

TENSOR DECOMPOSITION MODELS FOR KNOWLEDGE GRAPHS

by

Semih Akbayrak

B.S., Electrical and Electronics Engineering, Boğaziçi University, 2015

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 Computational Science and Engineering
Boğaziçi University

2018

TENSOR DECOMPOSITION MODELS FOR KNOWLEDGE GRAPHS

APPROVED BY:

Assoc. Prof. Ali Taylan Cemgil
(Thesis Supervisor)

Prof. Can Özturan

Assist. Prof. Sinan Yıldırım

DATE OF APPROVAL: 01.06.2018

ACKNOWLEDGEMENTS

Firstly, I would like to thank my supervisor Assoc. Prof. Ali Taylan Cemgil for his continuous support and mentorship. It was an honor for me to work with him. I also want to thank Prof. Can Özturan and Assist. Prof. Sinan Yıldırım for attending my thesis jury.

The thesis is mostly based on the model 'Sum Conditioned Poisson Factorization'. Gökhan Çapan and Taha Yusuf Ceritli had already studied well the original matrix factorization model when I joined them and my contribution to the original model is minor when compared to theirs. So I want to thank them for their friendship and partnership.

I want to express my gratitude to Melike Alayoğlu who has always supported me during the master years. I thank my dear friend and colleague Onur Poyraz. Our friendship started in high school and since then we grew up like brothers. During my master years, I have met great people like Burak Kurutmaz, Çağrı Sofuoğlu, Ali Caner Türkmen, İlker Gündoğdu, Mine Öğretir, Barış Kurt, Çağatay Yıldız. I also thank them.

My greatest thanks goes to my family. In every condition, they've always loved, encouraged and supported me.

I thank the Scientific and Technological Research Council of Turkey (TÜBİTAK). This thesis has been supported by the M.Sc. scholarship (2210-A) from the Scientific and Technological Research Council of Turkey (TÜBİTAK).

ABSTRACT

TENSOR DECOMPOSITION MODELS FOR KNOWLEDGE GRAPHS

Extracting new unknown facts from given facts in the format of triples(entity-relation-entity) is a popular statistical relational learning task and defined with the name of knowledge graph link prediction problem. Due to nature of the problem definition, tensors are widely preferred to represent existing datasets. In the presence of latent features for entities and relations, tensor factorization models are used to approximate to the original dataset tensor. These latent features of entities and relations are estimated/inferred during approximation and interaction between them reveals the probabilities of triple existences.

In this thesis, we propose the tensor extension of recently introduced Sum Conditioned Poisson Factorization, in order to use it in knowledge graph problems. Sum Conditioned Poisson Factorization is an alternative to Generalized Linear Models and can be used to model bounded data with L component Poisson Factorizations which are conditioned on their summation. Unlike GLMs which factorize canonical parameters, SCPF decomposes directly the moment parameters. For knowledge graph problems, we define two Poisson tensor factorizations by conditioning their summation to a tensor of ones. We introduce maximum likelihood parameter estimation with Expectation Maximization and Bayesian inference with variational inference and Gibbs sampling. We compare the predictive performance of SCPF models with the performance of state of the art Generalized Linear Model, Logistic Tensor Factorization on standard datasets (Nation, UMLS, and Kinship).

ÖZET

BİLGİ GRAFİKLERİ İÇİN TENSÖR AYRIŞTIRMA MODELLERİ

Üçlüler(varlık-ilişki-varlık) biçiminde ifade edilen gerçekleri kullanarak yeni ve bilinmeyen gerçekler çıkarsamak popüler bir istatistiksel ilişkişel öğrenme görevidir ve bilgi grafiği bağlantı tahmini problemi ismi ile tanımlanır. Problem tanımının doğası gereği mevcut veri setlerini temsil etmek için tensörler yaygın olarak tercih edilmektedir. Varlıklar ve ilişkiler için saklı özelliklerin varlığında, orijinal veri kümesi tensörüne yaklaşmak için tensör ayrıştırma modelleri kullanılır. Varlıkların ve ilişkilerin bu saklı özellikleri, yaklaşım sırasında kestirilir/çıkarsanır ve aralarındaki etkileşim, üçlülerin varoluş olasılıklarını ortaya çıkarır.

Bu tez çalışmasında, bilgi grafik problemlerinde kullanılmak üzere, yakın zaman önce tanıtılan Toplam Koşullu Poisson Ayrıştırması'nın tensör uzantısını önermekteyiz. Genelleştirilmiş Doğrusal Modeller'e alternatif olarak Toplam Koşullu Poisson Ayrıştırması değer aralığı sınırlı olan veriyi, toplamaları üzerinden koşullandırılmış L bileşen Poisson Ayrıştırması ile modellemek için kullanılabilir. Standart parametreleri ayırştıran Genelleştirilmiş Doğrusal Modeller'den farklı olarak, Toplam Koşullu Poisson Ayrıştırması doğrudan moment parametrelerini ayırştırır. Bilgi grafiği problemi için toplamaları birler tensörüne koşullanmış iki Poisson tensör ayrıştırması tanımlamaktayız. Beklenti Enbüyütme ile en büyük olabilirlik kestirimi, varyasyonel çıkarsama ve Gibbs örnekleme ile ise Bayesci çıkarsama sunuyoruz. Toplam Koşullu Poisson Ayrıştırması modellerinin öngörü performanslarını, standart veri kümeleri (Nation, UMLS, ve Kinship) üzerinde, en ileri Genelleştirilmiş Doğrusal Model olan Lojistik Tensör Ayrıştırması'nın performansı ile karşılaştırmaktayız.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	iii
ABSTRACT	iv
ÖZET	v
LIST OF FIGURES	viii
LIST OF TABLES	xi
LIST OF SYMBOLS	xii
LIST OF ACRONYMS/ABBREVIATIONS	xiv
1. INTRODUCTION	1
1.1. Tensor Representations for KG	3
1.2. Evaluation	3
1.3. Tensor Decomposition Models for KG	3
1.4. Sum Conditioned Poisson Factorization	5
1.5. Thesis Organization	7
2. THEORETICAL BACKGROUND	8
2.1. Rescal and Logistic Tensor Factorization	8
2.2. Poisson NMF	10
2.3. EM Algorithm	12
2.4. Variational Bayes	15
2.5. Gibbs Sampling	17
3. PROPOSED MODEL	19
3.1. Parafac Decomposition	19
3.2. RESCAL-like(Tucker-like) Decomposition	22
4. INFERENCE	25
4.1. EM Algorithm for Parafac Decomposition	25
4.1.1. E-step	25
4.1.2. M-Step	28
4.2. Variational Bayes for Parafac Decomposition	29
4.3. Gibbs Sampling for Parafac Decomposition	31
4.4. EM Algorithm for Rescal-like Decomposition	33

4.4.1. E-step	34
4.4.2. M-step	36
5. EXPERIMENTS	37
5.1. Nation Dataset	37
5.2. UMLS Dataset	45
5.3. Kinship Dataset	47
6. CONCLUSION	50
REFERENCES	51
APPENDIX A: SUM CONDITIONED POISSON VARIABLES	57
APPENDIX B: EM FOR PARAFAC DECOMPOSITION	59
APPENDIX C: VARIATIONAL BAYES FOR PARAFAC DECOMPOSITION	61
APPENDIX D: GIBBS SAMPLING FOR PARAFAC DECOMPOSITION	63

LIST OF FIGURES

Figure 1.1.	Sample triples from Nation dataset.	2
Figure 2.1.	Logistic Tensor Factorization graph representation. Original figure can be found in [1].	8
Figure 2.2.	Nonnegative Matrix X is approximated by multiplication of non- negative matrices W and H	11
Figure 2.3.	Poisson NMF introduces additional latent variables S	12
Figure 2.4.	Gibbs sampling for Parafac decomposition	18
Figure 3.1.	Parafac decomposition of Sum Conditioned Poisson tensor factor- ization.	19
Figure 3.2.	Elements of latent matrices interact with each other and generate $x_{l,ijk}$	20
Figure 3.3.	Illustraion of Tucker-like decomposition for Sum Conditioned Pois- son Factorization.	22
Figure 3.4.	An $R \times R \times K$ weight tensor for relations and an $I \times R$ latent features matrix for entities.	23
Figure 4.1.	Gibbs sampling for Parafac decomposition	31
Figure 5.1.	Visualization of Nation dataset.	37

Figure 5.2.	Area under the ROC curve comparison for training set.	39
Figure 5.3.	Area under the ROC curve comparison for test set.	40
Figure 5.4.	Entity latent features that are estimated with Parafac SCPF(EM)	41
Figure 5.5.	Relation latent feature examples that are estimated with Parafac SCPF(EM)	41
Figure 5.6.	Expectations of entity latent features that are inferred with Parafac SCPF(VI)	42
Figure 5.7.	Expectations of relation latent feature examples that are inferred with Parafac SCPF(VI)	43
Figure 5.8.	Entity latent features that are estimated with Rescal-like SCPF .	43
Figure 5.9.	Relation latent feature examples that are estimated with Rescal- like SCPF	44
Figure 5.10.	Entity latent features that are estimated with LTF	44
Figure 5.11.	Relation latent feature examples that are estimated with LTF . .	45
Figure 5.12.	Visualization of clipped UMLS dataset.	45
Figure 5.13.	Area under the ROC curve comparison for training set.	46
Figure 5.14.	Area under the ROC curve comparison for test set.	47
Figure 5.15.	Visualization of clipped Kinship dataset.	48

Figure 5.16. Area under the ROC curve comparison for training set.	48
--	----

Figure 5.17. Area under the ROC curve comparison for test set.	49
--	----

LIST OF TABLES

Table 4.1.	EM update equations for Parafac decomposition.	29
Table 4.2.	EM update equations for Rescal-like decomposition.	36
Table 5.1.	Entities and relations in Nation dataset.	38

LIST OF SYMBOLS

a^w	Hyperparameter for w
$A(\cdot)$	Log partition function
$D(\cdot \cdot)$	A divergence metric
\mathbf{I}	Identity matrix
I, J, K, R	Dimensionalities for vectors, matrices and tensors
$KL(\cdot \cdot)$	Kullback-Leibler divergence
\underline{M}	Mask tensor
\underline{N}	Tensor of ones
\underline{S}_l	Latent source tensor that generate \underline{X}_l
\underline{W}_l	Latent tensor
W_l, H_l, G_l	Latent matrices that generate \underline{X}_l
\mathbf{x}	Set of observed variables
x	Scalar
\underline{x}	Vector
X	Matrix
\underline{X}	Tensor
$\hat{\underline{X}}$	Approximating tensor
$X_{::k}$	k^{th} frontal slice of tensor \underline{X}
\underline{x}_i	Vector that keeps the elements of i^{th} column of matrix X
$\underline{x}_{i\cdot}$	Vector that keeps the elements of i^{th} row of matrix X
x_{ij}	Entity of matrix X located in i^{th} row and j^{th} column
x_{ijk}	Entity of tensor \underline{X} located in i^{th} row, j^{th} column and k^{th} tube
\underline{X}_l	l^{th} component tensor that generates \underline{N}
$x_{l,ij}$	Entity of l^{th} component matrix X_l located in i^{th} row and j^{th} column
$x_{l,ijk}$	Entity of l^{th} component tensor \underline{X}_l located in i^{th} row, j^{th} column and k^{th} tube
$\underline{x}^T \underline{y}$	Inner product of vectors
XY	Matrix multiplication of X and Y

$\underline{X} \odot \underline{Y}$	Element wise tensor multiplication
\mathbf{z}	Set of hidden variables
Z	Normalizer
$\Gamma(\cdot)$	Gamma function
θ	Model parameters
λ_A	Hyperparameter belongs to A
μ	Mean parameter
ξ	Canonical parameter
$\sigma(\cdot)$	Sigmoid function
ϕ	Variational parameters
ψ	Sufficient statistic
$\Psi(\cdot)$	Digamma function
$\mathcal{BE}(\cdot; \cdot)$	Bernoulli distribution
$\mathbb{E}_p[\cdot], \mathbb{E}[\cdot]_p, \langle \cdot \rangle_p$	Expectation with respect to function p
$\mathcal{G}(\cdot; \cdot, \cdot)$	Gamma distribution
$\mathcal{L}(q, \theta)$	Lower bound for a log-likelihood with distribution q and model parameters θ
$\mathcal{M}(\cdot; \cdot, \cdot)$	Multinomial distribution
$\mathcal{N}(\cdot; \cdot, \cdot)$	Normal distribution
$\mathcal{PO}(\cdot; \cdot)$	Poisson distribution

LIST OF ACRONYMS/ABBREVIATIONS

AUC	Area Under Curve
ELBO	Evidence Lower Bound
EM	Expectation Maximization
GLM	Generalized Linear Model
KG	Knowledge Graph
KL	Kullback-Leibler
L-BFGS	Limited Memory Broyden–Fletcher–Goldfarb–Shanno
LTF	Logistic Tensor Factorization
MAP	Maximum A Posteriori
MCMC	Markov Chain Monte Carlo
NMF	Nonnegative Matrix Factorization
ROC	Receiver Operating Characteristic
SCPF	Sum Conditioned Poisson Factorization
VI	Variational Inference

1. INTRODUCTION

Ability of gaining knowledge through reasoning is a crucial indicator of intelligence. A fully intelligent agent could obtain some structured knowledge from unstructured data [2], but the amount of information that is extracted by this way would be limited. Hence, intelligent agents are expected to discover and understand the relations between the components in the known information so that the new knowledge can be inferred through reasoning.

One way to represent the knowledge and the facts in a structured form is keeping them in triple forms. A triple is formed by two entities and one relation which connects entities to each other [3]. This connection can be directed or undirected but in either case, the problem is the same that is exploring new unknown triples. The triples constitute a knowledge graph together, and the problem of discovering new triples is called knowledge graph link prediction or knowledge graph completion problem which is an example of statistical relational learning problems [4].

In order to illustrate the problem and domain better, some known triples for Nation dataset is given in Figure 1.1. The dataset is a directed graph and the nations are subjects and objects of the facts which are connected to each other via relations. Although the number of triples in the figure is quite small, we are still able to extract some knowledge about these nations by taking our prior knowledge on concepts into account. As an example, by considering the economic aid of USA to the other countries, we may conclude that USA is a prosperous country. Beside of this, it is clearly seen that China and India have bad relationship. Most of the time, models that we are developing have no such prior knowledge about concepts so they have to infer the attributes of items and relations by themselves. For example, if a knowledge graph contains a relation like 'marriedTo', the model has to realize that it is a symmetric relation by looking at the facts with 'marriedTo' relation. So for a given triple ('David', 'marriedTo', 'Victoria'), the model is expected to propose a new triple ('Victoria', 'marriedTo', 'David') as a new fact. All the entity-relation-entity combinations are

potential facts, but only small amount of them are really facts and our objective is to discover them.

Entity1	Relation	Entity2
USA	economicaid	Egypt
USA	economicaid	Israel
China	warning	India
China	militaryactions	India
India	militaryactions	China
India	attackembassy	Indonesia
China	protests	India
USSR	protests	USA
UK	protests	USA
UK	tourism	USA
USA	tourism	UK

Figure 1.1. Sample triples from Nation dataset.

Knowledge graphs are being utilized in various areas including web search engines, social networks, e-commerce, biomedical, and chatbots. Google extracts triples by scanning web pages [5], and use these triples to improve its search engine. DBpedia serves a knowledge base to the Internet users by extracting triples from Wikipedia which contains rather organized knowledge thanks to its structural form [6]. Moreover, knowledge graphs were used to link social media profiles with concepts to understand users' interests better [7]. IBM improves Watson's question answering and chatbot system with a knowledge graph [8]. Walmart creates knowledge graphs for better advertisement and customer behaviour modeling [9]. Knowledge graphs are also useful in bioinformatics: several relational datasets exist such as protein-protein and genes-diseases interactions [10–13] which are widely used in disease treatment research. As it is evident from a wide spectrum of applications, the Knowledge Graph is a popular and generic representation for relational knowledge in a structural form, hence it is worth to study as a statistical object.

1.1. Tensor Representations for KG

For I entities and K relations, we can use an $I \times I \times K$ tensor (multidimensional array) \underline{X} to represent the data by signing the corresponding indices for the existing triples. For example, if we know that entity i is connected to entity j via relation k , we sign the x_{ijk} with 1, otherwise with 0. By keeping signing the corresponding indices, we can fulfill the tensor with 1s and 0s, and this is a structural way to represent the dataset.

Mainly, two different assumptions can be made for such tensorized datasets, open world assumption and closed world assumption [3]. In open world assumption, we assume that the given facts are known and represented by 1, but it is misleading to accept not given triples as false. Therefore, in open world assumption, we have only 1s in our hand and learning should be made over 1s, with no distinction between false and unknown relations. On the other hand, in the closed world assumption, one assumes that the absence of triples is an indicator of a false relationship, so it accepts not given triples as false, and we use both 1s and 0s in learning stage. Although open world assumption is typically a better way to represent a domain, it is conceptually harder to deal with datasets that only contain 1s, so most of the models in literature accept closed world assumption and in this thesis we also follow this trend.

1.2. Evaluation

All the entity-relation-entity combinations are potentially true facts. In the evaluation phase, a trained model assigns scores to the combinations. The scores reflect the model's belief in triples to be fact. Hence a threshold needs to be decided so that the triples which are scored above the threshold are labeled as facts.

1.3. Tensor Decomposition Models for KG

Due to nature of the data, approximate tensor decomposition models [14, 15] are widely used for knowledge graph completion task. In the presence of a divergence

metric $D(\cdot||\cdot)$, a 3-way array \underline{X} is approximated by low dimensional matrices W , H , and G . An additional core tensor can be used alongside W , H , and G , as well.

$$(W, H, G)^* = \arg \min_{W, H, G} D(\underline{X}||W, H, G)$$

In machine learning, approximating matrices W , H , and G are usually interpreted as latent variables of a graphical model [16] which defines some kind of dependency structure between the latent variables and observations. So, the problem of finding the optimal approximating matrices can be described as an inference problem in the graphical model. After inferring the latent variables by considering the observations and the graphical structure, one can estimate the unobserved tensor elements by running the forward model.

