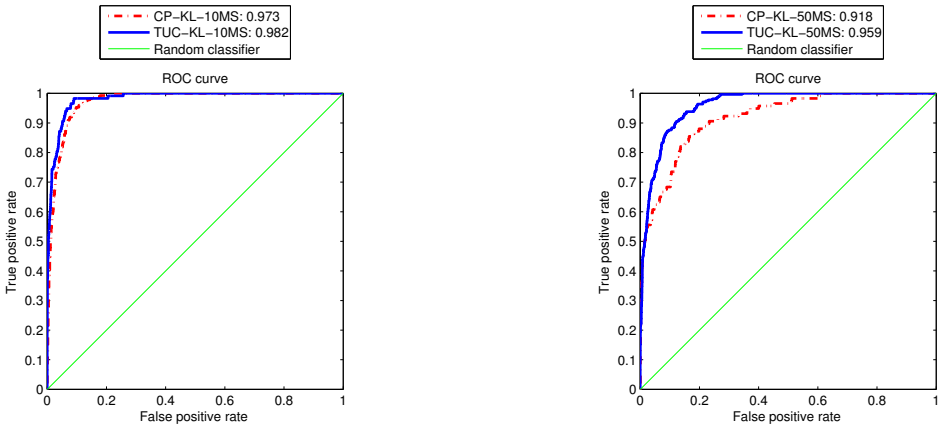
(a) CP vs Tucker - 40% missing



(b) CP vs Tucker - 90% missing

Figure 7.8. Comparison of Coupled CP and Tucker models with KL.

entry in X_1, i.e., a completely missing slice. It is not possible to reconstruct a missing slice of a tensor using its low-rank approximation. A similar argument is valid in the case of matrices for completely missing rows/columns [24]. In such cases, additional sources of information will be useful [63] to make recommendations to new users. We observe that our coupled models could predict the links when there is no information about a user in tensor X_1, by utilizing the additional sources of information. We test this case by setting randomly missing slices in X_1. Figure 7.9 demonstrates the performance of coupled models with KL divergence when 10 users' data and 50 users' data are missing. Also note that Tucker is superior to CP as the amount of missing data increases.



(a) CP and Tucker - 10MS (b) CP and Tucker - 50MS

Figure 7.9. Link prediction result with missing slices and KL cost.

7.2.2. Digg Dataset

In this section, we compare different tensor models and loss functions and show that selection of the tensor model and loss function is significant in terms of link prediction performance, especially when the fraction of unobserved elements is high.

7.2.2.1. Experimental Setting. We design experiments to evaluate the performance of our models given in Section 6.2.2 in terms of missing link prediction on Digg dataset. Based on the Digg scenario, we have two prediction tasks on Digg dataset: (i) comment prediction and (ii) digg prediction that are explained in Section 6.2.2

For the first prediction task, by setting different amounts of data to missing in user-story-comment tensor X_1, we compare the following models using Euclidean distance, KL and IS divergences as cost functions:

- *Low-rank approximations of a single tensor:* (i) CP and (ii) Tucker factorization of user-story-comment tensor X_1,
- *Coupled tensor factorizations:* (i) CP factorization of X_1 coupled with factorization of story-keyword-topic tensor X_2 (ii) Tucker factorization of X_1 coupled with factorization of X_2,
- *Coupled tensor factorizations:* (i) CP factorization of X_1 coupled with factoriza-

tion of X_2 and user-story matrix X_3 (ii) Tucker factorization of X_1 coupled with factorization of X_2 and X_3.

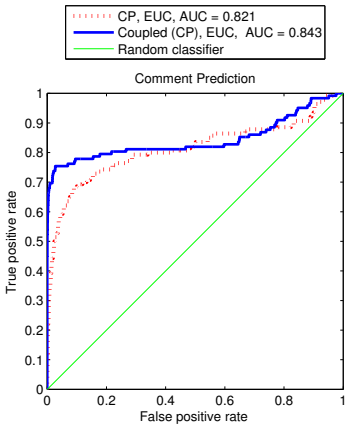
For the second prediction task, by setting different amounts of data to missing in user-story matrix X_1, we compare the following models using Euclidean distance, KL and IS divergences as cost functions:

- *Coupled tensor factorizations:* (i) Matrix factorization of X_1 coupled with CP factorization of story-keyword-topic tensor X_2 (ii) Matrix factorization of X_1 coupled with Tucker factorization of X_2.

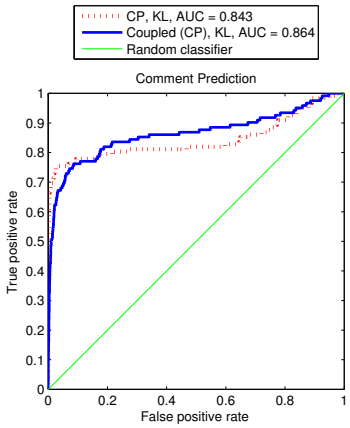
In all experiments, we use randomly missing entries and we set number of components, i.e., number of columns in each factor matrix, Z_i, to 5.

7.2.2.2. Results. In order to demonstrate the power of coupled analysis, we compared the link prediction performance of standard CP and Tucker models with coupled ones using EUC, KL and IS cost functions at different amounts, i.e., $\{40, 80, 90\}$, of randomly unobserved elements. Here, we show results of the experiments on *both comment and digg prediction tasks*. For all cases, coupled models outperform the standard models clearly. Figure 7.10 shows the comparison of CP and coupled CP models with different cost functions when 40% and 80% of the data are missing. As we can see, the coupled models that try to use as much additional information as possible to help alleviate the data sparsity issue perform better than the standard models; in particular, when the percentage of missing data is high. When the fraction of missing data was more than 80%, the standard models could not find a solution.

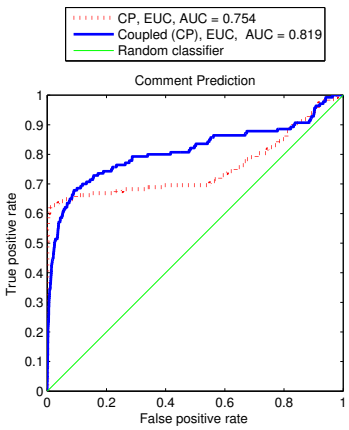
In order to demonstrate the effect of the cost function modeling the data, we have also carried out experiments on both coupled CP and Tucker models at different missing data fractions. For all cases, the IS cost function seems to perform better than EUC and KL for both of the prediction tasks, especially when the fraction of missing entries is high. Figure 7.11 and Figure 7.12 illustrates the performance of Euclidean



