Doctoral theses at NTNU, 2021:286

Doctoral thesis

Eliezer de Souza da Silva

Factorization models with relational and contextual information

Probabilistic factorization, point processes and neural sequential models

NTNU
Norwegian University of Science and Technology
Thesis for the Degree of
Philosophiae Doctor
Faculty of Information Technology and Electrical
Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Eliezer de Souza da Silva

Factorization models with relational and contextual information

Thesis for the Degree of Philosophiae Doctor

Trondheim, September 2021

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the Degree of Philosophiae Doctor

Faculty of Information Technology and Electrical Engineering
Department of Computer Science

ISBN 978-82-326-5463-5 (printed ver.)
ISBN 978-82-326-5273-0 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

Doctoral theses at NTNU, 2021:286

Printed by NTNU Grafisk senter

Preface

This doctoral thesis was developed mainly at the Norwegian University of Science and Technology (NTNU) under the supervision of prof. Helge Langseth and prof. Heri Ramampiaro. It includes as well work developed during research stays at Royal Melbourne Institute of Technology (RMIT), visiting prof. Mark Sanderson and prof. Yongli Ren, and at University of Helsinki, visiting prof. Arto Klami. Also, as part of the Norwegian Open AI Lab, I engaged in supervision and collaboration with master students, which resulted in some parts included in this thesis.

Eliezer de Souza da Silva
Trondheim
May 2021

Abstract

The increasing availability of interconnected multi-modal information sources motivates the development of novel probabilistic models for recommender system that can leverage context present in relational data. Thus, we seek to integrate contextual information that can be relevant for determining the users' information needs. In this thesis we focus on a set of techniques for modeling contextual information to factorization models, in particular models that uses implicit feedback such as event counts. Furthermore we propose analytical tools for those models, improving our capabilities with regards to find suitable hyparparameters. In order to model counts (for example, number of clicks in a page) as implicit user feedback, we chose to utilize the Poisson factorization as a building block. Then, we develop two Poisson factorization models that include social networks, item textual content and periodic time events as contextual information, incorporated into a joint matrix and tensor factorization model (in Chapters 3 and 4). Additionally, we develop a joint hierarchical recurrent neural networks and a temporal point process model for the problem of multi-session recommendations, where we observe sequences of items grouped into sequences of sessions, and create a model capable of providing itens recommendation and next-session time prediction (Chapter 5). Finally, we utilize and develop an approach based on the prior predictive distribution that allows us to set hyperparameters for Poisson factorization models without the need to fit the model to the data, obtaining both closed-form equations and an optimization algorithm for this task (Chapter 6). One relevant result here is a closed-form equation for the dimensionality of the latent space in Poisson factorization models. In general, we position this work as a contribution to probabilistic modeling in the context of recommender system utilizing multi-relational and count data as a signal for contextual information, with contributions ranging from model design, analysis and hyperparameter selection.

Acknowledgments

I am grateful to everyone that directly and indirectly contributed with my development as a person and researcher, and the development of the research included in this thesis. My sincere gratitude is extended to everyone I have interacted before and during the period of time that I developed this thesis, I have been enriched by so many different perspectives and experiences that it would be impossible to name individually everyone.

I am grateful to my supervisors at NTNU, they have given me the necessary support, guidance, trust and freedom to grow and develop. Helge, thank you for the time, mentorship and teaching, you have helped develop a deeper understanding of probabilistic models and machine learning, and sharpen my thinking around those subjects. Heri, thanks for being positive and fostering an environment full of support and energy for moving forward with multiple projects and ideas.

I am grateful for the researchers that hosted me in their groups and gave me the opportunity to collaborate with them. Visiting other groups has been a source of inspiration for novel work and challenges leading to growth. Thank you Arto, Harri, Mathias, Yongli and Mark for your generosity, understanding and openness.

I am grateful to the academic friends, collaborators and co-authors that I have met during my PhD. It has been invaluable to share ideas, aims, struggles and co-participate with you in the research journey. Tarik, Tomasz, Jan, Antonio, Joana, Marcelo, Stella, Malene, Shweta, Dirk, Donn, Anisia, Florian, Sardana, Christos, Vladi and many other friends, thank you for creating an environment where socializing, thinking and researching is intermixed in inspiring ways. Thank you Massimiliano, Erlend, Kenth, Bjørnar for the collaboration, sharing and development of research ideas. Thank you to the communities and friends of ProbAI, LxMLS, Deep|Bayes, ECML, WSDM, ACL and many others academic events and summer schools, which all consisted in incredible social and academic experiences contributing to cementing in me the idea that research is an exciting collective endeavour. To all my colleagues at NTNU, University of Helsinki, RMIT, Curious AI and Searis / Clarify thank you for providing an exciting and supporting working environment.

I very grateful to all my friends. Spending time with all of you have made me grow as a person, learning from you different perspectives, and giving me the space as well to contribute with what I could bring. To all my friends in Trondheim, Oslo, Helsinki, Amsterdam, Melbourne, Lisbon, Campinas, Rio, Vitoria, and many other cities that I lived or stayed for some period, thank you very much, you are part of who I am. To all my internet friends that I never met, I hope we can meet one day and thank you!

To my family and extended family, I can't thank you enough for all the love, care and nurturing received. Marta, José, Elienai, Nilcea, Juliana, and many others, thank you very much. To my son Samir Francisco, I can't thank you enough for coming into my life. One day you will read this thesis that was written during your initial years on this Earth and will know that you have been my compass and stronger motivation to keep going further, this thesis is dedicated to you!

I am grateful for receiving this life and strength to overcome the challenges and obstacles that emerge in this journey.

Contents

List of Figures

List of Tables

Acronyms

BN Bayesian Network
CDF Cumulative Density Function
CAVI Coordinate Ascent Variational Inference
DAG Direct Acyclic Graph
ELBO Evidence Lower BOund
HMM Hidden Markov Model
HP Hawkes Process
KL Kullbeck-Leibler divergence
LVM Latent Variable Model
MAP Maximum-A-Posteriory
MC Monte Carlo
MLE Maximum Likelihood Estimate
NN Neural Networks
PDF Probability Density Function
PF Poisson Factorization
PMF Probabilistic Matrix Factorization
TPP Temporal Point Process
PPD Prior Predictive Distribution
RNN Recurrent Neural Networks
VI Variational Inference

Introduction 1

"As we have said, nature's statistical tendency to disorder, the tendency for entropy to increase in isolated systems, is expressed by the second law of thermodynamics. We, as human beings, are not isolated systems. We take in food, which generates energy, from the outside, and are, as a result, parts of that larger world which contains those sources of our vitality. But even more important is the fact that we take in information through our sense organs, and we act on information received."

— Norbert Wiener, *The Human Use of Human Beings*

In our current technological landscape, we are in a state of abundance of interconnected users, systems and networks, generating and making available large amounts of data. This plethora of interactions can be observed and analyzed to infer the different agents' properties and behaviors, advancing our collective capabilities of data-backed decision making and planning. A big emerging challenge is the advancement of technologies and techniques that would unlock and utilize the great amount of "hidden" value in this ever-growing collection of data. The scale and complexity of data collections pushes the limit of what human intuition and analysis is capable of, thus creating the challenge of distilling the information content of large data collections in a way that is useful for the users of information systems.

This challenge can be decomposed in several problems for users, decision makers and designers of information system: for the end-users, it is the difficulty of finding items in the data collections that better suit their information needs, for decision-makers, the difficulty of obtaining insight about both aggregate and individualized user behavior and their effect on the systems, and for designer, it is the challenge of modeling the interfaces with the number of potential items to be interacted with is too large for any given user. In that context, an emerging solution has been *personalization* techniques, that are used in the development of adaptive systems with interfaces geared towards their users' *implicit* preferences, with the aim of predicting information needs, improving user experience, engagement and satisfaction. This collection of techniques shape different aspects of a system, presenting content and shaping interactions according to users personal needs, and creating an incentive structure of increasing engagement for both designers and users of the system. From the designer point-of-view, improvements in user engagement, loops back creating finer-grained data about users-needs and interests; from the user point-of-view, the system becomes more intuitive to interact with, leading to more motivation and less cognitive burden, as the load of explicitly informed needs is diminished. This incentive structure has the potential of improving productivity and overall capabilities for understanding large collections of interaction data, in the underlying dynamics of the agents generating the data. Nevertheless, it is not a risk-free endeavour, both in public discourse and academic investigation there is an increasing awareness of the risks involved, to name some: the so-called *echo chambers*, abuse of privacy and increased surveillance capabilities (by public and private institutions)[1]. With any new technology, risks and opportunities are to be evaluated with clarity, leading to a quest for deeper understanding of the techniques and how they can be changed and shaped. Having that in mind, our aim in this thesis is to progress in this understanding for some models and techniques in this ecosystem, with the assumption that with an expanded knowledge we ought to be able to make better choices regarding design, adoption and usage. We adopt the Bayesian modeling framework, given its natural fit in modeling uncertainty both in the observed data and the model itself, creating a reasoning framework in probabilistic terms for both the conclusions reached by the models, as well as the underlying parameters assumed by the models.

A central task is the development and analysis of algorithms that learn, infer and adapt based on the underlying preferences of users and agents interacting with the information systems – the so called *recommender systems* algorithms. Examples of those techniques are deployed on commercial platforms such as YouTube, Netflix, Google Search, Spotify, and many others, which use features such as search history, location, social network, demographics, item content and other contextual variables to train their models. Furthermore, this family of techniques has been found useful in diverse contexts such as healthcare (for example by deploying personalized treatments according to the patient history of diseases, genetics and other medicaments in use), drug discovery (modeling patterns of interactions between chemicals compounds and pathogens to predict best possible matches), public policy and governance (in various contexts where modeling interaction between individual or groups of citizens, decision-makers and services are beneficial for improving the matching of the policies and services provided) and even in automating ma-

[1]An interesting reflective piece on those issues by the ML researcher Jaan Altosaar can be found in `https://jaan.io/my-friend-radicalized-this-made-me-rethink-how-i-build-AI/`

chine learning pipelines (for example by modeling performance metrics for different combinations of algorithms and datasets).

A shared model design element across this work is the assumption that different sets of interactions can be represented using *factorization models*. In these models, complex interactions between entities of interest are represented using latent variables for each of those agents. The widespread adopted technique of Collaborative Filtering based in the matrix factorization model, where users and items interactions (for example: counts, ratings, number of page visits) are aggregated in a matrix with rows and columns indexed by users and items, and the matrix is factorized by assuming that each row and column of the matrix is represented by a latent vector. Furthermore, one could define an optimization problem by minimizing a loss-function related to the quality of the approximation of the original data-matrix by the latent-factors matrix, sometimes incorporating regularization terms in the latent factors in order to introduce *inductive bias* (for example for sparsity, non-negative, etc). This technique has been extended into a probabilistic model, with the main advantage being that models assumptions could be expressed in a unified probabilistic modeling language – both the distributional and structural aspects of the latent factors and observations can be reasoned in terms of choices for the prior distributions, independence structure and likelihood. One overarching advantage resulting from this approach is the ability to model generic relationships of multiple entities in the latent space, by sharing variables we can link distinct multimodal data and make inferences based on them by doing *data fusion* in the latent space. In recommender systems this creates an opportunity for *contextualized* recommendation, taking into account not only the user–item interactions, but other sources of data that are relevant for determining the user information need in different contexts, for example, location, social network, item content or time, using distinct relationships as indicators of context.

We are concerned with a family of probabilistic factorization models, called Poisson Factorization (PF), that are usually employed for count-data, can naturally incorporate (from the computational point-of-view) non-negativity and sparsity of the latent-factors level – induced by the choice of priors (Gamma distribution) – and the variational inference depends only on the non-zero entries of the data matrix (a property of the choice of Poisson-Gamma structure), allowing for scalable inference. A repeated motif and insight employed in these models is that we can use shared latent variables to couple different parts of a model in a modular fashion. Armed with this insight, first we introduce a model with coupled factorization of user–item, item–content and user social network, in order to obtain a version of PF-based recommendation capable of utilizing extra contextual information (in this case item contents, expressed in textual form, and user social network). Furthermore, we explore extensions of the Poisson model to include temporal dynamics. The first considers periodic-time chains of latent-factors with additive effects on the rate of counts, while keeping the same overall structure of the Poisson factorization model. The second utilizes temporal point processes with a parameterized function for the rate of events over time, and can be leveraged jointly with a recurrent neural networks to induce a model capable of predicting both interaction events and the timing of those events, based on past events data, with applications to the multi-session recommendation problem. Finally, we abstract away from many specific modeling choices of latent-variables to focus on a fundamental aspect of hierarchical modeling, which is the sensible specification of prior information. Inspired by the methodology of prior predictive checks from traditional Bayesian analysis, we explore how to use the prior predictive distribution of the model to obtain functional connections between hyperparameters of the model and prior expectations (or other summaries) about the data generated by the model. This approach applied to a generalized form of Poisson and compound Poisson factorization leads to closed-form equations, and for more general (differentiable) models it can be cast into an optimization problem that is solved via stochastic gradient-based methods. The overarching importance of this step is adding a more informative and intuitive view of the effect of the prior on the generative distribution assumed by the model, which is typically overshadowed by the complexity of the model.

Organization of the thesis. Chapter 2 introduces the concepts of Bayesian modeling for Machine Learning, including specific distributions and model family that we are focusing on this thesis. The basic notation used throughtout the thesis will be established, as well with the associated concepts. The generic formulation of the recommendation problem will be presented, with later specification for different settings will be presented in other chapters. In Chapter 3 we present the content-based social Poisson matrix factorization model, as well as the recommendation problem associated with the model, which takes into account social network of the user and the content information (in textual form) of the items as contextual information. We present the generative model, the inference equations using mean-field variational inference with coordinate ascent, empirical evaluation of the model, and discussions about the model design and results. Chapter 4 is dedicated to the periodic yime-aware Poisson factorization model, targeted to a recommendation problem in the context of periodic-time information, leading to a tensor factorization model of the user–item–time tensor, as well as a design of latent variables with periodic time dependencies. Two variations of the model is presented, one taking into account only the time context and another with an auxiliary item-context matrix. The inference is also based on mean-field variational inference with coordinate ascent. Motivated by the models presented in Chapter 3 and Chapter 4, and a characterization of generic relations between entities present in a dataset, in Section 4.5 we present generic modeling principles that can be used to guide the design choices for a family of contextualized matrix and tensor factorization models. In Chapter 5 we introduce an integrated temporal point process (TPP) model with a hierarchical Recurrent Networks (RNN) for a multi-session recommendation problem, where each session consist on a sequence of items, modelled by the RNN, and a latent layer couples different sections and the PP intensity function, allowing for next-items and inter-session time prediction. In Chapter 6 we focus on the problem of specifying priors and hyperparameters for hierarchical models, and develop tools leveraging the prior predictive distribution (PPD) in order to connect summaries or prior knowledge from the data with *virtual* summaries[2] generated from the model. This technique applied from Poisson and Compound Poisson Factorization results in closed-form expression for the dimensionality of the latent space, as well as hyperparameters associated with the Gamma priors of the model. Finally, Chapter 7 presents a summary of the main findings of this thesis, discussing the revelance of the models, methods and empirical results, as well as pointing to future lines of investigation.

Research questions

- **RQ1** Is there an overarching strategy for incorporating contextual information into factorization models for recommender system? What improvement are observed by adding contextual information such social networks and item textual content in a joint model for recommendation?

- **RQ2** How to incorporate implicit feedback using count data models in factorization models for recommender system and what are the advantages of doing so?

- **RQ3** How can we include periodic time information into matrix and tensor factorization models for recommender system and what are the observed gains from doing so?

- **RQ4** What is the effect of adding a temporal point process model in a sequential multi-session recommendation model?

- **RQ5** How to analyze the properties of Bayesian factorization models for recommender system in order to specify the hyperparameters of the model?

[2]We use the term *virtual* summaries in a sense of quantities generated from sampling from the PPD before seeing any data.

Main contributions

- Probabilistic models for contextualized recommendations, based on different sources of contextual information and carefully designed for settings with count data, sequential events and time.
- Proposing a generic modeling framework for multiple relations and multiple entities that can be represented as probabilistic matrix and tensor coupled factorization.
- Introducing a method that utilizes the prior predictive distribution for prior specification and hyperparameter setting, leading to closed-form expressions for hyperparameters of the Poisson and Compound Poisson Factorization models and an stochastic gradient-based optimization algorithm for generic differentiable models.

Publications

- **Journals**:
  - *In submission* (pre-print): Eliezer de Souza da Silva, Tomasz Kuśmierczyk, Marcelo Hartmann, and Arto Klami. Prior specification via prior predictive matching: Poisson matrix factorization and beyond. *CoRR*, 2019. URL `http://arxiv.org/abs/1910.12263`
- **Conferences**:
  - Eliezer de Souza da Silva, Helge Langseth, and Heri Ramampiaro. Content-based social recommendation with poisson matrix factorization. In *ECML/PKDD (1)*, volume 10534 of *Lecture Notes in Computer Science*, pages 530–546. Springer, 2017
  - Bjørnar Vassøy, Massimiliano Ruocco, Eliezer de Souza da Silva, and Erlend Aune. Time is of the essence: A joint hierarchical RNN and point process model for time and item predictions. In *WSDM*, pages 591–599. ACM, 2019
- **Workshops**[3]:

[3]These workshop papers contain initial ideas further developed in other publications, therefore they will not be discussed in depth in this thesis and are included here for the sake of completeness.

- Eliezer de Souza da Silva. New probabilistic models for recommender systems with rich contextual and content information. In *WSDM*, page 839. ACM, 2017[4]
- Eliezer de Souza da Silva and Dirk Ahlers. Poisson factorization models for spatiotemporal retrieval. In *GIR*, pages 3:1–3:2. ACM, 2017[5]

[4]Doctoral consortium paper included in the proceedings of the main conference.
[5]Position paper.

Preliminary material 2

"I can illustrate the ... approach with the ... image of a nut to be opened. The first analogy that came to my mind is of immersing the nut in some softening liquid, and why not simply water? From time to time you rub so the liquid penetrates better, and otherwise you let time pass. The shell becomes more flexible through weeks and months — when the time is ripe, hand pressure is enough, the shell opens like a perfectly ripened avocado!"

— Alexander Grothendieck*, Récoltes et Semailles*

In this chapter we introduce the concepts, notation and general problem setting that we will focus on this thesis. The models developed in this dissertation are inspired by the existing literature, therefore we start by focusing on their theoretical foundations. Our first step is to establish the notation used for both the equations and diagrams to represent probabilistic models. Then, we introduce the framework of Bayesian modeling in Section 2.1, including latent-variable models, prior and posterior predictive distributions, interfaces between probabilistic modeling, neural networks and inference methods. The overall framework will be useful for understanding the different models employed, as well as the methodological contributions for prior specification. In Section 2.2 we introduce probabilistic models for count data, discuss their assumptions, potential benefits and limitations, and in Section 2.3, the recommendation problem, addressing interconnections with relational modeling, representations in matricial and tensorial forms, as well the general formulation of the corresponding probabilistic models. Finally, in Section 2.4, we discuss the challenges and approaches for finding hyperparameters and specifying prior distributions in hierarchical models, introducing the framework of prior-predictive checks and related techniques.

Notation

Throughout this thesis we will use the following notation:

- Scalar values are represented by lower-case letters x. Generic random variables and sets are represented by upper-case letters X. When reasoning about a particular value that this random can assume a lower-case version is used, for example in $p(X = x)$, nevertheless, for convenience and when it is clear from the context, we will simply use $p(x)$. Often, when talking about latent-variables we will use greek letters, for example $p(X|\theta)p(\theta)$. The size of a set A is denoted by $|A|$.

- Vectors are represented with bold lowercase, for example $\boldsymbol{x} = [x_1, x_2, \ldots, x_D]^\top \in \mathbb{R}^D$, and are column-vectors. Given vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, the inner product can be calculated and denoted with transpose of $\boldsymbol{x}$ multiplied by $\boldsymbol{y}$, expanding with the vector elements we have $\boldsymbol{x}^\top \boldsymbol{y} = \sum_{i=1}^{D} x_i y_i$. Matrices and tensors are represented with bold uppercase, the distinction between them should be apparent from the context, and when specifying their elements and index set, for example $\boldsymbol{A} = (A_{i_1 i_2}) \in \mathbb{R}^{N_1 \times N_2}$ and $\boldsymbol{B} = (B_{i_1 i_2 i_3}) \in \mathbb{R}^{N_1 \times N_2 \times N_3}$. This notation is valid for any variable, random or non-random.

  - In order to simplify notation for indices sets, we use the shorthand $[n] := \{1, 2, \ldots, n\}$, for the finite set of positive integers $i \leq n$. A contigous subset of the index can be represented using the notation $\boldsymbol{A}_{i:j} = [A_i, A_{i+1}, \ldots, A_j]$, with $i < j$.
  - Variables representing indices appear as subscript, in some cases we use a comma between two different indices when using multiple indices to improve readability or give emphasis, for example, $A_{i_1, i_2 i_3} = A_{i_1 i_2, i_3} = A_{i_1 i_2 i_3}$. The summation symbol $+$ can be added as a subscript substituting an index, and it denotes summing over the respective index, for example, given the multi-index $i_1 i_2 i_3 \in \mathcal{I}_1 \times \mathcal{I}_2 \times \mathcal{I}_3$, we use $A_{i_1 i_2 +} = \sum_{i_3 \in \mathcal{I}_3} A_{i_1 i_2 i_3}$ or $A_{i_1 ++} = \sum_{i_2 \in \mathcal{I}_2} \sum_{i_3 \in \mathcal{I}_3} A_{i_1 i_2 i_3}$. Sets representing collections of indices will be represented using calligraphy formating and upper-case, for example $\mathcal{I}$, and the same format will be used sometimes for sets or collections of other sets or random variables of importance.
  - The Hadamard (or element-wise) product of two vectors, matrices or tensors $\boldsymbol{A}$ and $\boldsymbol{B}$ with same dimensionality is denoted with $\boldsymbol{A} \circ \boldsymbol{B}$, and it is defined such that $(\boldsymbol{A} \circ \boldsymbol{B})_{i_1 \ldots i_N} = \boldsymbol{A}_{i_1 \ldots i_N} \boldsymbol{B}_{i_1 \ldots i_N}$, where $i_1 \ldots i_N$ is a generic index. The symbol $\circ$ will be used as well for function composition

$f \circ g$, defined as $(f \circ g)(x) = f(g(x))$, with the difference defined by the context.

- We use the notation $\mathbb{1}\{e\}$ to convert the logical condition e to 1 (one) when the expression is true and 0 (zero) otherwise. When operating with multiple indices we use the Kronecker delta δ_{ij} defined as 1 when $i = j$ and 0 otherwise, or with the logical notation $\delta_{ij} = \mathbb{1}\{i = j\}$.

- Given a graph $G = (V, E)$, with $V = \{v_i\}_{i \in [|V|]}$ representing the set of vertices, the set of edges is $E = \{e | \text{endpoints}(e) = \{v_i, v_j\}\}$. Each edge is defined by the two distinct vertices in the endpoints in case of an undirected graph, and additionally with a direction, in case of a directed graph. We define the neighborhood of a vertex v_i as the set of vertices that have an edge connected to it, using the notation $N(v_i) = \{v_j \in V | \exists e \in E : v_i, v_j \in \text{endpoints}(e) \wedge (v_j \neq v_i)\}$. If we lift the restriction that edges should have only two endpoints, allowing for any finite number $n \geq 2$, we obtain an hypergraph, which can be a relevant representation of more complex relationships between entities. A directed graph with no directed cycles (no directed path with repeated nodes) is called a Direct Acyclic Graph (DAG), and it is of relevance for Bayesian Networks.

- A probabilistic model can be expressed either by writing down the joint probability densities function (PDF, for continuous variables) or probability mass function (for discrete variables)[1] in terms of how they are factorized, for example, $p(X, Y) = F(X|Y)G(Y)$ (where $p(.)$ is a generic notation for a density function and $p(.|.)$ the conditional density)[2], or the formal data generating process, for example $Y \sim G$ and $X \sim F(Y)$, using the notation "$\sim$" to tell that variable (on the left hand side) is sampled from a distribution (on the right hand side). When expressing the probability density and distribution that a variable is sampled from, we can overload the notation in the following way: $p(X|Y) = F(X|Y)$ will be associated with $X \sim F(Y)$, for example for a Normal distributed variable we have $p(X|\mu, \sigma) = \mathcal{N}(X|\mu, \sigma^2)$ and $X \sim \mathcal{N}(\mu, \sigma^2)$.

  In both cases we obtain the same structure: the variable Y is generated from a ***prior*** probability distribution and the variable X is generated from another distribution conditioned on Y, typically characterized via its ***likelihood*** function. Usually, simply stating the distributions or PDFs in the

[1] We make this distinction here for the sake of completeness, but this distinction will be avoided whenever it is clear from context or definition of the variables

[2] The notation $f(.)$ is a shorthand for defining a function f that has one argument. In general we use this notation by substituting the arguments of the function with a dot.

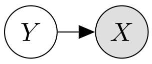
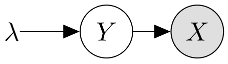
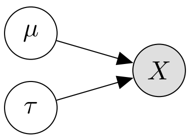
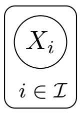
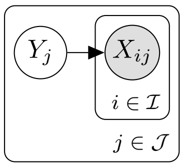
model is insufficient to fully specify the model, given that it might depend on fixed non-random variables, the ***hyperparameters*** of the model. To distinguish from conditioning on a random variable, we denote explicitly the hyperparameter with the symbol *";"*, in the above example it could be $p(X, Y) = p(X|Y; \lambda_1)p(Y; \lambda_2)$, with $\{\lambda_1, \lambda_2\}$ as the set of hyperparameters. Nevertheless, when it is implicit from the context, we will refrain from using that notation. For expected values $\mathbb{E}[.]$, variance $\mathbb{V}[.]$ and covariance $\mathrm{Cov}[.]$ we adopt the same notational convenience used for conditional probability and hyperparameters when denoting the expected value, for example $\mathbb{E}[X] := \mathbb{E}[X; \lambda]$ and $\mathbb{E}[X|Y] := \mathbb{E}[X|Y; \lambda]]$, variance $\mathbb{V}[X] := \mathbb{V}[X; \lambda]$ and covariance $\mathrm{Cov}[X, Y] := \mathrm{Cov}[X, Y; \lambda]$. The aforementioned concepts will have a detailed presentation in the next sections, here they are presented with the main purpose of clarifying the notations.

Plate notation and Probabilistic Graphical Networks (PGM). When presenting our models, we will both describe the model in hierarchical symbolic equations, as well as present their graphical representations, using Bayesian Networks (BN) and plate notations. This graphical notation is useful to represent concisely the independence structure of a probabilistic model, and although there are multiple variations on the notation and their representational power, we will focus on the notation that uses direct edges and does not allow directed cycles – the graphical model is a DAG (Pearl, 1988). Typically each variable of our model will be assigned a node in the graph, and direct edges are used to model the dependency structure of the variables. Given two nodes X and Y, a parent node has an edge pointing towards another node (the children node), for example if $Y \to X$, we say that $Y \in \mathrm{pa}(X)$ and $X \in \mathrm{ch}(Y)$, and define $\mathrm{pa}(X)$ as the set of all parents of X and $\mathrm{ch}(Y)$ as the set of all children of Y. This means that if X is in the neighborhood of Y, it is either a parent or a children of Y, formally $N(Y) = \mathrm{ch}(Y) \cup \mathrm{pa}(Y)$. Given this notation, and with the additional constraints that the resulting direct graph does not contain cycles, the joint probability of the model is derived from the BN graph as a product of conditional probabilities of each nodes conditioned on their parents[3]. In case we have an indexed family of variables $\{X_i\}_{i \in \mathcal{I}}$ that share a structure in our model, we can use the plate notation to simplify the graphical representation by enclosing the representation of the variables with a plate and indicating the index-set. The Markov Blanket $\mathrm{mb}(.)$ of a node X in a BN is the union-set of parent nodes, children nodes, and other parents of the children nodes (co-parents), formally defining it as $\mathrm{mb}(X) := \mathrm{pa}(X) \bigcup \mathrm{ch}(X) \bigcup_{Z \in \mathrm{ch}(X)} (\mathrm{pa}(Z) \setminus \{X\})$. We

[3]If $\mathrm{pa}(X) = \emptyset$, we say that $p(X|\,\mathrm{pa}(X)) = p(X)$

present the algebraic representation, text description and their equivalent graphical representation in Table 2.1.

Table 2.1: Examples with description of Bayesian networks and plate notation

Probability density	Description	Diagram
$p(X\|Y)p(Y)$	Latent variable Y and observable variable X	
$p(X\|Y)p(Y;\lambda)$	Observable variable X and latent variable Y with a fixed hyperparameter λ	
$p(X\|\mu,\tau)p(\mu)p(\tau)$	Observable variable X, and latent variables μ and τ	
$\prod_{i\in\mathcal{I}} p(X_i)$	Random variables $X_1, X_2, \dots, X_{\|\mathcal{I}\|}$ with indexset $\mathcal{I}$, represented with plate notation	
$\prod_{j\in\mathcal{J}} p(Y_j) \prod_{i\in\mathcal{I}} p(X_{ij}\|Y_j)$	For each observable variable X_{ij} there is latent variable Y_j	

2.1 Bayesian modeling in Machine Learning

A central topic in Machine Learning (ML) is the development of generic methods to solve different tasks by creating computational models with the ability to *learn* from data (relevant to the task), and consequently make data-based decisions, predictions or inferences. There are multiple approaches and frameworks relevant to achieve this goal, but we are going to focus on the approaches that use ideas from Bayesian statistics and probabilistic modeling. The most important advantage of using this approach is the ability to use a principled and logical methodology to reason and quantify *uncertainty* (Jaynes, 2003; Halpern, 2017; Pearl, 1988). Beyond those conceptual considerations, there has been recent progress in the computational tools facilitating both modeling and scalable inference of Bayesian models, which have been a bottleneck for larger scale adoption of this framework in the past. This section gives a general overview of Bayesian modeling applied in Machine Learning, introducing concepts, definition and results relevant for this thesis, the interested reader is refered to Bernardo and Smith (1994), Bishop (2006), Koller and Friedman (2009) and Gelman et al. (2013) for a detailed exposition.

A Bayesian model can be understood as description using probabilistic terms, distributions and expressions of a *data generation process*, in general with a structure containing conditional probabilities of the *observations* given certain *parameters*, and prior probability distributions for the parameters. This structure, aided by appropriate computational methods allows querying and characterization in probabilistic terms different variables and sets of variables of the model (observed and unobserved), that can be furthermore employed in tasks of inference, prediction or explanation. Mathematically one can express the different dependent components of this generative process using the laws of probability and Bayes equation. Given random variables X and θ, assuming X is the observed variable, and θ the parameters of the observation-model, we can express the joint probability in a factorized form using the conditional probability of X given θ, the *likelihood* of the observation, and the *prior* probability of choosing θ, described in Equation 2.1. The joint probability contains full information about different parts of our model, gives us probabilities of different values for the observations and parameters, and can be used to calculate marginal distributions. One resulting distribution of interest is the marginal probability of observations, the *evidence*, that can be obtained via integration (for continuous variables) or summation (for discrete variables) over the domain of those variables, represented in Equation 2.2. Bayes' theorem, represented in Equation 2.3, combines those results enabling the computation of the *posterior* distribution of parameters given the observed data, allowing us to answer the question: "given that we know the data, how can we update our estimates about the parameters?" – defining a process of updating our beliefs about our model given the observations. Once we have the posterior distribution, we can use it to compute distribution of future observations X' conditioned on the past observations X, given by the Equation 2.4, the *posterior predictive* distribution.

$$\begin{tikzcd}
p(\text{Parameter } \theta) \arrow[r, rightsquigarrow] & p(\text{Data } X|\text{Parameter } \theta) \arrow[r, rightsquigarrow] & \text{Data } X
\end{tikzcd}$$

FIGURE 2.1: Diagram of the generative view of Bayesian model. The observed data X is generated from a probabilistic process that specifies a model of the data $p(X|\theta)$ that depends on some parameter θ. Specifying these distributions will allow us to sample data from the model, and given a collection of observations, Bayes' Theorem allows us to characterize the settings that would generate those observations.

$$\overbrace{p(X,\theta)}^{\text{joint}} = \overbrace{p(X|\theta)}^{\text{conditional}}\ p(\theta) \tag{2.1}$$

$$p(X) = \int_\theta p(X,\theta)d\theta = \int_\theta p(X|\theta)p(\theta)d\theta \tag{2.2}$$

$$\overbrace{p(\theta|X)}^{\text{posterior}} = \frac{p(X,\theta)}{p(X)} = \frac{\overbrace{p(X|\theta)}^{\text{likelihood}}\overbrace{p(\theta)}^{\text{prior}}}{\underbrace{p(X)}_{\text{marginal or evidence}}} = \frac{p(X|\theta)p(\theta)}{\int_\theta p(X|\theta)p(\theta)d\theta} \tag{2.3}$$

$$p(X'|X) = \int p(X'|\theta)p(\theta|X)d\theta \tag{2.4}$$

As a simplified example, consider a binary classification task with a given dataset $\mathcal{D} = \{(\boldsymbol{x_i}, y_i)\}_{i\in[n]}$, feature vectors $\boldsymbol{x_i} \in \mathbb{R}^D$ and target values $y_i \in \{0,1\}$. One (traditional) approach would consist in proposing a parameterized mappings between the feature and target values $f_\theta : \mathbb{R}^D \to \{0,1\}$ (for example, a neural network, or a decision tree), a measure of discrepancy (denoted the *loss function*) between the target values and the values given by the mapping f_θ, seek an optimization or search strategy that would allow adjustments of the parameter θ in order to minimize the discrepancy, and finally apply this process of adjustments (the *training* process) using examples from the training dataset $\mathcal{D}$. The ultimate goal is *generalization*, meaning that the proposed model will display similar or superior performance on unseen examples from the same dataset and on the same task[4]. One limitation here is the inability to automatically express the *uncertainty* associated with the model, predictions of the model or the parameters inferred. Also, the choice of the space of possible functions can be *ad hoc* and present difficulties with supplying experts with mechanisms for integrating a priori knowledge into the model. Bayesian modeling offers an alternative with a principled methodology (based on Probability Theory and Statistics) to address those shortcoming: measuring different sources of uncertainty is reduced to characterizing the distributions associated, and by explicitly modeling different priors assumptions about the model in terms of random variables and their interdependencies, there is a natural language for reasoning about those assumptions, integrating with experts knowledge, and making inferences. One might distinguish between small and big data regimes, and the ability to reason about our uncertainty using Bayesian modeling becomes even more relevant in the small data regime, where experts or task specific knowledge can help guide the modelling when sufficient data is not available; in recommender systems literature this is related to the *cold start problem.* These considerations are nevertheless not absolute, and there are different tasks and contexts where they would be invalid.

In the previous example, a Bayesian modeling approach would start by proposing a representative structure to the probability of observing the target variables $y_i \in \{0, 1\}$, which can be modelled as Bernoulli distributed variable controlled by some parameter $p_i \in [0, 1]$, thus $y_i \sim \text{Ber}(p_i)$. The next step is to specify a possible distribution to the parameter, as well as how the features $\boldsymbol{x}_i$ will play a role in there, for example one could assume an unobserved vector parameter $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \sigma\boldsymbol{I}_D)$ normally distributed with zero mean and covariance given by the diagonal matrix $\sigma\boldsymbol{I}_D$, and $p_i = \phi(\boldsymbol{x_i}^\top \boldsymbol{w})$, where $\phi : \mathbb{R} \to [0, 1]$ is the sigmoid function (or some other function mapping $\boldsymbol{x_i}^\top \boldsymbol{w}$ to $[0, 1]$, for example the probit link function is another alternative). Our final model then is $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \sigma\boldsymbol{I}_D)$, and $y_i|x_i \sim \text{Ber}(\phi(\boldsymbol{x_i}^\top \boldsymbol{w}))$, represented in Figure 2.2, and conditioned on the dataset $\mathcal{D}$ and applying the appropriate inference method, either via analytical approximations or computational methods, using Bayes' theorem we are able to obtain the posterior distribution of the model parameters $p(\boldsymbol{w}|\mathcal{D})$. Furthermore, using the posterior and given a new feature vector $\boldsymbol{x'}$, we can compute the predictive distribution of the target y' given by $p(y'|\mathcal{D}, \boldsymbol{x'}) = \int \text{Ber}(y'|\phi(\boldsymbol{x'}^\top \boldsymbol{w}))p(\boldsymbol{w}|\mathcal{D})d\boldsymbol{w}$, which can be used to produce predictions as well as their associated uncertainty. Simplifying that expression, and targeting minimizing the misclassification errors, the classification rule reduces to $y' = 1$ if $0.5 \le p(y' = 1|\mathcal{D}, \boldsymbol{x'}) = \int \phi(\boldsymbol{x'}^\top \boldsymbol{w})p(\boldsymbol{w}|\mathcal{D})d\boldsymbol{w}$, and $y' = 0$,

[4]It is a common practice in ML to split the dataset into training and testing datasets, such that in a first phase the model is adjusted to the training examples, and consecutively evaluated on unseen examples from the testing dataset. In general, we would hope that this behavior would generalize to any new examples on the same task.