Parafac and Tucker decompositions are the most popular tensor decomposition methods and refer to two different graphical model structures. In Parafac decomposition, tensor \underline{X} is approximated by a summation of R rank 1 tensors where R is the rank of approximating tensor $\hat{\underline{X}}$ or model order. So, the approximation of an element of tensor \underline{X} can be expressed as the following

$$x_{ijk} \approx \hat{x}_{ijk} = \sum_r^R w_{ir} h_{jr} g_{kr}$$

In Tucker decomposition, multiplications and summations are applied not over the common index r but over the p, q, r which represent the indices of the vectors $\underline{w}_{i:}$, $\underline{h}_{j:}$, $\underline{g}_{k:}$ respectively. Scaling parameters λ_{pqr} can be introduced, as well.

$$x_{ijk} \approx \hat{x}_{ijk} = \sum_{p,q,r}^{P,Q,R} \lambda_{pqr} w_{ip} h_{jq} g_{kr}$$

Parafac and Tucker decompositions are used for knowledge graph link prediction problem in [17] and [18] respectively. [1, 19] use Tucker like decomposition and name it RESCAL. Beside of tensor factorization, matrix factorization models are used for knowledge graph completion [20]. Moreover, neural models have become so popular

in machine learning society recently, and one of its applications on knowledge graph completion can be found in [21]. Distance based models are also popular in knowledge graph completion problems [22–24].

This thesis focuses on tensor decomposition models, so the other methods mentioned above are out of scope. We especially studied Tensor extension of Sum Conditioned Poisson Factorization [25] which is built upon Poisson Nonnegative Matrix Factorization [26]. Additionally, we implemented Logistic Tensor Factorization which is state of the art method currently, for comparison.

1.4. Sum Conditioned Poisson Factorization

As statistical objects, tensor decomposition models introduce latent features for the items which interact with each other and generates the observations. Likelihood distribution of observations are generally chosen from exponential family distributions:

$$x_{ijk} \sim \exp(\psi(x_{ijk})\xi_{ijk} - A(\xi_{ijk})) \quad (1.1)$$

where ψ , ξ , A are sufficient statistics, canonical parameters and log-partition function respectively. One decomposition approach is factorization of the canonical parameters which refers to the tensor factorization extension of Generalized Linear Models (GLM). Generalized Linear Models [27] map the output of predictor to the expectation parameter of likelihood distribution via an inverse link function. As an example Logistic Tensor Factorization handles knowledge graph completion as a binary classification problem with sigmoid inverse link function and Bernoulli likelihood distribution. Equivalently, mean parameters of observation likelihoods can be mapped to the canonical parameters via link function, Logit in LTF case, and factorization is applied to the canonical parameters.

Alternatively, one may express the observation likelihood with moment parameterization and model the moment parameters:

$$x_{ijk} \sim \frac{1}{Z} \exp(-D(x_{ijk} || \mu_{ijk})) \quad (1.2)$$

where μ is the first central moment or the mean parameter of exponential family distribution and Z is a normalizer. Some models directly factorize the moment parameter μ and Poisson NMF [26] is one of them.

In moment parameterization, the mean parameter is constructed with the summation of latent sources which are generated by the interactions of latent features. In Poisson factorization case, the latent sources are non-negative and this makes the designed model highly interpretable. Moreover, predictive performance of a model may vary with the type of parametrization.

In order to investigate the effect of moment parameterization on knowledge graph problems, we present Sum Conditioned Poisson Factorization. Sum Conditioned Poisson Factorization introduces L component Poisson Factorizations by constraining their summation to a fixed value so that binary, ordinal, and multinomial data can be modelled with moment parametrizations.

Like Logistic Tensor Factorization, Sum Conditioned Poisson Factorization is also a latent variable model and finds latent features for entities and relations by utilizing given data points, then uses these features to estimate true and false triples in a test set.

Purpose of this thesis is completion of knowledge graph tensor, but latent feature approach enables us to propose solutions for the other problems related to knowledge graphs, i.e. clustering of entities and relations, and entity resolution which refers to detecting the same entities [3] such as US and USA.

1.5. Thesis Organization

Chapter 1 is the introduction part and provides the problem definition along with the motivations and previous works. Theoretical background that is required to understand the rest of the thesis is supplied in Chapter 2. SCPF models are proposed in Chapter 3. Chapter 4 focuses on the inference techniques for the proposed models. Chapter 5 exhibits the experiment results and comparisons. In Chapter 6, final comments and possible further research topics are provided.

2. THEORETICAL BACKGROUND

In this chapter, we review the required theoretical background for this thesis. Firstly, we describe RESCAL decomposition and its extension Logistic Tensor Factorization. Secondly, Poisson Nonnegative Matrix Factorization is introduced. Lastly, Expectation-Maximization Algorithm, Variational Bayes and Gibbs Sampling methods for inference are presented for this class of models.

2.1. Rescal and Logistic Tensor Factorization

RESCAL [19,28] is a tensor factorization model which is designed specifically for knowledge graphs. RESCAL factorization introduces global latent features for entities and relations. Independent of its role in the triples, the latent features of entity i are represented by a vector \underline{a}_i . The latent features of relation k are kept in matrix $B_{::k}$.

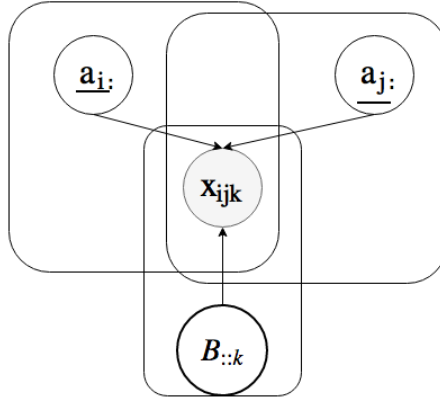


Figure 2.1. Logistic Tensor Factorization graph representation. Original figure can be found in [1].

Entity feature vectors \underline{a}_i and \underline{a}_j interact with each other via relation feature matrix $B_{::k}$ to generate the tensor element x_{ijk} . The type of this interaction is Tucker

like. Pairwise multiplications of entity vector elements are weighted by corresponding relation matrix values and the summation of weighted components generates a score ξ_{ijk} that is proportional to the triple's probability of being true. Based on the parameterization choice, ξ_{ijk} refers to a canonical parameter or a moment parameter.

$$\xi_{ijk} = \sum_{p,q} b_{pqk} a_{ip} a_{jq}$$

In the original RESCAL model, authors use Normal distribution to model x_{ijk} and they reach closed form update equations for alternating least squares algorithm. On the contrary, likelihood distribution of x_{ijk} in Logistic Tensor Factorization [1] is Bernoulli distribution which defines nature of the data better in return of slower learning stage.

$$x_{ijk} \sim \mathcal{BE}(x_{ijk}; \sigma(\xi_{ijk})) \quad (2.1)$$

where $\xi_{ijk} = \underline{a_{i:}}^T B_{::k} \underline{a_{j:}}$.

ξ_{ijk} is the canonical parameter of Bernoulli distribution and needs to be mapped to the mean parameter via an inverse link function. Sigmoid function is used as the inverse link function to map the value of ξ_{ijk} to a range of $(0, 1)$.

$$\sigma(\xi_{ijk}) = \frac{1}{1 + \exp(-\underline{a_{i:}}^T B_{::k} \underline{a_{j:}})}$$

LTF model accepts that the latent features of entities and relations are sampled from Normal distribution.

$$\begin{aligned} \underline{a_{i:}} &\sim \mathcal{N}(\underline{a_{i:}}; 0, \lambda_A \mathbf{I}) \\ B_{::k} &\sim \mathcal{N}(B_{::k}; 0, \lambda_B \mathbf{I}) \end{aligned}$$

Authors present Maximum A Posterior solution for this problem

$$\begin{aligned} \arg \max_{A, \underline{B}} p(A, \underline{B} | \underline{X}) &\propto p(\underline{X} | A, \underline{B}) p(A) p(\underline{B}) \\ \arg \max_{A, \underline{B}} \log p(A, \underline{B} | \underline{X}) &\propto \log p(\underline{X} | A, \underline{B}) + \log p(A) + \log p(\underline{B}) \end{aligned}$$

The problem can be converted to a loss minimization problem easily.

$$\text{Loss} = -\log p(A, \underline{B} | \underline{X})$$

$$\begin{aligned} \arg \min_{A, \underline{B}} \quad & -\sum_k [X_{::k} \odot \log \sigma(AB_{::k}A^T) + (1 - X_{::k}) \odot \log(1 - \sigma(AB_{::k}A^T))] \\ & + \lambda_A \|A\|_F^2 + \sum_k \lambda_B \|B_{::k}\|_F^2 \end{aligned}$$

Derivatives of loss function with respect to parameters A and $B_{::k}$ are derived below.

$$\begin{aligned} \frac{\partial \text{Loss}}{\partial A} &= \sum_k [(\sigma(AB_{::k}A^T) - X_{::k})AB_{::k}^T + (\sigma(AB_{::k}A^T) - X_{::k})^T AB_{::k}] + 2\lambda_A A \\ \frac{\partial \text{Loss}}{\partial B_{::k}} &= A^T(\sigma(AB_{::k}A^T) - X_{::k})A + 2\lambda_B B_{::k} \end{aligned}$$

Although the solution is not in closed form, the derivatives are derived and can be used in an unconstrained optimization method to solve this problem.

2.2. Poisson NMF

Originally proposed by Lee and Seung (1999), Non-negative Matrix Factorization(NMF) [29] is both a model and an algorithm to decompose a non-negative matrix X into non-negative matrices W and H . In their original paper, authors show that NMF finds more meaningful and interpretable bases when compared with Principal Component Analysis(PCA). Since then, NMF has become a popular matrix decomposition method and applied to various areas for various purposes including topic

modelling [30], clustering and manifold learning [31, 32], image analysis [33], source separation [34], and financial data analysis [35].

In [36], authors propose two different cost functions to be minimized and come up with multiplicative update rules for them.

- (i) minimize $\|X - WH\|^2$ w.r.t W and H for $W, H \geq 0$
- (ii) minimize $D(X||WH)$ w.r.t W and H for $W, H \geq 0$ where $D(A||B) = A \log \frac{A}{B} - A + B$

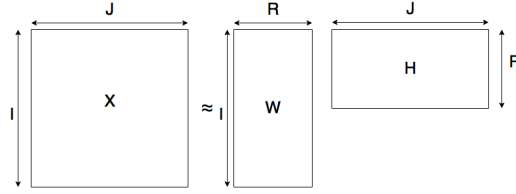


Figure 2.2. Nonnegative Matrix X is approximated by multiplication of nonnegative matrices W and H .

For given J data instances with I features, W is a template matrix which keeps the R different bases, and H is the excitation matrix which determines the weights of the bases for each of the data instances. Hence, a data instance \underline{x}_j can be approximated via $W\underline{h}_j$.

Poisson NMF [26] is a hierarchical generative model description of original NMF. Poisson NMF introduces additional latent sources $s_{1:I,1:J,1:R}$ to describe the problem better and try to minimize Kullback-Leibler divergence.

Below equations describe the generative model of Poisson NMF.

$$s_{ijr} \sim \mathcal{PO}(s_{ijr}; w_{ir}h_{jr})$$

$$x_{ij} = \sum_{r=1}^R s_{ijr}$$

	$\mathbf{h}_{1,r}$			$\mathbf{h}_{J,r}$
$\mathbf{w}_{1,r}$	$\mathbf{s}_{1,1,r}$			$\mathbf{s}_{1,J,r}$
$\mathbf{w}_{I,r}$	$\mathbf{s}_{I,1,r}$			$\mathbf{s}_{I,J,r}$

Figure 2.3. Poisson NMF introduces additional latent variables S .

In the article, author applies EM-algorithm for inference and comes up with update equations which are exactly same with the multiplicative update rules of original NMF model. This was the first time which showed that the algorithm of NMF is not only EM-like but exactly EM algorithm.

By defining prior distributions on w and h , a hierarchical model can be constructed. For easiness in Bayesian inference, conjugate prior of Poisson distribution namely Gamma distribution is chosen as prior distribution.

$$w_{ir} \sim \mathcal{G}(w_{ir}; a^w, b^w/a^w)$$

$$h_{jr} \sim \mathcal{G}(h_{jr}; a^h, b^h/a^h)$$

2.3. EM Algorithm

Both Maximum Likelihood and Maximum a Posteriori(MAP) parameter estimation methods are based on the idea of putting a probability distribution on data (\mathbf{x}) and tuning the parameters of the distribution (θ) such that the likelihood of observed data is maximized under this specified probability distribution. Maximum a Posteriori

solution requires an additional prior distribution on parameters. For numeric simplicity, we generally work in log scale which is a monotonic function that does not change the solution.

- (i) Maximum Likelihood solution: $\arg \max_{\theta} \log p(\mathbf{x}|\theta)$
- (ii) MAP solution: $\arg \max_{\theta} \log p(\theta|\mathbf{x}) = \arg \max_{\theta} \log p(\mathbf{x}|\theta) + \log p(\theta)$

In order to evaluate the likelihood of the data in latent variable models, the latent variables (\mathbf{z}) need to be marginalized out. The number of components that need to be marginalized out increases as the number of data instances increases and makes the marginalization ($\sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}|\theta)$) intractable. Because of the intractable marginalization, one can not compute the likelihood, directly and EM algorithm [37, 38] brings an iterative solution to this problem.

It would be easier to estimate the model parameters if the latent variables were known. EM algorithm replaces the unknown exact latent variable values with the soft values of latent variables. This is achieved in Expectation(E)-step by computing the posterior distributions of latent variables and using them to evaluate the expectation of log-joint distribution. In Maximization(M)-step, model parameters are learned.

As it is stated above, marginalization of latent variables ($\sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}|\theta)$) is intractable most of the time and this makes harder to work directly with log-likelihood. EM algorithm offers a way to work with lower bound to log-likelihood. In order to understand it better, we rewrite the log-likelihood with some arrangements.

$$\begin{aligned} \log p(\mathbf{x}|\theta) &= \log \sum_{\mathbf{z}} q(\mathbf{z}) \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z})} \\ &= \log \mathbb{E}_q \left[\frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z})} \right] \end{aligned}$$

Because logarithm is a concave function, Jensen's inequality can be applied to write the following.

$$\begin{aligned}\log p(\mathbf{x}|\theta) &\geq \mathbb{E}_q \left[\log \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z})} \right] \\ &= \mathbb{E}_q[\log p(\mathbf{x}, \mathbf{z}|\theta)] - \mathbb{E}_q[\log q(\mathbf{z})]\end{aligned}$$

A lower bound for log-likelihood is found above and represented with $\mathcal{L}(q, \theta)$ which is a function that depends on the choice of distribution q and the model parameters θ . We go further and rearrange it

$$\begin{aligned}\mathcal{L}(q, \theta) &= \sum_{\mathbf{z}} q(\mathbf{z})(\log p(\mathbf{z}|\mathbf{x}, \theta) + \log p(\mathbf{x}|\theta)) - \sum_{\mathbf{z}} q(\mathbf{z}) \log q(\mathbf{z}) \\ &= \sum_{\mathbf{z}} q(\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{x}, \theta)}{q(\mathbf{z})} + \sum_{\mathbf{z}} q(\mathbf{z}) \log p(\mathbf{x}|\theta) \\ &= \sum_{\mathbf{z}} q(\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{x}, \theta)}{q(\mathbf{z})} + \log p(\mathbf{x}|\theta)\end{aligned}$$

The first term above, is the negative KL divergence

$$\mathcal{L}(q, \theta) = -KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \theta)) + \log p(\mathbf{x}|\theta)$$

So, log-likelihood can be written as the summation of the KL divergence and a lower bound.

$$\log p(\mathbf{x}|\theta) = \mathcal{L}(q, \theta) + KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \theta)) \quad (2.2)$$

The KL divergence is a divergence metric between two probability distributions and its range is defined on non-negative values. The KL divergence is minimized when the distributions are equal. Therefore, we set $q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x}, \theta)$ to minimize the KL divergence and to maximize the lower bound $\mathcal{L}(q, \theta)$. By replacing $q(\mathbf{z})$ with $p(\mathbf{z}|\mathbf{x}, \theta)$,

we write the maximum $\mathcal{L}(q, \theta)$ for the old θ parameters.

$$L(p(\mathbf{z}|\mathbf{x}, \theta^{old}), \theta) = \sum_{\mathbf{z}} p(\mathbf{z}|\mathbf{x}, \theta^{old}) \log p(\mathbf{x}, \mathbf{z}|\theta) + \sum_{\mathbf{z}} p(\mathbf{z}|\mathbf{x}, \theta^{old}) \log p(\mathbf{z}|\mathbf{x}, \theta^{old})$$

The above equation completes the E-step and enables us to evaluate the log-likelihood for the parameters θ^{old} . The second term of the equation does not depend on θ , but the first term does. So it can be further increased by new θ values and this process refers to the M-step.

E and M steps can also be considered as inference and learning steps. In the E-step, we infer the posteriors of latent variables and in the M-step, we learn the new θ parameters.

Because the second term does not contain new θ parameters, we take the first term and denote it with $Q(\theta|\theta^{old})$. So, E and M steps can be organized as the following.

$$\text{E-step : } Q(\theta|\theta^{old}) = \mathbb{E}_{p(\mathbf{z}|\mathbf{x}, \theta^{old})}[\log p(\mathbf{x}, \mathbf{z}|\theta)]$$

$$\text{M-step : } \theta^{new} = \arg \max_{\theta} Q(\theta|\theta^{old})$$

2.4. Variational Bayes

In the E-step of EM algorithm, we assumed that $p(\mathbf{z}|\mathbf{x})$ can be calculated, but this is not always the case. When $p(\mathbf{z}|\mathbf{x})$ can't be calculated exactly, we approximate to it with a simpler, instrumental distribution $q(\mathbf{z}; \phi)$, where ϕ is variational parameters. Bayesian Inference of Sum Conditioned Poisson Factorization does not contain model parameters, instead, parameters replaced with random variables. Because of this reason, in this section, we do not take model parameters into account.