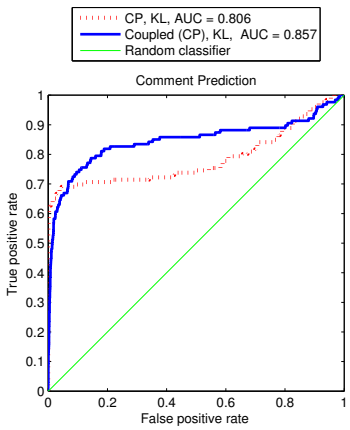
(a) EUC with 40% missing



(b) KL with 40% missing



(c) EUC with 80% missing

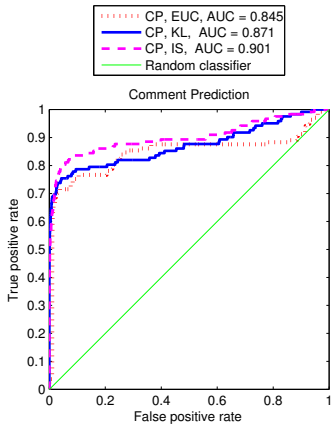


(d) KL with 80% missing

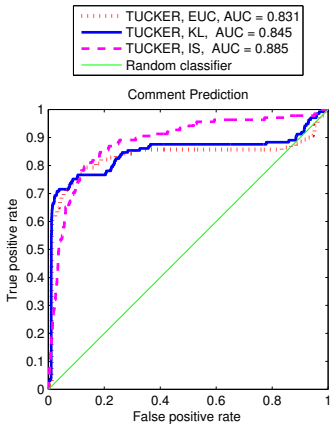
Figure 7.10. Comparison of CP and Coupled(CP) models for comment prediction.

distance, KL divergence and IS divergence for both coupled CP and Tucker models when 40% and 90% of the data is unobserved respectively. When the missing data rate become higher, the difference between performances of cost functions become clearer.

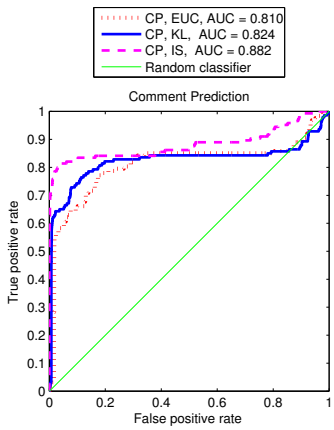
Then, we show the comparison of coupled CP and Tucker models in order to illustrate the tensor model that models the data best in Figure 7.13 and Figure 7.14. We can see that CP model outperforms the Tucker model in exploring the structural information embedded in the data. In addition, we compare the average timings of the two models. CP requires ∼568 sec. and TUCKER requires ∼2654 sec. for comment prediction while CP requires ∼260 sec. and TUCKER requires ∼1435 sec. for digg prediction in average.



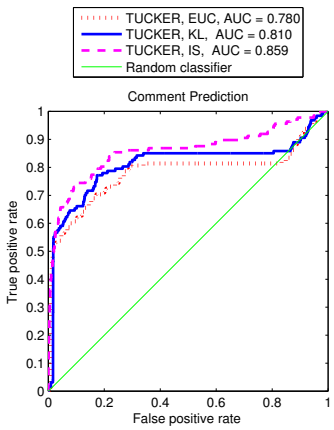
(a) CP - 40% missing



(b) Tucker - 40% missing



(c) CP - 90% missing

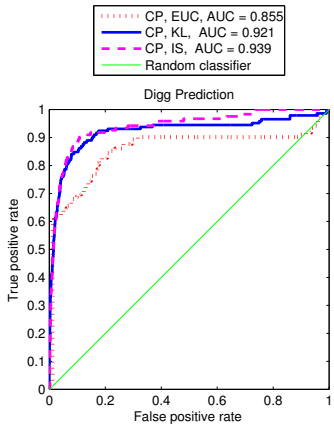


(d) Tucker - 90% missing

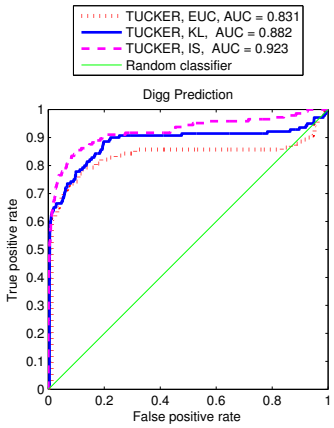
Figure 7.11. Comparison of EUC, KL and IS on comment prediction.

Afterwards, in order to demonstrate the effect of various relational context on comment prediction, we carried out experiments with different relational contexts. We observe that different combinations of the relations affect the prediction performance. Model of one of these combinations is given in Section 6.2.2. In this model, we incorporate the relation R4 with R1 and R5 to increase prediction performance on users' comment activities. Figure 7.15 shows the comparison of models given in Equation 6.23 and Equation 6.29 when 40% and 90% of the data is unobserved respectively.

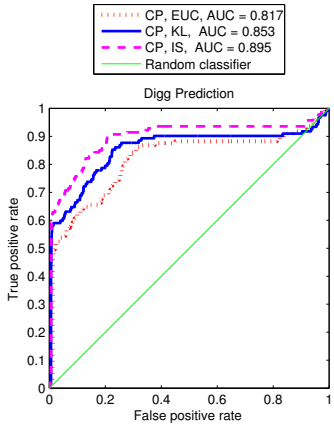
Finally, we compare the prediction performances of our models in Equation 6.23 and Equation 6.23 under 40% and 90% missing data with the prediction performance of Metafac (MFT) methods proposed by Lin *et al.* [18] in terms of P@10 metric. The results of digg and comment predictions are given in Table 7.2. We observe that our



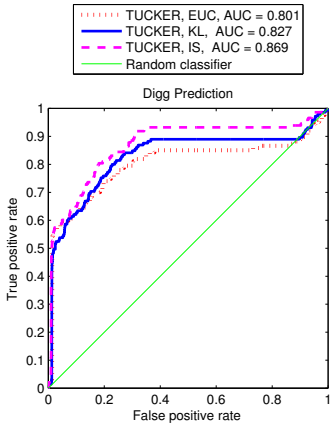
(a) CP - 40% missing



(b) Tucker - 40% missing

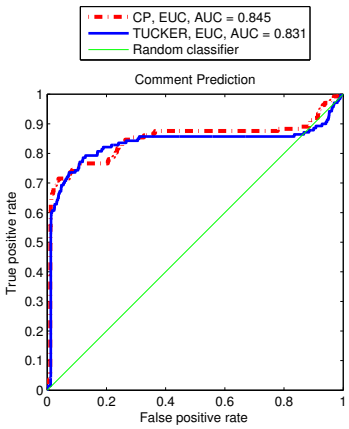


(c) CP - 90% missing

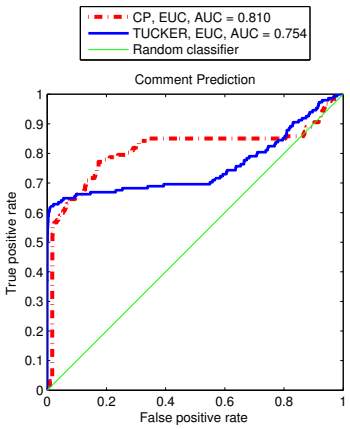


(d) Tucker - 90% missing

Figure 7.12. Comparison of EUC, KL and IS on comment prediction.