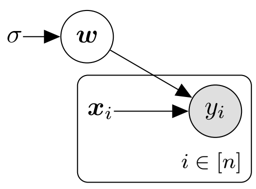


FIGURE 2.2: Example of a Bayesian model for a classification task

otherwise. With this final equation we have both a classification rule, as well the possibility of measure the uncertainty of associated with a given prediction. In order to turn this result into an algorithm, the steps necessary are the ones related to making the posterior computation and the predictive distribution feasible, this can be done either by manual calculation leading to close-form equations, approximations or general purpose computational methods available probabilistic programming languages and libraries such as Stan (Carpenter et al., 2017), PyMC3 (Salvatier et al., 2016) or Pyro (Bingham et al., 2019). In the following sections we will discuss in more details the different inference techniques and the context that they are desirable.

Independence. In probabilistic models with multiple interacting parts, one fundamental task is related to identifying and expressing when certain variables are influencing others and how this translates into structural assumptions in the model. Formally, we talk about *marginal independence* of variables X and Y, denoted as $X \perp\!\!\!\perp Y$, when the it holds that $p(X, Y) = p(X)p(Y)$, which means that the marginal distribution has all information to characterize the joint, in other words, each marginal density is acting as an independent dimension of the joint[5]. *Conditional independence*, denoted as $X \perp\!\!\!\perp Y|Z$, is defined by that property $p(X, Y|Z) = p(X|Z)p(Y|Z)$, asserting the independence only when conditioned to a certain variable, while implying a level coupling between the variables X and Y, mediated by variable Z that they are conditioned on[6]. Conditional independence can be defined as well when conditioning on a set of variables, and it can be applied to define the Markov Blanket as the set that contains all relevant probabilistic information for given variable of interest (Pearl, 1988; Bishop, 2006), meaning that conditioned on the Markov Blanket all other variables are independent, or in formal

[5]A related concept from information theory is mutual information between two variables, that according to its definition goes to zero when they are marginally independent.

[6]In fact it can be shown that $p(Y|X) = \int_Z p(Y|Z)p(Z|X)dZ$.

$$\begin{tikzcd}
X_i & \mathrm{mb}(X_i) \arrow[l] \arrow[d] \\
 & \mathcal{X} \setminus (\{X_i\} \cup \mathrm{mb}(X_i))
\end{tikzcd}$$

Figure 2.3: Ilustration of the conditional independence between the variable $X_i \in \mathcal{X}$ and all other variables in the collection $\mathcal{X}$ that are not in its Markov Blanket, conditioned on the Markov Blanket.

terms, $\forall X \notin \mathrm{mb}(Y) : Y \perp\!\!\!\perp X | \mathrm{mb}(Y)$ – Figure 2.3. This characterization simplifies the task of computing the *full conditionals* of a variable X, denoted as $p(X|*)$, which is the distribution of X conditioned on all other variables in the model. Full conditionals are relevant as building blocks for Gibbs samplings and variational inference with the mean field assumption. More formally, suppose that the model consists of a set of random variables $\mathcal{X} = \{X_1, \dots, X_n\}$, and given the conditional independence on the Markov Blanket, we can express the joint as

$$p(\mathcal{X}) = p(X_i | \mathrm{mb}(X_i)) \underbrace{p(\mathcal{X} \setminus (\{X_i\} \cup \mathrm{mb}(X_i)) | \mathrm{mb}(X_i)) p(\mathrm{mb}(X_i))}_{p(\mathcal{X} \setminus \{X_i\})}$$

for some $i \in [n]$ such that if we conditioned X_i on all the other variables $\mathcal{X} \setminus \{X_i\}$, it reduces to conditioning on $\mathrm{mb}(X_i)$. Reorganizing the above equation we obtain

$$p(X_i|*) = \frac{p(\mathcal{X})}{p(\mathcal{X} \setminus \{X_i\})} = p(X_i | \mathrm{mb}(X_i)) \tag{2.5}$$

2.1.1 Latent variable models

A common approach for adding expressity while balancing parsimony in a probabilistic model is to utilize latent variables models (LVM). The approach is based on modeling individual and group-level variability of the observations by postulating unobserved or *latent variables* acting on the model at individual and group level. For example, if we add a latent variable for each observation this would lead to a joint probability that factorizes in the following way $\prod_{i \in \mathcal{I}} p(X_i|\eta_i)p(\eta_i)$, for a set of observations $\{X_i\}_{i \in \mathcal{I}}$ and latent variables $\{\eta_i\}_{i \in \mathcal{I}}$. More broadly, the conditional structure of these models allows the use of *local* (per observation, or groups of

$$\begin{tikzcd}
& \mu \arrow[dl] \arrow[d] \arrow[dr] \arrow[ddl, bend right] \arrow[dd, bend right] \arrow[ddr] & \\
\eta_1 \arrow[r] \arrow[d] & \eta_2 \arrow[r] \arrow[d] & \eta_3 \arrow[d] \\
X_1 & X_2 & X_3
\end{tikzcd}$$

FIGURE 2.4: Diagram of a model with local interdependent latent variables and a global latent variable.

observations) and *global* (affecting the overall model) latent variables with various possibilities of structural independence between them (Hoffman et al., 2013). In general, we can have latent variables depending on each other and on a global variable, while the observations, on groups of latent variables (local and global). Put in formal terms, given a set observations $\{X_i\}_{i\in\mathcal{I}}$, local latent variables $\{\eta_j\}_{j\in\mathcal{J}}$ and global latent (multidimensional) variable $\boldsymbol{\mu}$, using the language of BN, one could describe the factorized join as $p(\boldsymbol{\mu})\prod_{j\in\mathcal{J}} p(\eta_j|\operatorname{pa}(\eta_j))\prod_{i\in\mathcal{I}} p(X_i|\boldsymbol{\mu}, \{\eta_j|\eta_j \in \operatorname{pa}(X_i)\})$. For example, it could be of interest to model sequential dependency on the level of latent variables, but not on the observations, which would be a stronger assumption. Figure 2.4 is an example of such type of model, with a joint probability that factorizes to $p(\mu)\prod_{i=1}^{3} p(X_i|\eta_i,\mu)p(\eta_i|\operatorname{pa}(\eta_i))$, where $\operatorname{pa}(\eta_1) = \{\mu\}$, and $\operatorname{pa}(\eta_i) = \{\mu, \eta_{i-1}\}$ (for $i \in \{2, 3\}$), and it is related to a family of models known as Hidden Markov Models (Bishop, 2006).

The use of latent variables models is widespread in different scientific fields, emerging from assuming the existence of simple underlying mechanisms that explain individual and group variability in the data. Another (related) reason is the need for *dimensionality reduction* of the data, which relies on assuming that the large variability in the observed data can be approximated by a small set of variables (Bartholomew et al., 2011). This is relevant to many scientific tasks, not only because of our limited cognitive capacity to grasp the structure of large quantities of variables without mapping them to small dimensions, but also because many theories rely on entities and abstractions having quantities which are not directly observed (for example, concepts such as personality traits, a topic in a collection of texts, user utility, preference or satisfaction are not *directly* measured)[7]. Statistical methods employed in those contexts utilize latent variables, model the

[7]A top-down approach could be proposing data generating models from those latent concepts to the observations, or to *proxy* measurable quantities. A buttom-up approach, could be one that

observations as dependent on those hypothetical quantities and find the settings for the latent variable that best fit the multivariate data.

Historical and bibliographical remarks. The modeling principles of LVM can be found already in the early works of Peirce (1884), Spearman (1904), Pearson (1901), Hotelling (1933) and Lazarsfeld (1950), that led to the development and widespread adoption of Latent Class Analysis (LCA), Principal Component Analysis (PCA) and Factor Analysis (FA) in the fields of applied statistics, social sciences, psychology and biometrics. Similarly, motivated by various scientific and engineering problems, mathematicians from the 19th and 20th centuries developed techniques related to finding solutions to linear system, low-rank approximations, bilinear and quadratic forms, culminating in a rich theory of matrix factorization methods such as Singular Value Decomposition (Stewart, 1993) and many others (see e.g. Hubert et al. (2000)). In fact, it is noted by Hubert et al. (2000) that historically those methods were developed and used in mathematics and applied statistics/psychometrics communities with distinct purposes: first as a way to decompose a difficult tasks into easier sub-problems (for example when solving large systems of equations), and secondly to reveal the fundamental structure present in large collection of observations (for example, when analysing large survey data). Nevertheless there is deep interconnection between the two tasks, and particularly in Machine Learning, factorization methods have been used both for their computational properties of simplifying larger tasks, as well as for the structural properties of finding hidden statistical patterns in the data. In our case, those are the main reasons for adopting the Poisson-Gamma factorization model as building block. For a broader review of latent variable models in psychology and social sciences the reader is refered to Bollen (2002) and Goodman (2002). Although different communities developed some latent variable models independently, mainstream statistics have been developing a convergence of those techniques, concepts and nomenclature, as well as extending and further developing them into a more general formulation; for a wider coverage and discussion of LVM models in statistics, we refer the reader to Skrondal and Rabe-Hesketh (2007) and Bartholomew et al. (2011).

Probabilistic Latent Variable Models in Machine Learning. Machine Learning, being a more recent field, received the scholarship previously developed, and extended it by developing efficient algorithms for learning and inference, as well as techniques for the utilization of large scale datasets. Furthermore, the

assumes latent variables, and only a posteriori conceptually interprest them to latent abstractions in certain theories.

use of Bayesian Networks both in modelling and inference, represents a step further in increasing expressivity of the models (Pearl, 1988), while maintaining and using general principles for design of inference algorithms (Ahmed et al., 2012; Hoffman et al., 2013; Wainwright and Jordan, 2008). Thus, continued progress on the field has been made on the development of Bayesian/probabilistic formulations of various LVM, together with efficient inference algorithms usually customized for each proposed model. This led to the proposal of various practical models and algorithms including Probabilistic PCA (Tipping and Bishop, 1999), Probabilistic Canonical Correlation Analysis (Bach and Jordan, 2005; Klami et al., 2013), Group Factor Analysis (Virtanen et al., 2012) and Latent Classification Models (Langseth and Nielsen, 2005; Vermunt and Magidson, 2003).

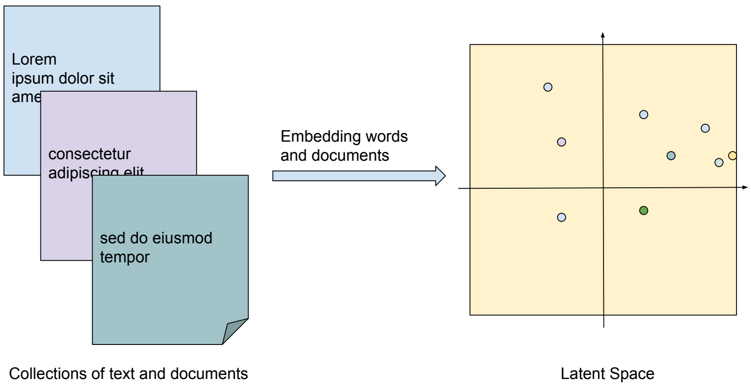


FIGURE 2.5: Embedding a collection of documents and words into a latent space

Vector Space Model and Latent Semantic Analysis. Particularly in text analysis and modeling, LVM have been introduced with the insight that the semantics of words and documents can be inferred from the relationships that emerge from their respective latent variables. This approach is present in the early work of Landauer and Dumais (1997) introducing the latent semantic analysis theory, with the idea that words can be represented in a (continuous) latent space, inducing a *representation* that is learned from the co-ocurrences patterns in large bodies of text (Figure 2.1.1). The main advantage of representing strings of text as numerical vectors is that it allows us to perform operations on the this latent representation using numerical algebra and algorithms in a consistent and unified way, so long as the structural relationships between those vectors (for example, distances, angles or inner products) captures structural relationships between the words they represent (for example, linguistics features, co-occurence patterns in sentences, semantics) – the survey work of Turney and Pantel (2010) discusses in depth the hypothesis, literature and models developed around the idea of representing textual data in a vector space. This insight has motivated further development of systems and algorithms for text analysis tasks, initially in Information Retrieval and text indexing with the proposal of Vector Space Model (VSM) and the development of the Latent Semantic Indexing (Berry et al., 1995; Deerwester et al., 1990) and Probabilistic Latent Semantic Indexing (Hofmann, 1999), widely adopted and deployed in modern computing technologies such as text-based internet search. A related family of models are the *topic models*, with the seminal work of Blei et al. (2003) introducing the Latent Dirichlet Allocation (LDA) model, sharing many features with LSI, but with an additional latent structure of *topics*, where each document is generated from a mixture of topics, and each word sampled from a given topic. Those models can be seen as early versions of methods for *word embedding* (Mikolov et al., 2013) and *representation learning* (Bengio et al., 2013), which are important sub-fields of Machine Learning research, that have been fueled in recents years by successful application of (deep) neural networks to Computer Vision (CV) and Natural Language Processing (NLP) tasks.

Neural networks and other applications. Probabilistic LVM have been sucessfully combined with neural networks, effectively creating mechanisms for parameterized non-linear transformations between the random variables in latent space and the observations. The seminal works of Kingma and Welling (2014) and Rezende et al. (2014) introduced the Variational Auto Encoder (VAE) model, and developed techniques for the computation of gradients through stochastic and deterministic computation graphs (Mohamed et al., 2020; Masegosa et al., 2021), leveraging the tools of automatic differentiation and optimization for probabilistic LVM. Other recent works by Mooij et al. (2010) and Kaltenpoth and Vreeken (2019) have shown the potential of latent-variable modeling for distinguishing between cause and effect from observational data. Kunin et al. (2019) develops a theoretical analysis of Linear Auto-Encoder, demonstrating equivalances between the critical points of linear auto-encoder model training via construction of regularized loss function, and the MAP of Probabilistic PCA, as well theoretical analysis of the geometry of those models.

Information theory and geometry. Another insightful perspective, inspired by Information Theory, is the idea that LVM can be seen as encoding the observations and compressing their representation. This binds together the statistical intuition that a latent variable is capturing sources of individual variability for the observation with formal concepts of information and coding theory, such as code length and entropy as measure of information. Examples of the link between Bayesian inference, either via Maximum-A-Posteriori, Variational Inference or model selection, information-theoretical measures of complexity and the Minimum Description Length principle are expressed in the works of Rissanen (1996), Honkela and Valpola (2004) and Graves (2011). The outline of this connection is that the expected value of the negative logarithm of the joint of a probability model or the posterior is proportional to measures of information (entropy), which is a lower bound for the code-length of a probabilistic source, binding together the maximization of probabilities with minimization of description lenghts. For an in-depth theoretical and empirical analysis of these conceptual interconnections we refer to Ullrich (2020), Grünwald (2007) and MacKay (2003).

2.1.2 Inference techniques

Given the generative description of a probabilistic model, which defines how the data is generated, we can use Bayes' Theorem to make inferences about the unobserved random variables of the model. A common task is then to characterize the distribution of the latent variables, conditioned on a set of observations – the posterior inference. These computations can be done analytically by direct application of Bayes' Theorem only for a limited set of models, and this difficulty in the general case is intrinsic to Bayes' Theorem itself, given that it depends on calculating the marginal probabilities of the observation, in fact in the work of Cooper (1990) and Roth (1996) it is shown that it is NP-hard for a large class of models and strategies. In order to overcome this challenge, many methods have been proposed in the literature and practice. One initial approach consists in relaxing the criteria of characterizing the full posterior distribution, relying only on point estimates of the relevant properties of that distribution, for example the mode of the posterior or the likelihood function – the Maximum-a-Posteriori (MAP) and Maximum-Likelihood Estimate (MLE) methods. This approach reduces the posterior inference problem to an optimization problem of a single point of interest of the posterior distribution. Another approach is to use an approximation for the posterior, generated via a sampling procedure, leading to a numerical approximation of the posterior that will converge to the true posterior in probability as the number of samples increase (and under certain assumptions) – the Markov Chain Monte Carlo (MCMC) family of methods (Casella and Berger, 2001; Gelman et al., 2013). Finally, the variational inference approach seeks an approximation for the posterior distribution by assuming that there is parameterized family of distributions that can be optimized to match the posterior (Bishop, 2006; Blei et al., 2017).

Let us consider a probabilistic model with local and global latent variables, represented in Figure 2.6, we can express the joint probability as $p(\{X_i\}, \{Y_i\}, Z; \sigma_Z, \sigma_Y) = p(Z; \sigma_Z) \prod_{i\in[n]} p(X_i|Y_i, Z)p(Y_i|Z, ; \sigma_Y)$. Given a dataset of observations $\mathcal{D} = \{X_1, \dots, X_n\}$, and denoting the set of latent variables as $\Theta = \{Z\} \cap \{Y_1, \dots, Y_n\}$ and hyperparameters $\sigma = \{\sigma_Z, \sigma_Y\}$, the posterior inference problem consist in computing $p(\Theta|\mathcal{D}; \sigma)$. Applying Bayes' Theorem we optain

$$p(\Theta|\mathcal{D}; \sigma) = \frac{p(\Theta, \mathcal{D}; \sigma)}{\int p(\Theta, \mathcal{D}; \sigma) d\Theta} = \frac{p(\mathcal{D}|\Theta)p(\Theta; \sigma)}{p(\mathcal{D}; \lambda)}$$

where $p(\mathcal{D}; \lambda) = \int p(Z; \sigma_Z) \prod_{i\in[n]} p(X_i|Y_i, Z)p(Y_i|Z; \sigma_Y) dZ dY_i$. In this case computing the MLE and MAP by maximizing the log-likelihood results in:

$$\begin{aligned}
\Theta_{MLE} &= \underset{\Theta}{\text{argmax}} \log p(\mathcal{D}|\Theta) = \underset{\{Z\}\cup\{Y_i\}}{\text{argmax}} \sum_{i\in[n]} \log p(X_i|Y_i, Z) \\
\Theta_{MAP} &= \underset{\Theta}{\text{argmax}} \{\log p(\mathcal{D}|\Theta) + \log p(\Theta; \sigma)\} \\
&= \underset{\{Z\}\cup\{Y_i\}}{\text{argmax}} \{\log p(Z; \sigma) + \sum_{i\in[n]} \log p(Y_i|Z; \sigma) + \log p(X_i|Y_i, Z)\}
\end{aligned}$$

In this way we can perform the MAP optimization as using the log of prior probabilisties to add regularizer terms to the MLE optimization.

Variational Inference. Given a measure of the discrepancy between probability distributions, usually the Kullback-Leibler divergence (KLD), and a parameterized family of distributions $Q(\Theta) := \{q(\Theta; \lambda)\}_{\lambda\in\mathcal{X}}$ that is assumed to approximate the posterior $p(\Theta|\mathcal{D}; \sigma)$, the variational inference approach consists in setting up an optimization problem to find parameters that minimizes the discrepancy in relation to the posterior:

$$q(\Theta; \lambda^*) = \underset{q(\Theta;\lambda)\in Q(\Theta)}{\text{argmin}} \text{KL}\{q(\Theta; \lambda), p(\Theta|\mathcal{D}; \sigma)\} \tag{2.6}$$

This approach consists in transforming the original problem of posterior inference into an optimization problem on the space of probability distributions. To make it feasible, it is necessary to define a discrepancy measure between distributions,

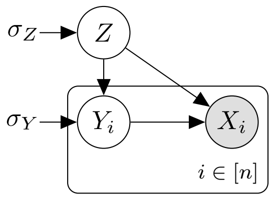


FIGURE 2.6: Diagram for a generic probabilistic model with local and global latent-variables and hyperparameters.

design a family of distributions to approximate the posterior and optimize the discrepancy measure over this family of distributions. The typical choice for the discrepancy measure is the Kullback-Leibler divergence (KLD), defined formally below.

Definition 2.1. *Given two probability distributions over the same space $\mathcal{X}$, with densities p and q, we define the Kullback-Leibler divergence as*

$$KL\{q(X), p(X)\} = \int_{x\in\mathcal{X}} q(x)\log\frac{q(x)}{p(x)}dx = \mathbb{E}_{q(X)}[\log\frac{q(X)}{p(X)}]$$

Substituting the variational family in the KLD we obtain the divergence between the variational approximation and the posterior.

$$\begin{aligned}
\mathrm{KL}\{q(\Theta;\lambda), p(\Theta|\mathcal{D};\sigma)\} &= \mathbb{E}_{q(\Theta;\lambda)}\left[\log\frac{q(\Theta;\lambda)}{p(\Theta|\mathcal{D};\sigma)}\right] \\
&= \mathbb{E}_q[\log q(\Theta;\lambda) - \log p(\Theta|\mathcal{D};\sigma)] \\
&= \mathbb{E}_q\left[\log q(\Theta;\lambda) - \log\frac{p(\Theta,\mathcal{D};\sigma)}{p(\mathcal{D};\sigma)}\right] \\
&= \underbrace{-\mathbb{E}_q\left[\log\frac{p(\Theta,\mathcal{D};\sigma)}{q(\Theta;\lambda)}\right]}_{\text{ELBO}(\lambda)} + \underbrace{\log p(\mathcal{D};\sigma)}_{\text{evidence}} \qquad (2.7)
\end{aligned}$$

Evidence Lower BOund (ELBO). In Equation 2.7 we observe that the KLD can be decomposed into a sum of two terms, the evidence $\log p(\mathcal{D};\sigma)$ and a term that depends on the variational approximation ELBO(λ). Observe that it is always true that $\mathrm{KL}\{q(\Theta;\lambda), p(\Theta|\mathcal{D};\sigma)\} \geq 0$, implying that $\mathrm{ELBO}(\lambda) \leq \log p(\mathcal{D};\sigma)$, which motivates the naming. Now this manipulation of the formula has simplified that task of optimizing the KLD, given that the ELBO depends only on the joint probability, while the KLD depends on the posterior. Additionally, the evidence is independent of the variational parameters, implying that the optimization of the KLD with respect to the variational parameters λ can be performed using the ELBO. Finally, the negative sign in front of the ELBO implies a inverse proportionality relationship with the KLD, meaning that minimization of the KLD is equivalent to maximization of the ELBO. Combining those observations we can rewrite the optimization problem described in Equation 2.6 in terms of the ELBO.

$$\begin{aligned}
q(\Theta;\lambda^*) &= \underset{q(\Theta;\lambda)\in Q(\Theta)}{\operatorname{argmin}} \ \mathrm{KL}\{q(\Theta;\lambda), p(\Theta|\mathcal{D};\sigma)\} \\
&= \underset{q(\Theta;\lambda)\in Q(\Theta)}{\operatorname{argmin}} \ \mathbb{E}_{q(\Theta;\lambda)}\left[\log\frac{q(\Theta;\lambda)}{p(\Theta|\mathcal{D};\sigma)}\right] \\
&= \underset{q(\Theta;\lambda)\in Q(\Theta)}{\operatorname{argmax}} \ \mathbb{E}_{q(\Theta;\lambda)}\left[\log\frac{p(\Theta,\mathcal{D};\sigma)}{q(\Theta;\lambda)}\right] \\
&= \underset{q(\Theta;\lambda)\in Q(\Theta)}{\operatorname{argmax}} \ \mathrm{ELBO}(\lambda) \qquad (2.8)
\end{aligned}$$

Coordinate Ascent Variational Inference (CAVI). The next component of the variational inference is the definition of the variational family of distributions $Q(\Theta)$. The simplest approach consist in using the mean-field approximation, which assigns a single approximation family for each latent variable, and defines the variational family as the product of all the individual approximations. Using the previous defined model, this would mean to define the variational family as $q(\Theta;\boldsymbol{\lambda}) = q(Z;\lambda_0)q(\boldsymbol{Y};\boldsymbol{\lambda}_{1:n}) = q(Z;\lambda_0)\prod_{i\in[n]} q(Y_i;\lambda_i)$, with $\boldsymbol{\lambda} := \{\lambda_0,\lambda_1,\dots,\lambda_n\}$. Given the mean-field assumption for the variational family, we proceed by looking for an optimal choice of variational family for each latent variable, which is possible because of the independence in the mean-field assumption. Substituting the variational family we can rewrite the optimization in terms of each individual factor of the mean-field approximation and iterated expectations, for example in the case of the global latent variable Z we would obtain

$$\begin{aligned}
\mathrm{ELBO}(\lambda) &= \mathbb{E}_q\left[\log\frac{p(Z,\boldsymbol{Y},\mathcal{D})}{q(Z;\lambda_0)\prod_{i\in[n]} q(Y_i;\lambda_i)}\right] \\
&= \mathbb{E}_q\left[\log p(Z,\boldsymbol{Y},\mathcal{D})\right] - \mathbb{E}_q[\log q(Z;\lambda_0)] - \sum_{i\in[n]} \mathbb{E}_q[\log q(Y_i;\lambda_i)]
\end{aligned}$$

Now focusing on optimizing the ELBO for the variational approximation of the global variable Z we can ignore the terms that are fixed when changing the variational distribution of Z, which in this case are the terms $\sum_{i\in[n]} \mathbb{E}_q[\log q(Y_i;\lambda_i)]$. We can also write the expectation in terms of iterated expectations, leading to the following optimization problem:

$$\underset{\lambda_0}{\operatorname{argmax}} \operatorname{ELBO}(\lambda) = \underset{\lambda_0}{\operatorname{argmax}} \mathbb{E}_{q(Z)}\left[\mathbb{E}_{q(\boldsymbol{Y})}[\log p(Z,\boldsymbol{Y},\mathcal{D})] - \log q(Z;\lambda_0)\right] \tag{2.9}$$

Notice that we can rewrite $f(Z) - \log q(Z)$ as $-\log \frac{q(Z)}{\exp f(Z)}$, leading to $\mathbb{E}_{q(Z)}[f(Z) - \log q(Z)] = -\operatorname{KL}\{q(Z), \exp f(Z)\}$. Applying this to Equation 2.9, using $f(Z) = \mathbb{E}_{q(\boldsymbol{Y})}[\log p(Z,\boldsymbol{Y},\mathcal{D})]$ we obtain

$$\begin{aligned}
\underset{\lambda_0}{\operatorname{argmax}} \operatorname{ELBO}(\lambda) &= \underset{\lambda_0}{\operatorname{argmax}} -\mathbb{E}_{q(Z)}\left[\log \frac{q(Z;\lambda_0)}{\exp \mathbb{E}_{q(\boldsymbol{Y})}[\log p(Z,\boldsymbol{Y},\mathcal{D})]}\right] \\
&= \underset{\lambda_0}{\operatorname{argmin}} \operatorname{KL}\{q(Z;\lambda_0), q^*(Z)\} \qquad (2.10) \\
\text{with } q^*(Z) &\propto \exp \mathbb{E}_{q(\boldsymbol{Y})}[\log p(Z,\boldsymbol{Y},\mathcal{D})]
\end{aligned}$$

Notice that the expectation on the exponential of $q^*(Z)$ is over all the other latent variables using the variational approximation, and we would need a normalizing constant to assure that $\int_z q^*(z)dz = 1$. Also, it can be shown that this expectation can be written in terms of the full conditional of the variable Z, in formal terms $\mathbb{E}_{q(\boldsymbol{Y})}[\log p(Z,\boldsymbol{Y},\mathcal{D})] = \mathbb{E}_{q(\boldsymbol{Y})}[\log p(Z|*)] + \text{const}$, such that when optimizing for Z we can ignore the constant terms. Now, on Equation 2.10 choosing λ_0 such that $q(Z;\lambda_0)$ would match $q^*(Z)$ is the optimal choice, and it can be done analytically when both distributions are from the same family of distributions.

A similar derivation can be done for each latent variable, and latent variables models in general with the mean-field assumption. Bishop (2006) and Blei et al. (2017) show the general result that given the mean field assumption the optimal solution is always proportional to the exponential of the expected log of the full conditional[8]. If we combine this generic result with Equation 2.5, showing that the full conditional can be expressed in terms of the Markov blanket, we will obtain a characterization valid for any BN for any given latent variable Z of a collection $\mathcal{Z}$ and expectations over all other variables $q(*) := q(\mathcal{Z} \setminus \{Z\})$.

$$\begin{aligned}
& q^*(Z) \propto \exp \mathbb{E}_{q(*)}[\log p(Z|*)] \\
\Rightarrow\; & q^*(Z) \propto \exp \mathbb{E}_{q(\operatorname{mb}(Z))}[\log p(Z|\operatorname{mb}(Z))]
\end{aligned} \tag{2.11}$$

[8]Also, this result can be rewritten using the geometric expected value, defined as $\mathrm{G}_q[X] = \exp(\mathbb{E}_q[\log(X)])$.

The CAVI algorithm consist in iterating for each latent variable the update for its variational distribution using Equation 2.11, while keeping the variational distribution for the remaining variables fixed, until a convergence criteria for the ELBO is reached. The update equation for the free parameter of the variational distribution of each latent variable can be computed in close-form when the variables of the model are *conditionally conjugate* (Wang and Blei, 2013; Blei et al., 2017). This presentation of the CAVI variational updates that relies on the Markov Blanket of the variable of interest, highlights the idea that when creating complex coupled matrix factorization models, one need to focus only on the immediate or local topology of the network, simplifying greatly the analysis and derivations necessary.

Probabilistic models and neural network models. A neural network (NN) can be seen as parametric mapping $f_\theta : \mathcal{X} \to \mathcal{Y}$, typically constructed by composing simpler functions $f_\theta = f^1_{\theta_1} \circ f^2_{\theta_2} \circ \cdots \circ f^H_{\theta_H}$. Those simpler functions are called neurons or cells, and are usually implemented as a combination of affine transformations (*hidden units*) of the input and a non-linear operation on the output (known as the *activaction function*). Furthermore, architectural choices regarding how to compose those simpler functions into bigger modules are also relevant and impactful in the final performance for a given task. The fitting process the parameters of the NN to the observation data is performed using the backpropagation technique to compute gradients with respect to the parameters of the NN, and the (stochastic) gradient descent algorithm to minimize a loss function. Aspects to be considered in the design of NN are the choices of network architecture, activation functions, dimensionality of the parameter and how many parameters in total, and those choices affect numerical stability and computational complexity of computing gradients and optimizing the loss function on the training phase. Specific design choices here have proven to be useful accross multiple tasks, generating *modules* such as the Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) which are a form of Recurrent Neural Network (Hochreiter and Schmidhuber, 1997; Chung et al., 2014), Convolutional Networks (LeCun et al., 1989), Attention and Transformers (Bahdanau et al., 2015; Vaswani et al., 2017; Luong et al., 2015), among others. Therefore, the field has grown by exploring various design choices for architecture, neurons, activation function and training algorithms for a number of different tasks. A more complete overview of the all those models and the recent advances in the field can be found in LeCun et al. (2015), Goodfellow et al. (2016) and Schmidhuber (2015).

Together with the neural network archictetural innovations, availability of large amounts of data, and effective training algorithm, another aspect that contributed to the recent growth of NN models and research is the emergence of computational tools and resources for large scale training of those models. One of the most successfully adopted tools has been Automatic Differentiation (AD) methods and frameworks. With Automatic Differentiation, a computer program can be written in general purpose language and consisting of multiple mathematical operations over sets of numerical variables, and the gradient with respect to any of those variables can be computed efficiently and automatically by the computer program. This can then be utilized in the gradient descent algorithm. For an overview of the topic, we refer to Baydin et al. (2017). Recent techniques for Variational Inference have targeted the development of gradient estimators for parameters of probabilistic models, this had led to the develpment of techniques such as Automatic Differentiation Variational Inference (Kucukelbir et al., 2017), Black Box Variational Inference (Ranganath et al., 2014) and other variations (Mohamed et al., 2020). The combination of AD for neural networks and variational inference of generic probabilistic models, together with the availability of software libraries with those capabilities, has led to the development of the Deep Probabilistic Programming framework (Bingham et al., 2019; Tran et al., 2017), synergetic combinations of modeling techniques, facilitated implementation and training of hybrid neural probabilistic models, a task that previously would involve laborious manual steps. The combination of neural networks and probabilistic models will be used in this thesis on Chapter 5 when designing a hybrid RNN and temporal point process model, and the techniques of gradient estimation for probabilistic models will be applied in Chapter 6 in the generic gradient-based optimization algorithm for prior specicification.

2.2 Models for count data

In many applications of interest the available data is in the forms of counts or discrete data. Examples of such type of information are the number of times a user heard a song, counts of visits to a given page, or interactions with other users or items. Also, some forms of explicit feedback such as rating come in binary forms (likes) or finite discrete scales. Nevertheless, many traditional methods for recommender systems assume a Gaussian distribution of those observations, this assumption is also implicit when utilizing the quadratic loss.

The choice of a distribution with a range on the set of natural numbers, or a finite subset of it, is therefore necessary in order to better fit count data. A common starting choice is the Poisson distribution, given its simplicity and properties, which can be extended by combining with other distributions in a hierarchical fashion. Furthermore, when observing count data over time, we need to model the variability of the rate of events over time, leading to a related set of models, known as Temporal

Point process (TPP) models. In the next subsections we will introduce those two model families, their relationships with other distributions and main properties with relevance for this thesis.

2.2.1 Poisson and compound Poisson models

A Poisson distributed variable is a random variable with a range on the natural numbers characterized by a positive continuous rate. It is the prototypical distribution for modeling counts, utilized usually in situations where the probability of a count event happening is constant over time or space. It can be seen as an approximation for the Binomial distribution as the number n of trials increases – also known as the *law of rare events* (Shiryaev and Boas, 1995). If we imagine an interval divided into n equally spaced sub-intervals, where events can ocurr at each sub-interval with the same probability $\frac{\lambda}{n}$, and at each interval there is an independent success/fail Bernoulli trial $\mathrm{Ber}(\frac{\lambda}{n})$ adding a count of at most 1, the total counts of events will have a Binomial distribution $\mathrm{Bin}(n, \frac{\lambda}{n})$. The *law of rare events* estabilish that as the number of sub-intervals n increases, therefore decreasing the probability of a single event at each sub-interval, the Binomial distributed total count of events will converge to the Poisson distribution with rate λ, that we can express with the equation $\lim_{n\to\infty} \mathrm{Bin}(n, \frac{\lambda}{n}) = \mathrm{Poisson}(\lambda)$. Furthermore, the Poisson distribution achieves good approximation rates for sum of n independent Bernoulli variables, expressed in terms of the KL divergence with an order of $O(n^{-2})$ (Bobkov et al., 2019).

Thus, in many real world scenarios with count data the Poisson distribution is a good initial fit, given that it can capture or approximate the total counts in situations with large number of events, with count increments happening with small (equal or similar) probabilities, under the condition that the sum of those probabilities converges to the rate of the associated Poisson distribution. In Definition 2.2 the probability mass function, the expected value and variance of the Poisson distribution is formally presented.

Definition 2.2 (Poisson distribution). *A Poisson distributed random variable $X \sim Poisson(\lambda)$ has its support on the set $\mathbb{N}_0 = \{0\} \cup \mathbb{N}$ and as parameter the positive continuous rate $\lambda > 0$. The probability mass function is given by*

$$P(X = x) = Poisson(x; \lambda) = \lambda^x \frac{e^{-\lambda}}{x!} \tag{2.12}$$

The expected value and variance are given by

$$\mathbb{E}[X; \lambda] = \mathbb{V}[X; \lambda] = \lambda \tag{2.13}$$

It is possible to increase the flexibility of a Poisson model by adding a prior to the rate parameter. The support of the prior distribution should be the positive real line $\mathbb{R}^+$, and if we impose the restriction of a conjugate prior, we conclude that it should be a Gamma distribution. We formally define the Gamma distribution in Definition 2.3.