As it is showed in Equation 2.2, loglikelihood can be expressed as summation of a lower bound(ELBO) and a KL divergence. In this section, we utilize the same

expression with little differences.

$$\log p(\mathbf{x}) = \mathcal{L}(q, \phi) + KL(q(\mathbf{z}; \phi) || p(\mathbf{z}|\mathbf{x}))$$

By considering the easiness, among all possible probability distributions, we choose the fully factorized ones as the probability distribution family for $q(\mathbf{z}; \phi)$ and this is called mean field approximation [38–40].

$$q(\mathbf{z}) = \prod_{i=1}^R q_i(\mathbf{z}_i)$$

where R is the number of latent factors. We put this expression into lower bound equation and then we solve it for each q_i one at a time, by accepting other q_i s are known. Below we show the solution for q_j .

$$\mathcal{L}(q, \phi) = \sum_{\mathbf{z}} \prod_i q_i (\log p(\mathbf{x}, \mathbf{z}) - \sum_i \log q_i)$$

We separate the terms that contain q_j , and the other terms are accepted as constants because they are not affecting our solution.

$$\begin{aligned} \mathcal{L}(q, \phi) &= \sum_{\mathbf{z}_j} q_j \left(\sum_{\mathbf{z}_{i \neq j}} \log p(\mathbf{x}, \mathbf{z}) \prod_{i \neq j} q_i \right) - \sum_{\mathbf{z}_j} q_j \log q_j + \text{const} \\ &= \sum_{\mathbf{z}_j} q_j \mathbb{E}_{q_{i \neq j}} [\log p(\mathbf{x}, \mathbf{z})] - \sum_{\mathbf{z}_j} q_j \log q_j + \text{const} \\ &= \sum_{\mathbf{z}_j} q_j \log \exp(\mathbb{E}_{q_{i \neq j}} [\log p(\mathbf{x}, \mathbf{z})]) - \sum_{\mathbf{z}_j} q_j \log q_j + \text{const} \end{aligned}$$

We want to find q_j which maximizes this expression. If we look at closer to this expression, it can be seen that it is nothing but the negative KL divergence between q_j and $\exp(E_{q_{i \neq j}} [\log p(\mathbf{x}, \mathbf{z})])$ plus some constant.

$$\mathcal{L}(q, \phi) \propto -KL(q_j || \exp(\mathbb{E}_{q_{i \neq j}} [\log p(\mathbf{x}, \mathbf{z})]))$$

So, we find that

$$q_j \propto \exp(\mathbb{E}_{q_{i \neq j}}[\log p(\mathbf{x}, \mathbf{z})]) \quad (2.3)$$

$q_j(\mathbf{z}_j)$ is a probability distribution, so this expression should be normalized. A simple way of proper calculation is associating this expression with a distribution from exponential family. In order to utilize this simple way, we choose the probability distribution family of $q_j(\mathbf{z}_j)$ carefully, by considering conjugacy.

2.5. Gibbs Sampling

As we will show in the upcoming chapters, expectations of random variables are critical in Bayesian inference, but sometimes due to intractable integrations, analytical computation of an expectation becomes impossible. In this case, we can use Monte Carlo integration to approximate to the expectation.

$$\mathbb{E}_{p(x)}[x] \approx \frac{1}{T} \sum_{i=1}^T x^{(i)}$$

where $x^{(i)}$ s are samples from probability distribution $p(x)$.

The above equation requires well distributed samples from target distribution. If the target distribution is not among well known probability distributions, it may be hard to sample from target distribution directly. Monte Carlo methods [41, 42] enable us to generate random samples from probability distributions which are difficult to sample from.

Gibbs sampling [43, 44] is a Markov Chain Monte Carlo(MCMC) [45, 46] algorithm. MCMC utilizes a Kernel $K(x^{(i)}|x^{(i-1)})$ which is basically transition probability Markov Chain of states for sampling. For healthy process, Markov Chain should be irreducible and aperiodic. Probability distribution of the new sample depends on the

old sample and transition probability kernel.

$$p(x^{(i)}) = \int K(x^{(i)}|x^{(i-1)})p(x^{(i-1)})dx^{(i-1)}$$

If we have a set of random variables x_1, x_2, \dots, x_D and full conditional probabilities of each random variables $p(x_d|x_{-d})$ are tractable, we can use Gibbs Sampling. Gibbs Sampling start with randomly initialized values for x_1, x_2, \dots, x_D , and then take samples x_d from $p(x_d|x_{-d})$ for $d = 1 : D$. After some burn in period, distributions of samples from full conditionals will be identical to the distributions of samples from joint distribution which we could not generate in the beginning [25].

Initialize $x_1^{(1)}, x_2^{(1)}, \dots, x_D^{(1)}$ randomly.

for i=2:T **do**

$$x_1^{(i)} \sim p(x_1|x_2^{(i-1)}, x_3^{(i-1)}, \dots, x_D^{(i-1)})$$

$$x_2^{(i)} \sim p(x_2|x_1^{(i)}, x_3^{(i-1)}, \dots, x_D^{(i-1)})$$

...

$$x_D^{(i)} \sim p(x_D|x_1^{(i)}, x_2^{(i)}, \dots, x_{D-1}^{(i)})$$

end for

Figure 2.4. Gibbs sampling for Parafac decomposition

3. PROPOSED MODEL

Sum Conditioned Poisson Factorization [25] is proposed as a general matrix and tensor decomposition framework to model binary, ordinal, and multinomial data using the moment parametrization which is expressed in the Chapter 1 together with the motivations. In this chapter, tensor extension of Sum Conditioned Poisson Factorization is introduced and designed for Knowledge Graph problem. Parafac and Rescal like(Tucker) decompositions are introduced respectively. Before diving into details of the model, reader can find the properties of Poisson random variables which are conditioned on summation of 1, in Appendix A.

3.1. Parafac Decomposition

Basically, the model contains two tensors \underline{X}_1 and its complement \underline{X}_2 . \underline{X}_1 is the real data tensor that we are working with and contains the observed triples, therefore it is formed by 1s and 0s. \underline{X}_2 is complement of tensor \underline{X}_1 and takes the values which make the summation of two tensors a tensor of ones which is stated with \underline{N} in Figure 3.1.

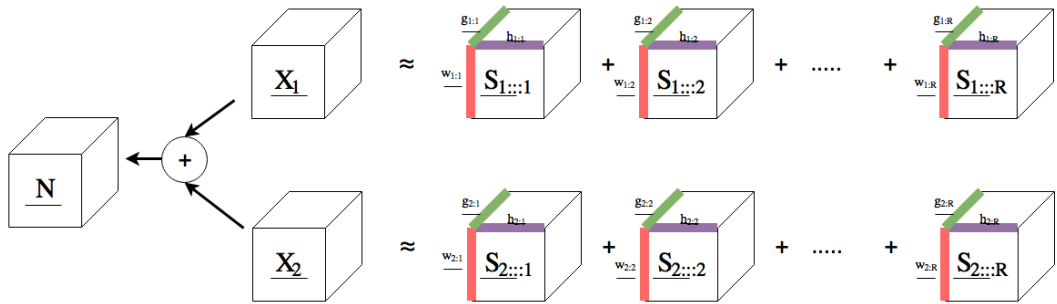


Figure 3.1. Parafac decomposition of Sum Conditioned Poisson tensor factorization.

From now on, \underline{X}_l will be used instead of \underline{X}_1 and \underline{X}_2 if the properties that we use are valid for both of the tensors and same notation will be valid for other representations.

\underline{X}_l is approximated by summation of R tensors and these tensors named as latent sources \underline{S}_l . Values in the tensors \underline{S}_l are sampled from Poisson distribution and the rate tensors are generated with outer product of latent vectors $\underline{w}_{l,:r}$, $\underline{h}_{l,:r}$, $\underline{g}_{l,:r}$ which are the columns of matrices W_l , H_l , G_l respectively. This process is illustrated in Figure 3.1 and Figure 3.2.

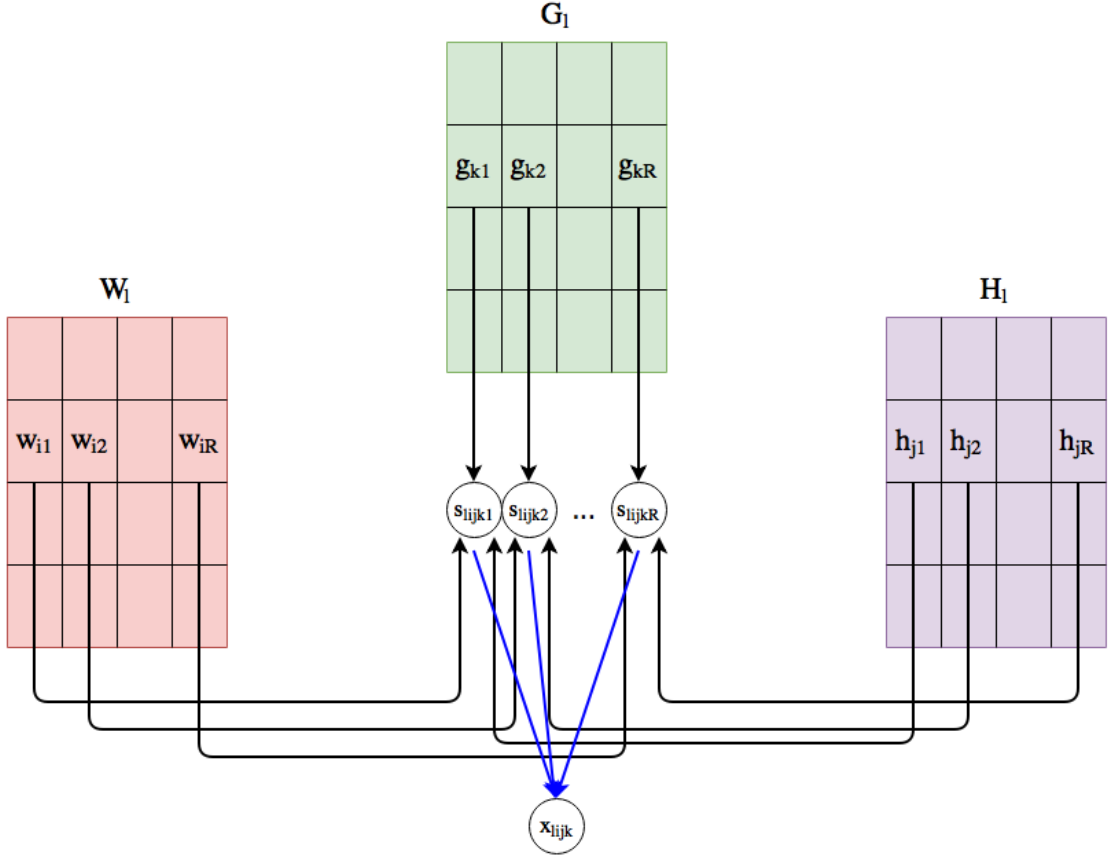


Figure 3.2. Elements of latent matrices interact with each other and generate $x_{l,ijk}$

To simplify the Bayesian inference, we define Gamma prior distributions on $w_{l,ir}$, $h_{l,jr}$, $g_{l,kr}$. Gamma distribution is conjugate prior of Poisson distribution and this makes Bayesian inference stage easier for us. When we use EM algorithm for parameter estimation, we no longer accept $w_{l,ir}$, $h_{l,jr}$, $g_{l,kr}$ as random variables but parameters and prior distributions on them becomes insignificant unless we apply Maximum A

Posterior(MAP) estimation. Full generative model is defined in the equations below.

$$\begin{aligned}
w_{l,ir} &\sim \mathcal{G}(w_{l,ir}; a^w, b^w/a^w) & s_{l,ijk} &\sim \mathcal{PO}(s_{l,ijk}; w_{l,ir} \times h_{l,jr} \times g_{l,kr}) \\
h_{l,jr} &\sim \mathcal{G}(h_{l,jr}; a^h, b^h/a^h) & x_{l,ijk} &= \sum_r s_{l,ijk} \\
g_{l,kr} &\sim \mathcal{G}(g_{l,kr}; a^g, b^g/a^g) & n_{ijk} &= \sum_l x_{l,ijk}
\end{aligned}$$

W_1 and H_1 matrices are latent feature matrices for entities according to their roles(etiher subject or object) in triples. G_1 is the latent feature matrix for relations. Each entities and relations have R dimensional latent features and these features are forming the rows of matrices.

Given their summation, conditional distribution of independent Poisson random variables is Multinomial distribution [47]. For the summation of 1, conditional distribution of two Poisson random variables turns into Bernoulli distribution which is a special case of Multinomial distribution. Thanks to the summation tensor \underline{N} which is a tensor of ones, we have the following property.

$$p(x_{l,ijk} | n_{ijk}, w_{:,i}, h_{:,j}, g_{:,k}) = \text{Ber}\left(x_{l,ijk}; \frac{\sum_r w_{l,ir} h_{l,jr} g_{l,kr}}{\sum_l \sum_r w_{l,ir} h_{l,jr} g_{l,kr}}\right) \quad (3.1)$$

The above property enables us to model knowledge graph problems with Sum Conditioned Poisson Factorization.

In Chapter 1, we stated that we accept closed world assumption. In order to apply this assumption in our models, we introduce a mask tensor \underline{M} which has the same dimensionality with \underline{X}_1 and \underline{X}_2 . m_{ijk} takes the value 1 if we observe the true or false relationship entity _{i} – relation _{k} – entity _{j} , and takes the value 0 if this relationship is not observed. Thus we assume that the tensors we observe are $\underline{X}_1 \odot \underline{M}$ and $\underline{X}_2 \odot \underline{M}$. The same rule is also valid for Rescal-like(Tucker-like) decomposition model.

3.2. RESCAL-like(Tucker-like) Decomposition

In Parafac decomposition model, we assumed that an entity has two different latent representations for its two different roles in triples which are either being object or subject. Now, we assume that an entity has a global latent representation and it does not vary with the role of entity in a triple.

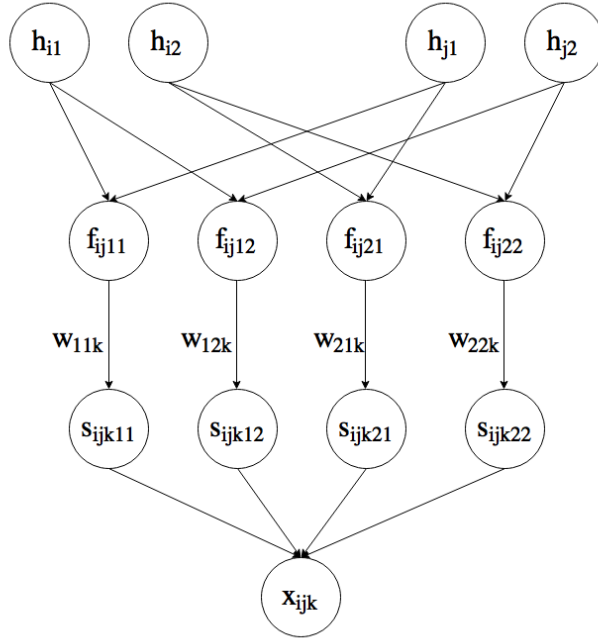


Figure 3.3. Illustration of Tucker-like decomposition for Sum Conditioned Poisson Factorization.

Rescal-like decomposition for Sum Conditioned Poisson Factorization is illustrated in Figure 3.3. In this illustration, rank R is taken 2, and we throw notation l away, because this scheme is valid for both $l = 1$ and $l = 2$. $\underline{h}_{i\cdot}$ is the latent feature vector for entity i and elements of $\underline{h}_{i\cdot}$ are multiplied with elements of $\underline{h}_{j\cdot}$. These multiplications are weighted with weights $W_{::k}$ and generates $S_{ijk::}$ values. An entity has a global latent feature which does not vary with the type of relation or the role of entity in the triple. The features of relations are the weights that determine in what way entity features interact with each other.

Generative model of Rescal-like decomposition for Sum Conditioned Poisson Factorization is given below.

$$\begin{aligned}
 w_{l,cdk} &\sim \mathcal{G}(w_{l,cdk}; a^w, b^w/a^w) & s_{l,ijkcd} &\sim \mathcal{PO}(s_{l,ijkcd}; w_{l,cdk} \times h_{l,ic} \times h_{l,jd}) \\
 h_{l,ir} &\sim \mathcal{G}(h_{l,ir}; a^h, b^h/a^h) & x_{l,ijk} &= \sum_{c,d} s_{l,ijkcd} \\
 n_{ijk} &= \sum_l x_{l,ijk}
 \end{aligned}$$

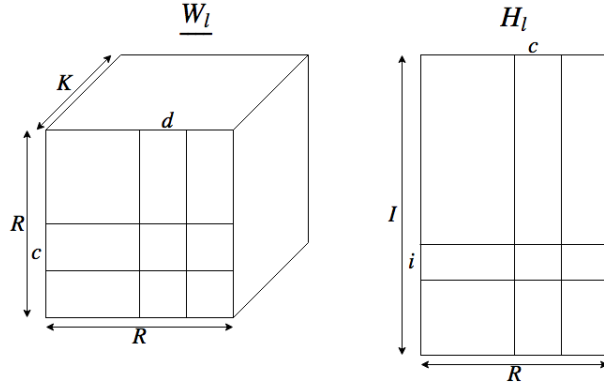


Figure 3.4. An $R \times R \times K$ weight tensor for relations and an $I \times R$ latent features matrix for entities.

In this formulation, \underline{W}_l is a $R \times R \times K$ tensor which keeps latent features or weights of relations, so each relation has its own $R \times R$ matrix shaped weights. H_l is $I \times R$ matrix which keeps latent features of entites, so each entity has its own R dimensional vector shaped features.

By conditioning on the summation of 1, we write the following equation.

$$p(x_{l,ijk} | n_{ijk}, w_{:::k}, h_{:i:}, h_{:j:}) = \text{Ber} \left(x_{l,ijk}; \frac{\sum_{c,d} w_{l,cdk} h_{l,ic} h_{l,jd}}{\sum_l \sum_{c,d} w_{l,cdk} h_{l,ic} h_{l,jd}} \right)$$

Rather than Rescal-like decomposition, a more general Tucker decomposition can be applied by introducing another matrix G_l to represent latent features of entities when their role is object in triples.