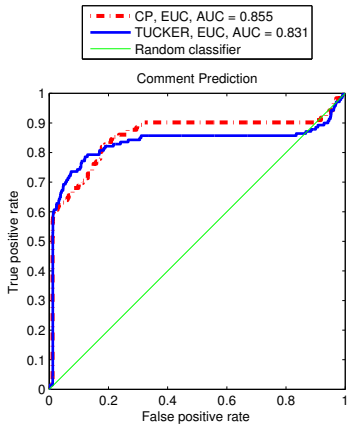


(a) CP vs Tucker - 40% missing

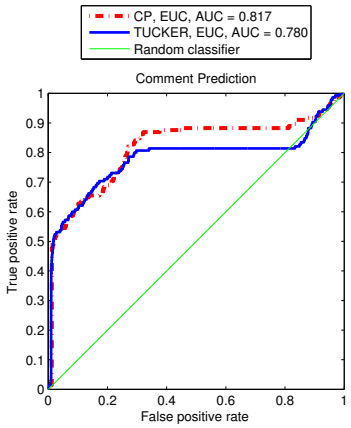


(b) CP vs Tucker - 90% missing

Figure 7.13. Comparison of Coupled CP and Tucker models with EUC on comment prediction.

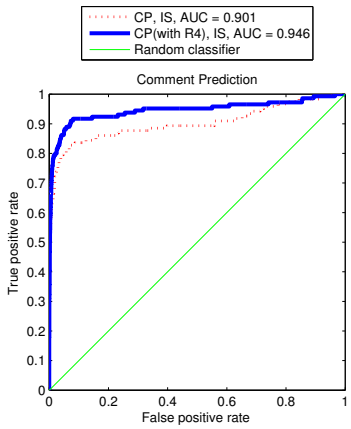


(a) CP vs Tucker - 40% missing

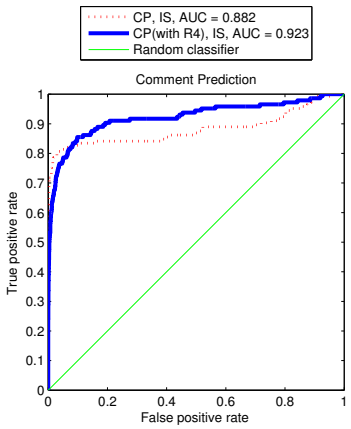


(b) CP vs Tucker - 90% missing

Figure 7.14. Comparison of Coupled CP and Tucker models with EUC on digg prediction.



(a) Model Comparison, IS - 40% missing



(b) Model Comparison, IS - 90% missing

Figure 7.15. Comparison of coupled models with different relations and EUC cost.

model significantly outperforms the MFT approach, which have outperformed many link prediction methods in [18]. Because, apart from their work, we use the integrated Digg dataset instead of the segmented data and we deal with missing link prediction problem.

7.3. Performance of Variational Bayesian Approach

In this section, we assess the performance of PLTF-VB and CTF-VB for both missing link prediction and temporal link prediction problem. We both use low-rank and coupled tensor factorization models in our experiments.

Table 7.2. The average prediction performance for digg and comment prediction, evaluated by P@10 values, with model in Equation 6.23 and Equation 6.36.

	Digg Prediction		Comment Prediction	
	40%	90%	40%	90%
GCTF-EUC	0.7154	0.6615	0.3632	0.3241
GCTF-KL	0.7414	0.6865	0.3751	0.3383
GCTF-IS	**0.7858**	**0.7479**	**0.4075**	**0.3846**

Table 7.3. The average prediction performance for digg and comment prediction, evaluated by P@10 values, with MFT.

	Digg Prediction	Comment Prediction
MFT [18]	0.543 ± 0.007	0.135 ± 0.001

7.3.1. DBLP Dataset

We study the temporal link prediction problem on the DBLP dataset. This time, we compare the performance of the proposed variational Bayesian PLTF (PLTF-VB) with the standard PLTF (PLTF-EM) in terms of temporal link prediction. For this prediction task, we use the model given in Equation 6.3.

7.3.1.1. Results. We compare the performance of standard and variational approaches of PLTF on proposed model given in Equation 6.3. Figure 7.16 shows that variational approach outperforms the standard approach clearly.

Then, we compare our method with the existing methods that are given in Section 7.1.3.2. Figure 7.17 shows that, our model with variational inference outperforms the other methods in terms of AUC.

Finally, in order to compare our method with the existing methods [6] (i.e. TSVD-CWT, TKatz-CWT, CP ...) and to understand the behavior of different link predictors better, we also compute how many correct links (true positives) are in the first top 1000

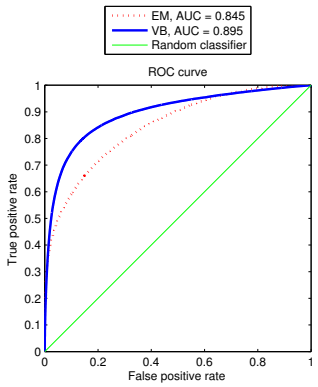


Figure 7.16. Comparison of PLTF-EM and PLTF-VB on DBLP dataset.

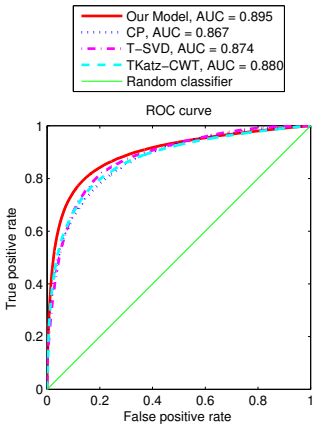


Figure 7.17. Average prediction result of several algorithms on DBLP data.

Table 7.4. The average prediction performance evaluated by $P@1000$ values.