Definition 2.3 (Gamma distribution)**.** *A Gamma distributed random variable $\lambda \sim Gamma(a, b)$ has its support on the positive real line $\mathbb{R}^+$ and parameters shape $a > 0$ and rate $b > 0$. The probability density function is given by*[9]

$$Gamma(\lambda; a, b) = \frac{b^a}{\Gamma(a)} \lambda^{a-1} e^{-b\lambda} \tag{2.14}$$

The expected value and variance are given by

$$\mathbb{E}[\lambda; a, b] = \frac{a}{b} \tag{2.15}$$

$$\mathbb{V}[\lambda; a, b] = \frac{a}{b^2} \tag{2.16}$$

A conjugacy analysis of the Poisson-Gamma model can be performed by assuming a model for a set of n Poisson distributed observations and a Gamma latent rate, $p(\boldsymbol{x}, \lambda) = \text{Gamma}(\lambda; a, b) \prod_{i=1}^{n} \text{Poisson}(x_i|\lambda)$, and computing the posterior distribution of the rate given the observations $p(\lambda|\boldsymbol{X})$. Applying the Bayes' Theorem and focusing on the terms that are related to λ we obtain

$$\begin{aligned}
p(\lambda|\boldsymbol{X}) &= \frac{1}{p(\boldsymbol{X})} \frac{b^a}{\Gamma(a)} \lambda^{a-1} e^{-b\lambda} \prod_{i=1}^{n} \lambda^{x_i} \frac{e^{-\lambda}}{x_i!} \\
\Rightarrow p(\lambda|\boldsymbol{X}) &\propto \lambda^{a+\sum_{i=1}^{n} x_i - 1} e^{-(b+n)\lambda} \\
\Rightarrow p(\lambda|\boldsymbol{X}) &= \text{Gamma}(\lambda; a + \sum_{i=1}^{n} x_i, b + n)
\end{aligned}$$

We can also compute the marginal expected value and variance of the observations using the law of total expectation and variance obtaining $\mathbb{E}[X_i] = \mathbb{E}[\mathbb{E}[X_i|\lambda]] = \frac{a}{b}$ and $\mathbb{V}[X_i] = \mathbb{E}[\mathbb{V}[X_i|\lambda]] + \mathbb{V}[\mathbb{E}[X_i|\lambda]]] = \frac{a}{b} + \frac{a}{b^2}$. By inspection of those formula we can observe that the resulting marginal distribution of the observations of our model is overdispersed – defined in terms of variance to mean ratio bigger than one (Hilbe, 2014). A similar analysis can be carried by marginalization of the rate λ in joint of the model, which results in a marginal Negative-Binomial distribution

[9]The function Γ is defined as $\Gamma(t) = \int_0^\infty x^{t-1} e^{-x} dx$ and have the property $\Gamma(t+1) = t\Gamma(t)$

$\text{NB}(a, \frac{b}{b+1})$ with the same mean and variance calculated previously (Hilbe, 2014). This result can be generalized to any hierarchical Poisson model with a prior rate distribution parameterized by mean μ and variance σ^2, resulting in marginal mean $\mathbb{E}[X_i] = \mu$ and variance $\mathbb{V}[X_i] = \mu + \sigma^2$, in other words an additive relationship between mean and variance $\mathbb{V}[X_i] = \mathbb{E}[X_i] + \sigma^2$. This approach of adding flexibility to the distribution via latent variables of the Poisson will be useful in our models, given that it allows for more complex models while also being used for coupling different observations, in the case of shared latent variables.

There are many properties of interest of Poisson models for the development of more complex count models. The *additivity* property is the fact that we can combine a set of Poisson models via summation into a single Poisson model with the rate given by the sum of rates of each individual model. The *decomposition* property, also known as Raikov's theorem (Raikov, 1938), states the converse, that if a Poisson random variable admits a decomposition as a sum of independent random variables, then each summand is Poisson distributed. In hierarchical models we can use these properties to justify the use of *latent counts* for a given Poisson model, which can be useful to simplify certain models as well as provide explanatory power, since we can interpret individually summand terms of complex models as generating counts. For example if $Y \sim \text{Poisson}(\sum_{k=1}^{K} \lambda_k)$, we can create an equivalent model $Y = \sum_{k=1}^{K} Z_k$ with latent counts $Z_k \sim \text{Poisson}(\lambda_k)$, for $k \in [K]$. The use of latent counts leads to another interpretation of Poisson-Gamma models as an *allocative* model (Schein, 2019; Yildirim et al., 2021), meaning that the latent rates define probabilities of allocation of counts or events in different buckets, given a total number of counts summing all buckets. This can be formalized when calculating the conditional distribution of latent counts $Z_1, \dots, Z_K$ given the observed total counts Y, which follows a Multinomial distribution $p(Z_1, \dots, Z_K|Y) = \text{Mult}(Z_1, \dots, Z_K; Y, \boldsymbol{p})$ with the probabilities proportional to the rates of each individual count. Thus, the total count Y represent a *budget* of counts that can be allocated to each individual latent count Z_k with a probability proportional to the rate of that individual count $p_k = \frac{\lambda_k}{\lambda}$. The allocative intuition of the latent counts can be used as well when thinking about latent variable models, in particular it helps one intuit about the role that different terms might be playing in your model. For example, if we assume a model with a rate $\lambda = \boldsymbol{\theta}^\top \boldsymbol{\eta} = \sum_{k=1}^{K} \theta_k \eta_k$, where $\boldsymbol{\theta}$ and $\boldsymbol{\eta}$ are high-dimensional non-negative vectors, latent counts will be allocated to each latent dimension with a probability $p_k = \frac{\theta_k \eta_k}{\boldsymbol{\theta}^\top \boldsymbol{\eta}}$, allowing us to interpret the value of each component of the latent vector as the strength of this allocation. To formalize these, we present the Multinomial distribution in Definition 2.4 and the aforementioned properties in the Proposition 2.1 and Proposition 2.2.

Definition 2.4 (Multinomial distribution)**.** *The random vector $\boldsymbol{z} \sim$ Mult$(n, \boldsymbol{p})$ is a*

K dimensional count vector $\boldsymbol{z} := [z_1, \dots, z_K] \in \mathbb{N}_0^K$ sampled from the Multinomial distribution defined by the parameters totals count $n \in \mathbb{N}$ and event probabilities vector $\boldsymbol{p} := [p_1 \dots, p_K] \in [0,1]^K$, with $\sum_{k=1}^K p_k = 1$ and $\sum_{k=1}^K z_k = n$. The probability mass function is given by

$$Mult(\boldsymbol{z}; n, \boldsymbol{p}) = n! \prod_{k=1}^K \frac{p_k^{z_k}}{z_k!} \mathbb{1}\{\sum_{k=1}^K z_k = n\}$$

The expected value and variance for each individual count variable is

$$\mathbb{E}[z_k; n, \boldsymbol{p}] = np_k \tag{2.17}$$

$$\mathbb{V}[z_k; n, \boldsymbol{p}] = np_k(1 - p_k) \tag{2.18}$$

Proposition 2.1 (additivity and decomposition)**.** *Given a set of K Poisson distributed random variables and rates $Z_k \sim Poisson(\lambda_k)$, the random variable $Y := \sum_{k=1}^K Z_k$ is Poisson distributed as $Y \sim Poisson(\lambda := \sum_{k=1}^K \lambda_k)$. The converse is also true, meaning that if we can decompose $Y = \sum_{k=1}^K Z_k \sim Poisson(\lambda)$ into individual separate counts, each summand Z_k will be Poisson distributed with rates λ_k that should sum to λ.*

Proof. Raikov (1938) □

Proposition 2.2. *Given a set of K Poisson distributed random variables and rates $Z_k \sim Poisson(\lambda_k)$ and the random variable $Y := \sum_{k=1}^K Z_k \sim Poisson(\lambda)$, with $\lambda = \sum_{k=1}^K \lambda_k$, the conditional distribution of the random vector $\boldsymbol{z} := [Z_1, \dots, Z_K]$ given the their sum Y is*

$$\boldsymbol{z}|Y \sim Mult(Y, \boldsymbol{p})$$
$$with\ p_k = \frac{\lambda_k}{\sum_{k=1}^K \lambda_k} = \frac{\lambda_k}{\lambda}$$

Proof. The joint probability can be written as

$$\prod_{k=1}^K \lambda_k^{z_k} \frac{e^{-\lambda_k}}{z_k!} \mathbb{1}\{\sum_{k=1}^K z_k = n\}$$

and conditioning on the sum with the probability given by $\lambda^n \frac{e^{-\lambda}}{n!}$ and reorganizing the terms we obtain

$$n! \prod_{k=1}^K \left(\frac{\lambda_k}{\lambda}\right)^{z_k} \frac{1}{z_k!} \mathbb{1}\{\sum_{k=1}^K z_k = n\}$$

□

Furthermore, we can define complex models using Poisson counts as latent variables, while choosing other distributions for the observations. The family of compound Poisson models consists of a Poisson distributed latent count $N \sim \text{Poisson}(\lambda)$ and a sum $Y = \sum_{i=1}^{N} X_i$ over N independent random variables from a fixed distribution G, with $X_i \sim G$. If we assume a mean $\mathbb{E}[X_i] = \mu_G$ and variance $\mathbb{V}[X_i] = \sigma_G^2$, we can apply iterated formulas for mean and variance to obtain the marginal mean $\mathbb{E}[Y] = \mathbb{E}[N\mu_g] = \lambda\mu_G$ and variance $\mathbb{V}[Y] = \mathbb{E}[N\sigma_G^2] + \mathbb{V}[N\mu_g] = \lambda(\mu_G^2 + \sigma_G^2)$, which indicates over-dispersion, but with an additive and multiplicative terms relating the mean and variance, namely $\mathbb{V}[Y] = \mathbb{E}[Y]\mu_G + \lambda\sigma_G^2$. In the context of Poisson matrix factorization models (Basbug and Engelhardt, 2016; Simsekli et al., 2013; Gouvert et al., 2019), one family of models that has been incorporated in the compound Poisson model is the exponential dispersion model (EDM) (Jorgensen, 1987) family. This model family includes Normal, Poisson, Gamma, Inverse-Gamma, and many other discrete and continous distributions (see Table 1 in Basbug and Engelhardt (2016) and Table 1 in Gouvert et al. (2019)). The EDM family also has the additivity property, which is convenient when building compound Poisson models, allowing the latent Poisson counts to be incorporated in the parameters of the EDM model, formally if $N \sim \text{Poisson}(\lambda)$ and $Y = \sum_{n=1}^{N} Z_n$, with $Z_n \sim \text{ED}(w, \kappa)$, then $Y \sim \text{ED}(w, N\kappa)$.

Definition 2.5 (exponential dispersion model)**.** *The random variable $Y \sim ED(w, \kappa)$ is sampled from an exponential dispersion model distribution, with natural parameter w, dispersion parameter $\kappa > 0$, log-partition $\psi(w)$ and base measure $h(Y, \kappa)$. The probability density function is given by*[10]

$$p(Y) = ED(Y; w, \kappa) = exp(Yw - \kappa\psi(w))h(Y, \kappa)$$

The expected value and variance is given by[11]

$$\mathbb{E}[Y_{ij}; w, \kappa] = \kappa\psi'(w) \tag{2.19}$$
$$\mathbb{V}[Y_{ij}; w, \kappa] = \kappa\psi''(w) \tag{2.20}$$

Proposition 2.3 (additivity)**.** *Given a set of EDM distributed random variables $Z_n \sim ED(w, \kappa)$, with $n \in \mathbb{N}$ and $n \leq N$, the random variable $Y := \sum_{n=1}^{N} Z_n$ is EDM distributed as $Y \sim ED(w, N\kappa)$.*

Proof. Jorgensen (1987) □

[10]The dependency on the log-partition function ψ and base measure function h is left implicit, since it is defined for each specific distribution that is part of the EDM family.

[11]We denote $\psi'(w) = \frac{d\psi}{dw}$, $\psi''(w) = \frac{d^2\psi}{dw^2}$

In this section we presented the main properties of Poisson and compound Poisson models that are of interest in the development of the models used in this thesis. We establish definitions and properties that will be used in development of new models, in the case of Poisson models, and the analysis of those models, in the case of compound Poisson models. For a broader treatment of the topic of models for count data, covering different distributions, over-dispersion, zero-inflation and many other topics, the reader can consult the reference works of Hilbe (2014) and Zelterman (2004).

2.2.2 Temporal point processes

Temporal point processes (TPP) are stochastic processes consisting of events realized over time. It can be seen as a generalization of count models with a temporal dimension, sometimes denoted *counting processes*. A realization of the TPP consists in a set of positive continuous numbers $t_n \in \mathbb{R}^+$, representing the times that a set of events occured (Aalen et al., 2008; Farajtabar, 2018). We can define the *history of events* until time t, given by the set of events up but not including the time t, formally given by $\mathcal{H}(t) := \{t_1, t_2, \dots, t_N\}$, with $t_n < t$ and $t_i \leq t_j$ if $i \leq j$. Given the history of events $\mathcal{H}(t)$ we can define the count process $N(t)$ given by the number of events that happened before time t defined as $N(t) := \sum_{t_i \in \mathcal{H}(t)} \mathbb{1}\{t_i < t\}$. Furthermore, considering the instantaneous increment of the count process at a time t in terms of events in an infinitesimal interval $[t, t+dt)$, which is given by $dN(t) = N(t+dt) - N(t)$ and $dN(t) \in \{0, 1\}$. The *conditional intensity*[12] $\lambda^*(t)$ is the conditional probability that a count event happens in the interval $[t, t+dt)$ divided by the length of the interval, expressing a measure of the rate of events per unit of time, in formal terms $\lambda^*(t)dt = P(dN(t) = 1|\mathcal{H}(t)) = \mathbb{E}[dN(t)|\mathcal{H}(t)]$. The likelihood of a temporal point process is fully determined by the conditional intensity and the history of events $\mathcal{H}(t)$ (Proposition 2.4).

Proposition 2.4. *Given a temporal point process with a history of events $\mathcal{H}(t) = \{t_1, t_2, \dots, t_N\}$, an observation interval $[0, t)$ with $t > t_N$ and with a defined conditional intensity function $\lambda^*(t)$. The likelihood function of the process is given by:*

$$L = \prod_{t_i \in \mathcal{H}(t)} \lambda^*(t_i)\, exp(-\int_0^t \lambda^*(s)ds) \tag{2.21}$$

Proof. Rasmussen (2018) □

[12]We use the convention $\lambda^*(t) := \lambda(t|\mathcal{H}(t))$, to indicate the dependency on the history of events.

The idea of the conditional intensity function as the average rate of events over time, and the fact that lilelihood is fully specified by it, motivates a modeling approach that focus on specifying the conditional intensity function. In this sense, the Poisson process is a TPP where the number of events that happens over a given time period is Poisson distributed with a rate that depends only on the said time interval with a conditional intensity function $\lambda^*(t) = \lambda(t)$ without any dependence with the history of events. The conditional intensity function of the *homogenous* Poisson process is a constant $\lambda^*(t) = \lambda \geq 0$, while the one for the *inhomogenous* Poisson process is a non-negative arbitrary function of the time $\lambda^*(t) = g(t) \geq 0$ (but not of the history) (Aalen et al., 2008).

The Hawkes process (Hawkes, 1971; Liniger, 2009) is defined as a process where each individual event in the event history $\mathcal{H}(t)$ contributes to increase the probability of future events with a probability that decreases with time, leading to a model with the property of clustered random events in time. Formally it is defined by the conditional intensity function $\lambda^*(t) = \mu + \alpha \sum_{t_i \in \mathcal{H}(t)} k_w(t, t_i)$, with $\mu \geq 0$ being a base-rate unaffected by time, a weighting factor $\alpha \geq 0$, and the kernel $k_w(t, t_i) \geq 0$ for each past event. A common choice for the kernel is the exponential kernel defined as $k_w(t, t_i) := \exp(-w(t - t_i))$.

Furthermore, recent works such as Mei and Eisner (2017), Du et al. (2016a), Jing and Smola (2017) have explored the conditional intensity modeling together with neural networks. A common approach in this case is to allow for generic conditional intensity functions, having them parameterized by a neural network, or in the case of Hawkes process, having the kernel function as neural network. Formally we define a model with $\lambda^*(t) = \mathrm{NN}_{\boldsymbol{W}}(t, \mathcal{H}(t))$, where $\mathrm{NN}_{\boldsymbol{W}}$ is a neural network with a parameters matrix $\boldsymbol{W}$, and with time t and history $\mathcal{H}(t)$ as inputs. In Chapter 5 we will develop this ideas in the context of a joint temporal point process model and a hierarchical recurrent neural network session-based recommender system.

For further details about temporal point process and their applications, the reader is referred to Aalen et al. (2008), Gomez-Rodriguez et al. (2013) and Farajtabar (2018).

2.3 Recommender Systems and Personalization: models and definitions

The basic setup of the recommender system problem consists in a scenario where a large group of users are interested and interacting (viewing, clicking, purchasing, consuming, reading) with a large set of items. We assume that there are certain items of interest for the users, but there is uncertainty about which items are those, introduced by the large amount of items, as well as latent preferences of the users.

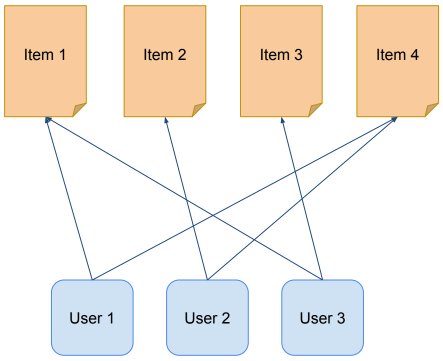


FIGURE 2.7: User and items interactions diagram

We assume that the individual user preferences are not known a priori, but they may be inferred by the patterns and contexts of the interactions. The decision problem from the system point of view is which items should be presented to the users, or how the items should be ranked in their presentation to the user, taking into account that the ranking of the presentation will influence user engagement and impact the user satisfaction with the system. One way to represent this problem computationally is by using a bipartite graph of user–items interactions, as it is represented Figure 2.7, which can be also represented as a user–item interaction matrix.

Problem definition. Given a user-set $\mathcal{U}$ of finite size $|\mathcal{U}|$, an item-set $\mathcal{I}$ of finite size $|\mathcal{I}|$, a user–item interaction matrix $\boldsymbol{R} = \{R_{ui}\} \in \mathbb{N}^{|\mathcal{U}|\times|\mathcal{I}|}$ and a training dataset consisting of observations of user–item interactions $\mathcal{D}_{\text{train}} := \{(u, d, R_{ui})\}$, with $|\mathcal{D}_{\text{train}}| = N_{\text{obs}} \ll |\mathcal{U}| \times |\mathcal{I}|$ we aim to learn a function f that estimates the value of each user–item interactions for all pairs of user and items $\mathcal{R}_{\text{complete}} := \{(u, d, f(u, d))\}$. In general to solve this problem we assume that users have a set of preferences, and (using matrix factorization) we model these preferences using latent vectors, using it to rank unseen items to the users. Furthermore, we might assume a collection of contextual information matrices $\boldsymbol{R}_{\mathcal{C}} = \{\boldsymbol{R}_c\}_{c\in[|\mathcal{C}|]}$, with shared rows or columns with the user–item interaction matrix and extra contextual information (for example, user–location or item–term matrices).

Probabilistic Matrix Factorization. The prototytical choice for a probabilistic model for recommender system is the Probabilistic Matrix Factorization (Salakhutdi-

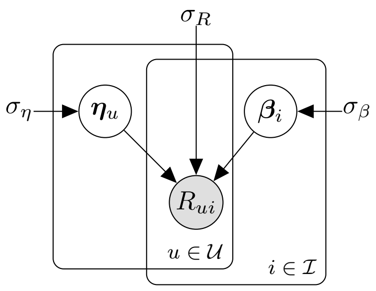


Figure 2.8: Diagram of Probabilistic Matrix Factorization model, with the user–item data matrix $\boldsymbol{R}$, latent vectors $\boldsymbol{\eta}_u$ and $\boldsymbol{\beta}_i$ for each user $u \in \mathcal{U}$ and item $i \in \mathcal{I}$

nov and Mnih, 2007) model that factorizes a sparse matrix of user–item interactions and uses the learned factors to predict unseen relationships, for example, using the factors to rank unseen items for each user. The hierarchical structure of the model is represented in Figure 2.8, and the complete specification of the model uses Gaussian distributions for the latent variables and likelihood. The following equations represent the generative model for PMF expressed in terms of observation variables for each observed entry of the user–item interaction matrix $\boldsymbol{R}$, corresponding latent K-dimensional vectors for each user $u \in \mathcal{U}$ and item $i \in \mathcal{I}$, using $\boldsymbol{I}_K$ to represent the identity matrix of size $K \times K$, $\mathbf{0}$ for the zero-vector and function $g : \mathbb{R} \to \mathcal{X}$, adjusting the values resulting from the inner product of the latent variables into the range of values of the observations[13].

$$\begin{aligned}
\boldsymbol{\eta}_u &\sim \mathcal{N}(\mathbf{0}, \sigma_\eta^2 \boldsymbol{I}_K) \\
\boldsymbol{\beta}_i &\sim \mathcal{N}(\mathbf{0}, \sigma_\beta^2 \boldsymbol{I}_K) \\
R_{ui} &\sim \mathcal{N}(g(\boldsymbol{\eta}_u^\top \boldsymbol{\beta}_i); \sigma_R^2)
\end{aligned}$$

We can express the joint likelihood of the observed entries adding an auxiliary indicator variable[14] $I_{ui} = 1$ for the observed user–item pairs, and $I_{ui} = 0$ otherwise,

[13]The function g was introduced because the inner product of the latent variables can have any value in the real line, while the observations typically would be value in a small range of numbers, such as counts, rating or a normalized metric.

[14]This indicator variable is used in a way that the non-observed variables will all have the same impact on the joint, namely $p(R_{ui}|g(\boldsymbol{\eta}_u^\top \boldsymbol{\beta}_i)) = 1$ if R_{ui} is not observed.

resulting in

$$p(\boldsymbol{R}, \boldsymbol{\eta}, \boldsymbol{\beta}; \sigma_\eta, \sigma_\beta, \sigma_R) = \prod_{u,i} \mathcal{N}(R_{ui}|g(\boldsymbol{\eta}_u^\top \boldsymbol{\beta}_i); \sigma_R^2)^{I_{ui}} \mathcal{N}(\boldsymbol{\eta}_u; \mathbf{0}, \sigma_\eta^2 \boldsymbol{I}_K) \mathcal{N}(\boldsymbol{\eta}_u; \mathbf{0}, \sigma_\beta^2 \boldsymbol{I}_K)$$

We can apply Bayes' theorem and calculate the posterior, which results in

$$p(\boldsymbol{\eta}, \boldsymbol{\beta}|\boldsymbol{R}) = \frac{\prod \mathcal{N}(R_{ui}|g(\boldsymbol{\eta}_u^\top \boldsymbol{\beta}_i); \sigma_R^2)^{I_{ui}} \mathcal{N}(\boldsymbol{\eta}_u; \mathbf{0}, \sigma_\eta^2 \boldsymbol{I}_K) \mathcal{N}(\boldsymbol{\eta}_u; \mathbf{0}, \sigma_\beta^2 \boldsymbol{I}_K)}{\int \prod \mathcal{N}(R_{ui}|g(\boldsymbol{\eta}_u^\top \boldsymbol{\beta}_i); \sigma_R^2)^{I_{ui}} \mathcal{N}(\boldsymbol{\eta}_u; \mathbf{0}, \sigma_\eta^2 \boldsymbol{I}_K) \mathcal{N}(\boldsymbol{\eta}_u; \mathbf{0}, \sigma_\beta^2 \boldsymbol{I}_K) d\boldsymbol{\eta}_u d\boldsymbol{\beta}_i}$$

Given the intractability of this computation, we can resort to the strategy of computing only point estimates of the posterior, in particular seeking the maximize the posterior we can obtain the MAP point estimates. Maximizing the posterior is equivalent to maximizing the log of the posterior, and ignoring terms that are do not affect the optimization results in:

$$\log p(\boldsymbol{\eta}, \boldsymbol{\beta}|\boldsymbol{R}) \propto -\sigma_R^{-2} \sum_{u,i} I_{ui}(R_{ui} - g(\boldsymbol{\eta}_u^\top \boldsymbol{\beta}_i))^2 - \sigma_\eta^{-2} \sum_u \boldsymbol{\eta}_u^\top \boldsymbol{\eta}_u - \sigma_\beta^{-2} \sum_i \boldsymbol{\beta}_i^\top \boldsymbol{\beta}_i$$

Exchanging the minus terms, we can then create a loss function proportional to $-\log\, p(\boldsymbol{\eta}, \boldsymbol{\beta}|\boldsymbol{R})$, which results in the following unconstrained minimization problem with the solution $\boldsymbol{\eta}_{\text{MAP}}, \boldsymbol{\beta}_{\text{MAP}}$ corresponding to the MAP:

$$L(\boldsymbol{\eta}, \boldsymbol{\beta}) = \sigma_R^{-2} \sum_{u,i} I_{ui}(R_{ui} - g(\boldsymbol{\eta}_u^\top \boldsymbol{\beta}_i))^2 + \sigma_\eta^{-2} \sum_u \boldsymbol{\eta}_u^\top \boldsymbol{\eta}_u + \sigma_\beta^{-2} \sum_i \boldsymbol{\beta}_i^\top \boldsymbol{\beta}_i$$

$$\boldsymbol{\eta}_{\text{MAP}}, \boldsymbol{\beta}_{\text{MAP}} = \underset{\boldsymbol{\eta}, \boldsymbol{\beta}}{\operatorname{argmin}}\, L(\boldsymbol{\eta}, \boldsymbol{\beta})$$

This optimization problem can be solved by gradient descent, either by explicitly calculating the partial derivatives with respect to each latent vector, or using Automatic Differentiation. We observe the correspondence between the MAP inference problem and the optimization approach applied in traditional matrix factorization (Koren et al., 2009) by comparing the resulting loss function, in both cases using a quadratic loss and L2 regularization terms for the latent-factors. This correspondence also highlights how the different modeling techniques can be used for imposing restrictions on the latent vectors. In probabilistic modeling any restrictions imposed on the latent vectors are to be expressed in terms of choice of prior distributions, for example, in this case the quadratic terms in the regularization are a direct result from the choice of Normal priors. If we had chosen for example, independent Laplace prior for each value of the prior, we would have obtained an L1 regularization. Similarly if a covariance structure Σ_η^{-1} were to be imposed on the latent vectors, this would appear in the regularization terms of the optimization problem in a quadratic form such as $\boldsymbol{\eta}_u^\top \boldsymbol{\Sigma}_\eta^{-1} \boldsymbol{\eta}_u$.

The standard technique assumes that the factors that explain the patterns of user–items interactions from the past are predictive for the future, which is valid in a static and non-contextual setting, but limited once we want to account for richers contextual information and dynamic changes. On top of that, typically it has been common to use Gaussian distributed variables for the observations, which fits poorly with the many real-life datasets consisting of count-data. Furthermore, since the user–item matrix is sparsily populated, with certain users without interactions with any items, this generates a problem for models such PMF, given that the respective latent factors will lack examples in the the dataset, known as the *cold start problem*. These challenges can be addressed by adding contextual information to the models, choices of likelihood that fits better the data and models customized for temporal dynamic data. In the following section we will discuss modeling techniques that address these challenges.

2.3.1 Poisson Factorization and Non-Negative Matrix Factorization

Poisson factorization. The basic Poisson factorization is a probabilistic model for non-negative matrix factorization based on the assumption that each user–item interaction R_{ui} can be modelled as a inner product of a user K dimensional latent vector $\boldsymbol{\eta}_u$ and item latent vector $\boldsymbol{\beta}_i$ representing the unobserved user preferences and item attributes (Gopalan et al., 2015), so that $R_{ui} \sim \text{Poisson}(\boldsymbol{\eta}_u^\top \boldsymbol{\beta}_i)$. Poisson factorization models for recommender systems have the advantage of principled modeling of implicit feedback, generating sparse latent representations, fast approximate inference with a sparse matrix and improved empirical results compared with the Gaussian-based models on count data interpreted as implicit feedback (Gopalan et al., 2014b, 2015). Nonparametric Poisson factorization model (BNPPF) (Gopalan et al., 2014b) extends basic Poisson factorization by drawing user weights from a *Gamma process*. The latent dimensionality in this model is estimated from the data, effectively avoiding the *ad hoc* process of choosing the latent space dimensionality K. Social Poisson factorization (SPF) (Chaney et al., 2015) extends basic Poisson factorization to accommodate preference and social based recommendations, adding a degree of trust variable and making all user–item interaction conditionally dependent on the user friends. With collaborative topic Poisson factorization (CTPF) (Gopalan et al., 2014a), shared latent factors are utilized to fuse recommendations with topic models using Poisson likelihood and Gamma variables for both.

Poisson matrix factorization. Poisson matrix factorization (PMF) (Cemgil, 2009; Gopalan et al., 2014a) with latent dimensionality K specifies a generative model for a count matrix $\mathbf{R} = \{R_{ui}\} \in \mathbb{N}^{N \times M}$, with each entry R_{ui} following a Poisson distribution with rate $\boldsymbol{\eta_u}^\top \boldsymbol{\beta_i}$, a sum-product of latent factors η_{uk} indexed by the rows and β_{ik} indexed by the columns. Each latent variable follows a prior Gamma(a, b).

$$\begin{aligned} &\eta_{uk} \sim \text{Gamma}(a_\eta, b_\eta), \quad \beta_{ik} \sim \text{Gamma}(a_\beta, b_\beta), \\ &R_{ui} \sim \text{Poisson}\left(\sum_{k=1}^{K} \eta_{uk}\beta_{ik}\right). \end{aligned} \tag{2.22}$$

The majority of the PMF literature assumes the priors to be gamma distributions using shape-rate parameterization, meaning that $\mathbb{E}[\eta_{uk}] = \frac{a_\eta}{b_\eta}$, $\mathbb{V}[\eta_{uk}] = \frac{a_\eta}{b_\eta^2}$. The choice of gamma priors leads to efficient variational inference equations given the conjugancy of gamma and poisson distributions. More generally we could choose different distributions as long as the resulting factor product $\boldsymbol{\eta_u}^\top \boldsymbol{\beta_i}$ is non-negative. Figure 2.9 presents the plate diagram for the PMF model.

Compound Poisson matrix factorization. Compound Poisson matrix factorization (CPMF) (Basbug and Engelhardt, 2016) extends PMF by incorporating an additive exponential dispersion model (EDM) (Jorgensen, 1987) in the observation model, while keeping the Poisson-Gamma factorization structure:

$$\begin{aligned} &\eta_{uk} \sim \text{Gamma}(a_\eta, b_\eta), \quad \beta_{ik} \sim \text{Gamma}(a_\beta, b_\beta) \\ &R_{ui} \sim \text{ED}(w, \kappa N_{ui}), \quad N_{ui} \sim \text{Poisson}(\sum_{k=1}^{K} \eta_{uk}\beta_{ik}), \end{aligned} \tag{2.23}$$

This results in an expected value $\mathbb{E}[R_{ij}|N_{ui}; w, \kappa] = \kappa N_{ui}\psi'(w)$ and variance $\mathbb{V}[R_{ui}; w, \kappa N_{ui}] = \kappa N_{ui}\psi''(w)$. The observation model of R_{ui} is a EDM distribution compounded using N_{ui} latent counts from a Poisson distribution. The EDM distribution is parameterized by the natural parameter w, dispersion κN_{ui}, and with a specific compounded distribution that is determined by the base log-partition function $\psi(w)$ and base-measure $h(R_{ui}, \kappa N_{ui})$. This model family includes Normal, Poisson, Gamma, Inverse-Gamma, and many other distributions (see Table 1 in Basbug and Engelhardt (2016) and Table 1 in Gouvert et al. (2019)), determined by different formulae for the log-partition and base-measure. Figure 2.10 presents the plate diagram for the compound PMF model.

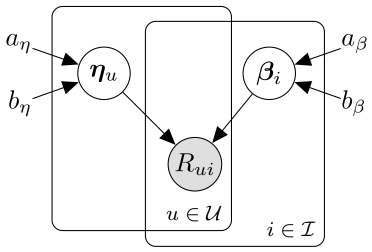


Figure 2.9: Diagram for Poisson Factorization

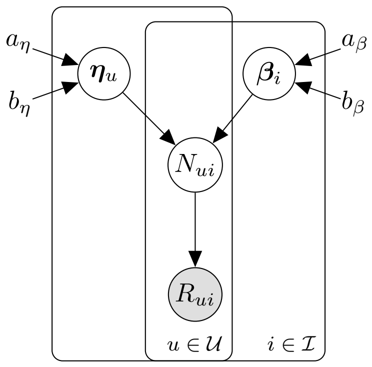


Figure 2.10: Diagram of Compound Poisson Factorization

Some early developments of Poisson factorization for recommender systems, text analysis and clustering include the following works:

- Ma et al. (2011a) proposes a Poisson-Gamma MF and Collective MF model for the problem of website recommendation. They develop only MAP inference, obtaining point-estimates for the latent-factors of the models and compare with baselines such as SVD, GaP (Canny, 2004), NMF, PMF, and others.

- Canny (2004) proposes GaP model for text mining applications, the model consider a document as a mixture of a loading factor matrix and a gamma-distributed latent vector, and a Poisson likelihood. It is in a fact a Poisson-Gamma fatorization model, but using a fixed mixture load matrix.

- Buntine and Jakulin (2005) Discrete Component Analysis, generalizes multiple latent-factor models that are applied to textual analysis into a unified modeling perspective and language, this includes GaP, LDA, and LSA.

2.3.2 Relations, matrices and tensors

As a general setting for the modeling tasks in this thesis, we start by assuming that there is a set with size $|\mathcal{E}|$ of entity groups $\mathcal{E} = \{\xi^i\}_{i \in [|\mathcal{E}|]}$, with each entity having a discrete (and usually finite) set of elements, and a set of labeled relations $\mathcal{R} = \{\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \ldots\}$ between elements of different entity groups, representing interaction data between those entities, for example in matrix or tensor form. The elements of each entity group are indexed such that we can distinghish objects within a group and consistently connect different groups of interest (via the relations matrix or tensors) for a given task.

In Figure 2.11 we present an example with *dyadic relations*[15], the set of entity types has size 4 and we could name them as: users ξ^1, documents ξ^2, words ξ^3 and locations ξ^4; with a set of of labeled interactions with the same size, consisting in user–item clicks $\boldsymbol{A}$, and item–words counts $\boldsymbol{B}$ (representing content of text documents associated with the item), item–location $\boldsymbol{C}$ and user–location $\boldsymbol{D}$ binary associations. This abstract *relational model* has a corresponding dataset consisting of matrices with count or binary data. Furthermore, we can also represent this abstract schematics as a bipartite directed graph $G = (\{\mathcal{E}, \mathcal{R}\}, E)$, with every directed edge being an arrow from an entity to a relation $\xi \to R$, with $\xi \in \mathcal{E}$ and $R \in \mathcal{R}$, and the additional constraint that every relation has at least two entities connected to it. This representation brings us closer to a Bayesian network, but to obtain a full probabilistic model of the data we need to add more assumptions, for example latent-factors for each entity, and probability distributions for the latent-factors and observations. Combining this relational perspective of multi-modal data with latent variable modeling is of central importance when designing the models on this thesis, and it serves the purpose of formalizing a procedure to generate novel models for contextualized recommender system, as well as other complex tasks involving multiple interrelated sources of data.

At this point we have only modeled the interconnection between different entities of interest, and described a diagramatic approach that can be seen as generating a skeleton or incomplete description of a probabilistic model for relational data. Nevertheless, it is a starting point, and can be used already to identify limitation

[15]Relations between two entity groups. In general, we can have relations with n-adicity (or arity), meaning between n entity groups that could be represented by a tensor

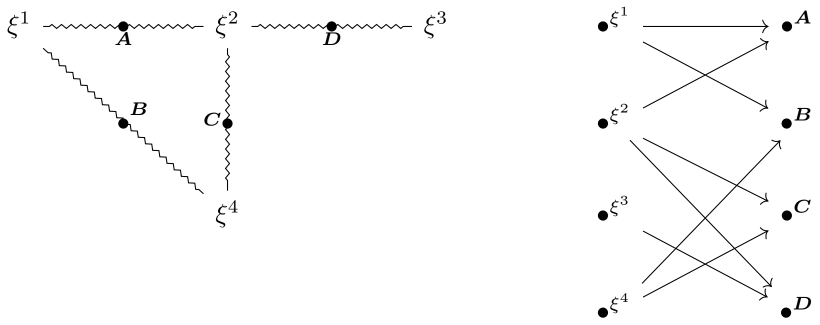


Figure 2.11: Example diagram of entities interconnected and the respective labeled relation (that can be represented as matrices) as connections

or mistakes in the design of the model. The next steps, consisting on adding latent-factors and distribution assumptions, will be discussed in the next section.

2.3.3 Collective Matrix and Tensor Factorization

In general the strategy for adding contextual information is to utilize shared latent variable between different matrix or tensors representing contextual information on the model. For example, it could mean adding a shared latent variable between a context–specific observation matrix and the user–preference model. Other context–specific constraints can be included in the model via prior distributions, for example defining the support of the latent variables (reals, integers or non-negative), or inducing grouping or correlation on the latent space.