4. INFERENCE

In this chapter, we present Maximum Likelihood parameter estimation for both Parafac decomposition and Rescal-like decomposition of Sum Conditioned Poisson Factorization. We derive update equations for parameters with Expectation-Maximization algorithm. Additionally, we present Bayesian inference for Parafac decomposition of SCPF via variational inference and Gibbs Sampling. The derivations follow closely [25, 26].

4.1. EM Algorithm for Parafac Decomposition

Original model accepts that $w_{l,ir}$, $h_{l,jr}$, $g_{l,kr}$ are random variables which are sampled from Gamma Distribution. In Maximum Likelihood learning, we put some restrictions on model such that $w_{l,ir}$, $h_{l,jr}$, $g_{l,kr}$ are not random variables but parameters.

4.1.1. E-step

In Chapter 2, we defined the E-step as evaluating the following expectation.

$$Q(\theta|\theta^{old}) = \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\theta^{old})} [\log p(\mathbf{x}, \mathbf{z}|\theta)]$$

For our model, this equation turns into

$$Q(W, H, G|W^{(t)}, H^{(t)}, G^{(t)}) = \mathbb{E}_{p(\underline{S}|\underline{N}, \underline{X}, W, H, G)} [\log p(\underline{N}, \underline{X}, \underline{S}|W, H, G)] \quad (4.1)$$

We will first evaluate $p(\underline{N}, \underline{X}, \underline{S}|W, H, G)$ and then continue with $p(\underline{S}|\underline{N}, \underline{X}, W, H, G)$ and $\mathbb{E}_{p(\underline{S}|\underline{N}, \underline{X}, W, H, G)} [\log p(\underline{N}, \underline{X}, \underline{S}|W, H, G)]$.

$$\begin{aligned}
p(\underline{N}, \underline{X}, \underline{S}|W, H, G) &= p(\underline{N}|\underline{X})p(\underline{X}|\underline{S})p(\underline{S}|W, H, G) \\
&= \prod_{l,i,j,k} \left[\delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) \delta \left(x_{l,ijk} - \sum_r s_{l,ijk r} \right) \right. \\
&\quad \left. \prod_r Po(s_{l,ijk r}; w_{l,ir} h_{l,jr} g_{l,kr}) \right] \tag{4.2}
\end{aligned}$$

Now, we start to evaluate the posterior of latent sources \underline{S} .

$$\log p(\underline{S}|\underline{N}, \underline{X}, W, H, G) = \log p(\underline{N}, \underline{X}, \underline{S}|W, H, G) - \log p(\underline{N}, \underline{X}|W, H, G) \tag{4.3}$$

In order to evaluate this posterior, we need to calculate $p(\underline{N}, \underline{X}|W, H, G)$ first. We may simply write $p(\underline{N}, \underline{X}|W, H, G) = \sum_{\underline{S}} p(\underline{N}, \underline{X}, \underline{S}|W, H, G)$, but there are some \underline{X} values in the tensor which are not observed, so we need to integrate them out as well. As it is stated in Chapter 3, we use mask parameters m_{ijk} to indicate whether we observe x_{ijk} or not.

$$\begin{aligned}
p(\underline{N}, \underline{X}|W, H, G) &= \\
&\left(\sum_{\underline{S}} \prod_{l,i,j,k} \left[\delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) \delta \left(x_{l,ijk} - \sum_r s_{l,ijk r} \right) \right. \right. \\
&\quad \left. \left. \prod_r Po(s_{l,ijk r}; w_{l,ir} h_{l,jr} g_{l,kr}) \right] \right)^{m_{ijk}} \left(\sum_{\underline{S}} \sum_{\underline{X}} \prod_{l,i,j,k} \left[\delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) \right. \right. \\
&\quad \left. \left. \delta \left(x_{l,ijk} - \sum_r s_{l,ijk r} \right) \prod_r Po(s_{l,ijk r}; w_{l,ir} h_{l,jr} g_{l,kr}) \right] \right)^{(1-m_{ijk})} \tag{4.4}
\end{aligned}$$

Superposition property of Poisson distribution [25, 26, 48] allows us to transform the above equation into a simpler form.

$$\begin{aligned}
\log p(\underline{N}, \underline{X}|W, H, G) &= \sum_{l,i,j,k} m_{ijk} \log Po(x_{l,ijk}; \sum_r w_{l,ir} h_{l,jr} g_{l,kr}) \\
&\quad + \sum_{i,j,k} (1 - m_{ijk}) \log Po(n_{ijk}; \sum_l \sum_r w_{l,ir} h_{l,jr} g_{l,kr}) \tag{4.5}
\end{aligned}$$

By using Equation 4.2 and Equation 4.4, we can derive $p(\underline{S}|\underline{N}, \underline{X}, W, H, G)$. Details of the derivation can be found in Appendix B.

$$\begin{aligned} \log p(\underline{S}|\underline{N}, \underline{X}, W, H, G) &= \sum_{l,i,j,k} m_{ijk} \log \mathcal{M}(s_{l,ijk}; x_{l,ijk}, p_{l,ijk}) \\ &\quad + \sum_{l,i,j,k} (1 - m_{ijk}) \log \mathcal{M}(s_{l,ijk}; n_{ijk}, q_{l,ijk}) \end{aligned} \quad (4.6)$$

$$\text{where } p_{l,ijk} = \frac{w_{l,ir} h_{l,jr} g_{l,kr}}{\sum_r w_{l,ir} h_{l,jr} g_{l,kr}} \text{ and } q_{l,ijk} = \frac{w_{l,ir} h_{l,jr} g_{l,kr}}{\sum_{l,r} w_{l,ir} h_{l,jr} g_{l,kr}}.$$

Our aim is to evaluate the expectation in the Equation 4.1, so we write it detaily.

$$\begin{aligned} \mathbb{E}_{p(\underline{S}|\underline{N}, \underline{X}, W, H, G)} [\log p(\underline{N}, \underline{X}, \underline{S}|W, H, G)] &= \mathbb{E} \left[\sum_{l,i,j,k} \left(\sum_r \log \mathcal{PO}(s_{l,ijk}; w_{l,ir} h_{l,jr} g_{l,kr}) \right. \right. \\ &\quad \left. \left. + \log \delta \left(x_{l,ijk} - \sum_r s_{l,ijk} \right) \right. \right. \\ &\quad \left. \left. + \log \delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) \right) \right]_{p(\underline{S}|\underline{N}, \underline{X}, W, H, G)} \\ &= \sum_{l,i,j,k} \left(\sum_r \left(\mathbb{E}[s_{l,ijk}] \log w_{l,ir} h_{l,jr} g_{l,kr} \right. \right. \\ &\quad \left. \left. - w_{l,ir} h_{l,jr} g_{l,kr} - \mathbb{E}[\log \Gamma(s_{l,ijk} + 1)] \right) \right. \\ &\quad \left. + \mathbb{E} \left[\log \delta \left(x_{l,ijk} - \sum_r s_{l,ijk} \right) \right] \right. \\ &\quad \left. + \log \delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) \right) \end{aligned} \quad (4.7)$$

We will optimize the objective function with respect to model parameters, so we keep the terms that include them and write the objective function.

$$Q(W, H, G|W^{(t)}, H^{(t)}, G^{(t)}) = \sum_{l,i,j,k,r} (\mathbb{E}[s_{l,ijk}] \log w_{l,ir} h_{l,jr} g_{l,kr} - w_{l,ir} h_{l,jr} g_{l,kr}) \quad (4.8)$$

Evaluating $\mathbb{E}[s_{l,ijk}]$ is easy now, because in Equation 4.6 we calculated the conditional distribution $p(\underline{S}|\underline{N}, \underline{X}, W, H, G)$ and it is nothing but combination of two multinomial distributions.

$$\mathbb{E}_{p(\underline{S}|\underline{N},\underline{X},W,H,G)} [s_{l,ijk}] = m_{ijk} \frac{w_{l,ir} h_{l,jr} g_{l,kr} x_{l,ijk}}{\sum_r w_{l,ir} h_{l,jr} g_{l,kr}} + (1 - m_{ijk}) \frac{w_{l,ir} h_{l,jr} g_{l,kr} n_{ijk}}{\sum_l \sum_r w_{l,ir} h_{l,jr} g_{l,kr}} \quad (4.9)$$

This completes the E-step and parameters need to be updated in the M-step.

4.1.2. M-Step

In the E-step of EM algorithm, we derived the objective function that we would like to maximize. In the M-step, we derive update equations for parameters, by simply taking derivative of $Q(W, H, G|W^{(t)}, H^{(t)}, G^{(t)})$ with respect to parameters $w_{l,ir}$, $h_{l,jr}$, $g_{l,kr}$ and equalizing to 0.

$$\frac{\partial Q}{\partial w_{l,ir}} = 0 \Rightarrow w_{l,ir}^{(t+1)} = \frac{\sum_{j,k} \mathbb{E}_{p(\underline{S}|\underline{N},\underline{X},W,H,G)} [s_{l,ijk}]}{\sum_{j,k} h_{l,jr} g_{l,kr}} \quad (4.10)$$

$$\frac{\partial Q}{\partial h_{l,jr}} = 0 \Rightarrow h_{l,jr}^{(t+1)} = \frac{\sum_{i,k} \mathbb{E}_{p(\underline{S}|\underline{N},\underline{X},W,H,G)} [s_{l,ijk}]}{\sum_{i,k} w_{l,ir} g_{l,kr}} \quad (4.11)$$

$$\frac{\partial Q}{\partial g_{l,kr}} = 0 \Rightarrow g_{l,kr}^{(t+1)} = \frac{\sum_{i,j} \mathbb{E}_{p(\underline{S}|\underline{N},\underline{X},W,H,G)} [s_{l,ijk}]}{\sum_{i,j} w_{l,ir} h_{l,jr}} \quad (4.12)$$

By using Equation 4.9, we rewrite the update equations in Table 4.1.

Table 4.1. EM update equations for Parafac decomposition.

Parameters	Update equations
$w_{l,ir}$	$\frac{w_{l,ir}^t}{\sum_{j,k} h_{l,jr}^t g_{l,kr}^t} \sum_{j,k} \left[\frac{m_{ijk} h_{l,jr}^t g_{l,kr}^t x_{l,ijk}}{\sum_r w_{l,ir}^t h_{l,jr}^t g_{l,kr}^t} + \frac{(1-m_{ijk}) h_{l,jr}^t g_{l,kr}^t n_{ijk}}{\sum_{l,r} w_{l,ir}^t h_{l,jr}^t g_{l,kr}^t} \right]$
$h_{l,jr}$	$\frac{h_{l,jr}^t}{\sum_{i,k} w_{l,ir}^t g_{l,kr}^t} \sum_{i,k} \left[\frac{m_{ijk} w_{l,ir}^t g_{l,kr}^t x_{l,ijk}}{\sum_r w_{l,ir}^t h_{l,jr}^t g_{l,kr}^t} + \frac{(1-m_{ijk}) w_{l,ir}^t g_{l,kr}^t n_{ijk}}{\sum_{l,r} w_{l,ir}^t h_{l,jr}^t g_{l,kr}^t} \right]$
$g_{l,kr}$	$\frac{g_{l,kr}^t}{\sum_{i,j} w_{l,ir}^t h_{l,jr}^t} \sum_{i,j} \left[\frac{m_{ijk} w_{l,ir}^t h_{l,jr}^t x_{l,ijk}}{\sum_r w_{l,ir}^t h_{l,jr}^t g_{l,kr}^t} + \frac{(1-m_{ijk}) w_{l,ir}^t h_{l,jr}^t n_{ijk}}{\sum_{l,r} w_{l,ir}^t h_{l,jr}^t g_{l,kr}^t} \right]$

4.2. Variational Bayes for Parafac Decomposition

In this section, we introduce W, H, G latent variables with gamma priors for each, as it is defined in the original model description in Chapter 3. In the presence of latent variables, computation of posterior distribution $p(\underline{S}, W, H, G | \underline{X}, \underline{N})$ becomes intractable. We introduce a fully factorized instrumental distribution $q(\underline{S}, W, H, G)$ in order to approximate to original posterior distribution under the reverse KL divergence $KL(q||p)$ metric.

$$q(\underline{S}, W, H, G) = q(\underline{S})q(W)q(H)q(G)$$

A general form of fixed point iteration is supplied in Equation 2.3. Same procedure is applied for SCPF.

$$\begin{aligned} q(\underline{S})^{(t+1)} &\propto \exp \left(\langle \log p(\underline{N}, \underline{X}, \underline{S}, W, H, G) \rangle_{q(W)^{(t)} q(H)^{(t)} q(G)^{(t)}} \right) \\ q(W)^{(t+1)} &\propto \exp \left(\langle \log p(\underline{N}, \underline{X}, \underline{S}, W, H, G) \rangle_{q(\underline{S})^{(t+1)} q(H)^{(t)} q(G)^{(t)}} \right) \\ q(H)^{(t+1)} &\propto \exp \left(\langle \log p(\underline{N}, \underline{X}, \underline{S}, W, H, G) \rangle_{q(\underline{S})^{(t+1)} q(W)^{(t+1)} q(G)^{(t)}} \right) \\ q(G)^{(t+1)} &\propto \exp \left(\langle \log p(\underline{N}, \underline{X}, \underline{S}, W, H, G) \rangle_{q(\underline{S})^{(t+1)} q(W)^{(t+1)} q(H)^{(t+1)}} \right) \end{aligned}$$

Log-joint distribution $\log p(\underline{N}, \underline{X}, \underline{S}, W, H, G)$ is given in Appendix C. By keeping the terms that include the random variables \underline{S} , we write the following.

$$q(s_{l,ijk:}) \propto \exp \left(\sum_r s_{l,ijk:r} (\langle \log w_{l,ir} \rangle + \langle \log h_{l,jr} \rangle + \langle \log g_{l,kr} \rangle) - \log \Gamma(s_{l,ijk:} + 1) \right) \\ \left(m_{ijk} \delta \left(x_{l,ijk} - \sum_r s_{l,ijk:r} \right) + (1 - m_{ijk}) \delta \left(n_{ijk} - \sum_{l,r} s_{l,ijk:r} \right) \right)$$

$$q(\underline{S}) \propto \prod_{l,i,j,k} [m_{ijk} \mathcal{M}(s_{l,ijk:}, x_{l,ijk}, p_{l,ijk:}) + (1 - m_{ijk}) \mathcal{M}(s_{l,ijk:}, n_{ijk}, q_{l,ijk:})] \quad (4.13)$$

The variational distribution of \underline{S} is Multinomial distribution. After recognizing distribution family, it is easy to find expectations which will be used later in derivations of fixed point iterations for w, h, g .

$$\langle s_{l,ijk:r} \rangle = m_{ijk} x_{l,ijk} p_{l,ijk:r} + (1 - m_{ijk}) n_{ijk} q_{l,ijk:r} \quad (4.14)$$

$$\text{for } p_{l,ijk:r} = \frac{\exp(\langle \log w_{l,ir} \rangle + \langle \log h_{l,jr} \rangle + \langle \log g_{l,kr} \rangle)}{\sum_r \exp(\langle \log w_{l,ir} \rangle + \langle \log h_{l,jr} \rangle + \langle \log g_{l,kr} \rangle)}, \quad q_{l,ijk:r} = \frac{\exp(\langle \log w_{l,ir} \rangle + \langle \log h_{l,jr} \rangle + \langle \log g_{l,kr} \rangle)}{\sum_{l,r} \exp(\langle \log w_{l,ir} \rangle + \langle \log h_{l,jr} \rangle + \langle \log g_{l,kr} \rangle)}$$

Similarly, we derive fixed point iterations for w .

$$q(w_{l,ir}) \propto \exp \left(\left(a_l^w + \sum_{j,k} \langle s_{l,ijk:r} \rangle - 1 \right) \log w_{l,ir} - \left(\frac{a_l^w}{b_l^w} + \sum_{j,k} \langle h_{l,jr} \rangle \langle g_{l,kr} \rangle \right) w_{l,ir} \right) \\ \propto \mathcal{G}(w_{l,ir}; \alpha_{l,ir}^w, \beta_{l,ir}^w)$$

where

$$\alpha_{l,ir}^w = a_l^w + \sum_{j,k} \langle s_{l,ijk:r} \rangle \quad \beta_{l,ir}^w = \left(\frac{a_l^w}{b_l^w} + \sum_{j,k} \langle h_{l,jr} \rangle \langle g_{l,kr} \rangle \right)^{-1} \\ \exp(\langle \log w_{l,ir} \rangle) = \exp(\Psi(\alpha_{l,ir}^w)) \beta_{l,ir}^w \quad \langle w_{l,ir} \rangle = \alpha_{l,ir}^w \beta_{l,ir}^w$$

$\Psi(\cdot)$ is Digamma function. Derivations for fixed point iterations of h and g are analogous to w and supplied in Appendix C.