	2001	2002	2003	2004	2005	2006	2007
VB-EUC	**0.131**	**0.148**	**0.138**	**0.135**	**0.135**	**0.139**	**0.148**
VB-KL	0.129	0.148	0.135	0.132	0.134	0.131	0.144
EM-EUC	0.098	0.106	0.095	0.101	0.094	0.089	0.111
EM-KL	0.094	0.095	0.097	0.094	0.091	0.086	0.115
CP [6]	0.087	0.097	0.078	0.099	0.116	0.091	0.093
TSVD [8]	0.104	0.124	0.96	0.105	0.117	0.110	0.95
TKatz-CWT [64]	0.107	0.124	0.102	0.105	0.117	0.109	0.99

scores when we aim to predict *previously unseen links*. We use seven of the eight test sets for this experiment because in [6], they use these seven sets. Table 7.4 shows these results for every test set and cost function: As we can see from the Table 7.4, variational approach gives the best result.

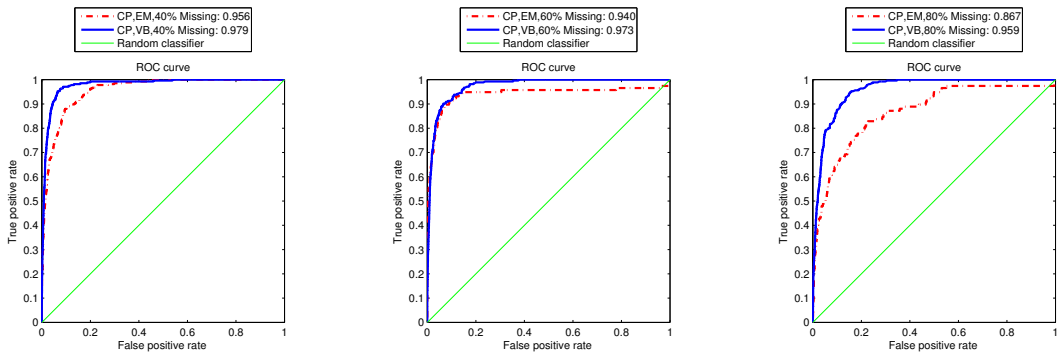
7.3.2. UCLAF Dataset

We study the missing link prediction problem and compare the performance of the proposed variational Bayesian PLTF (PLTF-VB) with the standard PLTF (PLTF-EM) in terms of missing link prediction recovery on the UCLAF dataset. We use a

standard CP tensor decomposition model and Tucker tensor decomposition model for this prediction task.

We now compare the standard PLTF, i.e., PLTF-EM, with the proposed variational method, i.e., PLTF-VB, on a missing link prediction task.

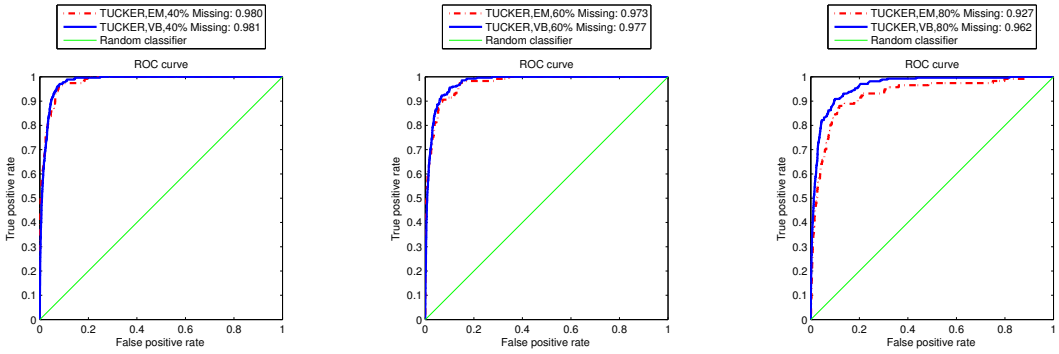
7.3.2.1. Results. We compare the performance of standard and variational approaches of PLTF on both CP and Tucker tensor factorization models at different amounts, i.e., $\{40, 60, 80\}$, of randomly unobserved elements. In these experiments, the incomplete tensor is factorized using either a CP or a Tucker model and the extracted factor matrices are used to construct the full tensor and estimate scores for missing links. For all cases, variational approach outperforms the standard approach clearly. Figure 7.18 and Figure 7.19 show the comparison of PLTF-VB and PLTF-EM methods for the CP model given in Equation 2.4 and the Tucker model given in Equation 2.5, respectively, when $\{40, 60, 80\}$ of the data is missing. As we can see, the variational methods due to implicit self-regularization effect [65], perform better than the standard methods; in particular, when the percentage of missing data is high. Furthermore, note that the Tucker model outperforms the CP model; because Tucker model is more flexible due to the full core tensor which is helpful for us to explore the structural information embedded in the data.



(a) CP, 40% missing (b) CP, 60% missing (c) CP, 80% missing

Figure 7.18. Comparison of PLTF-EM and PLTF-VB methods under missing data case with CP model.

Moreover, we study the performance of PLTF-EM and PLTF-VB in terms of



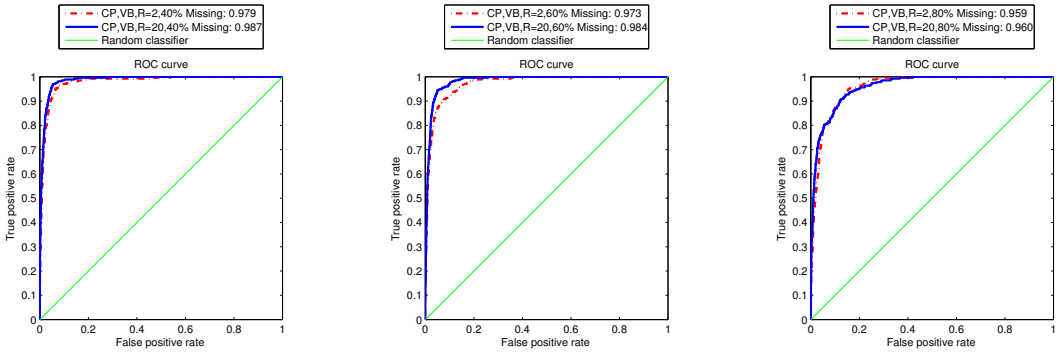
(a) Tucker, 40%missing (b) Tucker, 60% missing (c) Tucker, 80% missing

Figure 7.19. Comparison of PLTF-EM and PLTF-VB methods under missing data case with Tucker model.

robustness to model order selection. As model order increases, the prediction performance of PLTF-EM drops. This is as expected since PLTF-EM is prone to overfitting and the increase in model order causes an increase in the number of free parameters that, in turn, enlarges penalty term in PLTF-EM. On the other hand, the prediction performance of the variational approach is not very sensitive to the model order and is immune to overfitting since Bayesian approach alleviates over-fitting by integrating out all model parameters [50]. We compare the prediction performances of PLTF-EM and PLTF-VB methods for the CP tensor model when the component number R is equal to 2 and 20 and for different amounts of missing data, i.e., $\{40, 60, 80\}$ of the data is missing. Figure 7.20 and Figure 7.21 demonstrate that when the model order increases, the prediction performance of PLTF-VB approach stays almost same; however, the prediction performance of PLTF-EM approach declines as expected.