For example, one approach to integrate location and textual side information (*url* texts and queries) is to create a spatial grid over the location space and model the user–location–text (user u at location l using text w) interaction counts as a combination of latent Gamma variables and Poisson likelihood $C_{u,l,w} \sim \text{Poisson}(\sum_k U_{uk} L_{lk} W_{wk})$, where $U_{uk}, L_{lk}, W_{wk} \sim \text{Gamma}(a,b)$. To enforce the thematic coherence between the locations by constraining the values of the latent factors, we can include information about the location type c (e.g., retail, food court, navigational) of location l, i.e., we create a variable $T_{l,c} = 1$ if location l is of type c, and model this relationship $T_{l,c} = \mathbb{1}\{X_{l,c} \geq 1\}$, with $X_{l,c} \sim \text{Poisson}(\sum_k C_{ck} L_{lk})$, and $L_{lk}, C_{wk} \sim \text{Gamma}(a,b)$. The location-type relationship will induce the latent location factors of the same type to have similar values. Finally, we can

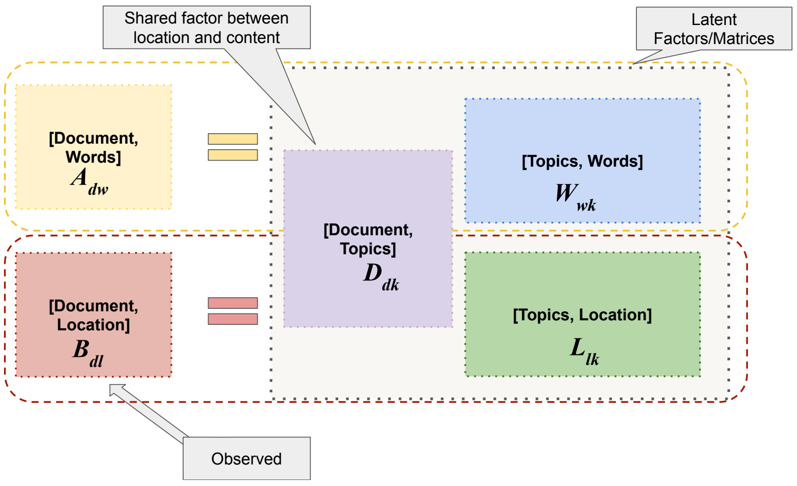

Shared factor between location and content
Latent Factors/Matrices
[Document, Words] A_{dw}
[Topics, Words] W_{wk}
[Document, Topics] D_{dk}
[Document, Location] B_{dl}
[Topics, Location] L_{lk}
Observed

FIGURE 2.12: Example of joint factorization of a document-word matrix and a document-location matrix, allowing for joint learning of aligned distributions of topics for document, words and locations

model the relationship user–location–time–url using a similar approach: $R_{u,l,t,h} \sim \text{Poisson}(\sum_k U_{uk} L_{lk} T_{tk} H_{hk})$, also with Gamma latent factors (and shared latent factors with the other contextual relationships). Notice that the latent gamma factors L_{lk} and U_{uk} are shared between other contextual relationships, which means that we are creating a particular instance of collective matrix-tensor factorization model, such as the example represented by Figure 3.1.

This design pattern of sharing latent variables between contexts is generic and we will use it in different models developed in this thesis. Similarly, it can be seen as emerging from the relational structure of the data or problem that we modeling, with the general rule that if an entity is connected to different relations, then a shared latent variable will influence both relations. With this insight we can move from a relational schema such as Figure 2.11 to a fully specified generative probabilistic model of the data.

2.4 Hyperparameters and prior specification

In any probabilistic modeling there are two types of unknown variables, random variables and hyperparameters. We can directly apply inference methods for the random variables, calculating for example the posterior distribution given the observations. In the case of the hyperparameters, being non-random quantities affecting overall properties of the model, it is not possible to use the same techniques, given that there is no distributional assumptions for them. The hyperparameters can be part of the prior probability of the model, the likelihood, or the specification of how the latent variables interact which each other. For example, in factorization models with latent vectors $\boldsymbol{\theta}$ and $\boldsymbol{\beta}$, interacting via inner product $\sum_{k=1}^{K} \theta_k \beta_k$, we have the dimensionality K of the latent space which is set a priori and affects the overall behavior of the model, as well as hyperparameters λ_θ and λ_β affecting the shape and location of the priors for the latent variables (assuming we have a prior distribution parameterized by shape and location). Estimating the impact of the hyperparameters of the model in the behavior of the model is non-trivial, specially in hierarchical models with multiple interacting latent-variables. Many methods attempting to solve this problems would rely on search strategies and evaluation of the posterior predictive distribution in partitions or samples of the dataset. The search can be automated with Bayesian optimization (Snoek et al., 2012), although it can turn into a costly method to apply. It is typically based on some proxy of the marginal likelihood, such as variational lower bound or leave-one-out cross validation (Vehtari et al., 2017), or directly on the performance in a downstream task, such as recommendation (Galuzzi et al., 2019).

Nevertheless, other strategies exist in the literature of traditional Bayesian analyis, and some of them, despite not giving the optimal setting according the the posterior predictive distribution, are able to help the practioner in choosing reasonable values for the hyperparameters. In particular we focus on the technique of Prior Predictive Checks (PPC), which consists of a collection of checks that are employed in the model before fitting it to any data. The rationale for this approach is that the specification of a probabilistic model allows for simulation of data, using samples generated from the model while marginalizing the latent variables. Thus, one can (manually) evaluate summaries of this simulated dataset before fitting any data, and validate certain prior assumptions. The advantage of following this approach is that forward simulation of a generative model is typically a cheap computational task, and we can span this simulations for multiple possible settings of the hyperparameters, and be able to aprioristically determine regions of the hyperaparamters configuration space that leads to observations that are not matching the experts opinions about the data, or some general idea for the expected summaries of the data. For example, imagine that we have a model for the volume of rain in a certain number of days, and if by the PPC we obtain observations that imply large volumes of rains in a period known for not having any rain (dry season), this would imply that the specified hyperparameters could be adjusted. The use of PPC in validating model assumptions is discussed in more details in the context of ideas of Bayesian workflow (Gabry et al., 2019; Schad et al., 2019; Gelman et al., 2020). Motivated by these ideas, in Chapter 6 we will introduce a new method that seeks to automatizes certain parts of PPC, obtaining both closed-form equations for Poisson-gamma factorization models and a generic algorithm for prior specifiction based on the prior predictive distribution.

Content-based social Poisson matrix factorization 3

"I've been working on three different ideas simultaneously, and strangely enough it seems a more productive method than sticking to one problem."

— Claude Shannon

Traditional recommender systems try to estimate a score function mapping each pair of user and item to a scalar value using the information of previous items already rated or interacted by the user (Adomavicius and Tuzhilin, 2005). Recent methods have been successful in integrating side information as content of the item, user context, social network, item topics, etc. For this purpose a variety of features should be taken into consideration, such as the routine, the geolocation, spatial correlation of certain preferences, mood and sentiment analysis, as well as social relationships such as "friendship" to others users or "belonging" to a community in a social network (Tang et al., 2013). In particular, a rich area of research has explored the integration of topic models and collaborative filtering approaches using principled probabilistic models (Wang and Blei, 2011; Gopalan et al., 2014a; Purushotham and Liu, 2012). Another group of models has been developed to integrate social network information into recommender systems using user–item ratings with extra dependencies (Chaney et al., 2015) or constraining and regularizing directly the user latent factors with social features (Ma et al., 2011b; Yuan et al., 2011). Finally, some models have focused on the collective learning of both social features and content features, constructing hybrid recommender systems (Purushotham and Liu, 2012; Kang and Lerman, 2013; Wang et al., 2013).

Our contribution is situated within all these three groups of efforts: we propose a probabilistic model that generalizes both previous models by jointly modeling content and social factors in the preference model applying Poisson-Gamma latent variable models to model the non-negativeness of the user–item ratings and induce sparse non-negative latent representation. Using this joint model we can generate recommendations based on the estimated score of non-observed items. We formulate the problem in the next paragraphs, in Section 3.1 we describe the proposed model, and in Section 3.2 present the variational inference algorithm, with the discussion the empirical results in Section 3.3. Our results indicate improved performance when compared to state-of-the-art methods including collaborative topic regression with social matrix factorization (CTR-SMF) (Purushotham and Liu, 2012).

Problem formulation Consider that given a set of observations of user–item interactions $R_{\text{train}} = \{(u, d, R_{ud})\}$, with $|R_{\text{train}}| = N_{\text{obs}} \ll U \times D$ (U is the number of users and D the number of documents), using additional item content information and user social network, we aim to learn a function f that estimates the value of each user–item interactions for all pairs of user and items $R_{\text{complete}} = \{(u, d, f(u, d))\}$. In general to solve this problem we assume that users have a set of preferences, and (using matrix factorization) we model these preferences using latent vectors.

Therefore, we have the documents (or items) set $\mathcal{D}$ of size $|\mathcal{D}| = D$, vocabulary set $\mathcal{V}$ of size $|\mathcal{V}| = V$, users set $\mathcal{U}$ of size $|\mathcal{U}| = U$, the social network given by the set of neighbors for each user $\{N(u)\}_{u\in\mathcal{U}}$. So, given the partially observed user–item matrix with integer ratings or implicit counts $\boldsymbol{R} = (R_{ud}) \in \mathbb{N}^{U\times D}$, the observed document–word count matrix $\boldsymbol{W} = (W_{dv}) \in \mathbb{N}^{D\times V}$, and the user social network $\{N(u)\}_{u\in\mathcal{U}}$, we need to estimate a matrix $\widetilde{\boldsymbol{R}} \in \mathbb{N}^{U\times D}$ to complete the user–item matrix $\boldsymbol{R}$. Finally, with the estimated matrix we can rank the unseen items for each user and make recommendations.

Related work

Collaborative Topic Regression (CTR): CTR (Wang and Blei, 2011) is a probabilistic model combining topic modeling (using Latent Dirichlet Allocation) and probabilistic matrix factorization (using Gaussian likelihood). Collaborative Topic Regression with Social Matrix Factorization (CTR-SMF) (Purushotham and Liu, 2012) builds upon CTR adding social matrix factorization, creating a joint model Gaussian factorization model with content and social side information. Limited Attention Collaborative Topic Regression (LA-CTR) (Kang and Lerman, 2013), is another approach with which the authors propose a joint model based on CTR integrating behavioral mechanism of attention. In this case, the amount of attention the user has invested in the social network is limited, and there is a measure of influence implying that the user may favor some friends more than others. In Wang et al. (2013), the authors propose a CTR model seamlessly integrated item–tags, item content and social network information. All the models mentioned above combine in some degree LDA with Gaussian based matrix factorization for recommendations. Thus the time complexity for training those models is dominated by LDA complexity, making them difficult to scale. Also, the combination of LDA and

Gaussian matrix factorization in CTR is a non-conjugate model that is hard to fit and difficult to work with sparse data.

Non-negative matrix and tensor factorization using Poisson models: Poisson models are also successfully utilized in more general models such as tensor factorization and relational learning, particularly where it can use count data and non-negative factors. In Hu et al. (2015), the authors propose a generic Bayesian non-negative tensor factorization model for count data and binary data. Furthermore, in Hu et al. (2016), the authors explore the idea of adding constraints between the model variables using side information with hierarchical information, while the approach in Acharya et al. (2015) uses graph side information jointly modeled with topic modeling with Gamma process – a joint non-parametric model of network and documents.

3.1 Poisson Matrix Factorization with Content and Social trust information (PoissonMF-CS)

The proposed model PoissonMF-CS (see Figure 3.2) is an extension and generalization of previous Poisson models, combining social factorization model (social Poisson factorization – SPF) (Chaney et al., 2015), and topic based factorization (collaborative topic Poisson factorization – CTPF) (Gopalan et al., 2014a).

The main idea is to employ shared latent Gamma factors for topical preference and trust weight variables in the user social network, combining all factors in the rate of a Poisson likelihood of the user–item interaction. We model both sources of information having an additive effect on the observed user–item interactions and add two global multiplicative weights for each group of latent factors. The intuition behind the additive effect of social trust is that users tend to interact with items presented by their peers, so we can imagine a mechanism of "peer pressure" operating, where items offered through the social network have a positive (or neutral) influence on the user. In other words, we believe there is a positive social bias more than an anti-social bias, and we factor this in PoissonMF-CS model.

In the case of Poisson models, this non-negative constraint results in sparseness in the latent factors and can help avoid over-fitting (in comparison the Gaussian-based models(Gopalan et al., 2015, 2014b)). Gamma priors on the latent factors, and the fact that the latent factors can only have a positive or a zero effect on the final prediction, induce sparse latent representations in the model. Hence, in the inference process we adjust a factor that decreases the model likelihood by making its value closer to zero.

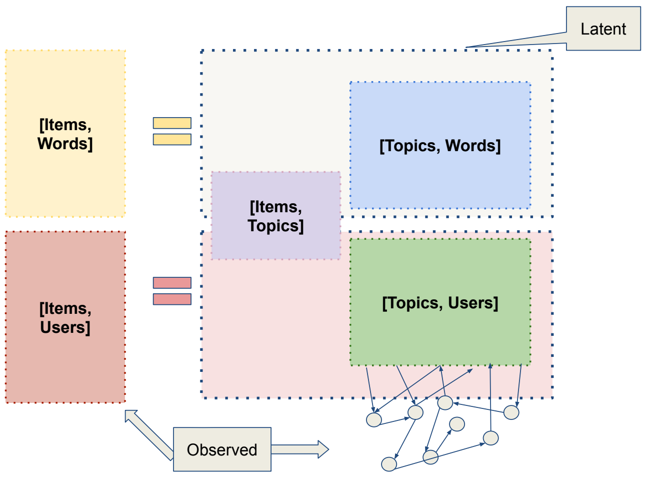


Figure 3.1: Diagram showing the view of PoissonMF-CS model as a joint factorization a user–item matrix and item–content matrix constrained by the user social network

3.1.1 Generative model

In this model, W_{dv} is a counting variable for the number of times word v appears in document d, $\boldsymbol{\beta}_v$ is a latent vector capturing topic distribution of word v and $\boldsymbol{\theta}_d$ is the document–topic intensity vector, both with dimensionality K. Count variable W_{dv} is parametrized by the linear combination of these two latent factors $\boldsymbol{\theta_d^\top \beta_v}$. The document–topic latent factor $\boldsymbol{\theta}_d$ influences also the user–document rating variable R_{ud}. Each user has a latent vector $\boldsymbol{\eta}_u$ representing the user–topic propensity, which interacts with the document topic intensity factor $\boldsymbol{\theta}_d$ and document topic offset factor $\boldsymbol{\epsilon}_d$, resulting in the term $\boldsymbol{\eta}_u^\top \boldsymbol{\theta}_d + \boldsymbol{\eta}_u^\top \boldsymbol{\epsilon}_d$. Here, $\boldsymbol{\eta}_u^\top \boldsymbol{\epsilon}_d$ captures the baseline matrix factorization, while $\boldsymbol{\eta}_u^\top \boldsymbol{\theta}_d$ connects the rating variable with the content-based part of the model (word–document variable W_{dv}). The trust factor τ_{ui} between user u to user i is equal to zero for all users that are not connected in the social network ($\tau_{ui} > 0 \Leftrightarrow i \in N(u)$). This trust factor adds dependency between social connected users: the user–document rating R_{ud} is influenced by the average rating to item d given by friends of user u in the social network, weighted by the trust user u assigns to his friends ($\sum_{i \in N(u)} \tau_{ui} R_{id}$). We model this social dependency using a conditional specified model, as in Chaney et al. (2015). The latent variables λ_C and λ_S are weight variables added in the model to capture and control the

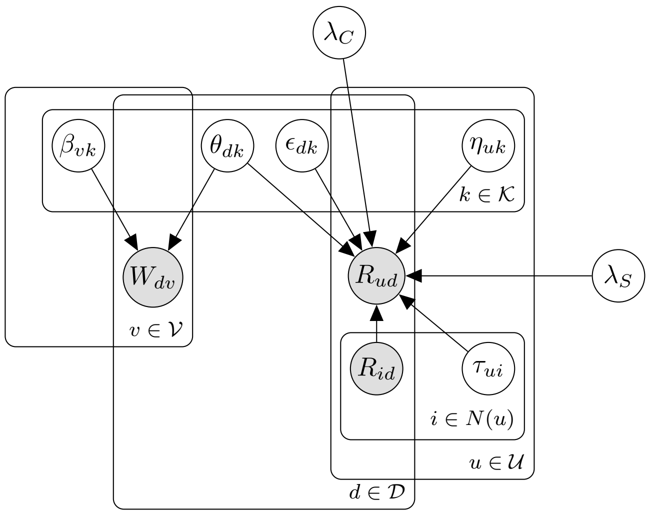


FIGURE 3.2: Plate diagram for PoissonMF-CS model

general weight of the content and social factors. These variables allow us to infer the importance of content and social factors according to the dataset or domain of usage. Also, instead of estimating these weights from the observed data, we may set λ_C and λ_S to constant values, thus controlling the importance of content and social parts of the model. Specifically if we set $\lambda_C = 0$ and $\lambda_S = 1$ we obtain the SPF model, while setting $\lambda_C = 1$ and $\lambda_S = 0$ result in CTPF, and $\lambda_C = 0$ and $\lambda_S = 0$ is equivalent to the simple Poisson matrix factorization without any side information (Gopalan et al., 2015).

Now we present the complete generative model assuming documents (or items) set $\mathcal{D}$ of size $|\mathcal{D}| = D$, vocabulary set $\mathcal{V}$ of size $|\mathcal{V}| = V$, users set $\mathcal{U}$ of size $|\mathcal{U}| = U$, the user social network given by the set of neighbors for each user $\{N(u)\}_{u \in \mathcal{U}}$ D documents, and K latent factors (topics) (with an index set $\mathcal{K}$).

1. Latent parameter distributions:

   a) for all topics $k \in \mathcal{K}$:

      - for all words $v \in \mathcal{V}$: $\beta_{vk} \sim \text{Gamma}(a^0_\beta, b^0_\beta)$
      - for all documents $d \in \mathcal{D}$: $\theta_{dk} \sim \text{Gamma}(a^0_\theta, b^0_\theta)$ and $\epsilon_{dk} \sim \text{Gamma}(a^0_\epsilon, b^0_\epsilon)$
      - for all users $u \in \mathcal{U}$: $\eta_{uk} \sim \text{Gamma}(a^0_\eta, b^0_\eta)$

- for all user $i \in N(u)$: $\tau_{ui} \sim \text{Gamma}(a^0_\tau, b^0_\tau)$

b) Content weight: $\lambda_C \sim \text{Gamma}(a^0_C, b^0_C)$

c) Social weight: $\lambda_S \sim \text{Gamma}(a^0_S, b^0_S)$

2. Observations probability distribution:

a) for all observed document–word pairs dv :

$$W_{dv}|\boldsymbol{\beta}_v, \boldsymbol{\theta}_d \sim \text{Poisson}(\boldsymbol{\beta_v}^\top \boldsymbol{\theta_d})$$

b) for all observed user–document pairs ud :

$$R_{ud}|\boldsymbol{R_{N(u),d}, \eta_u, \epsilon_d, \theta_d} \sim \text{Poisson}(\lambda_C \boldsymbol{\eta}_u^\top \boldsymbol{\theta}_d + \boldsymbol{\eta}_u^\top \boldsymbol{\epsilon}_d + \lambda_S \sum_{i \in N(u)} \tau_{ui} R_{id})$$

3.2 Inference

First, we add a set of auxiliary latent Poisson variables to facilitate the posterior inference of the model. By doing so, the extended model will be complete conjugate, and consequently have analytical equations for the complete conditionals and variational updates (Bishop, 2006). In Section 3.2.2 we show that those auxiliary variables can be seen as by-product of a lower bound on the expected value of the log sum of the latent random variables. Variable $Y_{dv,k}$ represent a topic specific latent count for a word–document pair, so that the observed word–document counts is a sum of the latent counts (a property of the Poisson distribution) [1]. We can perform a similar modification for the user–item counts, splitting the latent terms of R_{ud} rate into two groups of topic specific latent count allocation variables: $Z^M_{ud,k}$ for the item content part, $Z^N_{ud,k}$ for the collaborative filtering part and $Z^S_{ud,i}$ for the social trust part (for this part, the intuitive explanation for the latent dimension is the idea of friend specific allocation of trust). The sum of all those latent counts is

[1]The change consist in assigning a new Poisson variable to each sum-term in the latent rate of the Poisson likelihood, so if $S \sim \text{Poisson}(\sum_i X_i)$, we add variables $S_i \sim \text{Poisson}(X_i)$, and by the sum property of Poisson random variable $S = \sum_i S_i \sim \text{Poisson}(\sum_i X_i)$

the observed user–item interaction count variable R_{ud}.

$$\begin{aligned}
Y_{dv,k}|\beta_{vk},\theta_{dk} &\sim \text{Poisson}(\beta_{vk}\theta_{dk}) \\
Z^M_{ud,k}|\lambda_C,\eta_{uk},\theta_{dk} &\sim \text{Poisson}(\lambda_C\eta_{uk}\theta_{dk}) \\
Z^N_{ud,k}|\eta_{uk},\epsilon_{dk} &\sim \text{Poisson}(\eta_{uk}\epsilon_{dk}) \\
Z^S_{ud,i}|\lambda_S,\tau_{ui},R_{id} &\sim \text{Poisson}(\lambda_S\tau_{ui}R_{id})
\end{aligned} \tag{3.1}$$

$$\text{with} \sum_k Y_{dv,k} = W_{dv}, \text{ and} \sum_k Z^M_{ud,k} + Z^N_{ud,k} + \sum_{i\in N(U)} Z^S_{ud,i} = R_{ud}.$$

The inference problem consists on the estimation of the posterior distribution of the latent variables given the observed rating $\boldsymbol{R}$, the observed document–word counts $\boldsymbol{W}$, and the user social network $\{N(u)\}_{u\in\mathcal{U}}$, in other words, computing

$$p(\Theta|\boldsymbol{R},\boldsymbol{W},\{N(u)\}_{u\in\mathcal{U}}),$$

where $\Theta = \{\boldsymbol{\beta},\boldsymbol{\theta},\boldsymbol{\eta},\boldsymbol{\epsilon},\boldsymbol{\tau},\boldsymbol{y},\boldsymbol{z},\lambda_C,\lambda_S\}$ is the set of all latent variables. The exact computation of this posterior probability is intractable for any practical scenario, so we need approximation techniques for efficient parameter learning. In our case, we apply variational techniques to derive the learning algorithm. As an intermediate step towards the variational inference algorithm, we also derive the full conditional distribution for each latent variable. The full conditional distribution of each latent variable is also useful as update equations for Gibbs sampling, meaning that we could use the resulting equations to implement a sampling-based approximation. However, sampling methods are hard to scale and usually requires more memory, so as a design choice for the implementation of the learning algorithm, we refrained from applying the Gibbs sampling method and focus on the variational inference.

In the next paragraphs we present the full conditional distribution of each of the latent variables, and show the resulting update equation for the variational parameters in Section 4.2.0.1.

Full conditional distribution : the full conditional distribution of each of the latent variables is the distribution of a variable given all the other variables in the model, except the variable that we are considering. Given a set of indexed random variables X_k, we use the notation $p(X_k|*)$ (where $*$ means all the variables X_i such that $i \neq k$) to represent the full conditional distribution. Given the factorized structure of the model we can simplify the conditional set to the Markov blanket of the node we are considering (children nodes and co-parents nodes)[2] (Bishop, 2006). For conciseness, we show the derivations only for one Gamma latent variables and one Poisson latent count variable.

- **Gamma distributed variables:** We demonstrate how to obtain the full conditional distribution for Gamma distributed variable θ_{dk}, for the remaining Gamma distributed variables we only present the end result without the intermediate steps.

$$\begin{aligned}
p(\theta_{dk}|*) &= p(\theta_{dk}|\,\mathrm{mb}(\theta_{dk})) \\
&\propto p(\theta_{dk}) \textstyle\prod_{v=1}^{V} p(Y_{dv,k}|\beta_{vk},\theta_{dk}) \prod_{u=1}^{U} p(Z^M_{ud,k}|\lambda_C,\eta_{uk},\theta_{dk}) \\
&\propto \theta_{dk}^{a^0_\theta-1} e^{-b^0_\theta \theta_{dk}} \textstyle\prod_v \theta_{dk}^{Y_{dv,k}} e^{-\beta_{vk}\theta_{dk}} \prod_u \theta_{dk}^{Z^M_{ud,k}} e^{-\lambda_C \eta_{uk}\theta_{dk}} \\
&\propto \theta_{dk}^{a^0_\theta+\sum_v Y_{dv,k}+\sum_u Z^M_{ud,k}-1} e^{-\theta_{dk}(b^0_\theta+\sum_v \beta_{vk}+\lambda_C \sum_u \eta_{uk})}
\end{aligned} \tag{3.2}$$

Normalizing equation Eq. 3.2 over θ_{dk} we obtain the pdf of a Gamma variable with shape $a^0_\theta + \sum_v Y_{dv,k} + \sum_u Z^M_{ud,k}$ and rate $b^0_\theta + \sum_v \beta_{vk} + \lambda_C \sum_u \eta_{uk}$. The final solution is written in Eq. 3.3. Also, notice that because of the way the model is structured all other Gamma latent variable have similar equations, the difference being the set of variables in the Markov blanket.

$$\begin{aligned}
\theta_{dk}|* &\sim \mathrm{Gamma}(a^0_\theta + \textstyle\sum_v Y_{dv,k} + \sum_u Z^M_{ud,k}, b^0_\theta + \sum_v \beta_{vk} + \lambda_C \sum_u \eta_{uk}) \\
\beta_{vk}|* &\sim \mathrm{Gamma}(a^0_\beta + \textstyle\sum_d Y_{dv,k}, b^0_\beta + \sum_d \theta_{dk}) \\
\eta_{uk}|* &\sim \mathrm{Gamma}(a^0_\eta + \textstyle\sum_d Z^M_{ud,k} + Z^N_{ud,k}, b^0_\eta + \lambda_C \sum_d \theta_{dk} + \sum_d \epsilon_{dk}) \\
\epsilon_{dk}|* &\sim \mathrm{Gamma}(a^0_\epsilon + \textstyle\sum_u -Z^N_{ud,k}, b^0_\epsilon + \sum_u \eta_{uk}) \\
\tau_{ui}|* &\sim \mathrm{Gamma}(a^0_\tau + \textstyle\sum_d Z^S_{ud,i}, b^0_\tau + \lambda_S \sum_d R_{id}) \\
\lambda_C|* &\sim \mathrm{Gamma}(a_C + \textstyle\sum_{u,d,k} Z^M_{ud,k}, b_C + \sum_{u,d,k} \eta_{uk}\theta_{dk}) \\
\lambda_S|* &\sim \mathrm{Gamma}(a_S + \textstyle\sum_{u,d,i} Z^S_{ud,i}, b_S + \sum_{u,d,i} \tau_{ui} R_{id})
\end{aligned} \tag{3.3}$$

[2] We use the notation $\mathrm{mb}(X)$ to denote the Markov blanket of a variable X – the set of children, parents and co-parents nodes of variable X in the BN

- **Multinomial distributed (auxiliary) variables:** looking at the Markov blanket of $\boldsymbol{Y}_{dv}$ we obtain:

$$\begin{aligned} p(\boldsymbol{Y}_{dv}|*) &\propto \prod_{k=1}^{K} p(Y_{dv,k}|\beta_{vk},\theta_{dk}) = \prod_{k=1}^{K} \text{Poisson}(Y_{dv,k}|\beta_{vk}\theta_{dk}) \\ &\propto \prod_{k=1}^{K} \frac{(\beta_{vk}\theta_{dk})^{Y_{dv,k}}}{Y_{dv,k}!} \end{aligned} \tag{3.4}$$

Given that we know that $\sum_k Y_{dv,k} = W_{dv}$, this functional form is equivalent to the pdf of a Multinomial distribution with parameter probabilities proportional to $\beta_{vk}\theta_{dk}$.

$$\boldsymbol{Y}_{dv}|* \sim \text{Mult}(W_{dv}; \boldsymbol{\phi_{dv}}) \qquad \text{with } \phi_{dv,k} = \frac{\beta_{vk}\theta_{dk}}{\sum_k \beta_{vk}\theta_{dk}} \tag{3.5}$$

Similarly, $\boldsymbol{Z}_{ud}$ is a Multinomial with parameters proportional to the parent nodes of $\boldsymbol{Z}_{ud}$. For convenience in the previous section, we split $\boldsymbol{Z}_{ud}$ in three blocks of variables and parameters $\boldsymbol{Z}_{ud} = [\boldsymbol{Z}_{ud}^{M}, \boldsymbol{Z}_{ud}^{N}, \boldsymbol{Z}_{ud}^{S}]$ representing the different high-level parts of our model. The dimensionality of the first two blocks is the K, while for the last block is U, resulting that $\boldsymbol{Z}_{ud}$ has dimensionality $2K + U$. Similarly the parameters of the $\boldsymbol{Z}_{ud}$ full conditional Multinomial have a block structure $\boldsymbol{\xi}_{ud} = [\boldsymbol{\xi}_{ud}^{M}, \boldsymbol{\xi}_{ud}^{N}, \boldsymbol{\xi}_{ud}^{S}]$.

$$\boldsymbol{Z}_{ud}|* \sim \text{Mult}(R_{ud}; \boldsymbol{\xi_{ud}})$$

$$\text{with } \xi_{ud,k} = \begin{cases} \xi_{ud,k}^{M} = \frac{\lambda_C \eta_{uk}\theta_{dk}}{\sum_k \eta_{uk}(\lambda_C\theta_{dk}+\epsilon_{dk})+\lambda_S \sum_{i\in N(u)} \tau_{ui}R_{id}} \\ \xi_{ud,k}^{N} = \frac{\eta_{uk}\epsilon_{dk}}{\sum_k \eta_{uk}(\lambda_C\theta_{dk}+\epsilon_{dk})+\lambda_S \sum_{i\in N(u)} \tau_{ui}R_{id}} \\ \xi_{ud,i}^{S} = \frac{\lambda_S \tau_{ui}R_{id}}{\sum_k \eta_{uk}(\lambda_C\theta_{dk}+\epsilon_{dk})+\lambda_S \sum_{i\in N(u)} \tau_{ui}R_{id}} \end{cases}$$

We present in next section how to use these equations to derive a deterministic optimization algorithm for approximate inference using the *variational* method.

3.2.1 Variational inference

Given a family of surrogate distributions $q(\Theta|\Psi)$ for the unobserved variables (latent terms) parametrized by variational parameters Ψ, we want to find an assignment of the variational parameters that minimize the KL-divergence between $q(\Theta|\Psi)$ and $p(\Theta|\boldsymbol{R}, \boldsymbol{W})$ [3],

$$\operatorname*{argmin}_{\Psi} \mathrm{KL}\{q(\Theta|\Psi), p(\Theta|\boldsymbol{R}, \boldsymbol{W})\}.$$

Then, the optimal surrogate distribution can be used as an approximation the true posterior. However, the optimization problem using directly the KL divergence is not tractable, since it depends on the computation of the evidence $\log p(\boldsymbol{R}, \boldsymbol{W})$. This can be accomplished using a surrogate objective that is lower bounds the evidence – the Evidence Lower BOund (ELBO):

$$\operatorname*{argmax}_{\Psi} L(\Psi) = \mathbb{E}_q[\log p(\boldsymbol{R}, \boldsymbol{W}, \Theta) - \log q(\Theta|\Psi)]$$

Another ingredient in this approximation is the mean field assumption. It consists in assuming that all variables in the variational distribution $q(\Theta|\Psi)$ are mutually independent. As a result the variational surrogate distribution can be expressed as a factorized distribution of each latent factor (Eq. 4.2). Another implication is that we can compute the updates for each variational X_i factor using the complete conditional of the latent factor (Bishop, 2006). Finally, the inference algorithm consists in iterative updating of variational parameters of each factorized distribution until convergence is reached, resulting in the *coordinate ascent variational inference* algorithm based on the following equation:

$$q(X_i) \propto \exp\{\mathbb{E}_q[\log p(X_i|*)]\} \tag{3.6}$$

Using Eq. 3.6, we can take each complete conditional variable that we described in the previous section and create a respective proposal distribution for the variational inference. This proposal distribution is in the same family as the full conditional distribution of the latent variables, meaning that we have a group of Gamma and Multinomial variables. As long as we update the parameters of the variational distribution using Eq. 3.6, it is guaranteed to minimize the KL divergence between the surrogate variational distribution (Eq. 4.2) over the latent variables and the

[3] To simplify the notation, we use the short-handed $p(\Theta|\boldsymbol{R}, \boldsymbol{W})$ to denote the posterior distribution $p(\Theta|\boldsymbol{R}, \boldsymbol{W}, \{N(u)\}_{u\in\mathcal{U}})$. Also, we drop the explicitly notation indicating the dependency on the social network

posterior distribution of the model.

$$\begin{aligned} q(\Theta|\Psi) \quad &= q(\lambda_C|a_{\lambda_C}, b_{\lambda_C}) q(\lambda_S|a_{\lambda_S}, b_{\lambda_S}) \prod_{u,k,i} q(\tau_{ui}|a_{\tau_{ui}}, b_{\tau_{ui}}) q(\eta_{uk}|a_{\eta_{uk}}, b_{\eta_{uk}}) \\ &\times \prod_{d,v,k} q(\epsilon_{dk}|a_{\epsilon_{dk}}, b_{\epsilon_{dk}}) q(\theta_{dk}|a_{\theta_{dk}}, b_{\theta_{dk}}) q(\beta_{vk}|a_{\beta_{vk}}, b_{\beta_{vk}}) \\ &\times \prod_{d,v,u} q(\boldsymbol{Z}_{dv}|\boldsymbol{\phi}^*_{dv}) q(\boldsymbol{Y}_{ud}|\boldsymbol{\xi}^{M*}_{ud}, \boldsymbol{\xi}^{N*}_{ud}, \boldsymbol{\xi}^{S*}_{ud}) \end{aligned} \tag{3.7}$$

After applying Eq. 3.6 together with the expected value properties for each latent variable[4], we obtain the following update equations for the variational parameters.

- **Content and social weights**:

$$\begin{aligned} a_{\lambda_C} &= a_C + \textstyle\sum_{u,d,k} R_{ud}\xi^{M*}_{ud,k}, & b_{\lambda_C} &= b_C + \textstyle\sum_{u,d,k} \frac{a_{\eta_{uk}}}{b_{\eta_{uk}}}\frac{a_{\theta_{dk}}}{b_{\theta_{dk}}} \\ a_{\lambda_S} &= a_S + \textstyle\sum_u R_{ud}\xi^{M*}_{ud,k} + \sum_v W_{dv}\phi^*_{dv,k}, & b_{\lambda_S} &= b_S + \textstyle\sum_{u,d,i} R_{id}\frac{a_{\tau_{ui}}}{b_{\tau_{ui}}} \end{aligned}$$

- **Content v (topic/tags/etc) parameters**:

$$a_{\beta_{vk}} = a^0_\beta + \textstyle\sum_d W_{dv}\phi^*_{dv,k}, \quad b_{\beta_{vk}} = b^0_\beta + \sum_d \frac{a_{\theta_{dk}}}{b_{\theta_{dk}}}$$

- **Item d parameters**:

$$\begin{aligned} a_{\epsilon_{dk}} &= a^0_\epsilon + \textstyle\sum_u R_{ud}\xi^{N*}_{ud,k}, & b_{\epsilon_{dk}} &= b^0_\epsilon + \textstyle\sum_u \frac{a_{\eta_{uk}}}{b_{\eta_{uk}}} \\ a_{\theta_{dk}} &= a^0_\theta + \textstyle\sum_u R_{ud}\xi^{M*}_{ud,k} + \sum_v W_{dv}\phi^*_{dv,k}, & b_{\theta_{dk}} &= b^0_\theta + \mathbb{E}_q[\lambda_C]\textstyle\sum_u \frac{a_{\eta_{uk}}}{b_{\eta_{uk}}} + \sum_v \frac{a_{\beta_{vk}}}{b_{\beta_{vk}}} \end{aligned}$$

- **User u parameters**:

$$\begin{aligned} a_{\eta_{uk}} &= a^0_\eta + \textstyle\sum_d R_{ud}(\xi^{M*}_{ud,k} + \xi^{N*}_{ud,k}), & b_{\eta_{uk}} &= b^0_\eta + \textstyle\sum_d \mathbb{E}_q[\lambda_C]\frac{a_{\theta_{dk}}}{b_{\theta_{dk}}} + \frac{a_{\epsilon_{dk}}}{b_{\epsilon_{dk}}} \\ a_{\tau_{ui}} &= a^0_\tau + \textstyle\sum_d R_{ud}\xi^{S*}_{ud,i}, & b_{\tau_{ui}} &= b^0_\tau + \mathbb{E}_q[\lambda_S]\textstyle\sum_d R_{id} \end{aligned}$$

[4]Notice that, if $q(X) = \text{Gamma}(X|a_X, b_X)$ (parameterized by shape and rate) , then $\mathbb{E}_q[X] = \frac{a_X}{b_X}$ and $\mathbb{E}_q[\log X] = \Psi(a_X) - \log(b_X)$, where $\Psi(.)$ is the Digamma function. If $q(\boldsymbol{X}) = \text{Mult}(R|\boldsymbol{p})$, then $\mathbb{E}_q[X_i] = Rp_i$.