4.3. Gibbs Sampling for Parafac Decomposition

Gibbs Sampling algorithm is given in Chapter 2. The algorithm requires sampling from full conditional distributions. So, in this section, we supply the full conditional probability distributions of random variables s, w, h, g . In Figure 4.1, the Gibbs Sampling algorithm for Parafac Decomposition can be found.

```

Initialize  $s_{l,ijk}^{(1)}, w_{l,ir}^{(1)}, h_{l,jr}^{(1)}, g_{l,kr}^{(1)}$ , for  $i = 1 : I, j = 1 : J, k = 1 : K, l = 1 : 2, r = 1 : R$ 
for t=2:T do
  for each l,i,j,k,r do
     $s_{l,ijk}^{(t)} \sim p(s_{l,ijk} | W^{(t-1)}, H^{(t-1)}, G^{(t-1)}, \underline{S}_{-s_{l,ijk}}^{(t-1)}, \underline{X}, \underline{N})$ 
     $w_{l,ir}^{(t)} \sim p(w_{l,ir} | W_{-w_{l,ir}}^{(t-1)}, H^{(t-1)}, G^{(t-1)}, \underline{S}^{(t)}, \underline{X}, \underline{N})$ 
     $h_{l,jr}^{(t)} \sim p(h_{l,jr} | W^{(t)}, H_{-h_{l,jr}}^{(t-1)}, G^{(t-1)}, \underline{S}^{(t)}, \underline{X}, \underline{N})$ 
     $g_{l,kr}^{(t)} \sim p(g_{l,kr} | W^{(t)}, H^{(t)}, G_{-g_{l,kr}}^{(t-1)}, \underline{S}^{(t)}, \underline{X}, \underline{N})$ 
  end for
end for

```

Figure 4.1. Gibbs sampling for Parafac decomposition

Conditional probability distribution for \underline{S} is given in Equation 4.6, and it is nothing but combination of two Multinomial distributions. Therefore, sampling of $s_{l,ijk}$ can be done easily by sampling from one of the Multinomial distributions according to observation situation of x_{ijk} .

$$s_{l,ijk} \sim \begin{cases} M(s_{l,ijk}; x_{l,ijk}, p_{l,ijk}), & \text{if } m_{ijk} = 1 \\ M(s_{l,ijk}; n_{ijk}, q_{l,ijk}), & \text{otherwise} \end{cases}$$

$$\text{where } p_{l,ijk} = \frac{w_{l,ir} h_{l,jr} g_{l,kr}}{\sum_r w_{l,ir} h_{l,jr} g_{l,kr}} \text{ and } q_{l,ijk} = \frac{w_{l,ir} h_{l,jr} g_{l,kr}}{\sum_{l,r} w_{l,ir} h_{l,jr} g_{l,kr}}.$$

Sampling scheme for $s_{l,ijk}$ intuitively makes sense. If $x_{l,ijk}$ is observed, then its value is distributed to R piece of $s_{l,ijk}$ latent sources according to their weights $p_{l,ijk}$ and these R piece of $s_{l,ijk}$ latent sources generate $x_{l,ijk}$ indeed. If $x_{l,ijk}$ is not observed, then summed value n_{ijk} is distributed to $L \times R$ piece of $s_{l,ijk}$ latent variables according to their weights $q_{l,ijk}$ because these $L \times R$ latent variables generate the value of n_{ijk} .

In Chapter 3, model description has been made and prior distribution of random variables W, H, G was selected as Gamma distribution by emphasizing its easiness for the inference. Now, we will use it to derive full conditional distributions for W, H, G .

$$\begin{aligned}
\log p(w_{l,ir} | W_{-w_{l,ir}}, H, G, S, X, N) &\propto \log p(w_{l,ir}) + \log p(s_{l,i:r} | w_{l,ir}, h_{l,:r}, g_{l,:r}) \\
&= \log p(w_{l,ir}) + \sum_{j,k} \log p(s_{l,ijk} | w_{l,ir}, h_{l,jr}, g_{l,kr}) \\
&= \log \mathcal{G}(w_{l,ir}; a^w, b^w/a^w) \\
&\quad + \sum_{j,k} \log \mathcal{PO}(s_{l,ijk} | w_{l,ir} h_{l,jr} g_{l,kr}) \\
&= (a^w - 1) \log w_{l,ir} - w_{l,ir} (a^w/b^w) \\
&\quad - \log \Gamma(a^w) + a^w \log(b^w/a^w) \\
&\quad + \sum_{j,k} (-w_{l,ir} h_{l,jr} g_{l,kr} + s_{l,ijk} \log(w_{l,ir} h_{l,jr} g_{l,kr}) \\
&\quad \quad - s_{l,ijk}!) \\
&\propto (a^w + \sum_{j,k} s_{l,ijk} - 1) \log w_{l,ir} \\
&\quad - w_{l,ir} (a^w/b^w + \sum_{j,k} h_{l,jr} g_{l,kr})
\end{aligned}$$

$$\log p(w_{l,ir} | W_{-w_{l,ir}}, H, G, S, X, N) \propto \log \mathcal{G}\left(w_{l,ir}; a^w + \sum_{j,k} s_{l,ijk}, (a^w/b^w + \sum_{j,k} h_{l,jr} g_{l,kr})^{-1}\right) \quad (4.15)$$

Because Gamma distribution is conjugate prior of Poisson distribution, posterior distribution is in the form of Gamma distribution and sampling is really simple now.

Conditional distributions of H and G can be derived similarly, and details are given in the Appendix D.

$$\log p(h_{l,jr}|H_{-h_{l,jr}}, W, G, S, X, N) \propto \log \mathcal{G}\left(h_{l,jr}; a^h + \sum_{i,k} s_{likr}, (a^h/b^h + \sum_{i,k} w_{lir}g_{lkr})^{-1}\right) \quad (4.16)$$

$$\log p(g_{l,kr}|G_{-g_{l,kr}}, W, H, S, X, N) \propto \log \mathcal{G}\left(g_{l,kr}; a^g + \sum_{i,j} s_{lijkr}, (a^g/b^g + \sum_{i,j} w_{lir}h_{ljr})^{-1}\right) \quad (4.17)$$

The main task of Knowledge Graph is extraction of new facts, so it is a prediction task and we need to use our model for this purpose. To do that, we can use the sampled latent variables s_{lijkr} . Expected value of s_{lijkr} can be stated as the following.

$$\mathbb{E}[s_{l,ijk}] = \frac{1}{T - T_{\text{burn-in}}} \sum_{t=T_{\text{burn-in}}}^T s_{l,ijk}^{(t)} \quad (4.18)$$

We do not consider the samples which are gathered in the burn-in period, because Gibbs iterations need to converge so that the samples from full conditionals behave like if they were gathered from joint distribution.

By using Equation 3.1 and Equation 4.18, we can state the expectation of $x_{l,ijk}$ with the following equation.

$$\mathbb{E}[x_{l,ijk}] = \frac{\sum_r \mathbb{E}[s_{l,ijk}]}{\sum_{l,r} \mathbb{E}[s_{l,ijk}]} \quad (4.19)$$

4.4. EM Algorithm for Rescal-like Decomposition

EM algorithm for Rescal-like Decomposition is very much similar to EM algorithm of Parafac Decomposition, nevertheless we will supply all the details in this section.

4.4.1. E-step

We start with objective function that we want to maximize.

$$Q(W, H|W^{(t)}, H^{(t)}) = \mathbb{E}_{p(\underline{S}|\underline{N}, \underline{X}, W, H)} [\log p(\underline{N}, \underline{X}, \underline{S}|W, H)] \quad (4.20)$$

In order to calculate the objective function, we need to calculate the conditional distribution $p(\underline{S}|\underline{N}, \underline{X}, W, H)$ together with the expectation of conditional distribution $p(\underline{N}, \underline{X}, \underline{S}|W, H)$. We start with $p(\underline{N}, \underline{X}, \underline{S}|W, H)$.

$$\begin{aligned} p(\underline{N}, \underline{X}, \underline{S}|W, H) &= p(\underline{N}|\underline{X})p(\underline{X}|\underline{S})p(\underline{S}|W, H) \\ &= \prod_{l,i,j,k} \left[\delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) \delta \left(x_{l,ijk} - \sum_{c,d} s_{l,ijkcd} \right) \right. \\ &\quad \left. \prod_{c,d} Po(s_{l,ijkcd}; w_{l,cdk} h_{l,ic} h_{l,jd}) \right] \end{aligned} \quad (4.21)$$

As it is done in Section 4.1, we marginalize latent variables \underline{S} out. Additionally unobserved \underline{X} variables have to be marginalized out.

$$\begin{aligned} p(\underline{N}, \underline{X}|W, H) &= \\ &\left(\sum_{\underline{S}} \prod_{l,i,j,k} \delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) \delta \left(x_{l,ijk} - \sum_{c,d} s_{l,ijkcd} \right) \right. \\ &\quad \left. \prod_{c,d} \mathcal{PO}(s_{l,ijkcd}; w_{l,cdk} h_{l,ic} h_{l,jd}) \right)^{m_{ijk}} \left(\sum_{\underline{S}} \sum_{\underline{X}} \prod_{l,i,j,k} \delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) \right. \\ &\quad \left. \delta \left(x_{l,ijk} - \sum_{c,d} s_{l,ijkcd} \right) \prod_{c,d} Po(s_{l,ijkcd}; w_{l,cdk} h_{l,ic} h_{l,jd}) \right)^{(1-m_{ijk})} \end{aligned} \quad (4.22)$$

We use the superposition property of Poisson distribution to reach the below equation.

$$\begin{aligned} \log p(\underline{N}, \underline{X}|W, H) &= \sum_{l,i,j,k} m_{ijk} \log \mathcal{PO} \left(x_{l,ijk}; \sum_{c,d} w_{l,cdk} h_{l,ic} h_{l,jd} \right) \\ &\quad + \sum_{i,j,k} (1 - m_{ijk}) \log \mathcal{PO} \left(n_{ijk}; \sum_l \sum_{c,d} w_{l,cdk} h_{l,ic} h_{l,jd} \right) \end{aligned} \quad (4.23)$$

Posterior probability $p(\underline{S}|\underline{N}, \underline{X}, W, H)$ can be calculated by using Equation 4.21 and Equation 4.23.

$$\begin{aligned}
\log p(\underline{S}|\underline{N}, \underline{X}, W, H) &= \log p(\underline{N}, \underline{X}, \underline{S}|W, H) - \log p(\underline{N}, \underline{X}|W, H) \\
&= \sum_{l,i,j,k} m_{i,j,k} \log M(s_{l,ijk::}; x_{l,ijk}, p_{l,ijk::}) \\
&\quad + \sum_{l,i,j,k} (1 - m_{i,j,k}) \log M(s_{:,ijk::}; n_{ijk}, q_{:,ijk::})
\end{aligned} \tag{4.24}$$

where $p_{l,ijkcd} = \frac{w_{l,cdk} h_{l,ic} h_{l,jd}}{\sum_{c,d} w_{l,cdk} h_{l,ic} h_{l,jd}}$ and $q_{l,ijkcd} = \frac{w_{l,cdk} h_{l,ic} h_{l,jd}}{\sum_{l,c,d} w_{l,cdk} h_{l,ic} h_{l,jd}}$.

After calculating required components, we can evaluate the expectation.

$$\begin{aligned}
\mathbb{E}_{p(\underline{S}|\underline{N}, \underline{X}, W, H)} [\log p(\underline{N}, \underline{X}, \underline{S}|W, H)] &= \mathbb{E} \left[\sum_{l,i,j,k} \left(\sum_{c,d} \log \mathcal{PO}(s_{l,ijkcd}; w_{l,cdk} h_{l,ic} h_{l,jd}) \right. \right. \\
&\quad \left. \left. + \log \delta \left(x_{l,ijk} - \sum_r s_{l,ijk r} \right) \right. \right. \\
&\quad \left. \left. + \log \delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) \right) \right]_{p(\underline{S}|\underline{N}, \underline{X}, W, H)} \\
&= \sum_{l,i,j,k} \left(\sum_{c,d} \left(\mathbb{E}[s_{l,ijkcd}] \log w_{l,cdk} h_{l,ic} h_{l,jd} \right. \right. \\
&\quad \left. \left. - w_{l,cdk} h_{l,ic} h_{l,jd} - \mathbb{E}[\log \Gamma(s_{l,ijkcd} + 1)] \right) \right. \\
&\quad \left. + \mathbb{E} \left[\log \delta \left(x_{l,ijk} - \sum_{c,d} s_{l,ijkcd} \right) \right] \right. \\
&\quad \left. + \log \delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) \right)
\end{aligned} \tag{4.25}$$

So, the objective function becomes

$$Q(W, H|W^{(t)}, H^{(t)}) = \sum_{l,i,j,k,c,d} (\mathbb{E}[s_{l,ijkcd}] \log w_{l,cdk} h_{l,ic} h_{l,jd} - w_{l,cdk} h_{l,ic} h_{l,jd}) \tag{4.26}$$

By using Equation 4.24, we write

$$\mathbb{E}[s_{l,ijkcd}] = m_{ijk} \frac{w_{l,cdk} h_{l,ic} h_{l,jd} x_{l,ijk}}{\sum_{c,d} w_{l,cdk} h_{l,ic} h_{l,jd}} + (1 - m_{ijk}) \frac{w_{l,cdk} h_{l,ic} h_{l,jd} n_{ijk}}{\sum_l \sum_{c,d} w_{l,cdk} h_{l,ic} h_{l,jd}} \quad (4.27)$$

4.4.2. M-step

In this section we derive update equations for the parameters W and H . Because of the definition of Rescal-like decomposition model, objective function contains both $h_{l,ic}$ and $h_{l,jd}$ which are latent representations of entities in the role of object and subject respectively. We keep $h_{l,jd}$ constant and derive equations for $h_{l,ic}$.

$$\frac{\partial Q}{\partial w_{l,cdk}} = 0 \Rightarrow w_{l,cdk}^{(t+1)} = \frac{\sum_{i,j} \mathbb{E}[s_{l,ijkcd}]}{\sum_{i,j} h_{l,ic} h_{l,jd}} \quad (4.28)$$

$$\frac{\partial Q}{\partial h_{l,ic}} = 0 \Rightarrow h_{l,ic}^{(t+1)} = \frac{\sum_{j,k,d} \mathbb{E}[s_{l,ijkcd}]}{\sum_{j,k,d} w_{l,cdk} h_{l,jd}} \quad (4.29)$$

Table 4.2. EM update equations for Rescal-like decomposition.

Parameters	Update equations
$w_{l,cdk}$	$\frac{w_{l,cdk}^t}{\sum_{i,j} h_{l,ic}^t h_{l,jd}^t} \sum_{i,j} \left[\frac{m_{ijk} h_{l,ic}^t h_{l,jd}^t x_{l,ijk}}{\sum_{c,d} w_{l,cdk}^t h_{l,ic}^t h_{l,jd}^t} + \frac{(1-m_{ijk}) h_{l,ic}^t h_{l,jd}^t n_{ijk}}{\sum_{l,c,d} w_{l,cdk}^t h_{l,ic}^t h_{l,jd}^t} \right]$
$h_{l,ic}$	$\frac{h_{l,ic}^t}{\sum_{j,k,d} w_{l,cdk}^t h_{l,jd}^t} \sum_{j,k,d} \left[\frac{m_{ijk} w_{l,cdk}^t h_{l,jd}^t x_{l,ijk}}{\sum_{c,d} w_{l,cdk}^t h_{l,jd}^t h_{l,ic}^t} + \frac{(1-m_{ijk}) w_{l,cdk}^t h_{l,jd}^t n_{ijk}}{\sum_{l,c,d} w_{l,cdk}^t h_{l,jd}^t h_{l,ic}^t} \right]$

5. EXPERIMENTS

Predictive power of Sum Conditioned Poisson Factorization is tested on 3 standard knowledge graph datasets(Nation, UMLS, Kinship). These datasets are highly imbalanced and the number of negative examples is much more higher than the number of positive examples. As it is stated before, we accept closed world assumption by assuming that the given triples are positive examples and not given triples are negative examples. For each knowledge graph, we create 10 different datasets by randomly splitting the knowledge graph into training set(50%) and test set(50%). We use area under the ROC curve [49] as evaluation metric and compare the predictive performances of Sum Conditioned Poisson Factorization models and Logistic Tensor Factorization. Furthermore, we visualize the estimated features to examine if they are interpretable or not.

5.1. Nation Dataset

Nation dataset [50] is formed by 14 nations and 56 relations between them. This dataset is relatively small sized when compared with the other two datasets and ratio of positive examples to negative examples is higher than the other datasets’.

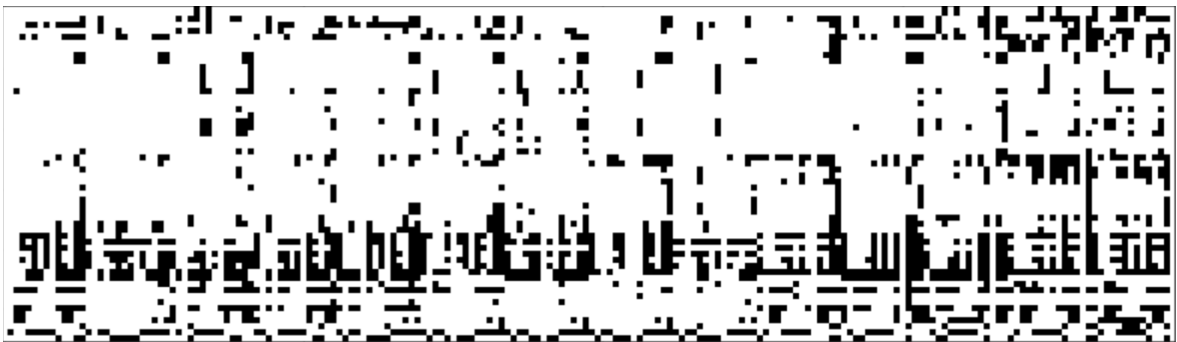


Figure 5.1. Visualization of Nation dataset.

In Figure 5.1, we visualize the Nation dataset by transforming $14 \times 14 \times 56$ dimensional tensor to a $14 \times (14 \times 56)$ dimensional matrix where black cells represent the known triples.

Table 5.1. Entities and relations in Nation dataset.