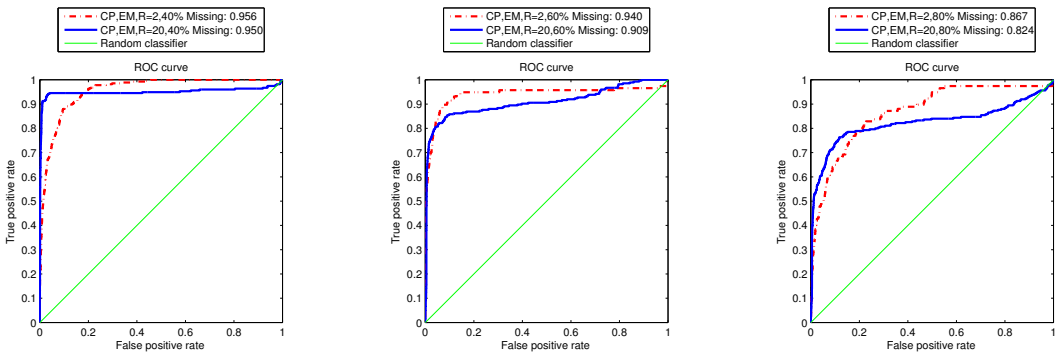
7.3.3. Digg Dataset

We study the missing link prediction problem and compare the performance of the proposed variational Bayesian CTF (CTF-VB) with the standard Generalized Coupled Tensor Factorization with KL cost (named as CTF-EM) in terms of missing link prediction recovery on the Digg dataset. We use models given in Equation 6.23, Equation 6.29 and Equation 6.36 for this prediction tasks.



(a) PLTF-VB, 40%missing (b) PLTF-VB, 60%missing (c) PLTF-VB, 80%missing

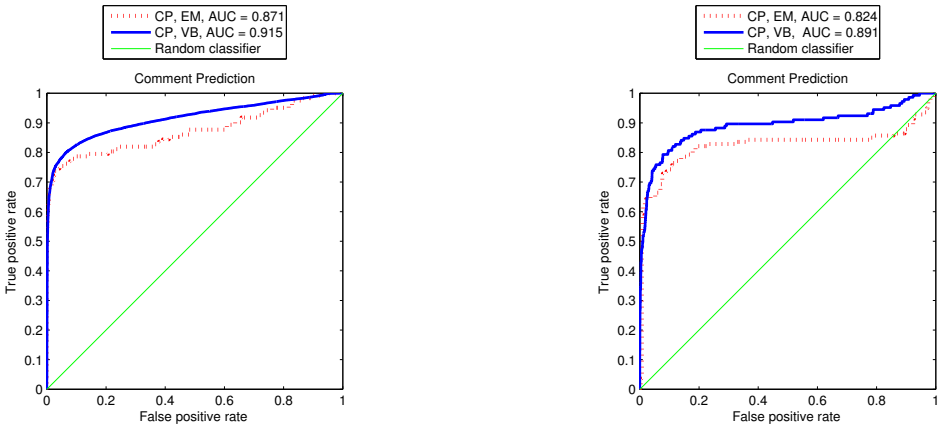
Figure 7.20. Effect of model order on the performance of PLTF-VB approach for CP model for different amounts of missing data with $R = 2$ and $R = 20$.



(a) PLTF-EM, 40%missing (b) PLTF-EM, 60%missing (c) PLTF-EM, 80%missing

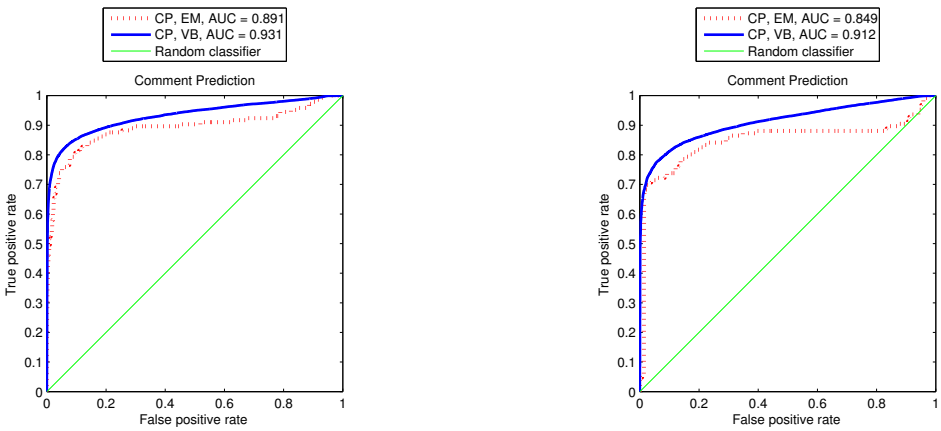
Figure 7.21. Effect of model order on the performance of PLTF-EM approach for CP model for different amounts of missing data with $R = 2$ and $R = 20$.

7.3.3.1. Results. We compare the performance of standard and variational approaches of coupled tensor factorizations on proposed models at different amounts, i.e., $\{40, 60, 80, 90\}$, of randomly unobserved elements. In these experiments, the incomplete tensor is factorized using the proposed models and the extracted factor matrices are used to construct the full tensor and estimate scores for missing links. For all cases, variational approach outperforms the standard approach clearly. Figure 7.22, Figure 7.23 and Figure 7.24 show the comparison of CTF-VB and CTF-EM methods for the models given in Equation 6.23, Equation 6.29 and Equation 6.36, respectively, when $\{40, 90\}$ of the data is missing.



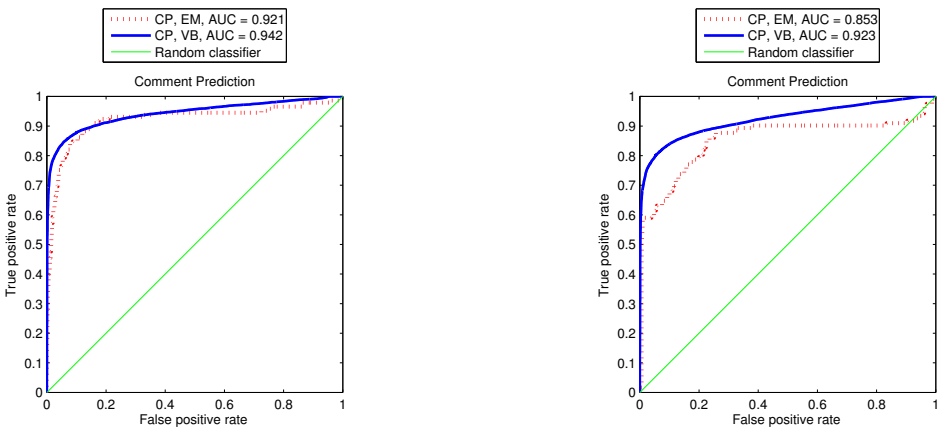
(a) Comment Prediction, 40% missing (b) Comment Prediction, 90% missing

Figure 7.22. Comparison of CTF-EM and CTF-VB methods under missing data case for comment prediction.