- **item–content** dv **parameters**:

$$\phi^*_{dv,k} \propto \frac{e^{\Psi\left(a_{\beta_{vk}}\right)}}{b_{\beta_{vk}}} \frac{e^{\Psi\left(a_{\theta_{dk}}\right)}}{b_{\theta_{dk}}} \quad \text{with } \textstyle\sum_k \phi_{dv,k} = 1$$

- **user–item** ud **parameters**:

$$\begin{aligned} \xi^{M*}_{ud,k} &\propto e^{E_q[\log\lambda_C]} \frac{e^{\Psi\left(a_{\eta_{uk}}\right)}}{b_{\eta_{uk}}} \frac{e^{\Psi\left(a_{\theta_{dk}}\right)}}{b_{\theta_{dk}}} & \xi^{N*}_{ud,k} &\propto \frac{e^{\Psi\left(a_{\eta_{uk}}\right)}}{b_{\eta_{uk}}} \frac{e^{\Psi\left(a_{\epsilon_{dk}}\right)}}{b_{\epsilon_{dk}}} \\ \xi^{S*}_{ud,i} &\propto e^{E_q[\log\lambda_S]} \frac{e^{\Psi(a_{\tau_{ui}})}}{b_{\tau_{ui}}} R_{id} & \text{with} \quad & \textstyle\sum_k \xi^{M*}_{ud,k} + \xi^{N*}_{ud,k} + \sum_i \xi^{S*}_{ud,i} = 1 \end{aligned}$$

ELBO: The variational updates calculated in the previous sections are guaranteed to non-decrease the ELBO. However, we still need to calculate this lower bound after each iteration to evaluate a stopping condition for the optimization algorithm. We briefly describe a particular lower-bounding for the ELBO involving the log-sum present in the Poisson rate.

Note also that the surrogate distribution is factorized using the mean field assumptions (Eq. 4.2), so we have a sum of terms corresponding to the expected log probability over the surrogate distribution. The terms comprising the log-probabilities of the Poisson likelihood display a expected value over a sum of logarithms of latent variables (for example $\mathbb{E}_q[\log(\sum_k \beta_{vk}\theta_{dk})]$), this is a challenging computation, but we can apply another lower-bound[5] and simplify it to Eq. 3.8.

$$\begin{aligned} \mathbb{E}_q[\log(\textstyle\sum_k \beta_{vk}\theta_{dk})] &\geq \textstyle\sum_k \phi^*_{dv,k} \left(\mathbb{E}_q[\log\beta_{vk}] + \mathbb{E}_q[\log\theta_{dk}]\right) \\ &\quad - \textstyle\sum_k \phi^*_{dv,k} \log\phi^*_{dv,k} \end{aligned} \tag{3.8}$$

This same simplification can be done to all Poisson terms independently because of the mean field assumptions. It is equivalent to using the auxiliary latent counts. So, for example, using the latent variable $Z_{dv,k}$, β_{vk} and θ_{dk}, the Poisson term in the ELBO results in Eq. 3.9.

$$\begin{aligned} \mathbb{E}_q\left[\log\frac{p(Z_{dv})}{q(Z_{dv})}\right] &= \textstyle\sum_k W_{dv}\phi^*_{dv,k}\,\mathbb{E}_q[\log(\beta_{vk}\theta_{dk})] \\ &\quad - \mathbb{E}_q[\beta_{vk}\theta_{dk}] - W_{dv}\phi^*_{dv,k}\log(\phi^*_{dv,k}) - \log(W_{dv}!) \end{aligned} \tag{3.9}$$

[5]this lower bound is valid for any $\phi^*_{dv,k}$, with $\sum_k \phi^*_{dv,k} = 1$, check Eq. 3.12 in Section 3.2.2 for details

For the Gamma terms, the calculations are a direct application of ELBO formula for the appropriate variable. For example, Eq. 3.10 describes the resulting terms for β_{vk}.

$$\begin{aligned} \mathbb{E}_q\left[\log \frac{p(\beta_{vk})}{q(\beta_{vk})}\right] &= \log \frac{\Gamma(a_{\beta_{vk}})}{\Gamma(a)} + a \log b + a_{\beta_{vk}}(1 - \log b_{\beta_{vk}}) \\ &+(a - a_{\beta_{vk}})\,\mathbb{E}_q[\log \beta_{vk}] - b\,\mathbb{E}_q[\beta_{vk}] \end{aligned} \tag{3.10}$$

Recommendations: Once we learn the latent factors of the model from the observations we can infer the user preference over the set of items using the expected value of the user–item rating $\mathbb{E}[R_{ud}]$. The recommendation algorithm ranks the unobserved items for each user according to $\mathbb{E}[R_{ud}]$ and recommend to top-M items. We utilize the variational distribution to efficiently compute $\mathbb{E}[R_{ud}]$ as defined in Eq. 3.11. This value can be broken down into three non-negative scores: $\mathbb{E}_q[\eta_u]^\top \mathbb{E}_q[\epsilon_d]$, representing the "classic" collaborative filtering matching of users preferences and items features, $\mathbb{E}_q[\lambda_C]\,\mathbb{E}_q[\eta_u]^\top \mathbb{E}_q[\theta_d]$ representing the content factors contribution and $\mathbb{E}_q[\lambda_S] \sum_{i\in N(u)} \mathbb{E}_q[\tau_{ui}]R_{id}$ the social influence contribution.

$$\mathbb{E}[R_{ud}] \approx \mathbb{E}_q[\eta_u]^\top(\mathbb{E}_q[\lambda_C]\,\mathbb{E}_q[\theta_d] + \mathbb{E}_q[\epsilon_d]) + \mathbb{E}_q[\lambda_S] \sum_{i\in N(u)} \mathbb{E}_q[\tau_{ui}]R_{id} \tag{3.11}$$

Complexity and convergence: the complexity of each iteration of the variational inference algorithm is linear on the number of latent factors K, non-zero ratings nR, non-zero word-document counts nW, users U, items D, vocabulary set W and neighbors for each user nS, in other words $O(K(nW+nR+nS+U+D+W))$. We have shown that we can obtain closed-form updates for the inference algorithm, which stems from the fact that the model is fully conjugate and in the exponential family of distributions. In this setting variational inference is guaranteed to converge, and we observed in the experiments the algorithm converging after 20 to 40 iterations.

3.2.2 A lower bound for $\mathbb{E}_q[\log \sum_k X_k]$

The function $\log(\cdot)$ is a concave, meaning that:

$$\log(px_1 + (1-p)x_2) \geq p \log x_1 + (1-p) \log x_2$$

$$\forall p : p \geq 0$$

By induction this property can be generalized to any convex combination of x_k ($\sum_k p_k x_k$ with $\sum_k p_k = 1$ and $p_k \geq 0$): $\log \sum_k p_k x_k \geq \sum_k p_k \log x_k$ Now using random variables we can create a similar convex combination by multiplying and dividing each random variable X_k by $p_k > 0$ and apply the sum of of expectation property:

$$\begin{aligned}
\mathbb{E}_q[\log \textstyle\sum_k X_k] &= \mathbb{E}_q[\textstyle\sum_k \log p_k \frac{X_k}{p_k}] \\
\log \textstyle\sum_k p_k \frac{X_k}{p_k} &\geq \textstyle\sum_k p_k \log \frac{X_k}{p_k} \\
\Rightarrow \mathbb{E}_q[\log \textstyle\sum_k p_k \frac{X_k}{p_k}] &\geq \textstyle\sum_k p_k \, \mathbb{E}_q[\log \frac{X_k}{p_k}] \\
\Rightarrow \mathbb{E}_q[\log \textstyle\sum_k X_k] &\geq \textstyle\sum_k p_k \, \mathbb{E}_q[\log X_k] - p_k \log p_k
\end{aligned} \tag{3.12}$$

The lower bound of Eq. 3.12 is applied in Eq. 3.8 and it is a general lower bound useful for the log–sum terms in the ELBO computation. If we want a tight lower bound, we should use Lagrange multipliers to choose the set of p_k that maximize the lower-bound given that they sum to 1.

$$\begin{aligned}
L(p_1, \ldots, p_K) &= (\textstyle\sum_k p_k \, \mathbb{E}_q[\log X_k] - p_k \log p_k) + \lambda \, (1 - \textstyle\sum_k p_k) \\
\frac{\partial L}{\partial p_k} &= \mathbb{E}_q[\log X_k] - \log p_k - 1 - \lambda = 0 \\
\frac{\partial L}{\partial \lambda} &= 1 - \textstyle\sum_k p_k = 0 \\
\Rightarrow \mathbb{E}_q[\log X_k] &= \log p_k + 1 + \lambda \\
\Rightarrow \exp \mathbb{E}_q[\log X_k] &= p_k \exp(1 + \lambda) \\
\Rightarrow \textstyle\sum_k \exp \mathbb{E}_q[\log X_k] &= \underbrace{\sum_k p_k}_{=1} \exp(1 + \lambda) \\
\Rightarrow p_k &= \frac{\exp\{\mathbb{E}_q[\log X_k]\}}{\sum_k \exp\{\mathbb{E}_q[\log X_k]\}}
\end{aligned} \tag{3.13}$$

The final formula for p_k in Eq. 3.13 is exactly the same that we can find for the parameters of the Multinomial distribution of the auxiliary variables in the Poisson model with sum of Gamma distributed latent variables, which demonstrates that the choice of distribution for the auxiliary variables is optimal for this lower-bound.

3.3 Evaluation

In this section, we analyze the predictive power of the proposed model with a real world dataset and compare it with state of the art methods.[6]

Datasets. to be able to compare with the state-of-art method Correlated Topic Regression with Social Matrix Factorization (Purushotham and Liu, 2012), we conducted experiments using the *hetrec2011-lastfm-2k* (Last.fm) dataset (Cantador et al., 2011). This dataset consists of a set of user–artists weighted interactions ("artists" is item set), a set of user–artists-tags and a set of user–user relations[7]. We process the dataset to create an artist–tags matrix by summing up all the tags given by all users to a given artist, this matrix is the item–content matrix in our model. Also, we discard the user–artists weight, considering a "1" for all observed cases. After the preprocessing, we sample 85% of the user–artists observation for training, and kept 15% held-out for predictive evaluation, selecting only users with more than 5 item ratings for the training part of the split.

Metric: Given the random splits of training and test, we train our model and use the estimated latent factors to predict the entries in the testing datasets. In this setting zero ratings can not be necessarily interpreted as negative, making it problematic to use the precision metric. Instead, we focus on recall metric to be comparable with previous work (Purushotham and Liu, 2012) and because the relevant items are available. Specifically, we calculate the recall at the top M items (recall@M) for a user, defined as:

$$\text{recall@}M = \frac{\text{number of items the user likes in Top } M}{\text{total number of items the user likes}} \tag{3.14}$$

Recall@M from Eq. 3.14 is calculated for each user, to obtain a single measure for the whole dataset we average it over all the users obtaining the Avg. Recall@M.

3.3.0.1 Experiments

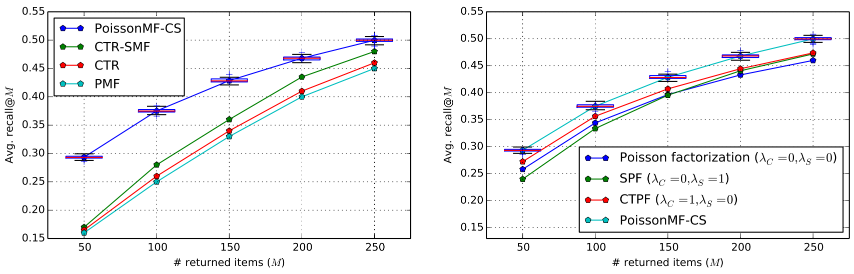
Initially we set all the Gamma hyperparameters to the same values a_{all}[8] and b_{all}[9] equal to 0.1, while varying the latent dimensionality K. For each value of K we

[6]Our C++ implementation of PoissonMF-CS with some of the experiments will be available this repository `https://github.com/zehsilva/poissonmf_cs`

[7]The statistics for the dataset are: 1892 users, 17632 artists, 11946 tags, 25434 user–user connections, 92834 user–items interactions, and 186479 user–tag–items entries

[8]$a_{all} = a^0_\beta = a^0_\eta = a^0_\theta = a^0_\epsilon = a^0_\tau = a_C = a_S = 0.1$

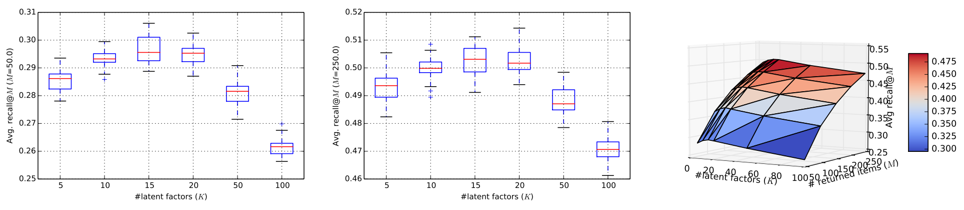
[9]$b_{all} = b^0_\beta = b^0_\eta = b^0_\theta = b^0_\epsilon = b^0_\tau = b_C = b_S = 0.1$



(a) PoissonMF-CS (K=10) and Gaussian based models

(b) PoissonMF-CS (K=10) and other PF models

Figure 3.3: Comparison of PoissonMF-CS with alternative models. Each subplot is the result of running the PoissonMF-CS recommendation algorithm over 30 random splits of the *Hetrec2011-lastfm* dataset for a fixed number of latent features K (in this case, $K = 10$). The values for CTR-SMF, CTR, and PMF was taken from (Purushotham and Liu, 2012), and according to the reported results, they are the best values found after a grid search.



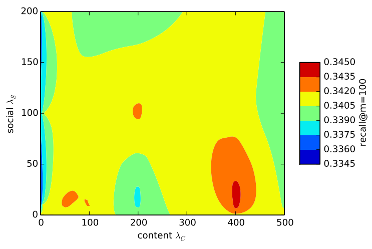
(a) M=50

(b) M=250

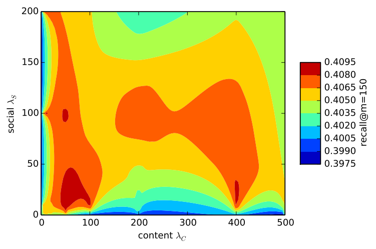
(c) 3D visualization

Figure 3.4: Impact of the number of latent variables (K) parameter on the Av. Recall@M metric for different number of returned items (M). Each subplot is the result of running the PoissonMF-CS recommendation algorithm over 30 random splits of the dataset with K varying in (5,10,15,20,50,100)

ran the experiments on 30 multiple random splits of the dataset in order to be able to generate boxplots of the final recommendation recall. We compare our results with the reported results in Purushotham and Liu (2012) for the same dataset and with optimal parameters. In this first experiment we let the algorithm estimate the optimal content weight λ_C and social weight λ_S. It is possible to see in Fig 3.3 that

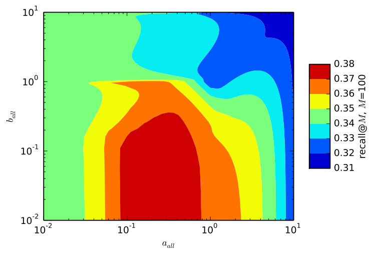


(a) M=100

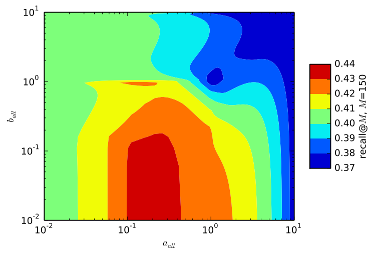


(b) M=150

FIGURE 3.5: Evaluation of the impact of content and social weight parameters (in all experiments in this figure $K = 10$)



(a) M=100



(b) M=150

FIGURE 3.6: Evaluation of the impact of latent Gamma hyperpriors on the recall (in all experiments in this figure $K = 10$)

PoissonMF-CS is consistently outperforming by large margin CTR-SMF and CTR (Fig. 3.3a), while outperforming other Poisson factorization methods (Fig. 3.3b) by a significant margin ($p \leq 1 \cdot 10^{-6}$ in Wilcoxon paired test for each M). . This may be indicative that both the choice of Poisson likelihood with non-negative latent factors and the modelling of content and social weights have positive impact in the predictive power of the model.

Model selection. Fig. 3.4 shows the resulting predictive performance of PoissonMF-CS with different values of number of latent factors K in *Hetrec2011-lastfm* dataset. We concluded that the optimal choice for K is 15. This result is important, indicating that the model is generating compact latent representations, given that the optimal choice of K reported for CTR-SMF in the same dataset is 200. In Fig. 3.6 we show the results for the latent variable hyperparameters. We ran one experiment varying the hyperparameters a_{all} and b_{all} to understand the impact of these hyperparameters in the final recommendation. We noticed that the optimal values for different values of M for both hyperparameters are between 0.1 and 1, a result consistent with the recommendations in the literature (Gopalan et al., 2014b,a; Chaney et al., 2015) and with the statistical intuition that Poisson likelihood with Gamma prior with shape parameter $a < 1$ favour sparse latent representation.

The next experiment was to set the content weight and social weight to fixed values and evaluate the impact of these weights on the result. In Fig 3.5 we can see that the resulting pattern for different values of M is not evident, but indicates that the resulting recall is less sensitive to change in the content and social weights parameters than on the hyperparameters a_{all} and b_{all}. This is also indicative that the importance of social and content factors is not the same at different points of the ranked list of recommendations.

Finally, we add the observation that in the recent work of (Xiao et al., 2019) our proposed method was compared with a variational deep matrix factorization approach using social information, comparing with eight alternative methods (including their approach), and PoissonMF-CS appear as second or third over different metrics and datasets, despite not including non-linear / deep transformations in the latent space.

3.4 Final remarks

We have described PoissonMF-CS, a joint Bayesian model for recommendations integrating three sources of information: item textual content, user social network, and user–item interactions. It generalizes existent Poisson factorization models for recommendations by adding both content and social features. Our experiment shows that the proposed model consistently outperforms previous Poisson models (SPF and CTPF) and alternative joint models based on Gaussian probabilistic factorization and LDA (CTR-SMF and CTR) on a dataset containing both content and social side information. These results demonstrate that joint modeling of social and content features using Poisson models improves the recommendations, can have scalable inference and generates more compact latent features. Although the batch variational inference algorithm is already efficient [10], one future improvement will be the design of Stochastic Variational Inference algorithm for very large scale inference.

[10]For example, it takes 12 minutes to train the best performing model in a desktop machine with the *Hetrec2011-lastfm* dataset in a single core without any parallelism

Tensor factorization with cyclic latent factors 4

"You show me continents, I see the islands, You count the centuries, I blink my eyes."

— Björk, *Oceania (Song)*

The proliferation of data intensive devices and systems has generated a growth of availability of users' behavioral data that includes spatiotemporal patterns of cyber–physical interactions (Ren et al., 2017). In the context of indoor public spaces, one example of this type of data is WiFi user logs, where individual users devices, location estimation, time and query and *URL* logs are available, among other sources of user information. Understanding and properly modeling users' behavior by taking into account specific features of cyber-physical data (e.g. periodicity that emerges from human habits) and designing systems capable of leveraging the interactions between the physical and cyber space, while improving personalization methods are of growing importance (Ren et al., 2018).

Within this context we focus on modeling multiple spatial-temporal sources of information to infer the user intent and information need. There are many specificities of this task, for example in many situations the location information is not explicit, but implicit and correlated with other features (such as the WiFi access point and the indoor region where the access point is located). More importantly, it is vital to account for periodic or cyclical patterns, specially in public spaces such as shopping malls, universities, public buildings, where recurrent events happens. In this context, we approach the problem by employing Poisson-Gamma factorization as a basic construction block, adding specific constraint for periodic time modeling and contextual side information.

Problem definition Given the set of user–item–time interactions; $S_{\text{train}} = \{(u, i, t, r_{ui,t})\}$ with $u, i, t, r_{ui,t} \in \mathbb{N}$, each tuple meaning that the user u interacted with the item i at periodic time t for $r_{ui,t}$ times. The time variable t is an index for a set of periodic timestamps, for example weekdays or hours within a day. We aim

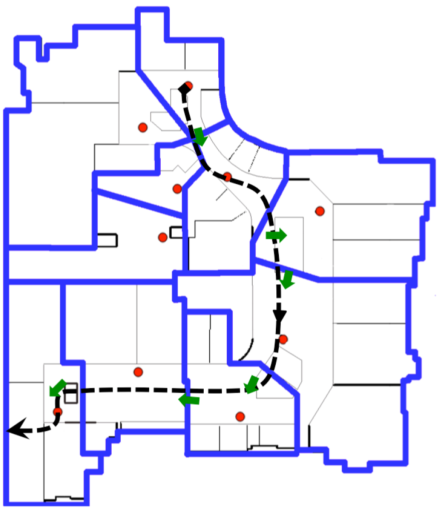


Figure 4.1: An example of indoor (retail) environment. It shows the floor map of a shopping mall: the red dots are WiFi APs; the blue lines are the rectified Voronoi cells for each AP; the green arrows show the actual walking directions of a user, and the dashed black line is the corresponding trajectories captured in WiFi logs in terms of AP associations.

to learn a function that estimates a score for new items for a user at a given time – the score will be used for ranking the items. More formally, given our training set S_{train}, we will train a probabilistic model, and use this model to derive a rate function for the interaction of user u with item j at time t, $p(r_{ui,t}|S_{\text{train}})$. Moreover, as an extra objective we want to equip the probabilistic model with side-information in a form of item–feature matrix, given that extra information about the item set can be available and correlated with spatial patterns. With the model at hand we want to be perform predictive task such as recommendation of items for a given user at a given time. The side information matrix is particularly useful for the users with limited number of interactions with items in the dataset (*cold start* problem).

For example, in the context of indoor recommendation using WiFi logs, we have a set of tuples with unique users identifiers, WiFi access points (AP), time-stamps and other side informations such as queries issued or categories related to the physical space that the access point is located (for example, the category of the store close to an AP in a shopping center). Figure 4.1 illustrates an example of indoor retail environment, including a user's walking trajectory through the mall. *In* *this setting the items are the WiFi APs that each user is associated with at a given time, which can be used as proxy for the physical location in indoor environments (a store within a shopping mall for example).* As *side information*, we have a matrix of the categories of the stores in the vicinity of each access point. This is defined via Voronoi cells, each centered on a single AP, which encompasses all the points that are closest to that AP. To better correspond with the frontages of physical stores, the cells can be manually rectified.

Related work

Some methods have been proposed for the indoor recommendation problem, however, given the novelty, in general most are tackling some different variations of the settings of the problem. For example, Orciuoli and Parente (2017) apply computational ontologies and cellular automata for designing a context-aware indoor recommender system, where the ontologies can be used to model context specific rules and logics. Fang et al. (2012) propose a mobile system, combining server-side location services and recommendation services into a single mobile application that is location-aware using Receiver Signal Strength (RSS) to infer indoor locations. Ren et al. (2018) develops a graph-based non-personalized recommender system leveraging query, location and web-domain contextual information for indoor recommendations. Our approach is distinguished from all the previous ones because we incorporate personalized contextual recommendation with periodic time modeling using Poisson factorization models – this allows implicit user count data to be harnessed into a spatio-temporal model.

4.1 Periodic Time-Aware Poisson Factorization

We extend Poisson factorization by allowing users and items latent preference factors to be indexed by time slices and imposing a constraint over consecutive time factors. In other words, instead of having a static latent factors for each user and item, we have latent factors that are indexed by time. Another contribution is a specific construction that binds consecutive time indexed factors (Figure 4.2). We connect the last factor to the first one in the time-ordered sequence and each consecutive factor, effectively creating cyclic influence on the posterior probability distribution of the latent factors.

Suppose there is a set of scalar Gamma latent variables for an user u indexed by time $\{\eta_{uk,1}, \cdots, \eta_{uk,T}\}$, with $\eta_{uk,t} \sim \text{Gamma}(a_\eta, b_\eta)$ ($t \in \{1, \cdots, T\}$ is the index of time-intervals in the data), it is possible to create a chain of cyclic dependency by adding an auxiliary dummy observation variable between each consecutive time

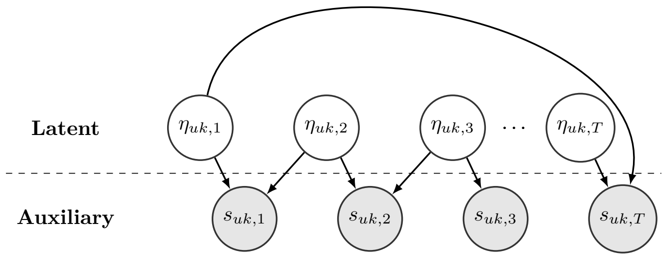


Figure 4.2: Diagram for the cyclic time-periodic latent variables probabilistic construction

factors, $s_{uk,t} \sim \text{Poisson}(\eta_{uk,t}\eta_{uk,t+1})$ and impose that $\eta_{uk,T+1} = \eta_{uk,1}$, $\eta_{uk,0} = \eta_{uk,T}$, $s_{uk,T+1} = s_{uk,1}$, and $s_{uk,0} = s_{uk,T}$. This means that the last time factor in the chain is connected to the first time factor in the chain and each consecutive factor is connected to each other through the auxiliary variable.

The complete posterior for each latent factor given by this construction is $p(\eta_{uk,t}|*) = \text{Gamma}(a_\eta + s_{uk,t-1} + s_{uk,t}, b_\eta + \eta_{uk,t-1} + \eta_{uk,t+1})$. This means that it is possible to propagate information across a period of time creating a cyclic dependence between the latent factor for a given period and granularity of time. Notice that $s_{u+,t} \sim \text{Poisson}(\sum_k \eta_{uk,t}\eta_{uk,t+1})$,: the rate of auxiliary variable $s_{u+,t}$ (sum of the auxiliary variables over the latent dimensions k for a fixed user and time) is given by the inner product between consecutive time latent vectors[1].

This construction also entails that the complete posterior for the set of auxiliary variables for all the latent dimensions (and fixed user u and time t) is a Multinomial $p((s_{u1,t}, \ldots, s_{uK,t})|*) = \text{Mult}(s_{u+,t}|\boldsymbol{\pi_{u,t}})$, with $\pi_{uk,t} = \frac{\eta_{uk,t}\eta_{uk,t+1}}{\sum_k \eta_{uk,t}\eta_{uk,t+1}}$ (Gopalan et al., 2014b; Zhou et al., 2012). If we assign a fixed value $s_{u+,t} = \lambda_s$ to the auxiliary variables $s_{u+,t}$ and use standard variational inference with mean-field approximation in each latent variable it is possible to obtain closed-form update equations for $\eta_{uk,t}$, using an approximate density $q(\eta_{uk,t}) = \text{Gamma}(a_{\eta_{uk,t}}, b_{\eta_{uk,t}})$, and writing $\mathbb{E}_q[.]$ and $\text{G}_q[.]$ for the arithmetic and geometric mean of the random

[1]Given a set of indexed variable $\{X_{ij}\}$ we use the subscript $+$ to denote a sum over the respective index. For example, if $X_{ij} \sim_{\text{iid}} \text{Poisson}(\lambda_j)$, then $X_{i+} = \sum_j X_{ij} \sim \text{Poisson}(\sum_j \lambda_j)$ because a sum of Poisson distributed variables is a Poisson with a rate resulting of the sum of individual rates.

variable[2]:

$$\begin{aligned} a_{\eta_{uk,t}} =& a_\eta + \lambda_s \Big(\frac{\mathrm{G}_q[\eta_{uk,t}]\,\mathrm{G}_q[\eta_{uk,t+1}]}{\sum_{k'} \mathrm{G}_q[\eta_{uk',t}]\,\mathrm{G}_q[\eta_{uk',t+1}]} \\ &+ \frac{\mathrm{G}_q[\eta_{uk,t}]\,\mathrm{G}_q[\eta_{uk,t-1}]}{\sum_{k'} \mathrm{G}_q[\eta_{uk',t}]\,\mathrm{G}_q[\eta_{uk',t-1}]} \Big) \\ b_{\eta_{uk,t}} =& b_\eta + \mathbb{E}_q[\eta_{uk,t-1}] + \mathbb{E}_q[\eta_{uk,t+1}] \end{aligned}$$

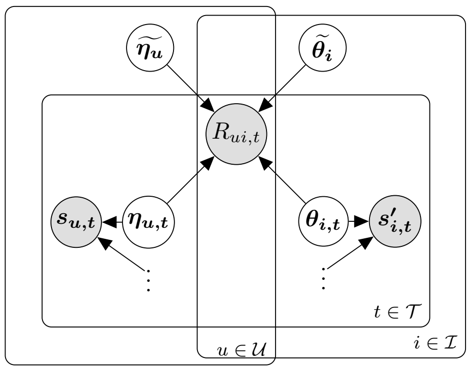
We can rewrite this update equation in terms of expected values, given that the expected value of a Gamma distributed variable is $\mathbb{E}_q[\eta_{uk,t}] = \frac{a_{\eta_{uk,t}}}{b_{\eta_{uk,t}}}$:

$$\mathbb{E}_q[\boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t}}] = \frac{a_\eta + \lambda_s \left(\frac{\mathrm{G}_q[\boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t}}] \circ \mathrm{G}_q[\boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t+1}}]}{\mathrm{G}_q[\boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t}}]^\top \mathrm{G}_q[\boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t+1}}]} + \frac{\mathrm{G}_q[\boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t}}] \circ \mathrm{G}_q[\boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t-1}}]}{\mathrm{G}_q[\boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t}}]^\top \mathrm{G}_q[\boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t-1}}]} \right)}{b_\eta + \mathbb{E}_q[\boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t-1}}] + \mathbb{E}_q[\boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t+1}}]} \tag{4.1}$$

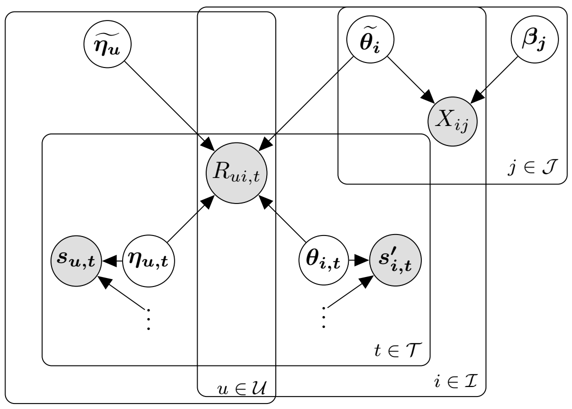
An intuitive way to look at Equation 4.1 is to notice that the numerator and denominator are estimates for the expected value of the latent variable at time t constructed by averaging the geometric and arithmetic expected value of the latent variable at time $t+1$ and $t-1$, working as a form of smoothing between adjacent time variables.

Adding side-information Given that all latent factors are non-negative Gamma factors, we can use shared factors to induce relative dependency between observations. Keeping all factors as Gamma latent variable will allow the model to be conjugate and facilitate closed-form updates latent inference. We will combine the idea of shared variables with the idea of time-dependent variables in a single model by having a set of global and local variables for each user and item, the local variables will be time specific, while the global variables will be shared across the user or item time-line. In Figure 4.3a, we display the proposed model without side-information, which is called Temporal Poisson Tensor Factorization (TPTF); in Figure 4.3b, the global items variables $\widetilde{\boldsymbol{\theta}}_i$ are shared between the user–item matrix and item–side-information matrix, which is called Collective Temporal Poisson Tensor Factorization (TPTF-C). More details will be discussed in the following section.

[2]$\mathbb{E}_q[X]$ is the expected value of the random variable X under the variational family q. While $\mathrm{G}_q[X]$ is the geometrical expected value, defined as $\mathrm{G}_q[X] = \exp(\mathbb{E}_q[\log(X)])$. Notice that $\mathrm{G}_q[XY] = \exp(\mathbb{E}_q[\log(X)])\exp(\mathbb{E}_q[\log(Y)]) = \mathrm{G}_q[X]\,\mathrm{G}_q[Y]$. Also, if X is a gamma distributed variable $X \sim \mathrm{Gamma}(a_X, b_X)$ (parameterized by shape and rate), then $\mathbb{E}_q[X] = \frac{a_X}{b_X}$ and $\mathrm{G}_q[X] = \frac{\exp(\Psi(a_X))}{b_X}$, where $\Psi(.)$ is the Digamma function.



(a) TPTF



(b) TPTF-C

Figure 4.3: Plate diagram for Temporal Poisson Tensor Factorization

4.1.1 Generative model

In general, TPTF and TPTF-C have a set of similar users and items variables, the main difference being that in TPTF-C there is a set of extra variables for the side-information matrix. A complete description of the generative model is presented in Table 4.1 assuming an item set $\mathcal{I}$ of size $|\mathcal{I}| = I$, users set $\mathcal{U}$ of size $|\mathcal{U}| = U$, periodic time slices set $\mathcal{T}$ of size $|\mathcal{T}| = T$, and set of side-information features $\mathcal{J}$ of size $|\mathcal{J}| = J$ and K latent factors (topics) (with an index set $\mathcal{K}$). For brevity, some details for the time-chain construction is omitted in Table 4.1 (particularly

Table 4.1: Specification of the generative model for TPTF

Temporal Poisson Tensor Factorization (TPTF)

1. Latent parameter prior distributions:
   a) for all topics $k \in \mathcal{K}$ and for all time $t \in \mathcal{T}$:
      - for all items $i \in \mathcal{I}$:
        - base factor: $\widetilde{\theta}_{ik} \sim \text{Gamma}(a^0_\theta, b^0_\theta)$
        - time-dependent factor: $\theta_{ik,t} \sim \text{Gamma}(a^1_\theta, b^1_\theta)$
        - periodic time auxiliary variables:
          * $s'_{ik,t} \sim \text{Poisson}(\theta_{ik,t}\theta_{ik,t+1})$
          * $s'_{i+,t} = \sum_k s'_{ik,t}$
      - for all users $u \in \mathcal{U}$:
        - base factor: $\widetilde{\eta}_{uk} \sim \text{Gamma}(a^0_\eta, b^0_\eta)$
        - time-dependent factor: $\eta_{uk,t} \sim \text{Gamma}(a^1_\eta, b^1_\eta)$
        - periodic time auxiliary variables:
          * $s_{uk,t} \sim \text{Poisson}(\eta_{uk,t}\eta_{uk,t+1})$
          * $s_{u+,t} = \sum_k s_{uk,t}$
2. Observations probability distribution:
   a) for all observed user–item–time tuples (u, i, t) :

$$R_{ui,t} \sim \text{Poisson}((\widetilde{\boldsymbol{\eta}}_{\boldsymbol{u}} + \boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t}})^\top(\widetilde{\boldsymbol{\theta}}_{\boldsymbol{i}} + \boldsymbol{\theta}_{\boldsymbol{i},\boldsymbol{t}}))$$

the connection between the end of the time-chain and the beginning), however this construction follows exactly the description detailed in previous section.