Entities	Relations
Brazil, Burma, China, Cuba, Egypt, India, Indonesia, Israel, Jordan, Netherlands, Poland, USSR, UK, USA	economicaid, releconomicaid, treaties, reltreaties, officialvisits, conferences, exportbooks, relexportbooks, booktranslations, relbooktranslations, warning, violentactions, militaryactions, duration, negativebehavior, severdiplomatic, expeldiplomats, boycottembargo, aidenemy, negativecomm, accusation, protests, unoffialacts, attackembassy, nonviolentbehavior, weightedunvote, unweightedunvote, tourism, reltourism, tourism3, emigrants, relemigrants, emigrants3, students, relstudents, exports, reexports, exports3, intergovorgs, relintergovorgs, ngo, relngo, intergovorgs3, ngoorgs3, embassy, reldiplomacy, timesincewar, timesinceally, lostterritory, dependent, independence, commonbloc0, blockpositionindex, militaryalliance, commonbloc1, commonbloc2

Parameter estimation with EM algorithm was applied to Parafac and Tucker SCPF models. We implemented also Bayesian Inference for Parafac SCPF with both variational inference and Gibbs sampling. We applied the same training procedures for latent ranks 2, 4, and 8. During training, latent features were estimated/inferred, and these features were used in test phase to extract new facts. A threshold value needs to be determined to label the positive and negative triples. A large threshold value causes most of the test examples to be labeled negative, in the opposite case,

most of the examples are labeled positive. We measured the true-positive-rates and false-positive-rates for various threshold values and we constructed ROC curves to use them as performance evaluation metric. Additionally, in the original LTF implementation, authors prefer to use L-BFGS algorithm [51] which is a quasi-Newton method for optimization. We reimplemented LTF by using Tensorflow [52] library which supplies automatic differentiation and optimization tools. Because L-BFGS algorithm is not available in Tensorflow, we preferred Stochastic Gradient Descent. For variational inference, we set the shape and the mean parameters of Gamma distribution to 1. and 0.5 respectively. For Gibbs sampling these parameters were set to 0.3 and 0.5 respectively. 1000 samples with 200 burn-in samples were gathered for Gibbs sampling.

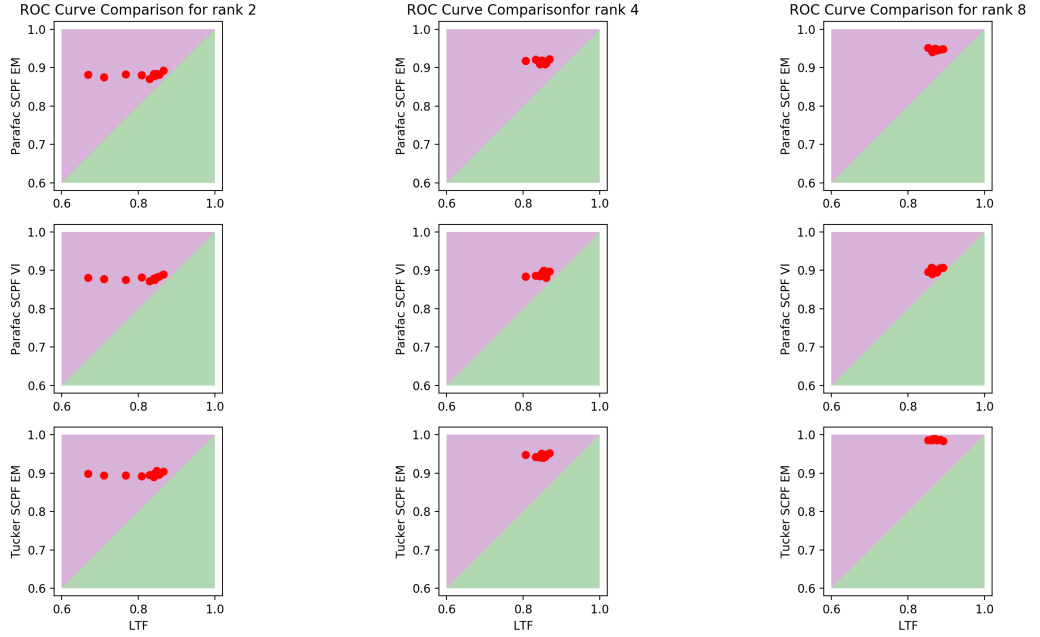


Figure 5.2. Area under the ROC curve comparison for training set.

We did not evaluate Gibbs sampling for training set because the samples for latent sources \underline{S} are used to estimate the masked \underline{X} values and these samples compose exactly the unmasked \underline{X} values.

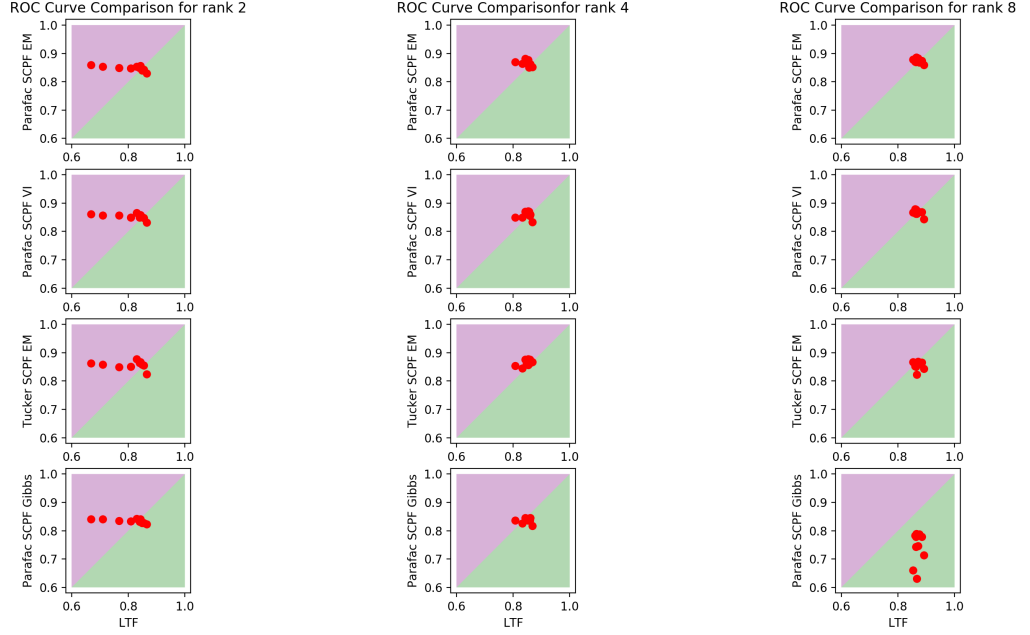


Figure 5.3. Area under the ROC curve comparison for test set.

Both Parafac and Tucker SCPF models are superior to LTF in training set evaluation. In testing phase, performances of SCPF models are better than or equal to LTF's, except the Gibbs sampling with rank 8.

In order to examine the interpretability of latent features, we visualized them. In Figure 5.4, entity latent features that are estimated with Parafac SCPF(EM) are visualized and the large values are represented by dark colors. Just by looking at these features, some similarities between countries can be detected e.g. USA and UK. Beside of latent features, we also visualized latent features for some of the relations in Figure 5.5.

In SCPF, latent feature values are all non-negative and their additive combinations generate the observations. This enables us to associate the nation feature vector indices with some global characteristics. For example, the first and the last subject vector indices are directly proportional to a nation's potential of economic aid to other

countries. Hence, these indices are somehow related to a nation's economic power and we can categorize USA, UK, and Netherlands as prosperous countries.

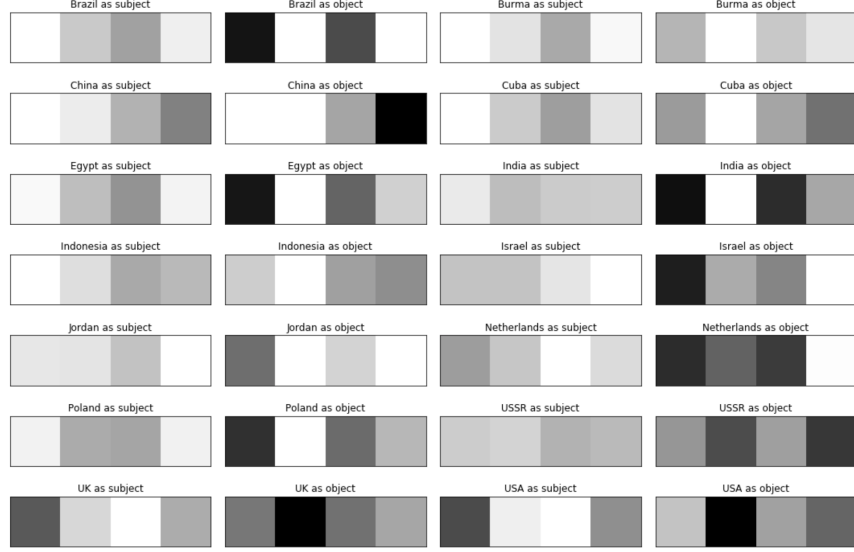


Figure 5.4. Entity latent features that are estimated with Parafac SCPF(EM)

An interesting observation is that the 'Export books' and the 'Economic aid' relations have similar representations. So one can arrive at a conclusion that a nation's economic power is directly proportional to its intellectual knowledge.



Figure 5.5. Relation latent feature examples that are estimated with Parafac SCPF(EM)

The countries that let in immigrants are the ones that has a large value in the second index of their object representation e.g. USA, UK, USSR, Netherlands. These

countries are also the ones that the other countries are dependent on. They are also popular with foreign students.

Most of the time, 'Military Alliance' defines symmetrical relationship between countries. So it has a more complicated feature representation when compared with the other ones.

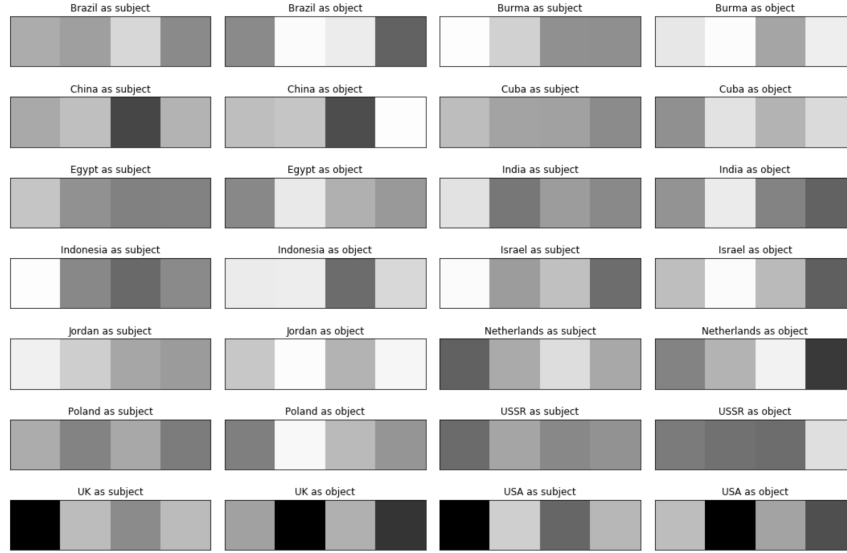


Figure 5.6. Expectations of entity latent features that are inferred with Parafac SCPF(VI)

We also visualize the expectations of latent variables that are inferred with variational inference in Figure 5.6 and Figure 5.7. For the latent feature priors, we used the same mean parameters 0.5, but different shape parameters 1. and 0.1 for items and relations respectively. Keeping the scale parameters small, we reach sparse representations for relation latent features as it can be seen in Figure 5.7. Similar interpretations that are made for the latent features in the maximum likelihood parameter estimation case can be made here as well. This time interpretation is easier because of sparse representations of relation latent features.

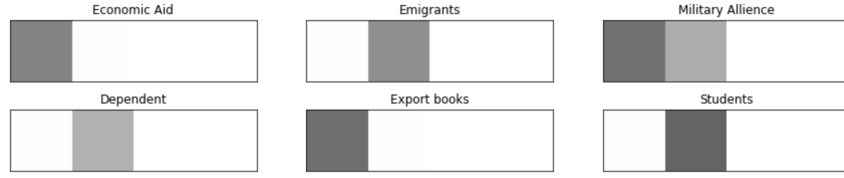


Figure 5.7. Expectations of relation latent feature examples that are inferred with Parafac SCPF(VI)

We compare the features that are estimated with Rescal-like SCPF and LTF. Both models use global vector representations for items and matrix representations for relations. Although it is too limited to approximate the original tensorized data, we chose rank 2 for simplicity.

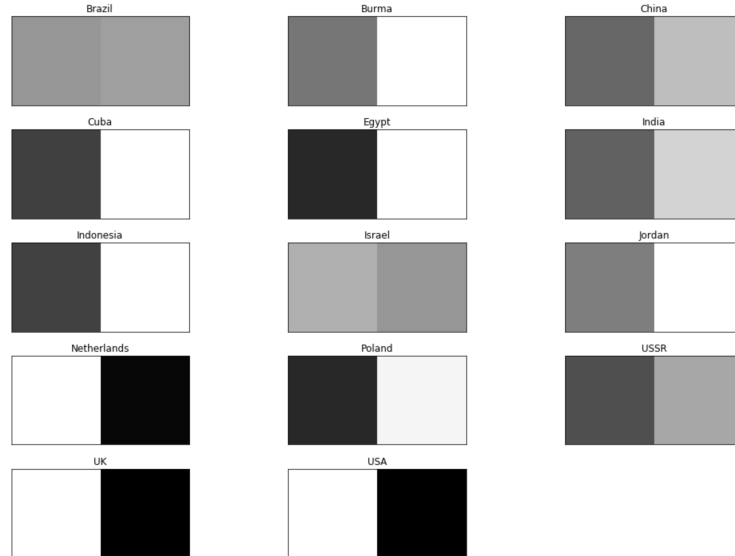


Figure 5.8. Entity latent features that are estimated with Rescal-like SCPF

Associating the economic power of nations with 'Economic Aid' relation, we can categorize USA, UK, and Netherlands as prosperous nations. Considering the representations of 'Economic Aid' and 'Dependent' relations, it can be stated that Poland, Jordan, Indonesia, Cuba, Egypt, and Burma need the economic aid of prosperous coun-

tries. Prosperous countries are also prone to accept immigrants and foreign students.

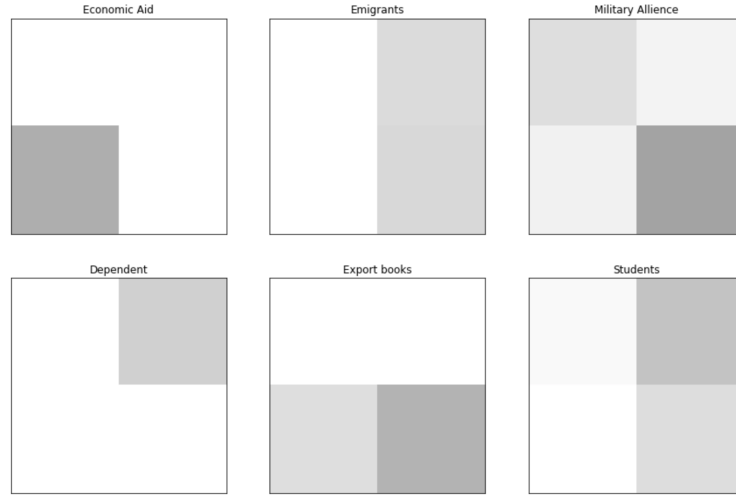


Figure 5.9. Relation latent feature examples that are estimated with Rescal-like SCPF

In Figure 5.10 and Figure 5.11, we visualize the features that are estimated with LTF. This time, features can take the negative values and the small values are represented by light colors.

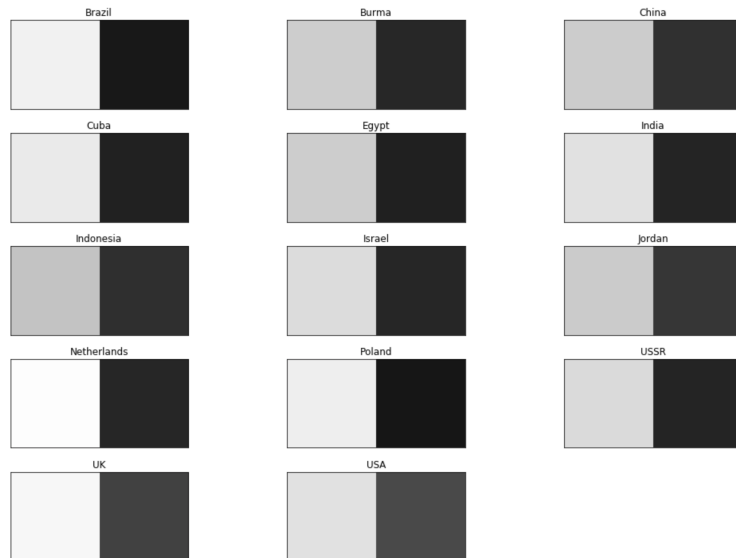


Figure 5.10. Entity latent features that are estimated with LTF

The latent features that are estimated with LTF are more complicated and harder to interpret. The latent feature visualizations reveal the superiority of moment parameterization to canonical parameterization in terms of interpretability.

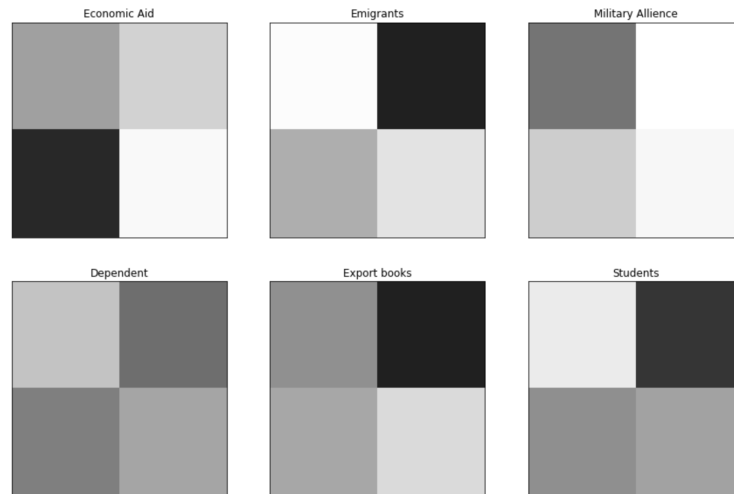


Figure 5.11. Relation latent feature examples that are estimated with LTF

5.2. UMLS Dataset

UMLS dataset [53] is a biomedical ontology dataset with 135 entities and 49 relations. 6752 triples are known facts and this refers to almost 0.75% of all possible triples.