(a) Comment Prediction with R4, 40% missing (b) Comment Prediction with R4, 90% missing

Figure 7.23. Comparison of CTF-EM and CTF-VB methods under missing data case for comment prediction with R4 relation.



(a) Comment Prediction with R4, 40% missing (b) Comment Prediction with R4, 90% missing

Figure 7.24. Comparison of CTF-EM and CTF-VB methods under missing data case for digg prediction.

8. CONCLUSIONS AND FUTURE WORK

In this study, we presented and compared several tensor based factorization models for link prediction problem. We deal with both the missing link prediction problem: the problem of predicting the existence of missing connections between entities of interest and temporal link prediction problem: where we have given a sequence of link data at various time steps as input, and our goal is to predict the links at the next time step. We use approaches which enable us to investigate alternative tensor models and cost functions for both single and coupled analysis of heterogeneous data.

First, we used an approach for link prediction problem based on probabilistic interpretation of tensor factorization models that enables one to incorporate domain specific information to any arbitrary factorization model and provides the update rules for multiplicative gradient descent and expectation-maximization algorithms using different loss functions.

Then, we have studied link prediction problem using coupled analysis of relational data sets represented as data sets in the form of matrices and higher-order tensors. The problem is formulated as simultaneous factorization of higher-order tensors/matrices extracting common latent factors from the shared modes. While most existing studies on coupled analysis have been developed to fit a specific type of a tensor model using a particular loss function, we have used GCTF framework, which enables us to develop coupled models for joint analysis of multiple data sets in a compact way using various tensor models and cost functions. In our coupled analysis for the link prediction problem, we have studied KL-divergence, IS-divergence and Euclidean distance-based cost functions as well as different tensor models. Our numerical results on both synthetic and real datasets demonstrate that selection of the tensor model and the loss function is important in terms of link prediction performance.

Furthermore, we have investigated variational inference for PLTF framework with KL cost from a full Bayesian perspective that also handles the missing data naturally.

In addition, we develop a practical way without incurring much additional computational cost to PLTF-EM approach for computing the approximation distribution and full conditionals; then, we estimate the model order in terms of marginal likelihood. By maximizing the bound on marginal likelihood, we have a method where all the hyperparameters can be estimated from data. Our experiments suggest that the variational bound seems to be reasonable approximation to the marginal likelihood and can guide model selection for PLTF. Afterwards, we extend our variational method in order to be able to make inference on coupled tensor factorization models where multiple observed tensors $(X_1, ..., X_K)$ can share a set of factors [35]. Factorization of multiple observed tensors simultaneously, alleviates the overfitting better than the standard variational Bayesian matrix factorization [66].

Through extensive experiments on both synthetic and real datasets, we assess the performance of various alternative tensor models and loss functions for the link prediction problem. Numerical experiments clearly demonstrate that not only joint analysis of data from multiple sources via coupled factorization improves the link prediction performance but also the selection of right loss function and tensor model is crucial for accurately predicting missing links.

As a next step of this work, we aim to extend our study in some directions listed below.

- Tensor factorization via MCMC inference: Markov Chain Monte Carlo is also a viable alternative for inference [67].
- Generalization of the model selection for arbitrary cost functions: The main limitation of our model selection and variational Bayesian approach is that we use only the KL cost. Although we sketch the outline for the general cost functions, we did not extract the expectation of the posterior distribution for each cost function.
- Symmetric Factorizations: In this thesis, we can estimate a set of different latent factor matrices in a particular factorization model such as $X \approx Z_1 Z_2$. However, there is certainly a need for carrying out a symmetric factorization model such

as $X \approx Z_1 Z_1^T$.

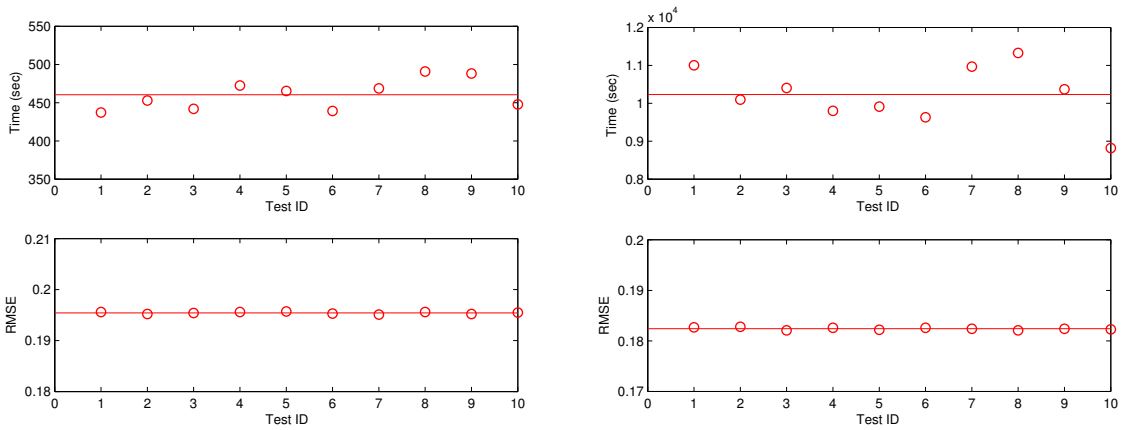
- Combination of different models: We use one latent structure model to represent a multidimensional data, i.e. CP and TUCKER, in this thesis. However, it would be interesting and useful to model the multidimensional data with combination of more than one (both using CP and TUCKER) latent structure such as $\hat{X} = C_1 + C_2$ where $C_1^{i,j,k} \approx \sum_r Z_1^{i,r} Z_2^{j,r} Z_3^{k,r}$ and $C_2^{i,j,k} \approx \sum_{p,q,r} Z_1^{i,p} Z_2^{j,q} Z_3^{k,r} Z_4^{p,q,r}$.

APPENDIX A: APPENDIX

A.1. Sparse Implementation

If a large number of entries is missing, then mask tensor M is sparse. In this case, there is no need to allocate storage for every entry of the tensor X. Instead, we can store and work with just the known values, making the method efficient in both storage and time. All of our algorithms can scale to sparse, large-scale data using specialized sparse data structures, significantly reducing the storage and computation costs. The time complexity of each iteration is roughly $O(N)$, which is linear in terms of the total number of non-missing entries N.