Augmented model : Furthermore, in order to facilitate posterior inference, we employ the technique developed in Gopalan et al. (2014b) and Zhou et al. (2012). The technique consists in adding a latent count variable for each term contributing to the rate of the Poisson counts observations, based on the fact that the sum of Poisson variables is also a Poisson variable. For $R_{ui,t}$, the augmented model has a set of $4K$

Table 4.2: Specification of the generative model for TPTF-C

Collective Temporal Poisson Tensor Factorization (TPTF-C)

1. Latent parameter prior distributions:

   a) for all topics $k \in \mathcal{K}$ and for all time $t \in \mathcal{T}$:

      - for all items $i \in \mathcal{I}$:
        - base factor: $\widetilde{\theta}_{ik} \sim \text{Gamma}(a^0_\theta, b^0_\theta)$
        - time-dependent factor: $\theta_{ik,t} \sim \text{Gamma}(a^1_\theta, b^1_\theta)$
        - periodic time auxiliary variables:
          * $s'_{ik,t} \sim \text{Poisson}(\theta_{ik,t}\theta_{ik,t+1})$
          * $s'_{i+,t} = \sum_k s'_{ik,t}$
      - for all users $u \in \mathcal{U}$:
        - base factor: $\widetilde{\eta}_{uk} \sim \text{Gamma}(a^0_\eta, b^0_\eta)$
        - time-dependent factor: $\eta_{uk,t} \sim \text{Gamma}(a^1_\eta, b^1_\eta)$
        - periodic time auxiliary variables:
          * $s_{uk,t} \sim \text{Poisson}(\eta_{uk,t}\eta_{uk,t+1})$
          * $s_{u+,t} = \sum_k s_{uk,t}$
      - for all features $j \in \mathcal{J}$:
        - global factor: $\beta_{jk} \sim \text{Gamma}(a_\beta, b_\beta)$

2. Observations probability distribution:

   a) for all observed user–item– tuples (u, i, t) :

$$R_{ui,t} \sim \text{Poisson}((\boldsymbol{\widetilde{\eta}_u} + \boldsymbol{\eta_{u,t}})^\top(\boldsymbol{\widetilde{\theta}_i} + \boldsymbol{\theta_{i,t}}))$$

   b) for all observed item–feature tuples (i, j) :

$$X_{ij} \sim \text{Poisson}(\boldsymbol{\widetilde{\theta}_i}^\top \boldsymbol{\beta_j})$$

latent counts variables since $R_{ui,t} \sim \text{Poisson}(\sum_k \left(\widetilde{\eta}_{uk}\widetilde{\theta}_{ik} + \eta_{uk,t}\widetilde{\theta}_{ik} + \widetilde{\eta}_{uk}\theta_{ik,t} + \eta_{uk,t}\theta_{ik,t}\right))$, while for X_{ij} it will need a set of K auxiliary counts.

$$\begin{aligned}
Z^A_{uik,t} &\sim \text{Poisson}(\widetilde{\eta}_{uk}\widetilde{\theta}_{ik}) \\
Z^B_{uik,t} &\sim \text{Poisson}(\eta_{uk,t}\widetilde{\theta}_{ik}) \\
Z^C_{uik,t} &\sim \text{Poisson}(\widetilde{\eta}_{uk}\theta_{ik,t}) \\
Z^D_{uik,t} &\sim \text{Poisson}(\eta_{uk,t}\theta_{ik,t}) \\
Y_{ijk} &\sim \text{Poisson}(\widetilde{\theta}_{ik}\beta_{jk}) \\
\text{with } R_{ui,t} &= \sum_k Z^A_{uik,t} + Z^B_{uik,t} + Z^C_{uik,t} + Z^D_{uik,t} \\
\text{and } X_{ij} &= \sum_k Y_{ijk}
\end{aligned}$$

The augmentation described above transform our model in a full conjugate model, with complete conditionals for the latent variables given by Multinomial and Gamma distributed variables. As detailed in the following section, we use this construction to develop an efficient variational inference coordinate ascent learning parameter algorithm.

4.2 Inference

Given the generative model of the data and a set of observations, our objective is to infer the parameters of the model. However, an exact computation of the posterior of these models given the data is intractable, which lead us to pursue approximation methods. We work with approximate batch variational inference. It is possible to augment Poisson-gamma factorization models by adding auxiliary Poisson count variables, obtaining an equivalent complete conjugate model (Gopalan et al., 2015), thus facilitating closed-form equation for coordinate ascent on the variational parameters.

4.2.0.1 Variational inference

The general approach of variational inference is to transform the original problem of posterior inference into an optimization problem. Suppose the joint density of the observations $\boldsymbol{O}$ and latent variables Θ is given by $p(\boldsymbol{O}, \Theta)$, assuming a family of surrogate approximate distributions for the latent terms $q(\Theta|\chi)$, parametrized by variational parameters χ, we look to minimize the KL-divergence between them. However, directly using the KL divergence is infeasible, instead we maximize of the

Evidence Lower BOund (ELBO) (Bishop, 2006):

$$\underset{\chi}{\operatorname{argmax}} \operatorname{ELBO}(\chi) = \mathbb{E}_q[\log p(\boldsymbol{O}, \Theta) - \log q(\Theta|\chi)]$$

Mean field approximation: Assuming one independent distribution for each latent factor in the model we obtain the coordinate ascent variational inference (CAVI) update algorithm for each factor as a function of the natural parameter. The resulting factorized variational distribution for TPTF is defined in Equation 4.2 (the resulting equation for TPTF-C is similar, the only difference would be due to the extra variables).

$$\begin{aligned} q(\Theta|\chi) &= \textstyle\prod_{k,i} q(\widetilde{\theta}_{ik}) \prod_{k,i,t} q(\theta_{ik,t}) q(\{s'_{ik,t}\}_k) \\ &\times \textstyle\prod_{k,u} q(\widetilde{\eta}_{uk}) \prod_{k,u,t} q(\eta_{uk,t}) q(\{s_{uk,t}\}_k) \\ &\times \textstyle\prod_{uik,t} q(\{Z^A_{uik,t}, Z^B_{uik,t}, Z^C_{uik,t}, Z^D_{uik,t}\}_k) \end{aligned} \tag{4.2}$$

Using the mean-field approximation, each variational distribution should be defined in the same distribution family as the complete conditional of the latent variable that it represents. Each variational natural parameter should be matched with the expected value of the natural parameter of the respective complete conditional on the original model (Bishop, 2006). Thus, the resulting update equations are compiled in Table 4.3 for TPTF and Table 4.4 for TPTF-C. The CAVI algorithm consist in iterative alternate updates of 1) user factors; 2) item factors and 3) latent count factors, for TPTF. In the case of TPTF-C the algorithm iterate between 1) user factors; 2) item factors; 3) side-information factors; and 4) latent count factors (for both the user–item–time tensor and item–feature side information matrix).

Complexity: Since the inference algorithm proceeds iteratively over all the parameters and the number of non-zeros in the sparse interactions count tensor and side-information matrix for all time periods and latent dimensions linearly, the complexity of each iteration is $O(KT(U + I + J)(nR + nX))$ (nR and nX is respectively the number of non-zero entries in $\boldsymbol{R}$ and $\boldsymbol{X}$).

Recommendation: Having specified the model for each user–item–time number of interactions $p(R_{ui,t}|\widetilde{\boldsymbol{\eta}}_{\boldsymbol{u}}, \boldsymbol{\eta}_{\boldsymbol{u},\boldsymbol{t}}, \widetilde{\boldsymbol{\theta}}_{\boldsymbol{i}}, \boldsymbol{\theta}_{\boldsymbol{i},\boldsymbol{t}})$, and equipped with variational approximations, we can use the knowledge of each latent user and item latent factors to estimate the expected value of all possible user–item–time numbers of interactions,

Table 4.3: Update equations of the variational parameters for TPTF

Temporal Poisson Tensor Factorization (TPTF)

Variable	Variational Distribution	Parameter update
$\widetilde{\eta}_{uk}$	$\text{Gamma}(a_{\widetilde{\eta}_{uk}}, b_{\widetilde{\eta}_{uk}})$	$a_{\widetilde{\eta}_{uk}} = a^0_\eta + \sum_{i,t} R_{ui,t} \left(\psi^A_{uik,t} + \psi^C_{uik,t}\right)$ $b_{\widetilde{\eta}_{uk}} = b^0_\eta + \sum_{i,t} \left(\mathbb{E}_q[\widetilde{\theta}_{ik}] + \mathbb{E}_q[\theta_{ik,t}]\right)$
$\widetilde{\theta}_{ik}$	$\text{Gamma}(a_{\widetilde{\theta}_{ik}}, b_{\widetilde{\theta}_{ik}})$	$a_{\widetilde{\theta}_{ik}} = a^0_\theta + \sum_{u,t} R_{ui,t} \left(\psi^A_{uik,t} + \psi^B_{uik,t}\right)$ $b_{\widetilde{\theta}_{ik}} = b^0_\theta + \sum_{u,t} \left(\mathbb{E}_q[\widetilde{\eta}_{uk}] + \mathbb{E}_q[\eta_{uk,t}]\right)$
$\eta_{uk,t}$	$\text{Gamma}(a_{\eta_{uk,t}}, b_{\eta_{uk,t}})$	$a_{\eta_{uk,t}} = a^1_\eta + \lambda_s \left(f_{uk,t-1} + f_{uk,t}\right)$ $+ \sum_i R_{ui,t} \left(\psi^B_{uik,t} + \psi^D_{uik,t}\right)$ $b_{\eta_{uk,t}} = b^1_\eta + \mathbb{E}_q[\eta_{uk,t-1}] + \mathbb{E}_q[\eta_{uk,t+1}]$ $+ \sum_i \left(\mathbb{E}_q[\widetilde{\theta}_{ik}] + \mathbb{E}_q[\theta_{ik,t}]\right)$
$\theta_{ik,t}$	$\text{Gamma}(a_{\theta_{ik,t}}, b_{\theta_{ik,t}})$	$a_{\theta_{ik,t}} = a^1_\theta + \lambda_s \left(g_{ik,t-1} + g_{ik,t}\right)$ $+ \sum_u R_{ui,t}(\psi^C_{uik,t} + \psi^D_{uik,t})$ $b_{\theta_{ik,t}} = b^1_\theta + \mathbb{E}_q[\theta_{ik,t-1}] + \mathbb{E}_q[\theta_{ik,t+1}]$ $+ \sum_u \left(\mathbb{E}_q[\widetilde{\eta}_{uk}] + \mathbb{E}_q[\eta_{uk,t}]\right)$
$(s_{u1,t}, \ldots, s_{uK,t})$	$\text{Mult}(\lambda_S \vert (f_{u1,t}, \ldots, f_{uK,t}))$	$f_{uk,t} = \frac{\text{G}_q[\eta_{uk,t}]\,\text{G}_q[\eta_{uk,t+1}]}{\sum_{k' \in \mathcal{K}} \text{G}_q[\eta_{uk',t}]\,\text{G}_q[\eta_{uk',t+1}]}$
$(s'_{i1,t}, \ldots, s'_{iK,t})$	$\text{Mult}(\lambda_S \vert (g_{i1,t}, \ldots, g_{iK,t}))$	$g_{ik,t} = \frac{\text{G}_q[\theta_{ik,t}]\,\text{G}_q[\theta_{ik,t+1}]}{\sum_{k' \in \mathcal{K}} \text{G}_q[\theta_{ik',t}]\,\text{G}_q[\theta_{ik',t+1}]}$
$(Z^A_{ui1,t}, \ldots, Z^A_{uiK,t}$ $Z^B_{ui1,t}, \ldots, Z^B_{uiK,t}$ $Z^C_{ui1,t}, \ldots, Z^C_{uiK,t}$ $Z^D_{ui1,t}, \ldots, Z^D_{uiK,t})$	$\text{Mult}(R_{ui,t} \vert \boldsymbol{\psi_{ui,t}})$ $\boldsymbol{\psi_{ui,t}} = (\boldsymbol{\psi^A_{ui,t}}, \boldsymbol{\psi^B_{ui,t}},$ $\boldsymbol{\psi^C_{ui,t}}, \boldsymbol{\psi^D_{ui,t}})$	$\psi^A_{uik,t} \propto \text{G}_q[\widetilde{\theta}_{ik}]\,\text{G}_q[\widetilde{\eta}_{uk}]$ $\psi^A_{uik,t} \propto \text{G}_q[\widetilde{\theta}_{ik}]\,\text{G}_q[\eta_{uk,t}]$ $\psi^A_{uik,t} \propto \text{G}_q[\theta_{ik,t}]\,\text{G}_q[\widetilde{\eta}_{uk}]$ $\psi^A_{uik,t} \propto \text{G}_q[\theta_{ik,t}]\,\text{G}_q[\eta_{uk,t}]$ $\sum_k \psi^A_{uik,t} + \psi^B_{uik,t} + \psi^C_{uik,t} + \psi^D_{uik,t} = 1$

Table 4.4: Update equations of the variational parameters for TPTF-C (including only the extra parameters for the context latent variables and the changes in $\widetilde{\beta}_{ik}$)

Collective Temporal Poisson Tensor Factorization (TPTF-C)		
Variable	Variational Distribution	Parameter update
β_{jk}	$\text{Gamma}(a_{\beta_{jk}}, b_{\beta_{jk}})$	$a_{\beta_{jk}} = a_\beta + \sum_i X_{ij}\phi_{ijk}$ $b_{\beta_{jk}} = b_\beta + \sum_i \mathbb{E}_q[\widetilde{\theta}_{ik}]$
$\widetilde{\theta}_{ik}$	$\text{Gamma}(a_{\widetilde{\theta}_{ik}}, b_{\widetilde{\theta}_{ik}})$	$a_{\widetilde{\theta}_{ik}} = a_\theta^0 + \sum_j X_{ij}\phi_{ijk}$ $+ \sum_{u,t} R_{ui,t}\left(\psi^A_{uik,t} + \psi^B_{uik,t}\right)$ $b_{\widetilde{\theta}_{ik}} = b_\theta^0 + \sum_j \mathbb{E}_q[\beta_{jk}]$ $+ \sum_{u,t}\left(\mathbb{E}_q[\widetilde{\eta}_{uk}] + \mathbb{E}_q[\eta_{uk,t}]\right)$
$(Y_{ij1}, \dots, Y_{ijK})$	$\text{Mult}(X_{ij} \mid \boldsymbol{\phi_{ij}})$	$\forall k \in \{1, \dots, K\}$, $\phi_{ijk} = \frac{\mathrm{G}_q[\widetilde{\theta}_{ik}]\,\mathrm{G}_q[\beta_{jk}]}{\sum_{k'}\mathrm{G}_q[\widetilde{\theta}_{ik'}]\,\mathrm{G}_q[\beta_{jk'}]}$

thus allowing us to create a ranking of items for each users at a given time using the expected value of the latent variables.

$$\mathbb{E}_q[R_{ui,t}] = \sum_k (\mathbb{E}_q[\widetilde{\eta}_{uk}] + \mathbb{E}_q[\eta_{uk,t}])(\mathbb{E}_q[\widetilde{\theta}_{ik}] + \mathbb{E}_q[\theta_{ik,t}]) \tag{4.3}$$

The recommendation algorithm for the M most likely items to be consumed would consist in, at each time period t and user u, applying Equation 4.3 for all the unseen items at that given time and returning the ones with highest ranking in a shortlist of size given by a parameter M.

4.3 Evaluation

Experiment Configuration: The data we used for experiments were collected via a free opt-in WiFi network in an inner city shopping mall in Sydney, Australia. The mall area is around 90,000 square meters, covered by 67 Wi-Fi access points; and it contains over 200 stores that belong to 34 shop categories as defined by the mall operator (as shown in Table 4.5).

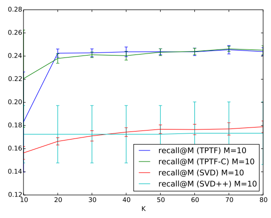
The WiFi trajectory logs were collected over 120,000 anonymized users between between September 2012 and October 2013. It contains 907,084 rows of WiFi

Table 4.5: Shop categories

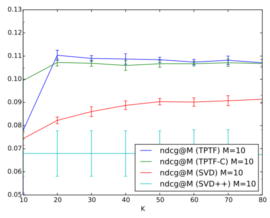
Cafe	Unisex Fashion	Women's Fashion
Men's Fashion	Fashion Accessories	Unisex Fashion
Fine Jewelery	Women's Footwear	General Footwear
Cosmetics	Restaurant	General Footwear
Groceries	Groceries	Bakeries / Patisseries
Men's Fashion	Watches	Gifts / Souvenirs
Costume Jewelery	Takeaway	Newsagents / Stationery
Hair & Beauty	Small / Major Appliances	Repairs & Maintenance
Travel	Mobile Phones & Accessories	Sport
Home Decor	Discount Cosmetics	Restaurant
Delicatessen	Gymnasiums	Pad Sites
Music / Videos / DVDs		

AP associations over 261,369 user visits to the mall. Specifically, it captures information about user physical behavior characterized by the following parameters: user device's MAC address uniquely identifying the associated device; the ID of the WiFi AP associated with the user's mobile device at a given point in time, used as a proxy for the user's location; the time-stamp of users' association/disassociation with the AP. To obtain the side information (the types of stores covered by an AP), floor plans of the mall were overlaid with AP positions and the service areas of the APs were approximated by Voronoi regions (as shown in Figure 4.1). We run each experiment by splitting the dataset in 90% training and 10% testing randomly, repeating that process for 10 times, collecting the metrics at each time, and calculating the average and error value over the 10 random splits.

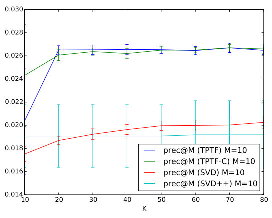
Metrics: Given the random splits of training and test datasets, we train our model and use the estimated latent factors to predict the entries in the testing



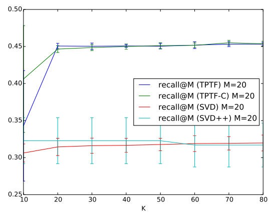
(a) Recall@10



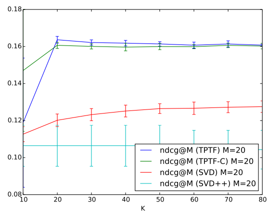
(b) NDCG@10



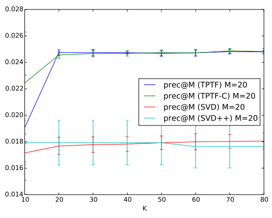
(c) Prec@10



(d) Recall@20



(e) NDCG@20



(f) Prec@20

Figure 4.4: Comparison of predictive performance of TPTF, TPTF-C and baseline models for varying number of latent factor (K) using different metrics measured at top-10 and top-20 recommendations

datasets for each user–time pair present in the testing dataset. After creating a ranking of the top-M recommendation for the pair user–time, we compare the recommendations with the respective item set in the testing set evaluation using precision, recall and normalized discounted cumulative gain (NDCG) (Lim et al., 2015). We then average each of the metrics for all users and different number of returned items, generating a table of averaged results for Recall@M, NDCG@M and Precision@M. We report average metrics with an error bar, calculated over all users in the test set and repeated for 10 random splits of the dataset.

Baselines: In order to evaluate the proposed model we will focus on the time-aware recommendation task. This task consists in recommending items for each user at a given time moment, in other words, given user u and time t, the algorithm returns a ranked list of items $\{i_1, \ldots, i_M\}$. We compare our approach with the baseline methods SVD and SVD++ (Hu et al., 2008) on this task.

Discussion and results: We report our results on the time-aware recommendation task in Figure 4.4. In the first row we display the averaged results when the top-10 items are returned for each user at a given time. It is possible to observe, that when the dimensionality is low, all methods tend to converge to the same performance (although, TPTF-C seems to be more robust). Also, we observe that after a certain dimensionality, there is a stabilization of the metrics, meaning that there is little gain on the predictive power from increasing the dimensionality. Examining the results from top-20 returned items, we observe that the advantage of TPTF and TPTF-C over the baseline increases, confirming the insight that Poisson-Gamma models tend to capture better the long tail behavior from sparse data. As observed before, by looking both at the behavior of the methods as we change the dimensionality and the error bars across the reported results, there is an indication that the extra contextual matrix is making the results more robust, across the variations of the settings (dimensionality K) and the random initialization performed in order to create the error bars.

4.4 Final remarks

We proposed a new Poisson matrix-tensor factorization model tailored to factorize a count data tensor of user–item–time interactions with the assumption of periodic user–item interactions and an optional auxiliary count-feature matrix for the items. Also, we present close-form updates for the coordinate ascent variational inference of the latent parameters of the model, which is efficient because it depends only on the non-zero observations (can operate as well with sparse data). We evaluate the predictive performance of the proposed models on a dataset of WiFi logs in an indoor environment, showing improved performance over the baselines for time-aware recommendations.

$$\begin{tikzcd}
\xi^1 \arrow[d] \arrow[dr] & \xi^2 \arrow[d] \arrow[dr] & \xi^3 \arrow[dll] \arrow[d] & \xi^4 \arrow[dl] \\
\boldsymbol{R}_1 & \boldsymbol{R}_2 & \boldsymbol{R}_3 &
\end{tikzcd}$$

Figure 4.5: Diagram of a generic model with multiple entities and relations that can be represented with coupled matrix and tensor factorization

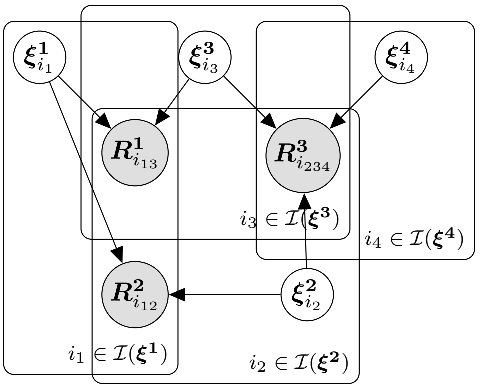

$\boldsymbol{\xi}^{\mathbf{1}}_{i_1}$
$\boldsymbol{\xi}^{\mathbf{3}}_{i_3}$
$\boldsymbol{\xi}^{\mathbf{4}}_{i_4}$
$\boldsymbol{R}^{\mathbf{1}}_{i_{13}}$
$\boldsymbol{R}^{\mathbf{3}}_{i_{234}}$
$i_3 \in \mathcal{I}(\boldsymbol{\xi}^{\mathbf{3}})$
$i_4 \in \mathcal{I}(\boldsymbol{\xi}^{\mathbf{4}})$
$\boldsymbol{R}^{\mathbf{2}}_{i_{12}}$
$\boldsymbol{\xi}^{\mathbf{2}}_{i_2}$
$i_1 \in \mathcal{I}(\boldsymbol{\xi}^{\mathbf{1}})$
$i_2 \in \mathcal{I}(\boldsymbol{\xi}^{\mathbf{2}})$

Figure 4.6: Equivalent probabilistic model diagram for the example of a generic coupled matrix and tensor factorization model

4.5 Towards generalized collective matrix and tensor factorization

The modeling design presented in Chapter 3 and Chapter 4 can be generalized for multiple relationships and entities, going beyond the particular use cases presented in those chapters. We have not implemented this generic model, but will present the basic specification of that model. A generic relational model can be represented by a bipartite directed graph $G = (\{\mathcal{E}, \mathcal{R}\}, E)$, with every directed edge being an arrow from an entity to a relation $\xi \to R$, with $\xi \in \mathcal{E}$ and $R \in \mathcal{R}$, and the additional constraint that every relation R has at least two entities connected to it. These properties can be formalized using parents and children sets, namely for $R \in \mathcal{R}$, we have $\mathrm{pa}(R) \subseteq \mathcal{E}$, $\mathrm{ch}(E) = \emptyset$ and $|\,\mathrm{pa}(R)| \geq 2$, for $\xi \in \mathcal{E}$, we have $\mathrm{pa}(\xi) = \emptyset$, $\mathrm{ch}(\xi) \subseteq \mathcal{R}$ and $|\,\mathrm{ch}(\xi)| \geq 1$.

Furthermore, each relationship R can be represented by a matrix or tensor with an (multi)index set $\boldsymbol{i}$, with the number of indices being the same as the number of parent entities to the relationship. Alternatively the index set of each relation matrix or tensor is constructed by the concatenation of the index set of each of the parent entities. Now to convert our construction to full probabilistic model we can perform the following steps, basically assigning observations for each relationship, and latent-variables for each entities, taking care to instantiate shared latent variables for the entities connected to more than one relation.

- For each group of entities $\xi^e \in \mathcal{E}$ create a set of latent-variables $\boldsymbol{\xi}^e_{\boldsymbol{i_e}}$ according the the indices $\boldsymbol{i_e} \in \mathcal{I}(\xi^e)$.
  - Sample $\boldsymbol{\xi}^e_{\boldsymbol{i_e}} \sim \mathrm{Gamma}(a^e, b^e)$
- For each relation $R^n \in \mathcal{R}$ create a set of observable variables $\boldsymbol{R}^n_{\boldsymbol{i_n}}$, with the indices constructed by concatenated (cartesian product of) the parent entities indices $\boldsymbol{i_n} \in \bigtimes_{\xi \in \mathrm{pa}(R^n)} \mathcal{I}(\xi)$
  - Combine the parent latent variables $\lambda_{\boldsymbol{i_n}} = \prod_{\xi^e \in \mathrm{pa}(R^n)} \boldsymbol{\xi}^e_{\boldsymbol{i_e}}$, with $\boldsymbol{i_n}$ being the concatenation of the parent indices $\boldsymbol{i_e}$.
  - Sample $\boldsymbol{R}^n_{\boldsymbol{i_n}} \sim \mathrm{Poisson}(\lambda_{\boldsymbol{i_n}})$

We could extend this model by including different types of observations utilizing the compound Poisson distributions and different choices of distributions to be compounded. Furthermore we can allow the entities to have local (generated for for each relationship) and global (shared across all connected relationship) latent variables. Both modifications would not change drastically the inference algorithm, only increasing the number of parameters, as well the model flexibility. On Figure 4.5 and Figure 4.6 we represent the relational model and its equivalent plate notation for the generic factorization model. Further research and development is necessary to evaluate to potential is this model design.

Time modelling using point process 5

"'I seek through comprehensive anticipatory design science and its reductions to physical practices to reform the environment instead of trying to reform humans, being intent thereby to accomplish prototyped capabilities of doing more with less..."

— R. Buckminster Fuller, *Earth, Inc*

A guiding principle in recommender systems is to apply predictive modeling to infer users' preferences using their history of interactions with the system. In many cases, however, it is not possible to track a long history of interactions, instead, there is only information about recent *sessions* of interactions. Due to increased focus on data privacy over the last years, this situation is likely to become more prevalent. Session-based recommender systems address this issue by focusing on using information available in the sessions themselves. In particular, the predictive problem is cast as a *sequence prediction problem*, where a history with a sequence of actions and interactions taken by the users in a set of sessions are applied to predict the next k actions in a given session or a new session (Hidasi et al., 2016a).

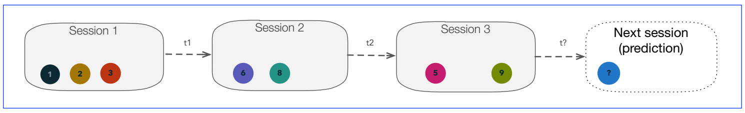


FIGURE 5.1: Representation of multiple sequential sessions with sequential item clicks and time between sessions

Return time prediction is another important challenge that has been explored using different strategies. The main challenge here is developing a time prediction model that uses previous sessions information to infer *when* the user is going to return or start a new session in a given application or service. This task is important for two reasons: 1.) Modeling user retention rate in web-services through their return time dynamics can offer deep insight into the service at hand, allowing the service decision-makers to better optimize their service to maximize time spent on the service. This is a valuable metric within web economy, which is fueled by advertisement (Kapoor et al., 2014). 2.) In the context of recommender systems, there is an interplay between temporal information and recommendations that can be exploited to harness predictive models and better understanding of users engagement with the recommender system (Wu et al., 2017).

The model developed tackle both tasks jointly, based on the assumption that the interplay between users interactions with items and their temporal information is synergetic in delivering predictors for both tasks. We propose a new neural model capable of delivering next–item predictions using Recurrent Neural Networks (RNNs) and return–time predictions using Temporal Point Processes (TPP). For example, consider the context represented in Figure 5.1, where there are a sequence of sessions for a given user and within each session a series on user–item interactions (clicks, purchases, etc) with a series of items (represented by the colored circles). Here, there is also information about the time before the user returns to the service and start a new session. Given the motivation given above, it is valuable to build a model with capabilities for capturing inter- and intra sessions user–item dynamics and using the past user–item dynamics and the temporal information to predict when a new session would start, as well as deliver recommendations.

To summarize, the main contributions of this chapter are:

- Defining and introducing a Temporal Hierarchical Recurrent Neural Network (THRNN): a joint model for inter-session and intra-session recommendations and return-time prediction. It consists in three major components, a module for sequence of sessions of representations, a module for sequences of user-item interactions within each session, a module for the time interval between the sessions. The first two modules consists on a Hierarchical Recurrent Neural Network (HRNN) architecture and the third module is given by a TPP which shares representations with the HRNN.

- A tuning mechanism in the training process that allows the time model to modulate the focus on short, medium or long time prediction. The mechanism consists of adding a control parameter in the loss function that allows us to control the importance of temporal information at training time.

Problem formulation

Given the set of user–item interactions; $S_{\text{train}} = \{(u, i, j, t_j) | u, i, j \in \mathbb{N}, t \in \mathbb{R}^+\}$, each tuple meaning that the user u interacted with the item i in a session j at time t_j.

We aim to learn a function that estimates a score for new items within a session, a score for an initial recommendation in the next session, and the time gap prediction between the current and next session. More formally, given our training set S_{train}, and an index j for a position within the session, we want a function $f(S_{\text{train}}, j)$ that will output a tuple $(s_{j+1}, f_{j+1}, g_{t_{j+1}})$, with the intra-session recommendation score s_{j+1}, the next session initial item score f_{j+1} and the time-gap $g_{t_{j+1}}$.

Thus the objective is to build a model that can predict new items within a session and for the next session, as well as predict the time-gap for the next session (return–time prediction).

Related Work

In recent years, numerous deep learning techniques have been successfully employed for recommendations. In particular the use of RNNs has been shown to be promising for session-based recommendation.

Session-based recommendation have been historically handled by item-to-item/ neighborhood-based methods (Sarwar et al., 2001) without considering the sequential nature of session data. A natural choice for modeling this sequential nature are RNN-based models such as those given in Hidasi et al. (2016a), Ruocco et al. (2017), Quadrana et al. (2017), Tan et al. (2016) and Jannach and Ludewig (2017). The GRU2REC architecture presented in Hidasi et al. (2016a) is widely credited to be the first to apply a RNN-based recommendation model and achieving state of the art results. The authors assume that the users with attached sessions are anonymous without any user-history, and assume that the sessions are independent of each other. Each session is modeled as a single layer of GRU units followed by a single feed-forward-layer. The GRU units, having recurrent connections, provide the state and captures the temporal dynamics of the sessions, while the feed-forward layer outputs the scores for each item. The model proposed was tested on two different datasets and achieved a 20-30% gain in the evaluated measures over the results of Rendle et al. (2009b), which is a nearest-neighbor-based model and was presented as the best performing baseline. A systematic comparison of multiple methods for session-based recommendation and the GRU2REC in Hidasi et al. (2016a) is presented in Ludewig and Jannach (2018). An extension of the GRU2REC model is presented in Ruocco et al. (2017), with a second level of RNNs that tries to capture inter-session dynamics. The proposed model is a hierarchical RNN where one level considers inter-session dynamics and the other considers intra-session dynamics. As opposed to assuming fully anonymous users and totally independent sessions, this extension allows some simple and low-cost user history to be considered in order to provide better recommendations for live sessions. The user history is in the form of abstract representations of previous sessions. The inter-session RNN is fed a finite number of the most recent session-representations in chronological order, and the output is used as the initial hidden state of the intra-session RNN. The motivation behind this is to handle the cold-start problem. In contrast to Hidasi et al. (2016a), Ruocco et al. (2017) achieved the best results when adding an embedding layer for the items. Additionally Ruocco et al. (2017) did not use the interleaved session mini-batching scheme, instead opting for padded sessions. Nor did they sample the output or use any pairwise losses. The work in Quadrana et al. (2017) proposes an architecture that is very similar to the model in Ruocco et al. (2017). They also propose an inter-session RNN layer in order to handle the cold-start problem when one has access to a limited user history in the form of previous sessions. The main difference between this hierarchical model and the one presented in Ruocco et al. (2017) is that Quadrana et al. (2017) only considered a session representation scheme based on the last hidden state of the intra-session RNN. Additionally, these session representations were created by passing the last hidden state to a final single layer feed-forward layer with a hyperbolic tangent activation function. They also experiment with propagating output of the inter-session network to all time-steps in the intra-session RNN, which complicates the model somewhat, but achieve slightly better results for one of the datasets evaluated. The hierarchical model proved to be the best performing overall by significant margin. In Tan et al. (2016), the approach proposed in Hidasi et al. (2016a) is improved by proposing a data-augmentation pre-processing step for improving the robustness of the model as well as proposing to output an item embedding, instead of the individual scores for each item, to make the recommendations faster. The work in Jannach and Ludewig (2017) presents an approach to adapt a nearest neighbor method for session-based recommendation task by finding sessions that are similar to the live session at the first step using an item cosine similarity measure. For predictions they use a weighted sum of nearest neighbour recommendations and RNN recommendations.

Other notable works on using RNNs have focused on how to best handle context- and feature rich input (Liu et al., 2016; Smirnova and Vasile, 2017; Hidasi et al., 2016b). In Liu et al. (2016) a simple intra-session RNN model is extended by training and evaluate using different sets of weights based on the context of the input. This context can for instance be time and date of different granularities, location, or weather at the time of an event. In Hidasi et al. (2016b) the focus is to feed feature-rich input in the RNN input, while inn Smirnova and Vasile (2017), the authors propose a new class of Contextual Recurrent Neural Networks for Recommendation (CRNNs) taking into account contextual information in two different ways. One consists in combining the context embedding and input embedding, similar to Twardowski (2016), and the other consists in incorporating it in the model dynamics. Attention mechanisms have also been used within the RNN setting to provide recommendations (Li et al., 2017). The idea here is to use an attention mechanism over hidden states to provide to capture the dynamics and dependence between the states.

Even though RNNs are inherently able to capture some temporal dependencies, this is largely based on the order of the events in a sequence. Aspects like time gap between events or sessions, the season, the year, the weekday, and the time of the day, are all temporal aspects that may influence the ideal recommendation in many domains, but which cannot easily be captured by the order of events alone. There have been attempts in both making time-aware recommendation by directly feeding such information in the RNN for the simple recommendation task (Liu et al., 2016), as well as modeling time for the task of predicting the return time of the next session (Du et al., 2016b; Jing and Smola, 2017; Zhu et al., 2017).

The model proposed in Du et al. (2016b) – based on a single layer RNN – attempts to predict the return-time of a user in addition to recommend the next item. The time prediction is modeled as a marked point process with intensity function conditioned on the history and the time to the next event. Similarly in combining recommendation and time-modeling, the model proposed in Jing and Smola (2017) use survival analysis to model the time instead of a marked point process. Another difference is that this model is used for next-basket recommendation task, and not just a single sequence of items. Furthermore, the time modeling is on the inter-basket/session time-gaps, and not the time between each selection. It means using a single level RNN for inter-session modeling directly, in contrast to Ruocco et al. (2017) and Quadrana et al. (2017), both implementing some form of inter-session modeling in one of the two levels of their hierarchical RNN models. The final model's recommendation capabilities was compared with two factorization based baselines and one based on neural networks. It was shown to outperform all of these on two different datasets. The time prediction was compared with expressive point processes, one of which was a Hawkes point process, and achieved smaller time errors than all of these. Finally, in Zhu et al. (2017), starting from the observation RNN units are good at modeling orders of entities, but does not offer any inherent support of time intervals between the entities, they proposed to incorporate the temporal aspect in the LSTM through explicit time gate.

Also, there has been efforts in using point process intensity modeling as an inspiration for hybrid Recurrent Neural Networks capable of time modeling. The approach in Du et al. (2016a) consist in creating a new type of recurrent neural unit with two outputs, one for the time model and another one for the general prediction model. The main contribution here is the proposal of a modeling approach for the intensity function using a combination of the hidden layers for long time dependency and recent time prediction, and combination the usual prediction loss with the negative log-likelihood of the point process based on the intensity function. This results in a model capable of predicting markers and time of the markers, with an intensity function for the time model that is learned via a neural architecture. In Xiao et al. (2017) a similar approach is taking, modeling the intensity function of a point process via neural architecture, however in this case two RNN's are employed: one for time-series data, capturing background intensity rate, and another one for event data, capturing long range relationship between events. Finally, Mei and Eisner (2017) is proposing a continuous time LSTM using Hawkes process as a starting point and making changes in the internal dynamics of the LSTM to achieve that goal. The resulting model is a LSTM with similar properties as the Hawkes process.

5.1 Hierarchical multi-session RNN

We propose a system that aims to predict the return time for the next session and to recommend the next item. It is based on a hierarchical RNN (HRNN) model enhanced with a time model part. The inputs of the HRNN are the representations of previous sessions together with contextual information, followed by the item representations in the session. Figure 5.2 shows an overview of the system.

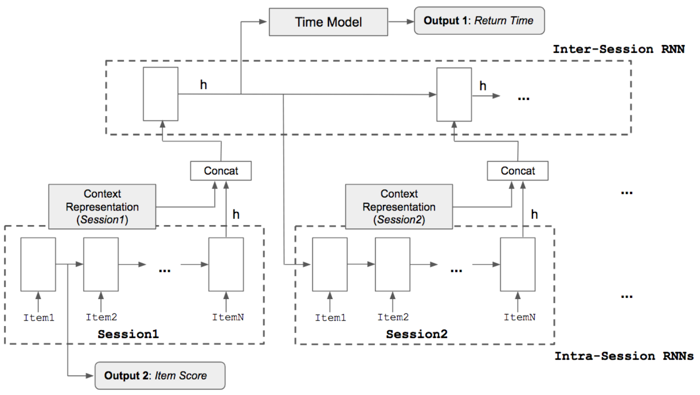


Figure 5.2: Schematics of the proposed model consisting of a hierarchical multi-session RNN with a time model with point processes

The HRNN is inspired by Ruocco et al. (2017), and consists of an *inter-session RNN* and an *intra-session RNN*. GRU units are used in both the intra-session and the inter-session RNNs. GRU was chosen due to its ability to remedy the vanishing gradient problem and since it was found to work better than LSTM for this problem and model structure. Like in Ruocco et al. (2017), the inter-session RNN is fed with a fixed number of preceding session-representations. Moreover, in the proposed model, relevant session-related contextual information is concatenated to this input.