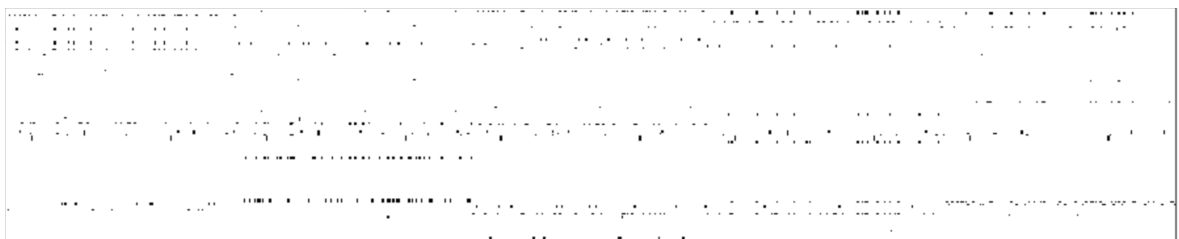


Figure 5.12. Visualization of clipped UMLS dataset.

In Figure 5.12, we visualized UMLS dataset with a $135 \times (135 \times 5)$ dimensional matrix by selecting 5 relations for easiness in visualization and black cells in the matrix refer to known triples.

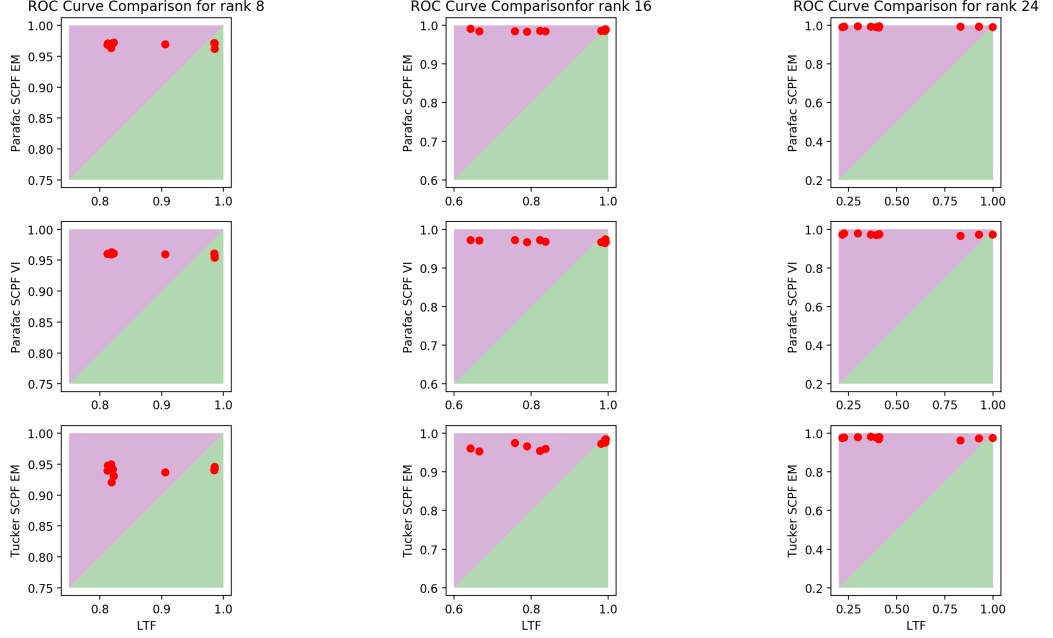


Figure 5.13. Area under the ROC curve comparison for training set.

Implementation details for UMLS dataset is same with the implementation for Nation dataset. Differently, latent ranks are specified as 8,16, and 24. Because Gibbs sampling implementation is computationally time consuming, we did not apply Gibbs sampling, instead we used variational inference for Bayesian inference.

In Figure 5.13 and Figure 5.14, we compared the performances of models with the metric of area under the ROC curve for training and test sets respectively. SCPF models perform better than LTF both in training and test sets. For higher order ranks, number of parameters that need to be estimated increases and the dimensionality of the space that we work on increases as well. In this high dimensional space, number of

local optimum points is higher and it complicates the objective of finding local optimum that is close to global optimum. Considering the inconsistency in performance of LTF, we can state that LTF starts to get stuck in local minima of loss function space which are not close to global optimum points. On the other hand, our models do not encounter such a problem and deliver consistent results by outperforming LTF.

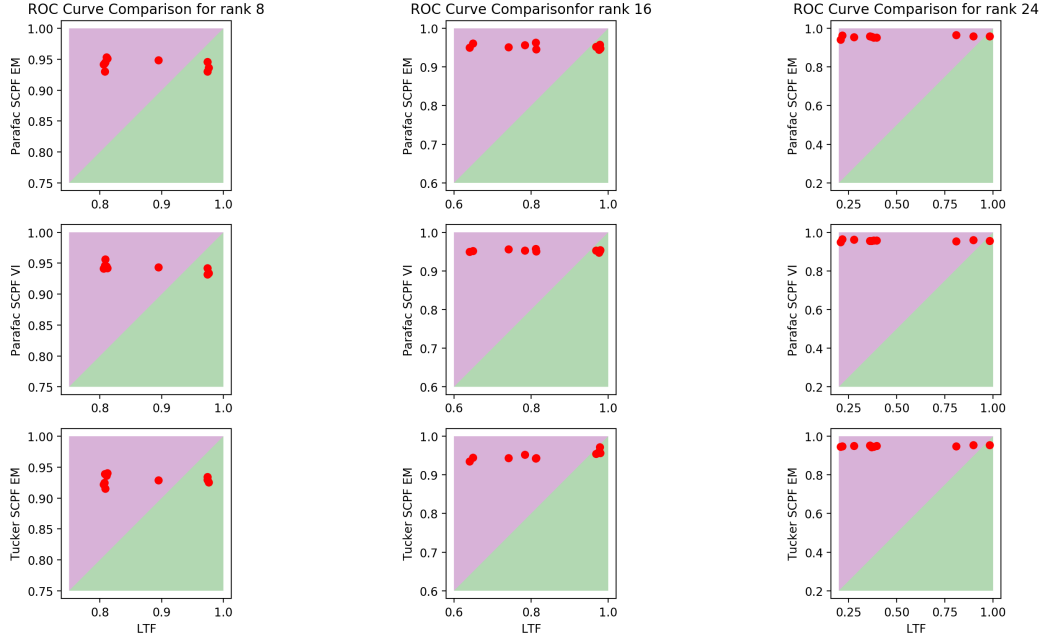


Figure 5.14. Area under the ROC curve comparison for test set.

5.3. Kinship Dataset

This dataset [54] includes kinship relations within the Alwayarra tribe with 104 entities and 26 relations. For easiness in visualization, we selected 5 relations and marked the known triples with the black color on $104 \times (104 \times 5)$ dimensional matrix in Figure 5.15. By keeping the implementation procedures same, we tested SCPF models and LTF on Kinship dataset except Gibbs sampling.

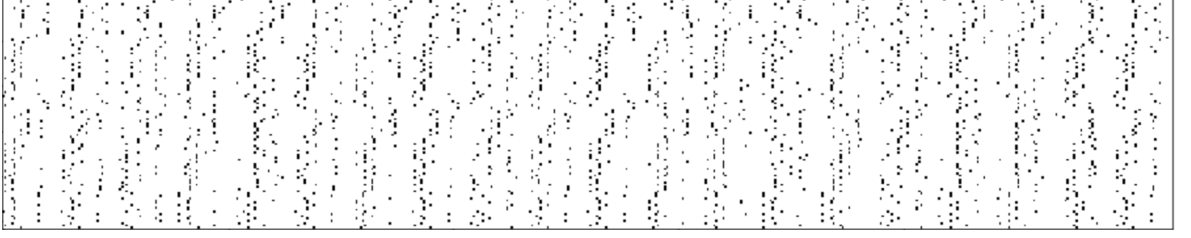


Figure 5.15. Visualization of clipped Kinship dataset.

Unfortunately, Tucker SCPF do not converge, so it is not involved in the results section. As it can be seen in Figure 5.17 and Figure 5.16, SCPF outperforms LTF on both training and test sets for all ranks.

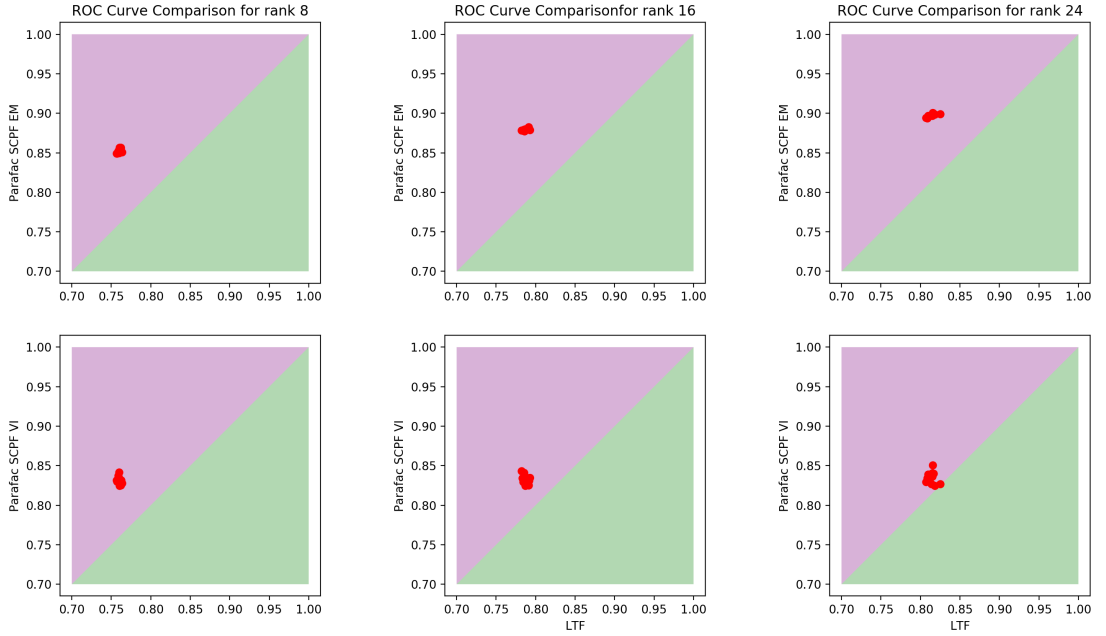


Figure 5.16. Area under the ROC curve comparison for training set.

Although the results are getting better as we increase the rank during the training phase, the same observation cannot be made for the test set. Hence, we observe overfittings as we increase the ranks.

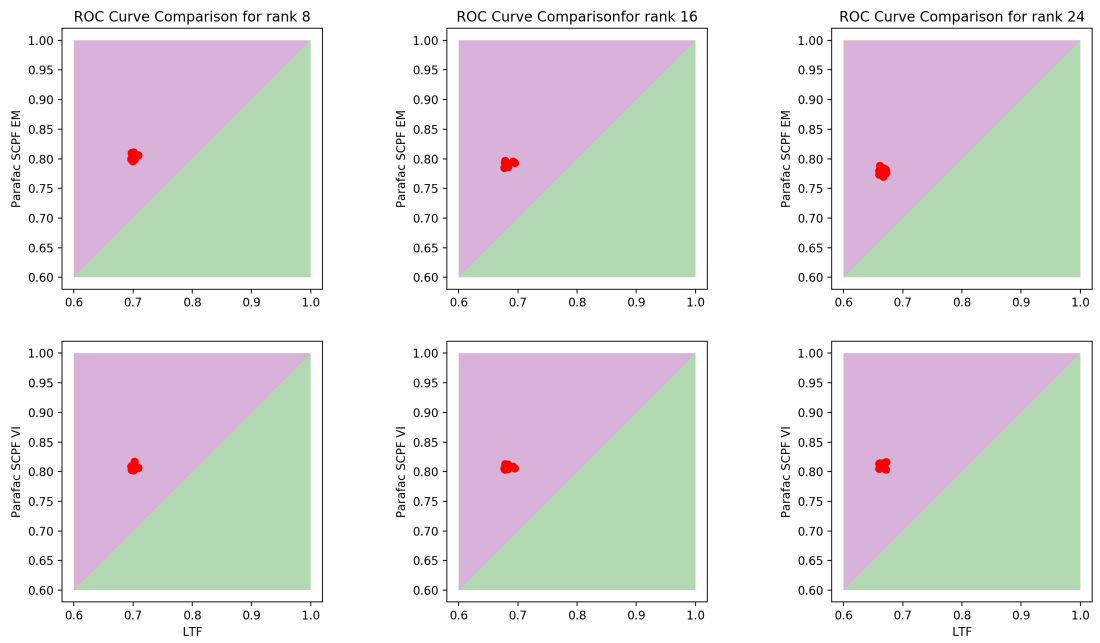


Figure 5.17. Area under the ROC curve comparison for test set.

6. CONCLUSION

In this thesis, we presented Parafac and Rescal-like tensor decompositions for Sum Conditioned Poisson Factorization with the aim of link prediction in knowledge graphs. Sum Conditioned Poisson Factorization is presented as a better alternative to Generalized Linear Model based tensor factorizations. In order to evaluate the predictive power of Sum Conditioned Poisson Factorization, we tested its performance on standard knowledge graph datasets. For comparison, we chose Logistic Tensor Factorization which is a state of the art Generalized Linear Model developed for knowledge graph link prediction problem, specifically.

We presented maximum likelihood parameter estimation for both Parafac and Rescal-like Sum Conditioned Poisson Factorizations with Expectation Maximization algorithm. Additionally, we introduced Bayesian inference for Parafac SCPF with variational inference and Gibbs sampling.

The experiments on standard datasets(Nation,UMLS,Kinship) revealed that Sum Conditioned Poisson Factorization is superior to Logistic Tensor Factorization, in terms of prediction performance. Moreover, we showed the effect of the parameterization choice on interpretability of latent variables, by visualizing the estimated/inferred latent variables. SCPF models estimated/inferred highly interpretable and meaningful latent feature representations for entites and relations.

The estimated latent feature representations can be used to solve the other knowledge graph related problems such as clustering and entity resolution. Clustering refers to grouping the entities or the relations that have the similar features. The entity resolution is detecting the entities which are named differently but refer to the same item. Furthermore, interpretability of latent features enable us to analyze in what way entities or relations are similar. We made such an analysis on the Nation dataset by grouping the countries and the relations according to their features.

REFERENCES

1. Nickel, M. and V. Tresp, “Logistic tensor factorization for multi-relational data”, *arXiv preprint arXiv:1306.2084*, 2013.
2. McCallum, A., “Information extraction: Distilling structured data from unstructured text”, *Queue*, Vol. 3, No. 9, p. 4, 2005.
3. Nickel, M., K. Murphy, V. Tresp and E. Gabrilovich, “A review of relational machine learning for knowledge graphs”, *arXiv preprint arXiv:1503.00759*, 2015.
4. Getoor, L. and B. Taskar, *Introduction to statistical relational learning*, MIT press, 2007.
5. Dong, X., E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun and W. Zhang, “Knowledge vault: A web-scale approach to probabilistic knowledge fusion”, *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 601–610, ACM, 2014.
6. Auer, S., C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak and Z. Ives, “Dbpedia: A nucleus for a web of open data”, *The semantic web*, pp. 722–735, 2007.
7. Yang, Z., J. Tang and W. W. Cohen, “Multi-Modal Bayesian Embeddings for Learning Social Knowledge Graphs.”, *IJCAI*, pp. 2287–2293, 2016.
8. Ferrucci, D., E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager *et al.*, “Building Watson: An overview of the DeepQA project”, *AI magazine*, Vol. 31, No. 3, pp. 59–79, 2010.
9. Deshpande, O., D. S. Lamba, M. Tourn, S. Das, S. Subramaniam, A. Rajaraman, V. Harinarayan and A. Doan, “Building, maintaining, and using knowledge bases: a report from the trenches”, *Proceedings of the 2013 ACM SIGMOD International*

- Conference on Management of Data*, pp. 1209–1220, ACM, 2013.
10. Messina, A., H. Pribadi, J. Stichbury, M. Bucci, S. Klarman and A. Urso, “BioGrakn: A Knowledge Graph-Based Semantic Database for Biomedical Sciences”, *Conference on Complex, Intelligent, and Software Intensive Systems*, pp. 299–309, Springer, 2017.
 11. Bonnici, V., F. Russo, N. Bombieri, A. Pulvirenti and R. Giugno, “Comprehensive reconstruction and visualization of non-coding regulatory networks in human”, *Frontiers in bioengineering and biotechnology*, Vol. 2, 2014.
 12. Pareja-Tobes, P., R. Tobes, M. Manrique, E. Pareja and E. Pareja-Tobes, “Bio4j: a high-performance cloud-enabled graph-based data platform”, *bioRxiv*, p. 016758, 2015.
 13. Fiannaca, A., M. La Rosa, L. La Paglia, A. Messina and A. Urso, “BioGraphDB: a new GraphDB collecting heterogeneous data for bioinformatics analysis”, *Proceedings of BIOTECHNO*, 2016.
 14. Kolda, T. G. and B. W. Bader, “Tensor decompositions and applications”, *SIAM review*, Vol. 51, No. 3, pp. 455–500, 2009.
 15. Rabanser, S., O. Shchur and S. Günnemann, “Introduction to Tensor Decompositions and their Applications in Machine Learning”, *arXiv preprint arXiv:1711.10781*, 2017.
 16. Koller, D. and N. Friedman, *Probabilistic graphical models: principles and techniques*, MIT press, 2009.
 17. Franz, T., A. Schultz, S. Sizov and S. Staab, “Triplerank: Ranking semantic web data by tensor decomposition”, *The Semantic Web-ISWC 2009*, pp. 213–228, 2009.
 18. Erdos, D. and P. Miettinen, “Discovering facts with boolean tensor tucker decom-

- position”, *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pp. 1569–1572, ACM, 2013.
19. Nickel, M., V. Tresp and H.-P. Kriegel, “Factorizing yago: scalable machine learning for linked data”, *Proceedings of the 21st international conference on World Wide Web*, pp. 271–280, ACM, 2012.
 20. Riedel, S., L. Yao, A. McCallum and B. M. Marlin, “Relation Extraction with Matrix Factorization and Universal Schemas.”, *HLT-NAACL*, pp. 74–84, 2013.
 21. Socher, R., D. Chen, C. D. Manning and A. Ng, “Reasoning with neural tensor networks for knowledge base completion”, *Advances in neural information processing systems*, pp. 926–934, 2013.
 22. Lin, Y., Z. Liu, M. Sun, Y. Liu and X. Zhu, “Learning Entity and Relation Embeddings for Knowledge Graph Completion.”, *AAAI*, pp. 2181–2187, 2015.
 23. Bordes, A., J. Weston, R. Collobert and Y. Bengio, “Learning structured embeddings of knowledge bases”, *Conference on artificial intelligence*, EPFL-CONF-192344, 2011.
 24. Wang, Z., J. Zhang, J. Feng and Z. Chen, “Knowledge Graph Embedding by Translating on Hyperplanes.”, *AAAI*, pp. 1112–1119, 2014.
 25. Ceritli, T., *Modeling Bounded Data with Sum Conditioned Poisson Factorization*, Master’s Thesis, Bogazici University, 2017, <https://tahaceritli.github.io/files/ms-thesis.pdf>, accessed at May 2018.
 26. Cemgil, A. T., “Bayesian inference for nonnegative matrix factorisation models”, *Computational Intelligence and Neuroscience*, Vol. 2009, 2009.
 27. McCullagh, P., “Generalized linear models”, *European Journal of Operational Re-*

- search*, Vol. 16, No. 3, pp. 285–292, 1984.
28. Nickel, M., V. Tresp and H.-P. Kriegel, “A three-way model for collective learning on multi-relational data”, *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 809–816, 2011.
 29. Lee, D. D. and H. S. Seung, “Learning the parts of objects by non-negative matrix factorization”, *Nature*, Vol. 401, No. 6755, pp. 788–791, 1999.
 30. Arora, S., R. Ge and A. Moitra, “Learning topic models—going beyond SVD”, *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pp. 1–10, IEEE, 2012.
 31. Shen, B. and L. Si, “Non-Negative Matrix Factorization Clustering on Multiple Manifolds.”, *AAAI*, pp. 575–580, 2010.
 32. Lazar, C. and A. Doncescu, “Non negative matrix factorization clustering capabilities; application on multivariate image segmentation”, *Complex, Intelligent and Software Intensive Systems, 2009. CISIS'09. International Conference on*, pp. 924–929, IEEE, 2009.
 33. Sandler, R. and M. Lindenbaum, “Nonnegative matrix factorization with earth mover’s distance metric for image analysis”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 33, No. 8, pp. 1590–1602, 2011.
 34. Ozerov, A. and C. Févotte, “Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation”, *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 18, No. 3, pp. 550–563, 2010.
 35. de Fréin, R., K. Drakakis, S. Rickard and A. Cichocki, “Analysis of financial data using non-negative matrix factorization”, *International Mathematical Forum*, Vol. 3, pp. 1853–1870, Journals of Hikari Ltd, 2008.