We consider two situations for the experiments: (i) $500 \times 500 \times 500$ with 99% missing data (1.25 million known values), and (ii) $1000 \times 1000 \times 1000$ with 98% missing data (20 million known values). Figure A.1 shows that, in the $500 \times 500 \times 500$ case, all ten problems were solved with an Root Mean Square Error (RMSE) score greater than 0.20, with solve times ranging between 400 and 500 seconds and in the $1000 \times 1000 \times 1000$ case, all ten problems were solved with an RMSE score greater than 0.20. The solve times ranged from 8000 to 12000 seconds, approximately 20 times slower than the $500 \times 500 \times 500$ case, which had 16 times more non-missing entries.



(a) 40% missing (b) 80% missing

Figure A.1. Results of PLTF-EM algorithm for large-scale problems with 95% missing data. The means are shown as solid lines.

REFERENCES

1. Getoor, L. and C. P. Diehl, "Link Mining: A Survey", *SigKDD Explorations Special Issue on Link Mining*, 2005.

2. Menon, A. K. and C. Elkan, "Link Prediction via Matrix Factorization", *Machine Learning and Knowledge Discovery in Databases*, Vol. 6912, pp. 437–452, 2011.

3. Hasan, M. A. and M. J. Zaki, "A Survey of Link Prediction in Social Networks", Aggarwal, C. C. (editor), *Social Network Data Analytics*, pp. 243–275, Springer US, 2011.

4. Koren, Y., R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems", *Computer*, Vol. 42, No. 8, pp. 30–37, 2009.

5. Kashima, H., T. Kato, Y. Yamanishi, M. Sugiyama, and K. Tsuda, "Link Propagation: A Fast Semi-supervised Learning Algorithm for Link Prediction", 2009.

6. Dunlavy, D. M., T. G. Kolda, and E. Acar, "Temporal Link Prediction using Matrix and Tensor Factorizations", *ACM Transactions on Knowledge Discovery from Data (ACM TKDD)*, Vol. 5, No. 2, pp. 10–70, 2011.

7. Hasan, M. A., V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning", *SDM 06 workshop on Link Analysis, Counterterrorism and Security*, 2006.

8. Liben-Nowell, D. and J. Kleinberg, "The link-prediction problem for social networks", *Social Information in Science and Technology*, Vol. 58, No. 7, pp. 1019–1031, May 2007.

9. Clauset, A., C. Moore, and M. E. J. Newman, "Hierarchical structure and the prediction of missing links in networks", *Nature*, Vol. 453, pp. 98–101, 2008.

10. Getoor, L., "Link Mining: A New Data Mining Challenge", *SIGKDD Explorations*, pp. 1–6, 2003.

11. O'Madadhain, J., J. Hutchins, and P. Smyth, "Prediction and ranking algorithms for event-based network data", *SIGKDD Exploration Newsletter*, Vol. 7, No. 2, pp. 23–30, December 2005.

12. Singh, A. P. and G. J. Gordon, "Relational learning via collective matrix factorization", *Knowledge Discovery and Data Mining (KDD'08)*, 2008.

13. Long, B., Z. M. Zhang, X. Wu, and P. S. Yu, "Spectral Clustering for Multi-type Relational Data", *International Conference on Machine Learning (ICML'06)*, pp. 585–592, 2006.

14. Yoo, J., M. Kim, K. Kang, and S. Choi, "Nonnegative matrix partial co-factorization for drum source separation", *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2010, 14-19 March 2010, Sheraton Dallas Hotel, Dallas, Texas, USA*, pp. 1942–1945, IEEE, 2010.

15. Alter, O., P. O. Brown, and D. Botstein, "Generalized singular value decomposition for comparative analysis of genome-scale expression data sets of two different organisms", *Proceedings of the National Academy of Sciences (PNAS)*, Vol. 100, pp. 3351–3356, 2003.

16. Smilde, A., J. A. Westerhuis, and R. Boque, "Multiway multiblock component and covariates regression models", *Journal of Chemometrics*, Vol. 14, pp. 301–331, 2000.

17. Banerjee, A., S. Basu, and S. Merugu, "Multi-way Clustering on Relation Graphs", *Proceedings of the SIAM International Conference on Data Mining (SDM-2007)*, 2007.

18. Lin, Y.-R., J. Sun, P. Castro, R. Konuru, H. Sundaram, and A. Kelliher, "MetaFac: community discovery via relational hypergraph factorization", *Knowledge Discovery and Data Mining (KDD '09)*, pp. 527–536, 2009.

19. Acar, E., T. G. Kolda, and D. M. Dunlavy, "All-at-once Optimization For Coupled Matrix and Tensor Factorizations", *Knowledge Discovery and Data Mining (KDD'11) Workshop Proceedings*, 2011.

20. Taskar, B., M. fai Wong, P. Abbeel, and D. Koller, "Link Prediction in Relational Data", *in Neural Information Processing Systems*, 2003.

21. Popescul, A. and L. H. Ungar, "Statistical Relational Learning for Link Prediction", *International Joint Conferences on Artificial Intelligence (IJCAI'03)*, 2003.

22. Getoor, L., N. Friedman, D. Koller, and B. Taskar, "Learning Probabilistic Models of Link Structure", *Journal of Machine Learning Research*, Vol. 3, pp. 679–707, 2002.

23. Cichocki, A., R. Zdunek, A. Phan, and S. Amari, *Nonnegative Matrix and Tensor Factorization*, Wiley, New York, 2009.

24. Candès, E. J. and Y. Plan, "Matrix Completion With Noise", *Proceedings of the IEEE*, Vol. 98, pp. 925–936, 2010.

25. Acar, E., D. Dunlavy, T. Kolda, and M. Morup, "Scalable Tensor Factorizations for Incomplete Data", *Chemometrics and Intelligent Laboratory Systems*, Vol. 106, pp. 41–56, 2011.

26. Gandy, S., B. Recht, and I. Yamada, "Tensor completion and low-n-rank tensor recovery via convex optimization", *Inverse Problems*, Vol. 27, p. 025010, 2011.

27. Acar, E., D. M. Dunlavy, and T. G. Kolda, "Link Prediction on Evolving Data Using Matrix and Tensor Factorizations", *Proceedings of the 2009 IEEE Interna-*

tional Conference on Data Mining Workshops, ICDMW '09, pp. 262–269, IEEE Computer Society, Washington, DC, USA, 2009.

28. Chi, Y. and S. Zhu, "FacetCube: a framework of incorporating prior knowledge into non-negative tensor factorization", *Conference on Information and Knowledge Management (CIKM)*, pp. 569–578, 2010.

29. Harshman, R., "Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis", *UCLA Working Papers in Phonetics*, Vol. 16, 1970.