The final hidden state of the inter-session RNN is propagated to the intra-session RNN mainly representing the inter-session information, as well as employed for the return-time prediction task. The intra-session RNN uses the last hidden state of the inter-session RNN as initial state, and item representations in input. For each item embedding in the input, the corresponding output of the RNN is passed to a linear layer which outputs scores for each target items. The recommendation is given by selecting the items with highest scores.

5.1.1 Context Representation

Additional session-related contextual information, is concatenated with the last hidden state of the intra-session RNN, representing the input of the inter-session RNN. We consider three types of embeddings used in the main setup of the model: 1) Item embeddings, 2) inter-session gap-time embeddings and 3) user embeddings. The purpose of such embeddings is to learn finer dynamics and representations of the embedded entities. For instance, if two different artist often are listened to by users with similar tastes, an embedding layer would most likely learn artist representations that in some sense are similar to each other. By using RNNs, such representations can learn temporal dynamics in addition to a more general "similarity measure". A simplified example of such representational knowledge could be: "Artist A is almost always listened to before Artist B". Embeddings can also learn dissimilarity and non-linearity. For instance when thinking about gap times between sessions, there might be a higher correlation between gap sizes of 24 and 48 hours (periodic daily behavior) than between 12 and 24 hours.

5.1.1.1 Item embeddings

. These embeddings represent each unique item in the dataset. The learned embeddings are directly handled by the intra-session RNN and consequently only trained by the resulting loss of this part of the model. This means that the item embeddings will affect earlier parts of the network, but then as input, not computational graphs. Hence, their gradient are unable to flow back to the embedding layer, and cannot be used to train the embeddings further. Due to the large number of items present in the considered datasets, the dimensionality of this embedding is the largest by far.

5.1.1.2 Inter-session gap-time embedding

These embeddings represent the time gaps between sessions. The time-gap is first normalized and then divided into discrete buckets. The resulting bucket IDs can then be used to index embedding tables/layers to propagate the corresponding embedding. Two different normalization schemes were examined. Both are first given an upper bound, which sets a threshold of the gap time after which we don't consider the user active enough to be provided accurate time predictions. The gap-times that are greater than this bound is set to the upper-bound. The first normalization scheme, divides the gap-time range in uniformly large buckets. The benefit of this is that all the values in the gap-time range will belong to a bucket of equal size, causing no gap-time to be in the same bucket as a much higher or lower gap-time. A disadvantage of using this is that the earlier "popular" buckets can be overcrowded and the later ones can end up being almost empty, which can make such embeddings hard to train. One also needs a high resolution to cover the finer differences in the smaller time-gap ranges, which further increases the problem of sparse buckets. In the second normalization scheme, the gap times are first transformed with a log function, before the transformed range is divided uniformly into buckets. This results in a more evenly distributed number of gap times in the different buckets, but at the cost of cruder resolution for larger gap-times where the corresponding buckets cover much more time than the earlier ones. We observed that the second scheme is performing better with small resolutions, but was overall out-performed by the uniform scheme with higher resolution. Since having high resolution is a non-issue, both with regards to model performance and run-times, the uniform was deemed the better option.

5.1.1.3 User embedding

These embeddings are mainly inspired by Jing and Smola (2017), and learning the user behaviour beyond the history of session representations is especially useful with long user histories. For instance, if a user behaves a bit unusual and sporadic in the last few sessions, a model with user embeddings can have information about long term user behavior, which can help making sense of/override the recent noisy behavior.

5.1.2 Time Model

The main goal of this model is to predict the time until the next session start, given a fixed length history in the form of abstract session representations and corresponding contextual temporal information. The time modeling is heavily inspired by the work in Du et al. (2016a), where time modeling is used to both predict the the time of the next item recommendation given a single sequence of previous items, as well as to improve the recommendation. In their model, the gap-times between selected items are considered to be drawn from a marked point process. The parameterization of the marked point process is defined by the authors and is dependent on the previous selection history modeled as a RNN, with corresponding inter-selection gap-times, and on the time of the last selection. In our system the history is based on session representations and not individual items. Consequently, the corresponding time-gaps are the times between the session representations in the history. This inter-session modeling, as well as the concatenated embeddings, is more similar to what was done in Jing and Smola (2017). In the point process, we define the intensity function as:

$$\lambda^*(t) = \exp(v^{t\top} \cdot h_j + w^t \cdot g_j + b^t) \tag{5.1}$$

where h_j is the j-th hidden state, $g_j = t - t_j$ - the time since the last session (in which t_j is the last timestamp in the last session and t is the time variable), v is a vector of weights with the same dimensionality as h_j, w^t is a single weight and b^t is a bias term. $v^{t\top} \cdot h_j$ comprises the historical influence on the intensity function, while $w^t \cdot g_j$ is the current influence. The full conditional density function is then defined as follows:

$$f^*(t) = \lambda^*(t)\exp\biggl(-\int_{t_j}^{t} \lambda^*(\tau)d\tau\biggr) \tag{5.2}$$

finally, where the intensity Equation 5.1 has been substituted into the conditional density function Equation 5.2, the full expression of the marked point process is:

$$\begin{aligned} f^*(t) = \exp\biggl\{ & v^{t\top} \cdot h_j + w^t \cdot g_j + b^t + \frac{1}{w^t}\exp(v^{t\top} \cdot h_j + b^t) \\ & - \frac{1}{w^t}\exp(v^{t\top} \cdot h_j + w^t \cdot g_j + b^t)\biggr\} \end{aligned} \tag{5.3}$$

The expected return time $\hat{t}_{j+1}$ is computed as the expected value in form of weighted area over the probability distribution as follows:

$$\hat{t}_{j+1} = \int_0^\infty t \cdot f^*(t)dt \tag{5.4}$$

The integration of the density distribution of the point process (Equation 5.3) does not have an analytic solution. Thus, prediction has to be approximated by numerical integration. To handle the infinite upper integration bound, a simple upper cut-off time is defined. While this is another approximation, the approximation becomes negligible when setting the cut-off time a bit higher than the vast majority of gap-times found in the data.

5.1.3 Loss

Creating a loss function is a matter of combining a recommendation loss and a return time loss. The latter is simply the log-likelihood of the time-gap point process distribution. On top of that, in order to control and tune the *importance* of the short, medium and long term data points in the training process, we changed the loss by adding the exponent $\alpha \in (0,1)$ to the time-step variable g_j.

Calculating the negative log-likelihood from Equation 5.3 and taking the derivative with respect to the weight w^t we obtain

$$\nabla_{w^t}[-\log f^*(t)] = -g_j + \frac{e^{v^{t\top} \cdot h_j + b^t}}{(w^t)^2}(1 + e^{w^t \cdot g_j}(w^t \cdot g_j - 1)) \tag{5.5}$$

The addition of the exponent on the time-step variable comes from looking at the gradient of the negative log-likelihood in relation to w^t. The gradient has two basic components related to the time-gap information, a linear component coming from the current session and a exponential component that is related to the past interactions in the TPP model (see discussion of Equation 5.1). This led to the conclusion that a simple mechanism to modulate the emphasis between shorter time prediction or longer time prediction is to exponentiate the time-interval variable g_j, since this operation would affect the linear and exponential part differently. Another way to see the same intuition is to observe that the negative log-likelihood is approximately linear on small time-gaps and exponential on large time-gaps, meaning that exponentiating the time-gaps to a number closer to zero would make the shorter time-gaps dominate the loss more than the larger time-gaps. This addition proved to be valuable for tuning trade-offs between short-medium-long time predictions as we will see in Section 5.2.5. The final time loss is:

$$\begin{aligned} L_{time}(g_j, h_j, w) = -\Bigg(v^{t\top} \cdot h_j + w^t \cdot g_j^{\alpha} + b^t + \\ \frac{1}{w^t}\exp(v^{t\top} \cdot h_j + b^t) \\ - \frac{1}{w^t}\exp(v^{t\top} \cdot h_j + w^t \cdot g_j^{\alpha} + b^t)\Bigg) \end{aligned} \tag{5.6}$$

The recommendation loss, $L_{rec}(s_{j+1}, i)$, is simply the softmax for item i given session representation at time $j+1$, i.e. s_{j+1}. We combine these losses using a weighted mean:

$$L_{total} = \beta_1 L_{time}(g_j, h_j, w) + \beta_2 L_{rec}(s_{j+1}, i) \tag{5.7}$$

Here, i is the target item of the intra-session recommendation. s_{j+1} is the score of the intra-session recommendations and g_j is the target time-gap.

5.2 Experimental Setting

In order to ensure the reproducibility of the experiments, and for further implementation details, we make our code available on a github repository [1].

5.2.1 Datasets

The evaluation is performed on two different datasets as in Ruocco et al. (2017): the *LastFM* dataset (Bertin-Mahieux et al., 2011) containing listening habits of users on the music website Last.fm and the *Reddit* dataset [2], on user activity on the social news aggregation and discussion website Reddit. Last.fm is a music website where users can keep track of the songs they listen to as well as sharing this with their peers. The data is in the form of tuples containing *user-id*, *artist*, *song* and *timestamp*, each representing a single listening event. Reddit is a popular forum/discussion website, where people can share and comment on different news, creations, pictures and other topics. Its structure is divided into different subforums, named subreddits, which define the topics/interests/allegiance of the posts to be posted there. This data is in the form of tuples containing a *user*, a *subreddit* and a *timestamp*, and each of these represents a single event where the user has commented on a post within the specific subforum at the given timestamp.

5.2.2 Data Preprocessing

The data were first preprocessed by removing noisy and irrelevant data and defining markers in the data following the steps in Ruocco et al. (2017). The Reddit dataset contains a log of user interaction on different subreddits (sub-forums), with timestamps. Here, an interaction is when a user adds a comment to a thread. Since the dataset does not split the events into sessions, we did this manually by specifying an inactivity time limit. Using the timestamps, we let consecutive actions that happened within the time limit belong to the same session. That is, for a specified time limit Δ_t, and a list of a user's interactions $\{a_{t_0}, a_{t_1}, \ldots, a_{t_n}\}$, ordered by their timestamps t_i, two consecutive interactions a_{t_i} and $a_{t_{i+1}}$ belong to the same session if $t_{i+1} \leq t_i + \Delta_t$. We set the time limit to 1 hour (3600 seconds) for the LastFM and 30 minutes (1800 seconds) for the Reddit dataset as in Ruocco et al. (2017). For

[1] `https://github.com/BjornarVass/Recsys`

[2] Subreddit interactions dataset: https://www.kaggle.com/colemaclean/subreddit-interactions

	Reddit	Last.fm
Number of users	18.271	977
Number of sessions	1.135.488	630.774
Sessions per user	62,1	645,6
Average session length	3,0	8,1
Number of items	27.452	94.284

Table 5.1: Statistics for the datasets after preprocessing

both dataset we then removed all L consecutively repeating items, reducing them to only one occurrence. Following (Ruocco et al., 2017) we set the maximum length, L, of a session to $L = 20$, split sessions that had a length $l > L$ into two sessions and removed sequences of length more than $2L$. As (Ruocco et al., 2017), we also simplified the LastFM dataset by ignoring the specific song of each user interaction and only use the artists. When modeling the inter-session time-gaps, the sessions that are split because their length are not considered separate sessions, since this would introduce noise. We solved this problem by setting the last timestamp in the first half and the first timestamp in the second half, to be the start time of the full session, resulting in a gap-time equal to 0. The contribution will then be masked away from the time loss, making sure the model does not to train on these gap-times. Finally, the datasets were split into a training set and a test set on a per user basis. Each user's sessions were sorted by the timestamp of the earliest event in the session, and the earliest 80% of his sessions were placed in the training set, while the remaining 20% of the sequences, that contain the most recent sessions of each user, were allocated to the test set. Table 5.1 shows statistics for the two datasets after preprocessing (before splitting into training and test sets).

5.2.3 Baselines

In order to validate the performance of our model for both the tasks of return session time prediction and next item recommendation the THRNN model is compared to the following baselines, in addition to the intra-session Hierarchical RNN itself

(HRNN). These baselines have been shown to be the strongest ones for such tasks as demonstrated in Ruocco et al. (2017) and Quadrana et al. (2017) for the session based recommendation and Du et al. (2016b) for time prediction.

GRU4REC [3]: implementation based on the seminal work of Hidasi et al. (2016a) that uses session-parallel mini-batch training process and ranking-based loss functions for learning a model using solely the items sequence information within a session.

HRNN The Hierarchical RNN (also Inter- and Intra-Session RNN), is a simpler version of the architecture used in this article. Namely, take the architecture presented in Figure 5.2, remove the time model and the user embeddings. The resulting model is the one given in Ruocco et al. (2017). The HRNN baseline parametrization is the same as in this article where it has been shown to otuperform the GRU4REC and other strongest baselines, such as Item K-NN (Linden et al., 2003) and BPR-MF (Rendle et al., 2009a)

Hawkes process The Hawkes process is a self-exciting point process with intensity given by

$$\lambda(t|\mathcal{H}_t) = \gamma_0 + \alpha_H \sum_{t_j \in \mathcal{H}_t} \gamma(t, t_j) \tag{5.8}$$

for some constants γ_0, α_H, a kernel γ and the history of events $\mathcal{H}_t$ (Hawkes, 1971). γ_0 defines a baseline intensity, while γ defines the propensity with which the intensity is excited by events in the process itself. The kernel typically has the property that if a number of events happen with short time gaps, the intensity λ becomes larger than if the same number of events happen with larger gaps. The history of event $\mathcal{H}_t$ has the list of event times up to time t, $\{t_1, ..., t_n | \forall j \leq n : t_j < t\}$. For the baseline, we fit one Hawkes process for each user (using the last 15 sessions), and set the kernel to an exponential function. Return-time predictions are expected values of the Hawkes process primed by the time gaps of the 15 most recent sessions for each user. We also used a *Hawkes long-term* fitting procedure, for which the entire per-user training dataset is used for estimating the parameters in the Hawkes process.

5.2.4 Evaluation Metrics and Hyper-Parameters tuning

We used *Recall@k* and *MRR@k* with $k = 5, 10, 20$ to evaluate all models for the recommendation task and *Mean Absolute Error* (MAE), for the return-time

[3] `https://github.com/hidasib/GRU4Rec`

	Reddit	Last.fm
Item Embedding size	50	100
User Embedding Size	10	10
Time-Gap Embedding Size	5	5
Learning rate	0.001	0.001
Learning rate-time	0.0001	0.0001
Dropout rate	0	0.2
Max. recent session representations	15	15
Mini-batch size	100	100
Number of GRU layers, intra-session level	1	1
Number of GRU layers, inter-session level	1	1

Table 5.2: Best configurations for the RNN models.

prediction. In addition to the baselines already discussed, we also compared the THRNN with other models on the two presented datasets. We experimented with mini-batch sizes, embedding sizes, learning rate, dropout rate, using multiple GRU layers, and number of session representations to find the best configurations for each dataset. The best configurations we found are summarized in Table 5.2. *Learning-rate time* refers to the learning rate of w and v^t respectively, in the Equation 5.6. This value had to be reduced in order to stop this loss from diverging, mainly due to the exponential function as well as the scaling with the time-gap in the formula. We then find the best scaling factors for the loss function in Equation 5.7 to be $\beta_1 = 0.45$ and $\beta_2 = 0.45$.

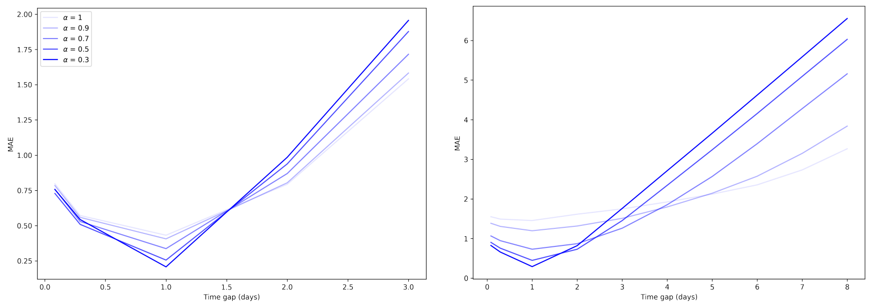


FIGURE 5.3: Time prediction MAE with different values of parameterα evaluated on LastFM dataset (left) and Reddit dataset (right). Runs with $\alpha = 0.1$ are not included since time-specific gradients diverged for this initialization

5.2.5 Effect of parameter α

In order to show the effect of the parameter α in Equation 5.6, we evaluate the model by varying $\alpha \in [0.3, 0.5, 0.7, 0.9, 1.0]$ for the task of return time prediction. In Figure 5.3 we see performance, in term of MAE, of the proposed architecture for both LastFM and Reddit dataset. In both cases we observe that with α close to 1.0 the MAE decrease for longer time gap, showing an increase of accuracy in the time prediction for such gaps. In particular, for the LastFM dataset the performance of all the initializations seems to meet at 1.5 days time-gaps which is the middle point between the $[0.5 - 1.5]$ days interval and $[1.5 - 2.5]$ days interval. We can also observe that the plot looks more linear the smaller α is set. For example, for $\alpha = 0.3$ the plot is similar to the one produced by predicting an averaged time-gap for every single prediction. For the LastFM dataset, we see that by decreasing the value of α, the focus of the model is to have better performance on smaller time-gaps but unlike the Reddit dataset, on the long-term gap, the performance of the model tends to deteriorate faster. This is probably due to differences in distribution of the time gaps for the two datasets.

For both datasets, Figure 5.3 clearly illustrates the trade-off between error for the most frequent time-gaps and the effective prediction range of the model. Choosing α should be based on the service for which we are modeling return times. For some services, modeling shorter time gaps are more important that modeling longer ones.

5.2.6 Return time prediction: Long Term vs Short Term

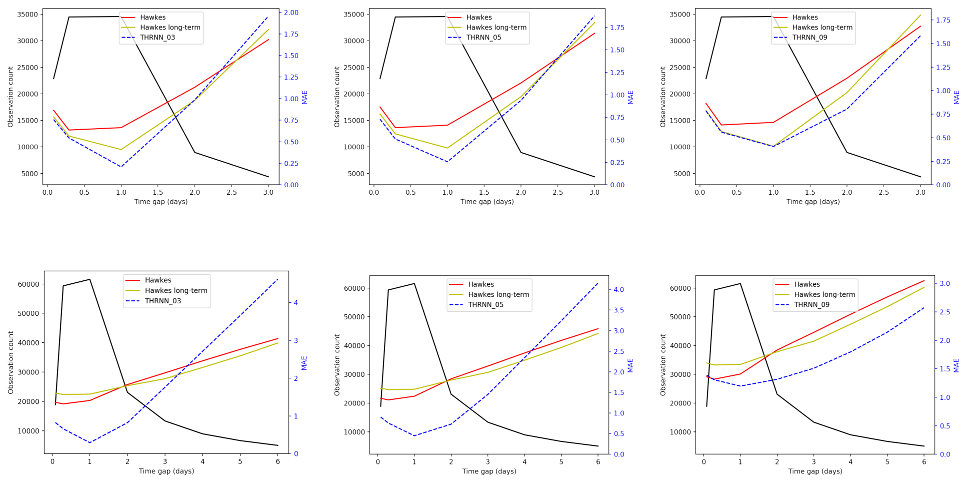


Figure 5.4: Time prediction MAE compared with Hawkes baselines with different values of parameter α. LastFM dataset (top row): α values of 0.3 (left), 0.5 (center), 0.9 (right) Reddit dataset (bottom row): α values of 0.3 (left), 0.5 (center), 0.9 (right).

In Figure 5.4, we illustrate the performance of the proposed model for return time predictions versus two baseline Hawkes models, with tuning parameter $\alpha \in [0.3, 0.5, 0.9]$ in Equation 5.6. For short time-gaps, the joint model of session embeddings and point process intensity gives better predictions than the baselines. The results are plotted alongside the number of observations for each time-gap, which gives insightful information about how the user–item interaction within a session is relevant for the model. We observe that the joint model outperforms the baseline consistently at time-gaps where there are more user–item observations. This is intuitive because of the shared learned parameters between the time model and the intersession model, in the sense that those parameters are not only fitting the time-gaps, but also the user–item interactions dynamics.

We see that tuning α allows us to control which time-gaps are most important to the model. Tuning for the most important time-gaps, however, comes at the cost of having worse return-time predictions for the time-gaps that are deemed less important.

	R@5	R@10	R@20	MRR@5	MRR@10	MRR@20
GRU4REC	0.1349 ± 0.0004	0.184 ± 0.0002	0.2474 ± 0.0002	0.086 ± 0.0003	0.0925 ± 0.0002	0.0969 ± 0.0002
HRNN	0.1415 ± 0.0005	0.1993 ± 0.0007	0.2751 ± 0.0006	0.0876 ± 0.0004	0.0952 ± 0.0004	0.1004 ± 0.0004
	(+4.9%)	(+8.3%)	(+11.2%)	(+1.8%)	(+2.9%)	(+3.7%)
THRNN	**0.1437 ± 0.0004**	**0.2026 ± 0.0006**	**0.2795 ± 0.0006**	**0.0889 ± 0.0002**	**0.0967 ± 0.0003**	**0.102 ± 0.0003**
	(+6.6%)	(+10.1%)	(+13.0%)	(+3.4%)	(+4.5%)	(+5.3%)

Table 5.3: Table with the recall and MRR on the LastFM dataset.

	R@5	R@10	R@20	MRR@5	MRR@10	MRR@20
GRU4REC	0.3208 ± 0.0004	0.3959 ± 0.0005	0.475 ± 0.0003	0.2346 ± 0.0006	0.2445 ± 0.0006	0.25 ± 0.0006
HRNN	0.4432 ± 0.0013	0.5316 ± 0.0009	0.616 ± 0.0012	0.317 ± 0.0016	0.3288 ± 0.0016	0.3347 ± 0.0015
	(+38.1%)	(+34.3%)	(+29.7%)	(+35.1%)	(+34.5%)	(+33.9%)
THRNN	**0.4468 ± 0.0013**	**0.5366 ± 0.001**	**0.6228 ± 0.0009**	**0.3191 ± 0.0015**	**0.3311 ± 0.0014**	**0.3371 ± 0.0014**
	(+39.3%)	(+35.6%)	(+31.1%)	(+36.0%)	(+35.4%)	(+34.8%)

Table 5.4: Table with the recall and MRR on the Reddit dataset.

5.2.7 Impact on actual recommendation

The experimental results for the recommendation task are summarized on Table 5.3 (for the LastFm dataset) and Table 5.4 (for the Reddit dataset). We observe an improvement in both metrics (Recall and MRR) at distinct levels (5, 10 and 20). This indicates that there is a win-win situation by incorporating intensity-based time modeling in the HRNN model, improving both recommendations and return time prediction when compared to respective baselines. However, it is worth noting that the difference between THRNN and HRNN is not just the time modeling, but also some added use of contexts in the THRNN model. It is also noticeable that the improvements of THRNN over HRNN, although significant, are less salient in absolute value than the improvements over GRU4REC. The simplest hypothesis is that the inter-session layer of HRNN is already capturing some of temporal dynamics between the sessions, for example it is possible that the latent representation are encoding time information correlated with the changes of user–item interactions from the end of one session to the beginning of the next session (for example if there are different profiles for distinct time-gaps of items that are typically accessed when finishing a session and items that typically are accessed in the beginning of the next session).

Prior specification in hierarchical models 6

"Pure reason therefore is that which contains the principles of knowing something entirely a priori. (...) For since such a science must contain completely both analytic and synthetic a priori knowledge, it is, as far as our present purpose is concerned, much too comprehensive. We will be satisfied to carry the analysis only so far as is indispensably necessary in order to understand in their whole range the principles of a priori synthesis, with which alone we are concerned."

— Immanuel Kant, *Critique of Pure Reason*

The choice of the particular priors, as well as other hyperparameters like the number of factors or components, has notable impact on the overall performance of hierarchical models. It is not an easy task to determine the effect of the prior in the overall model using classical principles: the priors may not have intuitive interpretation and for complex hierarchical models the relationship between the priors and data is poorly understood, ruling out subjective prior knowledge. For example, the behavior of the hierarchical Poisson matrix factorization model of Gopalan et al. (2015) depends on seven hyperparameters (six for defining the priors and one for the number of latent factors) in a non-trivial manner.

The hyperparameters are typically chosen heuristically or by an iterative process that explicitly evaluates the quality of multiple choices. The search can be automated with Bayesian optimization (Snoek et al., 2012), typically based on some proxy of the marginal likelihood, such as variational lower bound or leave-one-out cross validation (Vehtari et al., 2017), or directly on the performance in a downstream task, such as recommendation (Galuzzi et al., 2019). Both require carrying out posterior inference for every considered set of hyperparameters, adding significant computational burden and increasing overall training time by orders of magnitude. Furthermore, the result is only optimal for the chosen measure and inference method, unnecessarily tying model specification with inference.

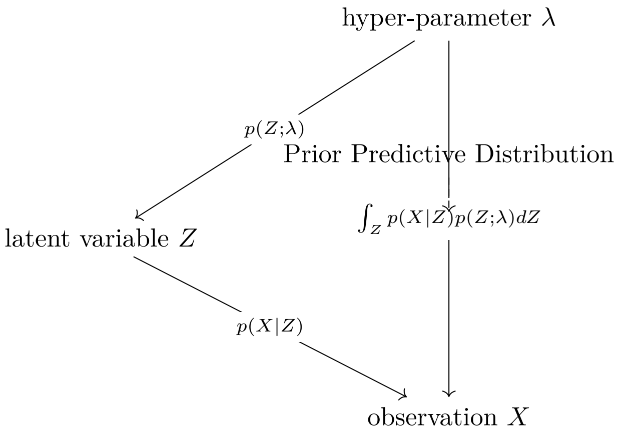

hyper-parameter λ
$p(Z;\lambda)$
Prior Predictive Distribution
latent variable Z
$\int_Z p(X|Z)p(Z;\lambda)dZ$
$p(X|Z)$
observation X

Figure 6.1: Diagram of a generic hierarchical Bayesian model

To overcome this, we turn attention to the statistical literature on the *prior predictive distribution* (PPD) – the marginal distribution of observables before seeing any data. The PPD is routinely used during the statistical modeling pipeline in form of prior predictive checks, to qualitatively access whether the model and the priors are reasonable (Schad et al., 2019; Gabry et al., 2019). PPD has also been used for prior elicitation, to convert knowledge an expert has on the properties of data into prior distributions (Kadane et al., 1980; Akbarov, 2009; Hartmann et al., 2020). We turn those ideas into a tool for automatic learning of hyperparameters, by directly optimizing for a good match between *virtual statistics* of the PPD and statistics of the data[1]. The tool can be used in two ways: (1) the target statistics are provided by the expert (user) as prior knowledge on data, or (2) the target statistics are estimated from (subset of) the actual data. The former is related to use of PPD for prior elicitation (Kadane et al., 1980), extended here for practical use with Bayesian ML models with large number of latent variables, whereas the latter is similar to empirical Bayes (Casella, 1985).

The proposed *prior predictive matching* approach, described in Section 6.1, finds good hyperparameters without needing posterior inference. When the true

[1]In order to distinguish from statistics of the observed data, we use the term *virtual statistics* to refer to summary quantities calculated for hypothetical data sampled from PPD, inspired by the phrase *virtual counts* used sometimes for the hyperparameters in count-data models.

data generating process is within the assumed model family, the approach provides hyperparameters that are optimal with respect to the selected statistics, and we show empirically that the method is robust for small model misspecification. If the data fits poorly with the assumed model family the approach may return unreasonable choices (e.g. recommending use of only a few factors for a recommender engine, when the common tradition is to operate with tens or hundreds of factors), which can be interpreted as sign of model mismatch and need for model refinement.

6.1 Prior specification via prior predictive matching

Priors for Bayesian MF Bayesian matrix factorization (BMF) is an important class of Bayesian ML models used, e.g., in recommender engines (Salakhutdinov and Mnih, 2007), for dimensionality reduction (Bai et al., 2013; Xu et al., 2003), community detection (Psorakis et al., 2011), and modeling relationships between data modalities (Klami et al., 2013). Importantly, it is a family for which the prior distributions are difficult to specify, as will be clarified in Section 6.1.1. We start by characterizing two concrete models building on Poisson distribution, for which the effect is emphasized (Cemgil, 2009).

Poisson Matrix Factorization. Poisson matrix factorization (PMF) (Cemgil, 2009; Gopalan et al., 2014a) with latent dimensionality K specifies a generative model for a count matrix $\mathbf{Y} = \{Y_{ij}\} \in \mathbb{R}^{N \times M}$, with each entry Y_{ij} following a Poisson distribution with rate $\theta_{ik}\beta_{jk}$, a product of latent factors θ_{ik} indexed by the rows and β_{jk} indexed by the columns.

Each latent variable follows a prior $f(\mu, \sigma^2)$, parameterized here using mean μ and standard deviation σ

$$\begin{aligned} \theta_{ik} &\overset{\text{iid}}{\sim} f(\mu_\theta, \sigma_\theta^2), \quad \beta_{jk} \overset{\text{iid}}{\sim} f(\mu_\beta, \sigma_\beta^2), \\ Y_{ij} &\overset{\text{iid}}{\sim} \text{Poisson}\left(\sum_{k=1}^{K} \theta_{ik}\beta_{jk}\right). \end{aligned} \tag{6.1}$$

The majority of the PMF literature assumes the priors to be gamma distributions (using shape-rate parameterization, $f(\mu_\theta, \sigma_\theta^2) = \text{Gamma}(a, b)$ and $f(\mu_\beta, \sigma_\beta^2) = \text{Gamma}(c, d)$, with $\mu_\theta = \frac{a}{b}$, $\sigma_\theta^2 = \frac{a}{b^2}$, $\mu_\beta = \frac{c}{d}$ and $\sigma_\beta^2 = \frac{c}{d^2}$) for efficient posterior inference, but we use the more general notation to extend the analysis to all scale-location priors.

The priors and the number of factors K control the *sparsity* and *magnitude* of the latent representation (Cemgil, 2009), via the expected mean and variance of the rates. However, these effects are hard to separate from each other and in practice the match can only be checked against the observed data *a posteriori*.

Compound Poisson Matrix Factorization. Compound Poisson matrix factorization (CPMF) (Basbug and Engelhardt, 2016) extends PMF by incorporating an additive exponential dispersion model (EDM) (Jorgensen, 1987) in the observation model, while keeping the Poisson-Gamma factorization structure:

$$\begin{aligned} &\theta_{ik} \sim f(\mu_\theta, \sigma_\theta^2), \quad \beta_{jk} \sim f(\mu_\beta, \sigma_\beta^2) \\ &Y_{ij} \sim \mathrm{ED}(w, \kappa n_{ij}), \quad n_{ij} \sim \mathrm{Poisson}(\sum_{k=1}^{K} \theta_{ik}\beta_{jk}), \end{aligned} \tag{6.2}$$

where we have $p(Y_{ij}|n_{ij}; w, \kappa) = \exp(Y_{ij}w - \kappa n_{ij}\psi(w))h(Y_{ij}, \kappa n_{ij})$, $\mathbb{E}[Y_{ij}|n_{ij}; w, \kappa] = \kappa n_{ij}\psi'(w)$ and $\mathbb{V}[Y_{ij}; w, \kappa n_{ij}] = \kappa n_{ij}\psi''(w)$, and n_{ui} is a Poisson distributed latent count[2]. $\mathrm{ED}(w, \kappa n_{ij})$ represents an EDM distribution, with natural parameter given by w and dispersion given by κn_{ij}, and the particular distribution determined by the base log-partition function $\psi(w)$ and base-measure $h(Y_{ij}, \kappa n_{ij})$. This model family includes Normal, Poisson, Gamma, Inverse-Gamma, and many other distributions (see Table 1 in Basbug and Engelhardt (2016)).

The data generating distribution is influenced by both the chosen EDM distribution and the hyperparameters, now including also κ and w, and a precise a priori reasoning about their joint effect is beyond feasible even for well-versed practitioners. An intuitive view of this model is that it allows us to decouple the sparsity or dispersion from the response model (controlled by the choice of distribution to be compounded). In this sense κ would give an indication about variability of the responses, while w would be related to the natural parameterization of the response distribution (see Table 1 in Basbug and Engelhardt (2016)). Determining specific values for these parameters to achieve desired or expected characteristics for the data is, however, extremely difficult.

6.1.1 On Priors for BMF

Despite the vast literature on BMF models and their inference algorithms, surprisingly little effort has been dedicated to the choice of the priors. Instead, most authors work with heuristic choices, possibly motivated by the general rule of selecting priors before observing any data, even though this may be ill-suited for general-purpose models. For example, Brouwer and Lio (2017) present a compendium of BMF models with wide range of likelihoods and priors, but despite

[2]We denote $\psi'(w) = \frac{d\psi}{dw}$, $\psi''(w) = \frac{d^2\psi}{dw^2}$

focusing on small data applications still set the hyperparameters by either trying values from a regular grid or setting them to fixed values based on claims of insensitivity. Similarly, Gopalan et al. (2015) solved the problem of choosing hyperparameters for hierarhical PMF by setting all of them to 1 or 0.3 to encourage sparsity, and Basbug and Engelhardt (2016) used a combination of empirically tested and heuristically chosen hyperparameter values for CPMF. Finally, Tan and Fevotte (2013) optimized for the latent dimensionality K, but used ah-hoc values for other hyperparameters. Importantly, we stress that these examples should not be seen as weakness in these particular works, but rather as examples of a common practice motivating our research – we have also published articles where we followed the same convention.

We argue that the common practice of heuristic choices is not because BMF models are particularly insensitive to the priors, but because selecting them is not easy. Even simple models have multiple hyperparameters that typically do not have clear meaning, ruling out subjective prior knowledge. For example, the only prior information for setting the variance parameters σ^2 of (6.1) would be based on what kind of values have worked before when applying the model for other data sets, rather than an expert being able to somehow quantify a real subjective knowledge based on domain knowledge. Furthermore, the hyperparameters are typically not even identifiable due to the latent variables relating to data only via the product $\theta_i^T \beta_j$. In general, this is a characteristic of hierarchical, high-dimensional and complex Bayesian models, where the interplay between prior specification and the final model properties is difficult to intuit aprioristically and can only be understood in connection to the likelihood and the predictions that come from it, as is argued by Gelman et al. (2017).

Global optimization for BMF, in turn, is hard because (1) training/validation split is non-trivial for structured data, (2) posterior inference is slow for large data, and (3) the optimization surface is difficult. Figure 6.4 illustrates the last point for the PMF model by evaluating the predictive quality of the mean-field variational approximation as measured by PSIS-LOO (Vehtari et al., 2017) for a range of hyperparameter choices (see Section 6.5.1 for details). There is large areas of inappropriate choices and the sharp border between those and the feasible region makes global optimization hard. Furthermore, as illustrated on the right-most bottom plot in Figure 6.4 (an example of a 1D slice plotted using three different scalings for the y-axis), the optimization surface characteristics depend on the inspection scale; the small neighborhood with optimal scores is lost at coarses scales and is difficult to find with most optimization strategies.

In Section 6.3 we will describe analytic solution for learning the hyperparameters for this model by matching the virtual statistics of PPD with target values. The example already illustrates how that method, which here does not require any computation besides simple equation, finds a solution surface within the feasible region, selecting also the latent dimensionality automatically. It does not give the hyperparameters optimal for this specific evaluation metric and inference algorithm, nor should it. Instead, the result captures essential properties of the data with the PPD, resulting in an appropriate starting point for posterior inference.

6.2 Prior Predictive Matching

Having illustrated the challenges with BMF, we now proceed to provide a new method for determining hyperparameters of any Bayesian machine learning model, building on the idea of matching virtual statistics of prior predictive distribution as explained in detail below.

6.2.1 General Idea

Our goal is to select good hyperparameters λ for a probabilistic model $p(Y, Z, \lambda)$, where Z denotes actual model parameters and latent variables collectively, without directly computing the posterior quality of any particular model fit. That is, we want to avoid costly and potentially difficult global optimization requiring the selection of specific evaluation criterion for the quality of the final solution as well as training/validation split for the data – which can be challenging especially for MF models. Instead, we prefer to optimize an overall match between the model and the data characteristics.