36. Lee, D. D. and H. S. Seung, “Algorithms for non-negative matrix factorization”, *Advances in neural information processing systems*, pp. 556–562, 2001.
37. Dempster, A. P., N. M. Laird and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm”, *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.
38. Bishop, C. M., *Pattern recognition and machine learning*, Springer, 2006.
39. Jaakkola, T., “10 tutorial on variational approximation methods”, *Advanced mean field methods: theory and practice*, p. 129, 2001.
40. Blei, D. M., A. Kucukelbir and J. D. McAuliffe, “Variational inference: A review for statisticians”, *Journal of the American Statistical Association*, Vol. 112, No. 518, pp. 859–877, 2017.
41. MacKay, D. J., “Introduction to monte carlo methods”, *NATO Asi Series D Behavioural and Social Sciences*, Vol. 89, pp. 175–204, 1998.
42. Cemgil, A. T., “A tutorial introduction to Monte Carlo methods, Markov Chain Monte Carlo and particle filtering”, *Academic Press’ Library in Signal Processing*, 2012.
43. Geman, S. and D. Geman, “Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images”, *IEEE Transactions on pattern analysis and machine intelligence*, Vol. 6, No. 6, pp. 721–741, 1984.
44. Casella, G. and E. I. George, “Explaining the Gibbs sampler”, *The American Statistician*, Vol. 46, No. 3, pp. 167–174, 1992.
45. Roberts, G. O. and J. S. Rosenthal, “Markov-chain monte carlo: Some practical implications of theoretical results”, *Canadian Journal of Statistics*, Vol. 26, No. 1, pp. 5–20, 1998.

46. Andrieu, C., N. De Freitas, A. Doucet and M. I. Jordan, “An introduction to MCMC for machine learning”, *Machine learning*, Vol. 50, No. 1-2, pp. 5–43, 2003.
47. Steel, G. *et al.*, *Relation between Poisson and multinomial distributions*, Tech. rep., Cornell University, 1953.
48. Kingman, J. F. C., *Poisson processes*, Wiley Online Library, 1993.
49. Bradley, A. P., “The use of the area under the ROC curve in the evaluation of machine learning algorithms”, *Pattern recognition*, Vol. 30, No. 7, pp. 1145–1159, 1997.
50. Kim, D., *Nation Dataset*, 2016, <https://github.com/dongwookim-ml/kg-data/tree/master/nation>, accessed at May 2018.
51. Liu, D. C. and J. Nocedal, “On the limited memory BFGS method for large scale optimization”, *Mathematical programming*, Vol. 45, No. 1-3, pp. 503–528, 1989.
52. Abadi, M., A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, ..., X. Zheng, *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, 2015, <https://www.tensorflow.org>, accessed at May 2018.
53. Kim, D., *UMLS Dataset*, 2016, <https://github.com/dongwookim-ml/kg-data/tree/master/umls>, accessed at May 2018.
54. Nickel, M., *Kinship Dataset*, 2013, <https://github.com/mnick/rescal.py/tree/master/data>, accessed at May 2018.

APPENDIX A: SUM CONDITIONED POISSON VARIABLES

The idea of SCPF is based on conditioning the summation of Poisson random variables to a fixed value. In this thesis, we specifically interested in Knowledge Graph problem, so we fixed the summation to 1. To handle the property, we introduce two Poisson random variables x_1 and x_2 such that

$$\begin{aligned} x_1 &\sim Po(x_1; \lambda_1) = \exp(-\lambda_1) \frac{\lambda_1^{x_1}}{x_1!} \\ x_2 &\sim Po(x_2; \lambda_2) = \exp(-\lambda_2) \frac{\lambda_2^{x_2}}{x_2!} \\ y &= x_1 + x_2 = 1 \end{aligned}$$

We evaluate the probability of y

$$\begin{aligned} p(y = 1) &= p(x_1 = 1)p(x_2 = 0) + p(x_1 = 0)p(x_2 = 1) \\ &= \lambda_1 \exp(-\lambda_1) \exp(-\lambda_2) + \lambda_2 \exp(-\lambda_1) \exp(-\lambda_2) \\ &= (\lambda_1 + \lambda_2) \exp(-\lambda_1 - \lambda_2) \end{aligned}$$

Conditioned on $y = 1$, we find the probability of $x_1 = 1$

$$p(x_1 = 1|y = 1) = \frac{p(x_1 = 1, y = 1)}{p(y = 1)} = \frac{\lambda_1}{\lambda_1 + \lambda_2}$$

Similarly we find the conditional probability $p(x_2 = 1|y = 1)$

$$p(x_2 = 1|y = 1) = \frac{p(x_2 = 1, y = 1)}{p(y = 1)} = \frac{\lambda_2}{\lambda_1 + \lambda_2}$$

By looking at these probabilities, we see that sum 1 conditioned Poisson variables are actually Bernoulli variables as the followings

$$p(x_1 = 1|y = 1) = Ber\left(x_1; \frac{\lambda_1}{\lambda_1 + \lambda_2}\right)$$
$$p(x_2 = 1|y = 1) = Ber\left(x_2; \frac{\lambda_2}{\lambda_1 + \lambda_2}\right)$$

APPENDIX B: EM FOR PARAFAC DECOMPOSITION

EM algorithm for Parafac Decomposition is explained in Chapter 3, but some details have been left to Appendix.

$$\begin{aligned}
\log p(\underline{S}|\underline{N}, \underline{X}, W, H, G) &= \sum_{l,i,j,k} \left[\log \delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) + \log \delta \left(x_{l,ijk} - \sum_r s_{l,ijk r} \right) \right. \\
&\quad \left. + \sum_r \log \mathcal{PO}(s_{l,ijk r}; w_{l,ir} h_{l,jr} g_{l,kr}) \right] \\
&\quad - \sum_{l,i,j,k} m_{ijk} \log \mathcal{PO}(x_{l,ijk}; \sum_r w_{l,ir} h_{l,jr} g_{l,kr}) \\
&\quad - \sum_{i,j,k} (1 - m_{ijk}) \log \mathcal{PO}(n_{ijk}; \sum_l \sum_r w_{l,ir} h_{l,jr} g_{l,kr}) \\
&= \sum_{l,i,j,k} \left[\log \delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) + \log \delta \left(x_{l,ijk} - \sum_r s_{l,ijk r} \right) \right. \\
&\quad \left. + \sum_r m_{ijk} \log \mathcal{PO}(s_{l,ijk r}; w_{l,ir} h_{l,jr} g_{l,kr}) \right. \\
&\quad \left. - m_{ijk} \log \mathcal{PO} \left(x_{l,ijk}; \sum_r w_{l,ir} h_{l,jr} g_{l,kr} \right) \right] \\
&\quad + \sum_{i,j,k} \left(\sum_{l,r} [(1 - m_{ijk}) \log \mathcal{PO}(s_{l,ijk r}; w_{l,ir} h_{l,jr} g_{l,kr})] \right. \\
&\quad \left. - \log \mathcal{PO} \left(n_{ijk}; (1 - m_{ijk}) \sum_{l,r} w_{l,ir} h_{l,jr} g_{l,kr} \right) \right) \\
&= \sum_{l,i,j,k} m_{ijk} \left[\sum_r \left(s_{l,ijk r} \log \frac{w_{l,ir} h_{l,jr} g_{l,kr}}{\sum_r w_{l,ir} h_{l,jr} g_{l,kr}} - \log \Gamma(s_{l,ijk r} + 1) \right) \right. \\
&\quad \left. + \log \Gamma(x_{l,ijk} + 1) \right] + \log \delta \left(x_{l,ijk} - \sum_r s_{l,ijk r} \right) \\
&\quad + \log \delta \left(n_{ijk} - \sum_l x_{l,ijk} \right) \\
&\quad + \sum_{i,j,k} \left[(1 - m_{ijk}) \sum_{l,r} s_{l,ijk r} \log \frac{w_{l,ir} h_{l,jr} g_{l,kr}}{\sum_r w_{l,ir} h_{l,jr} g_{l,kr}} + \log \Gamma(n_{ijk} + 1) \right]
\end{aligned}$$

$$\begin{aligned} \log p(\underline{S}|\underline{N}, \underline{X}, W, H, G) &= \sum_{l,i,j,k} m_{ijk} \log \mathcal{M}(s_{l,ijk}; x_{l,ijk}, p_{l,ijk}) \\ &\quad + \sum_{i,j,k} (1 - m_{ijk}) \log \mathcal{M}(s_{:,ijk}; n_{ijk}, q_{:,ijk}) \end{aligned}$$

where $p_{l,ijk} = \frac{w_{l,ir} h_{l,jr} g_{l,kr}}{\sum_r w_{l,ir} h_{l,jr} g_{l,kr}}$ and $q_{l,ijk} = \frac{w_{l,ir} h_{l,jr} g_{l,kr}}{\sum_{l,r} w_{l,ir} h_{l,jr} g_{l,kr}}$.

APPENDIX C: VARIATIONAL BAYES FOR PARAFAC DECOMPOSITION

Joint distribution needs to be evaluated for Variational Bayes

$$\begin{aligned}
\log p(\underline{N}, \underline{X}, \underline{S}, W, H, G) &= \log p(\underline{N}|\underline{X}) + \log p(\underline{X}|\underline{S}) + \log p(\underline{S}|W, H, G) \\
&+ \log p(W) + \log p(H) + \log p(G) \\
&= \sum_{i,j,k} \left[m_{ijk} \left(\log \delta \left(n_{ijk} - \sum_l x_{lijk} \right) \right. \right. \\
&+ \sum_l \log \delta \left(x_{lijk} - \sum_r s_{lijkr} \right) \left. \right) \\
&+ (1 - m_{ijk}) \log \delta \left(n_{ijk} - \sum_{l,r} s_{lijkr} \right) \left. \right] \\
&+ \sum_{l,i,j,k,r} -w_{lir} h_{ljr} g_{lkr} + s_{lijkr} \log(w_{lir} h_{ljr} g_{lkr}) - \log \Gamma(s_{lijkr} + 1) \\
&+ \sum_{l,i,r} (a_l^w - 1) \log w_{lir} - \frac{a_l^w}{b_l^w} w_{lir} - \log \Gamma(a_l^w) - a_l^w \log(b_l^w / a_l^w) \\
&+ \sum_{l,j,r} (a_l^h - 1) \log h_{ljr} - \frac{a_l^h}{b_l^h} h_{ljr} - \log \Gamma(a_l^h) - a_l^h \log(b_l^h / a_l^h) \\
&+ \sum_{l,k,r} (a_l^g - 1) \log g_{lkr} - \frac{a_l^g}{b_l^g} g_{lkr} - \log \Gamma(a_l^g) - a_l^g \log(b_l^g / a_l^g)
\end{aligned}$$

Fixed point iterations for w is supplied in Chapter 4. By symmetry, derivations for h and g is supplied.

$$\begin{aligned}
\alpha_{ljr}^h &= a_l^h + \sum_{i,k} \langle s_{lijkr} \rangle & \beta_{ljr}^h &= \left(\frac{a_l^h}{b_l^h} + \sum_{i,k} \langle w_{lir} \rangle \langle g_{lkr} \rangle \right)^{-1} \\
\exp(\langle \log h_{ljr} \rangle) &= \exp(\Psi(\alpha_{ljr}^h)) \beta_{ljr}^h & \langle h_{ljr} \rangle &= \alpha_{ljr}^h \beta_{ljr}^h
\end{aligned}$$

$$\begin{aligned}\alpha_{lkr}^g &= a_l^g + \sum_{i,j} \langle s_{lijkr} \rangle & \beta_{lkr}^g &= \left(\frac{a_l^g}{b_l^g} + \sum_{i,j} \langle w_{lir} \rangle \langle h_{ljr} \rangle \right)^{-1} \\ \exp(\langle \log g_{lkr} \rangle) &= \exp(\Psi(\alpha_{lkr}^g)) \beta_{lkr}^g & \langle g_{lkr} \rangle &= \alpha_{lkr}^g \beta_{lkr}^g\end{aligned}$$

APPENDIX D: GIBBS SAMPLING FOR PARAFAC DECOMPOSITION

Full conditional distribution for H is given

$$\begin{aligned}
\log p(h_{l_{jr}} | W, H_{-h_{l_{jr}}}, G, \underline{S}, \underline{X}, \underline{N}) &\propto \log p(h_{l_{jr}}) + \log p(s_{l:j:r} | w_{l:r}, h_{l_{jr}}, g_{l:r}) \\
&= \log p(h_{l_{jr}}) + \sum_{i,k} \log p(s_{lijkr} | w_{lir}, h_{l_{jr}}, g_{lkr}) \\
&= \log \mathcal{G}(h_{l_{jr}}; a^h, b^h / a^h) \\
&\quad + \sum_{i,k} \log \mathcal{PO}(s_{lijkr} | w_{lir} h_{l_{jr}} g_{lkr}) \\
&= (a^h - 1) \log h_{l_{jr}} - h_{l_{jr}} (a^h / b^h) - \log \Gamma(a^h) \\
&\quad + a^h \log(b^h / a^h) + \sum_{i,k} (-h_{l_{jr}} w_{lir} g_{lkr} \\
&\quad + s_{lijkr} \log(w_{lir} h_{l_{jr}} g_{lkr}) - s_{lijkr}!) \\
&\propto (a^h + \sum_{i,k} s_{lijkr} - 1) \log h_{l_{jr}} \\
&\quad - h_{l_{jr}} (a^h / b^h + \sum_{i,k} w_{lir} g_{lkr})
\end{aligned}$$

$$\log p(h_{l_{jr}} | W, H_{-h_{l_{jr}}}, G, \underline{S}, \underline{X}, \underline{N}) \propto \log \mathcal{G}\left(h_{l_{jr}}; a^h + \sum_{i,k} s_{lijkr}, (a^h / b^h + \sum_{i,k} w_{lir} g_{lkr})^{-1}\right)$$

Similarly,

$$\begin{aligned}
\log p(g_{lkr}|W, H, G_{-g_{lkr}}, \underline{S}, \underline{X}, \underline{N}) &\propto \log p(g_{lkr}) + \log p(s_{l:j:r}|w_{l:r}, h_{l:r}, g_{lkr}) \\
&= \log p(g_{lkr}) + \sum_{i,j} \log p(s_{lijkr}|w_{lir}, h_{ljr}, g_{lkr}) \\
&= \log \mathcal{G}(g_{lkr}; a^g, b^g/a^g) \\
&\quad + \sum_{i,j} \log \mathcal{PO}(s_{lijkr}|w_{lir}h_{ljr}g_{lkr}) \\
&= (a^g - 1) \log g_{lkr} - g_{lkr}(a^g/b^g) - \log \Gamma(a^g) \\
&\quad + a^g \log(b^g/a^g) + \sum_{i,j} (-g_{lkr}w_{lir}h_{ljr} \\
&\quad + s_{lijkr} \log(w_{lir}h_{ljr}g_{lkr}) - s_{lijkr}!) \\
&\propto (a^g + \sum_{i,j} s_{lijkr} - 1) \log g_{lkr} \\
&\quad - g_{lkr}(a^g/b^g + \sum_{i,j} w_{lir}h_{ljr})
\end{aligned}$$

$$\log p(g_{lkr}|W, H, G_{-g_{lkr}}, \underline{S}, \underline{X}, \underline{N}) \propto \log \mathcal{G}\left(g_{lkr}; a^g + \sum_{i,j} s_{lijkr}, (a^g/b^g + \sum_{i,j} w_{lir}h_{ljr})^{-1}\right)$$