30. Carroll, J. D. and J. J. Chang, "Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition", *Psychometrika*, Vol. 35, pp. 283–319, 1970.

31. Hitchcock, F. L., "Multiple invariants and generalized rank of a p-way matrix or tensor", *Journal of Mathematics and Physics*, 1927.

32. Tucker, L. R., "Implications of factor analysis of three-way matrices for measurement of change", Harris, C. W. (editor), *Problems in measuring change.*, pp. 122–137, University of Wisconsin Press, Madison WI, 1963.

33. Yilmaz, Y. K. and A. T. Cemgil, "Probabilistic Latent Tensor Factorization", *International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, pp. 346–353, 2010.

34. Yilmaz, Y. K., "Generalized Tensor Factorization", Ph.D. Thesis, Bogazici University, 2012.

35. Yilmaz, Y. K., A. T. Cemgil, and U. Simsekli, "Generalised Coupled Tensor Factorisation", *The Neural Information Processing Systems (NIPS)*, 2011.

36. Cichocki, A., R. Zdunek, A. H. Phan, and S. ichi Amari, *Nonnegative Matrix and*

Tensor Factorizations - Applications to Exploratory Multi-way Data Analysis and Blind Source Separation, Wiley, New York, 2009.

37. Paatero, P. and U. Tapper, "Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values", *Environmetrics*, Vol. 5, No. 2, pp. 111–126, 1994.

38. Lee, D. D. and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization.", *Nature*, Vol. 401, pp. 788–791, 1999.

39. Manning, C. D., P. Raghavan, and H. Schtze, *Introduction to Information Retrieval*, Cambridge University Press, New York, NY, USA, 2008.

40. Kolda, T. G. and B. W. Bader, "Tensor Decompositions and Applications", *SIAM Review*, Vol. 51, No. 3, pp. 455–500, September 2009.

41. Acar, E. and B. Yener, "Unsupervised multiway data analysis: A literature survey", *IEEE Transactions on Knowledge and Data Engineering*, 2008.

42. Cattell, R. B., "The three basic factor-analytic research designs - their interrelations and derivatives", *Psychological Bulletin*, 1952.

43. Carroll, J. and J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition", *Psychometrika*, pp. 283–319, 1970.

44. Kiers, H. A. L., "Towards a standardized notation and terminology in multiway analysis", *Journal of Chemometrics*, Vol. 14, p. 105–122, 2000.

45. Banerjee, A., S. Merugu, I. S. Dhillon, and J. Ghosh, "Clustering with Bregman Divergences", *Journal of Machine Learning Research*, Vol. 6, pp. 1705–1749, 2005.

46. Bishop, C. M., *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer, New York, 2007.

47. Ghahramani, Z. and M. J. Beal, "Propagation Algorithms for Variational Bayesian Learning", *The Neural Information Processing Systems (NIPS)*, pp. 507–513, 2000.

48. Milton, J. S. and J. C. Arnold, *Introduction to Probability and Statistics: Principles and Applications for Engineering and the Computing Sciences*, McGraw-Hill Higher Education, 4th edition, 2002.

49. Salakhutdinov, R. and A. Mnih, "Probabilistic Matrix Factorization", *The Neural Information Processing Systems (NIPS)*, 2007.

50. Cemgil, A. T., "Bayesian Inference in Non-negative Matrix Factorisation Models", *Computational Intelligence and Neuroscience*, 2009.

51. Beal, M., *Variational Algorithms for Approximate Bayesian Inference*, Ph.D. thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.

52. Fevotte, C. and A. T. Cemgil, "Nonnegative matrix factorisations as probabilistic inference in composite models", *Proc. 17th European Signal Processing Conference (EUSIPCO'09)*, 2009.

53. Simsekli, U. and A. T. Cemgil, "Markov Chain Monte Carlo Inference for Probabilistic Latent Tensor Factorization", *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2012.

54. Spiegel, S., J. Clausen, S. Albayrak, and J. Kunegis, "Link prediction on evolving data using tensor factorization", *Proceedings of the 15th international conference on New Frontiers in Applied Data Mining*, PAKDD'11, pp. 100–110, Springer-Verlag, Berlin, Heidelberg, 2012.

55. Uhlig, S., B. Quoitin, J. Lepropre, and S. Balon, "Providing public intradomain traffic matrices to the research community", *SIGCOMM Computer Communication Review*, Vol. 36, No. 1, pp. 83–86, January 2006.

56. Zheng, V. W., B. Cao, Y. Zheng, X. Xie, and Q. Yang, "Collaborative Filtering Meets Mobile Recommendation: A User-centered Approach", *AAAI'10: Proceedings of the Twenty-Fourth Conference on Artificial Intelligence*, 2010.

57. Choudhury, M. D., H. Sundaram, A. John, and D. D. Seligmann, "Social Synchrony: Predicting Mimicry of User Actions in Online Social Media", *IEEE International Conference on Social Computing (CSE)*, pp. 151–158, 2009.

58. Watanabe, S., "Algebraic Analysis for Non-identifiable Learning Machines", *Neural Computation*, Vol. 13, pp. 899–933, 2000.

59. Yamazaki, K. and S. Watanabe, "Singularities In Mixture Models And Upper Bounds Of Stochastic Complexity", *International Journal of Neural Networks*, Vol. 16, pp. 1029–1038, 2003.

60. Fawcett, T., "An introduction to ROC analysis", *Pattern Recognition Letter*, Vol. 27, No. 8, pp. 861–874, June 2006.

61. Stäger, M., P. Lukowicz, and G. Tröster, "Dealing with Class Skew in Context Recognition", *International Conference on Distributed Computing Systems (ICDCS) Workshops*, p. 58, 2006.

62. Sanderson, M., "Test Collection Based Evaluation of Information Retrieval Systems.", *Foundations and Trends in Information Retrieval*, Vol. 4, No. 4, pp. 247–375, 2010.

63. Narita, A., K. Hayashi, R. Tomioka, and H. Kashima, "Tensor Factorization Using Auxiliary Information", *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, pp. 501–516, 2011.

64. Katz, L., "A new status index derived from sociometric analysis", *Psychometrika*, Vol. 18, No. 1, pp. 39–43, 1953.

65. Nakajima, S. and M. Sugiyama, "Implicit Regularization in Variational Bayesian Matrix Factorization", *International Conference on Machine Learning (ICML)*, pp. 815–822, 2010.

66. Yoo, J. and S. Choi, "Bayesian Matrix Co-Factorization: Variational Algorithm and Cramér-Rao Bound", *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, pp. 537–552, 2011.

67. Schmidt, M. N. and S. Mohamed, "Probabilistic non-negative tensor factorization using Markov chain Monte Carlo", Aug 2009.