To achieve this, we consider the prior predictive distribution

$$p(Y|\lambda) = \int p(Y|Z, \lambda)p(Z|\lambda)dZ,$$

which integrates out the parameters, and search for hyperparameters for which it matches the data distribution well. PPD is typically used for validating prior and modeling choices as part of the statistical modeling pipeline (Schad et al., 2019; Gelman et al., 2020), often by visual comparison of prior predictive samples and the data,. e.g., so that large deviation between the two is interpreted as an indication that the model should be modified (Gabry et al., 2019). We extend the idea to automate the prior choice, by optimizing for λ for which virtual statistics of PPD match sufficiently well with either prior knowledge of the user or empirical statistics for the available data. Even though we do not have analytic expression for PPD for most models of interest (for example, Bouveyron et al. (2019) demonstrates how tedious the derivations are even for MF with Gaussian likelihood), we can draw samples from it and use those for evaluating (and hence optimizing) the fit.

Sometimes we can also derive analytic expressions for certain moments of PPD, which can be utilized for more efficient algorithms. The central contribution here is an automatic process similar to prior predictive checks, providing a method that directly links prior knowledge (or estimates) about the data generating process and specification of hyperparameters of the model.

6.2.2 Method

The gist of our proposed method is to search for λ such that the PPD $p(Y|\lambda)$ and the data distribution $p(Y)$ or user's prior beliefs about $p(Y)$ (when following strictly the principles of Bayesian modeling framework) match as well as possible. We quantify the match using a collection of statistics T that capture the essential properties of the data, for example in form of central moments. The goal is to find λ such that the virtual statistics $\hat{\mathrm{T}}_\lambda$ of the PPD match some target statistics T^*. In ideal case, we find the optimal match where $\hat{\mathrm{T}}_\lambda = \mathrm{T}^*$. We use the phrase *virtual statistic* for $\hat{\mathrm{T}}_\lambda$ to emphasize that it does not correspond to any particular observed data, but can instead be thought of as the corresponding statistic computed for a hypothetical – or virtual – data set sampled from PPD.

This general formulation depends on two elements: (1) the choice of the statistics T (and associated discrepancy measure) used for evaluating the match, and (2) the choice of the specific target statistic values T^*. Together they define the optimality. Importantly, these two objects are fundamentally linked with each other: a richer set of statistics T leads to hyperparameter choice likely to be good in broader set of applications, but at the same time implies the need to be more careful when providing the target values T^*, while often making computation more difficult as well. Finally, let us note that computational algorithms solving for λ are agnostic to how T^* were obtained, but to clarify the broad scope of the developed machinery we explain three common use-cases with different way of defining T^*:

1. **Principled statistician:** Following the strict Bayesian principle, the target statistics may be provided by a domain expert, in form of the expected values for the statistics. When used in this form, the proposed method essentially becomes a prior elicitation method; the expert provides subjective information on what is to be expected regarding the data, and this is used for indirectly defining the prior over the model parameters, similar to e.g. Kadane et al. (1980) and Hartmann et al. (2020). Importantly, the expert only needs to provide statistics of the data and not of the model parameters, not necessarily needing to understand the model in detail.

2. **Held-out validation:** A somewhat more pragmatic approach is to use actual observed statistics T of a separate validation data as the target values T^*.

For example, in the case of a recommender engine we might use a subset of the users and items to estimate the target statistics T^* and find λ for which the virtual statistics of the PPD best match the observed ones. After this, this data subset is discarded and the remaining data is used for posterior inference and possible further computation steps with the hyperparameters fixed to the selected ones.

3. **Automatic inference:** Finally, the method can also be used in a fashion where we use the observed statistics T of all available data Y as the targets T^*, loosely following the concept of empirical Bayes (Casella, 1985). This breaks the fundamental idea of specifying the priors before observing the data, but in typical cases the statistics are of low dimensionality and only characterize the data roughly. We argue most practitioners should consider this a valid procedure, and we believe this will be the most common way of using the method. For example, if only using the mean and variance as statistics we are merely making sure the range of the data is approximately correct, and the method can be considered simply as an automatic replacement for the manual model refinement that would be carried out by inspecting whether the PPD roughly matches the observed data or not (Schad et al., 2019; Gabry et al., 2019; Gelman et al., 2020). Note that this reasoning would no longer hold if using very rich statistics, in the extreme case directly using individual data entries so that $\mathrm{T}^* = Y$, but in this work we only consider problems where T consists of a few low-order moments.

Throughout this work we use moments such as mean and variance as the statistics T, since they lead to computationally efficient algorithms applicable for reasonably broad model families, but the method would work for other choices as well. In particular, we go through details of two practical algorithms for different scenarios, demonstrated in the context of Bayesian MF models. In Section 6.3, we first look at cases for which we can compute certain low-order moments of the PPD analytically and hence can find a closed-form expression for λ corresponding to the optimal match $\hat{\mathrm{T}}_\lambda = \mathrm{T}^*$. This is ideal in terms of computation, but restricted in scope to specific models and statistics. Hence, in Section 6.4, we proceed to provide a general-purpose algorithm applicable to considerably broader family of models and statistics, formulated as explicit optimization of a discrepancy measure between the PPD and target statistics, using sampling-based estimates for the virtual statistics and stochastic gradient-descent (SGD) optimization. The method is applicable for all continuous hyperparameters and requires only ability to sampled from PPD, but alternative forms of optimization could be considered to extend support also for discrete hyperparameters.

6.3 Matching Moments for PMF and CPMF

PMF as specified in (6.1) allows us to derive analytic expression for certain moments of the PPD, to be used as virtual statistics. If we denote by Y a virtual data matrix following the PPD, we can compute the mean $\mathbb{E}[Y_{ij}|\lambda]$, the variance $\mathbb{V}[Y_{ij}|\lambda]$, and the correlations $\rho[Y_{ij}, Y_{tl}|\lambda]$ in closed form. Here the hyperparameters are $\lambda = \{K, \mu_\theta, \sigma_\theta^2, \mu_\beta, \sigma_\beta^2\}$, and we drop explicit conditioning on λ, writing $\mathbb{E}[Y_{ij}] := \mathbb{E}[Y_{ij}|\lambda]$. The detailed derivations, provided in the Section 6.6, build primarily on the laws of total expectation, variance and covariace for marginalizing out θ and β.

Proposition 6.1. *For any entry of the count matrix $Y = \{Y_{ij}\} \in \mathbb{R}^{N\times M}$, the mean and variance is given by:*

$$\mathbb{E}[Y_{ij}] = K\mu_\theta\mu_\beta \tag{6.3}$$

$$\mathbb{V}[Y_{ij}] = K[\mu_\theta\mu_\beta + (\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2] \tag{6.4}$$

Proposition 6.2. *For any pair of entries Y_{ij} and Y_{tl} of matrix Y, their correlation is given by:*

$$\rho[Y_{ij}, Y_{tl}] = \begin{cases} 0, \textit{if } i \neq t \ \ \& \ \ j \neq l \\ 1, \textit{if } i = t \ \ \& \ \ j = l \\ \rho_1, \textit{if } i = t \ \ \& \ \ j \neq l \\ \rho_2, \textit{if } i \neq t \ \ \& \ \ j = l \end{cases} \tag{6.5}$$

$$\rho_1 = \frac{(\mu_\beta\sigma_\theta)^2}{\mu_\theta\mu_\beta + (\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2}$$

$$\rho_2 = \frac{(\mu_\theta\sigma_\beta)^2}{\mu_\theta\mu_\beta + (\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2}$$

Given Propositions 6.1 and 6.2 and some target values for the moments, we can directly solve e.g. for the number of latent factors K. Denoting $\tau = 1 - (\rho_1 + \rho_2)$, we obtain:

$$K = \frac{\tau\,\mathbb{V}[Y_{ij}] - \mathbb{E}[Y_{ij}]}{\rho_1\rho_2}\left(\frac{\mathbb{E}[Y_{ij}]}{\mathbb{V}[Y_{ij}]}\right)^2. \tag{6.6}$$

We also obtain formulas for relationships between means and standard deviations of the priors that can be used to set e.g. the Gamma hyperparameters a, b, c and *d*.

$$a = \frac{\rho_2 \, \mathbb{V}[Y_{ij}]}{\tau \, \mathbb{V}[Y_{ij}] - \mathbb{E}[Y_{ij}]} \tag{6.7}$$

$$c = \frac{\rho_1 \, \mathbb{V}[Y_{ij}]}{\tau \, \mathbb{V}[Y_{ij}] - \mathbb{E}[Y_{ij}]} \tag{6.8}$$

$$bd = \frac{\mathbb{E}[Y_{ij}]}{\mathbb{V}[Y_{ij}]} \sqrt{\frac{ac}{\rho_1 \rho_2}}. \tag{6.9}$$

Compound Poisson matrix factorization For CPMF we have

$$K = \frac{\tau \, \mathbb{V}[Y_{ij}] - \left(\kappa \psi'(w) + \frac{\psi''(w)}{\psi'(w)}\right) \mathbb{E}[Y_{ij}]}{\rho_1 \rho_2} \left(\frac{\mathbb{E}[Y_{ij}]}{\mathbb{V}[Y_{ij}]}\right)^2 \tag{6.10}$$

for the latent factors and the following relationships for other terms:

$$\begin{aligned}
\sigma_\theta \sigma_\beta &= \frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}] \kappa \psi'(w)} \sqrt{\rho_1 \rho_2} \\
\mathbb{E}[Y_{ij}] &= \kappa \psi'(w) K \mu_\theta \mu_\beta \\
\mathbb{V}[Y_{ij}] &= \kappa \psi''(w) K \mu_\theta \mu_\beta + [\kappa \psi'(w)]^2 K [\mu_\theta \mu_\beta + (\mu_\beta \sigma_\theta)^2 + (\mu_\theta \sigma_\beta)^2 + (\sigma_\theta \sigma_\beta)^2].
\end{aligned}$$

The derivations are provided in Section 6.6 in Propositions 6.6, 6.7, 6.9, 6.10 and 6.13.

Observations The above equations provide closed-form expressions that determine priors optimal in terms of virtual statistics of PPD matching the target statistics. They can be computed instantaneously, bypassing the need for optimizing λ, and provide the first computationally efficient way of automatically determining the number of factors for PMF and CPMF. The result is not necessarily optimal for any particular task, especially when the data does not follow the model well, but as illustrated experimentally in Section 6.5 tends to be a good choice.

6.3.1 Empirical Estimates for the Moments

As explained in Section 6.2, one way of using the method is based on matching the virtual statistics with the true statistics of the observed data. For MF models, we only have a single matrix representing one (often partial) observation, and hence need to estimate the statistics by averaging over the rows and columns of this one observation instead of averaging over multiple matrices. The mean and variance can

Algorithm 1: Empirical correlations ρ_1 and ρ_2 for a observed matrix $\mathbf{Y} = \{Y_{ij}\} \in \mathbb{R}^{N \times M}$

```
input  : observed matrix $\mathbf{Y} = \{Y_{i,j}\} \in \mathbb{R}^{N \times M}$, number of samples for the
         estimator $S$
output : $\rho_1$ and $\rho_2$
Initialize arrays $\mathbf{A} = \{A_{i,j}\} \in \mathbb{R}^{S \times 2}$ and $\mathbf{B} = \{B_{i,j}\} \in \mathbb{R}^{S \times 2}$;
for $s \in \{1, \cdots, S\}$ do
    Sample $i \sim \text{Unif}(\{1, \cdots, N\})$;
    Sample $j_1 \sim \text{Unif}(\{1, \cdots, M\})$;
    Sample $j_2 \sim \text{Unif}(\{1, \cdots, M\} \setminus \{j_1\})$;
    $A_{s,1} \leftarrow Y_{i,j_1}$;
    $A_{s,2} \leftarrow Y_{i,j_2}$;
    Sample $j \sim \text{Unif}(\{1, \cdots, M\})$;
    Sample $i_1 \sim \text{Unif}(\{1, \cdots, N\})$;
    Sample $i_2 \sim \text{Unif}(\{1, \cdots, N\} \setminus \{i_1\})$;
    $B_{s,1} \leftarrow Y_{i_1,j}$;
    $B_{s,2} \leftarrow Y_{i_2,j}$;
end
Calculate and return the Pearson correlation of columns of $\mathbf{A}$ and $\mathbf{B}$;
```

be easily estimated over the independent matrix entries, but it is not immediate nor intuitive how to compute the correlations. One remark is that there are two values of correlations, one for row and another one for columns, hence we can levarage this property of the model to make an estimator. To estimate the correlation we derived an estimator that samples two elements from the same row or column and uses them to calculate the correlation of elements sharing a row or column index, independent of the specific index. The estimator is described in Algorithm 1.

6.4 Gradient-based Approach

Deriving analytic expressions for even simple models and moments is tedious, error-prone, and often impossible. For users willing to give up the convenience and robustness of analytic expressions, we next provide an optimization-based alternative that does not require model-specific derivations.

6.4.1 Formulation as an Optimization Problem

Instead of directly equating the target and virtual statistics, we solve for λ that minimizes a discrepancy measuring the difference between the requested quantities T^* (either expert's prior expectation or empirical estimate) and the virtual statistics $\hat{\mathrm{T}}_\lambda$ of the PPD

$$\min_\lambda \mathrm{d}\left(\mathrm{T}^*, \hat{T}_\lambda\right), \tag{6.11}$$

where for brevity we write $\hat{\mathrm{T}}_\lambda$ instead of the complete $\hat{T}(\mathbb{E}[g(Y)|\lambda])$. As before, T is a collection of statistics (e.g. central moments) defining which aspects are used for learning the hyperparameters. Intuitively, a richer set allows more accurate prior specification, but requires more careful choice of discrepancy as well. For example, to match at the same time an expected value E^* and variance V^* we can use $\mathrm{d} := (\mathrm{E}^* - \mathbb{E}[Y])^2 + (\mathrm{V}^* - (\mathbb{E}[Y^2] - \mathbb{E}[Y]^2))^2$, where $g(Y) = (Y, Y^2)$ and $\hat{\mathrm{T}}(E_1, E_2) = (E_1, E_2 - E_1^2)$.

The process builds on repeatedly drawing samples from PPD to estimate the virtual statistics $\hat{\mathrm{T}}_\lambda$, and solving for λ using some iterative algorithm. Importantly, this only requires the model code providing the samples and the target statistics T^*, and hence allows solving for the priors as part of the modeling pipeline without needing to consider any particular data.

6.4.2 Differentiable Moments' Estimators

We optimize (6.11) with stochastic gradient descent, using Monte Carlo approximation (Mohamed et al., 2020) for the prior predictive moments and automatic differentiation with reparameterization gradients (Figurnov et al., 2018) wherever available and using REINFORCE (log derivative trick) (Williams, 1992) elsewhere. For gradient-based optimization we require that $\mathrm{d}(\cdot)$ and $\hat{\mathrm{T}}$ are differentiable w.r.t their arguments, and that we can propagate gradient ∇_λ through $\mathbb{E}[g(Y)]$. We next show that this is possible for a rather general structure of hierarchical Bayesian models with outputs Y and latent variables Z (see Figure 6.2 for a conceptual illustration of the procedure).

The procedure is based on recursively applying the law of total expectation. The unconditional expectation of $g(Y)$ can be obtained by integrating out latent variables Z, but since an analytical form of it is not available, we proceed by performing a numerical approximation, where each of the integrals over latent variables $Z_1, \dots Z_l \dots Z_L$ is replaced by a sum over samples from respective (conditional) distributions. An estimate of the required gradient $\nabla_\lambda \mathbb{E}[g(Y)]$ is then obtained by propagating estimates of the gradients $\nabla_\lambda \mathbb{E}[g(Y)|Z_l]$ and $\nabla_\lambda \log p(Z_l|\dots;\lambda)$ backward through the computation graph.

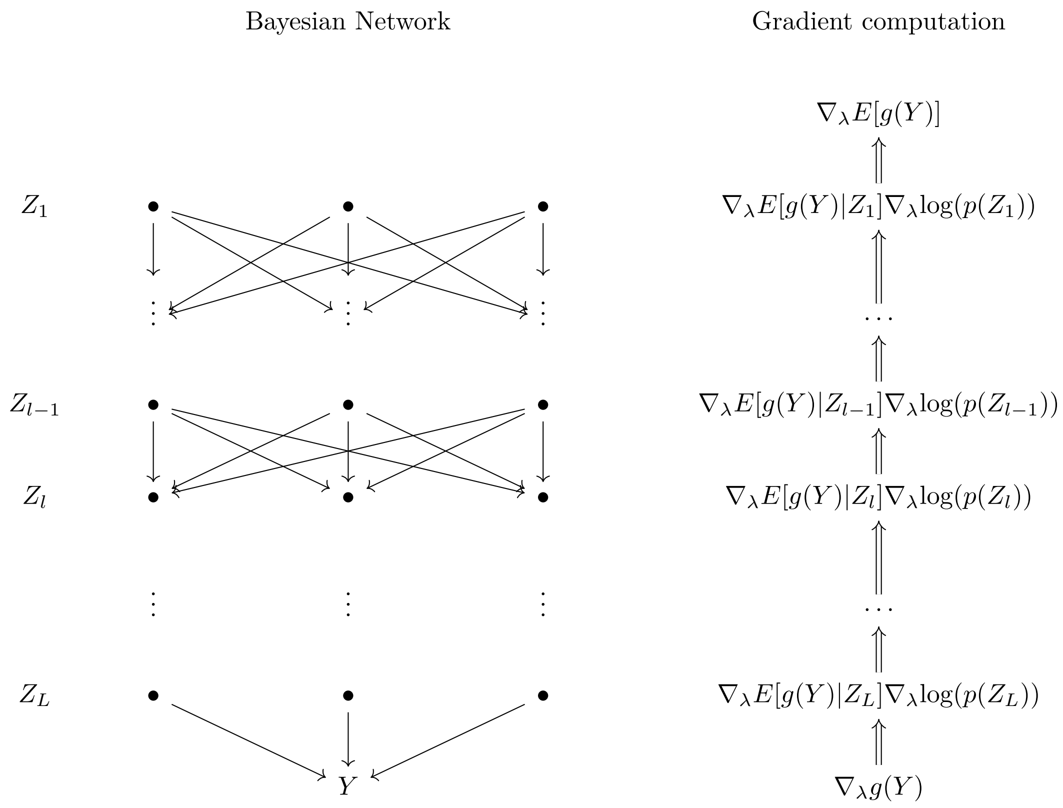

Bayesian Network
Gradient computation
$\nabla_\lambda E[g(Y)]$
Z_1
$\nabla_\lambda E[g(Y)|Z_1]\nabla_\lambda \log(p(Z_1))$
$\dots$
Z_{l-1}
$\nabla_\lambda E[g(Y)|Z_{l-1}]\nabla_\lambda \log(p(Z_{l-1}))$
Z_l
$\nabla_\lambda E[g(Y)|Z_l]\nabla_\lambda \log(p(Z_l))$
$\dots$
Z_L
$\nabla_\lambda E[g(Y)|Z_L]\nabla_\lambda \log(p(Z_L))$
Y
$\nabla_\lambda g(Y)$

FIGURE 6.2: Conceptual illustration of how moments' gradients can be estimated for Bayesian networks.

6.4.3 Example: Derivation for PMF and HPF

To demonstrate the rather generic presentation above and to link it to the BMF use-case, we show as an example the MC estimate for the expectation $\mathbb{E}[Y_{ij}]$ of the PMF model

$$\mathbb{E}[Y] \approx \frac{1}{S_\theta \cdot S_\beta} \sum_{\epsilon_\theta \sim p_0} \sum_{\epsilon_\beta \sim p_0} \mathbb{E}[Y|\theta(\epsilon_\theta, \lambda)^T \beta(\epsilon_\beta, \lambda)], \tag{6.12}$$

where we reparametrize both β and θ. For clarity, we also dropped indices i and j in Y_{ij}, θ_i and β_j. The internal conditional expectation in (6.12) we expand as

$$\mathbb{E}[Y|\theta^T\beta] \approx \frac{1}{C} \sum_{y\sim \text{Poisson}(\theta^T\beta)} y \cdot \mathit{Poisson}(y|\theta^T\beta), \tag{6.13}$$

where *Poisson* denotes Poisson probability mass distribution. Similarly, variance of Y we estimate with $\mathbb{V}[Y] = \mathbb{E}[Y^2] - \mathbb{E}[Y]^2$, where the estimator of $\mathbb{E}[Y^2]$ we obtain by substituting y with y^2 in (6.13).

To demonstrate the flexibility of the model-independent algorihm we also apply it on hierarchical Poisson factorization (HPF) model of Gopalan et al. (2015), for which we do not have closed-form expressions for the moments. This model adds one level of hierarchy to PMF, and hence matches our general formulation:

$$\begin{aligned} \theta &\sim \text{Gamma}(a,\xi), \quad & \xi &\sim \text{Gamma}(a', a'/b') \\ \beta &\sim \text{Gamma}(c,\eta), \quad & \eta &\sim \text{Gamma}(c', c'/d'), \end{aligned}$$

where the new continuous variables ξ, η we reparametrize (and sample) as follows:

$$\begin{aligned} \theta &:= \theta(\epsilon_\theta, a, \xi), \quad & \xi &:= \xi(\epsilon_\xi, a', a'/b') \\ \beta &:= \beta(\epsilon_\beta, c, \xi), \quad & \eta &:= \eta(\epsilon_\eta, c', c'/d'), \end{aligned}$$

Then, for HPF (6.12) takes the form of

$$\mathbb{E}[Y] \approx \frac{1}{S_\xi \cdot S_\eta} \sum_{\epsilon_\xi\sim p_0} \sum_{\epsilon_\eta\sim p_0} \underbrace{\frac{1}{S_\theta \cdot S_\beta} \sum_{\epsilon_\theta\sim p_0} \sum_{\epsilon_\beta\sim p_0} \mathbb{E}[Y|\theta(\epsilon_\theta, \lambda, \underline{\eta(\epsilon_\eta,\lambda)})^T \beta(\epsilon_\beta, \lambda, \underline{\eta(\epsilon_\eta,\lambda)})]}_{\mathbb{E}[Y|\xi,\eta]},$$

6.5 Experiments

In this section we present and discuss some of experimental validation for the proposed methods. We perform those validation using both synthetic data, in this case assuming the true model with different values for the hyperparameters, as well as predictive performance on a real-world dataset.

Setup of the experiments Table 6.1 contains initial configurations of hyperparameters used in the experiments. The legends in Figures 6.5 and 6.3 refer to these letters.

For PMF and CPMF, equations (6.50) and (6.10) provide analytic expressions for hyperparameters given target moments. We demonstrate them in an empirical

Table 6.1: Considered sets of hyperparameters.

	a	b	c	d	μ_θ	σ_θ	μ_β	σ_β	$\mathbb{E}[Y]$	$\mathbb{V}[Y]$
A	10	1	10	1	10.0	3.16	10.0	3.16	2500.00	55000.00
D	0.1	1	0.1	1	0.1	0.32	0.1	0.32	0.25	0.55
F	1	1	0.1	0.1	1.0	1.0	1.0	3.16	25.00	550.00

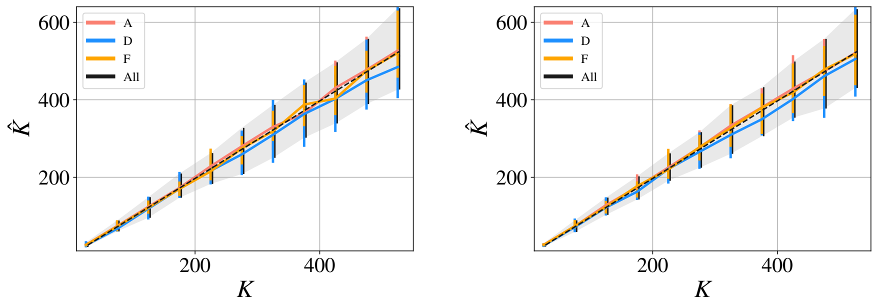


FIGURE 6.3: Prior predictive matching provides accurate estimates $\hat{K}$ for all true latent factor dimensionalities K and prior configurations (colored lines), as analytic expression of empirical moments for both Poisson MF (left) and Compound Poisson MF (right).

Bayes scenario, where estimates of the observed data are used as targets, to show the results are robust to estimation error. We sample a data matrix (of size $10^3 \times 10^3$) from the model for 30 scenarios where the true hyperparameters λ^* are set at different values. We repeat this for a range of values for the true K, and for each data compute the empirical estimates required for estimating the number of factors using (6.50) and (6.10). Figure 6.3 shows the estimates (mean and 95% confidence interval over the 30 replications) accurately match the ground truth when the data follows the model, for both PMF and CPMF with observation model $Y_{ij} \sim \sum_{i=1}^{n_{ij}} \mathcal{N}(1,1)$.

6.5.1 Posterior Quality

The main use for the approach is as part of a modeling process where we eventually carry out posterior inference using the selected hyperparameters. The quality of the

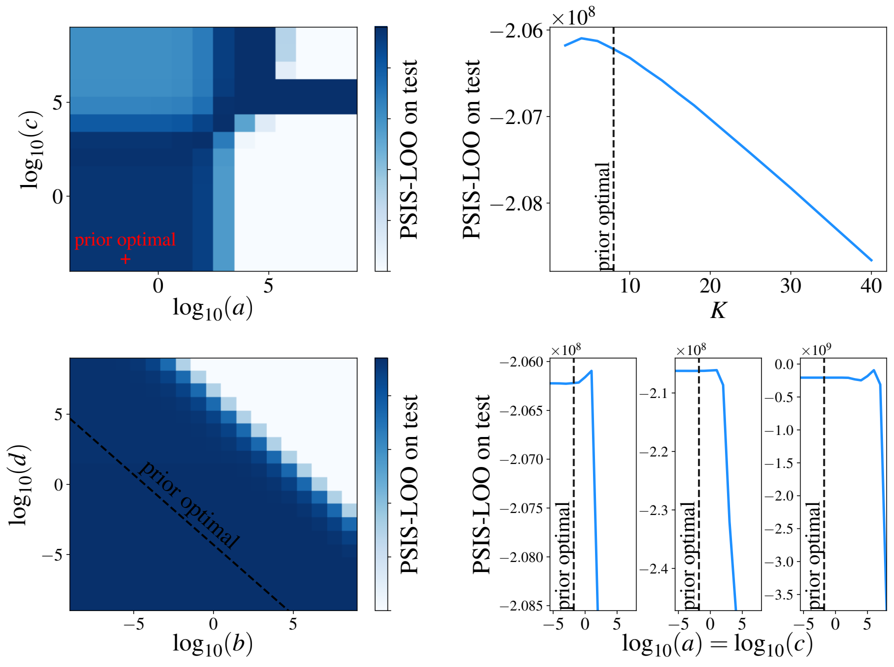


Figure 6.4: Illustration of difficulty of selecting good priors for Poisson matrix factorization, evaluated by predictive quality of a variational approximation on the `hetrec-lastfm` dataset. We show 2D (left) and 1D (right) slices of the loss surface in the five-dimensional hyperparameter space, with all other values fixed to prior optimal ones. The proposed prior predictive matching approach provides closed-form solution (indicated by "prior optimal"), including the latent dimensionality K (top right), within the area of reasonable values.

solution can hence only be evaluated by inspecting how the final model performs. We do this by fitting a PMF model to the `user-artists` data of the `hetrec-lastm` dataset (Cantador et al., 2011)[3], using an efficient implementation of coordinate ascent variational inference[4] fitted to randomly selected 90% subset of the data. We evaluate the quality of the model using the PSIS-LOO criterion (Vehtari et al., 2017) on the remaining unseen 10% data. PSIS-LOO is here considered as an

[3]`http://files.grouplens.org/datasets/hetrec2011`

[4]We extended `http://github.com/dawenl/stochastic_PMF` to support sparse data.

example of a typical task-agnostic metric a practitioner would be likely to use, but the experiment is not sensitive to the specific choice.

Figure 6.4 already illustrated the quality surface via explicit enumeration of the hyperparameter choices in a regular grid, shown as slices of the five-dimensional surface where all remaining parameters were fixed to the optimal ones provided by our method. Some prior choices have better PSIS-LOO scores – for example, slightly smaller K would be better – but the crucial observation is that prior predictive matching provides sufficiently good solution in an instant. It is also important to understand that these results depend on the specific inference algorithm and evaluation metric used, and the optimal solution would change – possibly by a lot – if the variational approximation was replaced, e.g., with MCMC and PSIS-LOO by a another metric. Hence, exactly matching whatever choice happens to be optimal here would not even be correct.

6.5.2 Sensitivity to Model Mismatch

In most applications, the data does not follow any model in the assumed model family. Since we compute the statistics conditional on the model, it is unclear how well the approach works when the model mismatch is severe. We conducted two experiments to evaluate this, by controlling the amount of mismatch on artificial data while assuming the PMF model and using the analytic expression (6.50) for setting the number of factors K. For both experiments we generated data with $K \in \{25, 50, 75, 100, 125, 150\}$ for two hypermateter configurations and generated 20 realizations for each configuration, and we summarize the results using relative error $\frac{\hat{K}-K}{K}$ to average over the data sets with different true K.

A typical mismatch with count MF models relates to sparsity; the data has more zeroes than expected under the model. To investigate the effect of this, we independently sample for each entry Y_{ij} (sampled from PMF) a Bernoulli variable $X_{ij} \sim \text{Ber}(p_{obs})$ controlling whether the entry is observed, so that our final observation is given by $\tilde{Y}_{ij} = X_{ij} \times Y_{ij}$. Decreasing p_{obs} increases model mismatch, and Figure 6.5 (left) shows this also increases relative error in the estimate $\hat{K}$, but the decline is graceful. For some configurations (blue) the relative error stays below 25% even after dropping 10% of observations. Note that for this kind of model mismatch the retrieved hyperparameter is consistently larger than the true one; additional components are required to explain the increased variance caused by the excess zeroes.

The second experiment considers another prototypical model mismatch, a scenario where a too simple model class is used. We use PMF as the model, but generate the data so that the Poisson likelihood is replaced with negative binomial (NB) with varying rate of overdispersion that PMF cannot account for. More

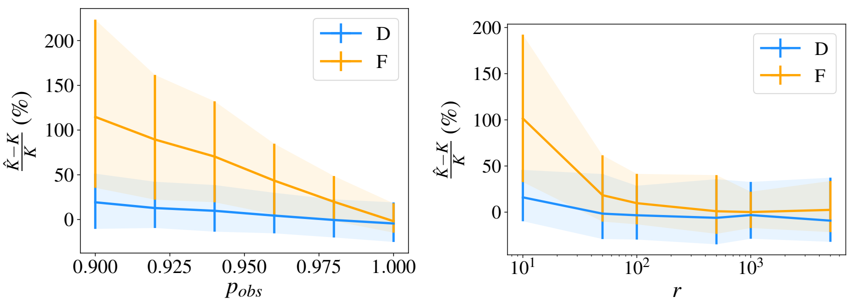


Figure 6.5: Sensitivity to model mismatch on zero-inflated (left) and overdispersed (right) data. For both cases increasing model mismatch (smaller p_{obs} or r) increases the error monotonically, implying the approach is robust for small model mismatch but may give misleading results if the assumed model family fits the data very poorly. The y-axis represent the relative error $\frac{\hat{K}-K}{K}$ and its associate empirical error-bar (0.025-97.5 percentile), and D and F refer to two different true hyperparameter configurations.

specifically, we sample data from NB distribution so that the conditional mean is given by the MF, but the variance is controlled with overdispersion parameter r, using the parameterization NB(r, p) where r is the number of failures until the experiment is stopped and $1-p$ is probability of failure. Using the notation of Eq. 6.1, our data is hence generated by $\lambda_{ij} = \left(\sum_{k=1}^{K} \theta_{ik}\beta_{jk}\right)$ and $Y_{ij} \sim \text{NB}(r, \frac{\lambda_{ij}}{\lambda_{ij}+r})$. This implies that $\mathbb{E}[Y_{ij}; \lambda_{ij}; r] = \lambda_{ij}$ and $\mathbb{V}[Y_{ij}; \lambda_{ij}; r] = \lambda_{ij} + \frac{\lambda_{ij}^2}{r}$, so the smaller the r, the more overdispersed the distribution is. We vary $r \in \{10, 50, 100, 500, 1000, 5000\}$, and Figure 6.5 (right) shows that again the procedure is relatively robust for the mismatch.

Both experiments indicate that the analytic expression for determining K is robust for small model mismatch, and that the relative error grows as a function of the mismatch. Here both examples correspond to overdispersed data compared to the assumed model – which is often the case – which results in overestimating the number of factors. In general, it is difficult to anticipate how specific kinds of model mismatches influence the results, but as a general rule we suggest interpreting odd hyperparameters as signs of potential mismatch worth investigating in more detail.

6.6 Proofs and calculations

6.6.1 Preliminaries

We will make use of Total Expectation Law, Total Variance Law and Total Covariance Law and the following propositions without proofs:

$$\mathbb{E}[Y] = \mathbb{E}[\mathbb{E}[Y|X]] \tag{6.14}$$
$$\mathbb{V}[Y] = \mathbb{E}[\mathbb{V}[Y|X]] + \mathbb{V}[\mathbb{E}[Y|X]] \tag{6.15}$$
$$\mathrm{Cov}[X,Y] = \mathbb{E}[\mathrm{Cov}[X,Y|Z]] + \mathrm{Cov}[\mathbb{E}[X|Z],\mathbb{E}[Y|Z]] \tag{6.16}$$
$$\mathbb{V}[XY] = \mathbb{E}[X^2]\,\mathbb{E}[Y^2] - \mathbb{E}[X]^2\,\mathbb{E}[Y]^2 \text{ if X and Y are independent} \tag{6.17}$$
$$\mathbb{V}[XY] = \mathbb{E}[X]^2\,\mathbb{V}[Y] + \mathbb{E}[Y]^2\,\mathbb{V}[X] + \mathbb{V}[X]\,\mathbb{V}[Y] \text{ if X and Y are independent} \tag{6.18}$$
$$\mathbb{E}[X^2] = \mathbb{V}[X] + \mathbb{E}[X]^2 \tag{6.19}$$
$$\mathbb{V}[\sum_k X_k] = \sum_k \mathbb{V}[X_k] + 2\sum_{k<k'} \mathrm{Cov}[X_k, X_{k'}] \tag{6.20}$$

These relations will be useful in the task of marginalizing out the latent parameters of the hierarchical model, as we shall see in the following section.

6.6.2 Intermediate results

We will start by computing some intermediate results that are useful in different steps for the final results.

Proposition 6.3. *For any combination of valid value for the indexes i,j,t and l, if the latent indexes $k \neq k'$, then $Cov[\theta_{ik}\beta_{jk}, \theta_{tk'}\beta_{lk'}] = 0$*

Proof. By definition of the covariance

$$\mathrm{Cov}[\theta_{ik}\beta_{jk}, \theta_{tk'}\beta_{lk'}] = \mathbb{E}[\theta_{ik}\beta_{jk}\theta_{tk'}\beta_{lk'}] - \mathbb{E}[\theta_{ik}\beta_{jk}]\,\mathbb{E}[\theta_{tk'}\beta_{lk'}]$$

Given that $k \neq k'$, this implies (for any combination of the other indices) $\mathbb{E}[\theta_{ik}\beta_{jk}\theta_{tk'}\beta_{lk'}] = \mathbb{E}[\theta_{ik}\beta_{jk}]\,\mathbb{E}[\theta_{tk'}\beta_{lk'}]$ □

Proposition 6.4. *For any combination of valid value for the indexes i,j and k the following equations hold:*

1. $\mathbb{E}[\sum_k \theta_{ik}\beta_{jk}] = K\mu_\theta\mu_\beta$

2. $\mathbb{V}[\sum_k \theta_{ik}\beta_{jk}] = K[(\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2]$

3. $Cov[\sum_k \theta_{ik}\beta_{jk}, \sum_k \theta_{tk}\beta_{lk}] = K[\delta_{it}(\mu_\beta\sigma_\theta)^2 + \delta_{jl}(\mu_\theta\sigma_\beta)^2 + \delta_{it}\delta_{jl}(\sigma_\theta\sigma_\beta)^2]$

Proof. :

1. For the first equation we apply the summation property of the expected value and the fact that θ_{ik} and β_{jk} are independent.

2. For $\mathbb{V}[\sum_k \theta_{ik}\beta_{jk}]$, we start by using Eq. 6.20, thus resulting in $\mathbb{V}[\sum_k \theta_{ik}\beta_{jk}] = \sum_k \mathbb{V}[\theta_{ik}\beta_{jk}] + 2\sum_{k<k'} \mathrm{Cov}[\theta_{ik}\beta_{jk}, \theta_{ik'}\beta_{jk'}]$. However, from Proposition 6.3 we know the covariance terms where the indexes k and k' are not the same should be zero, resulting in $\mathbb{V}[\sum_k \theta_{ik}\beta_{jk}] = \sum_k \mathbb{V}[\theta_{ik}\beta_{jk}]$. Now using Equation 6.18 for the variance of the product of random variables we obtain

$$\begin{aligned}\mathbb{V}[\theta_{ik}\beta_{jk}] &= \mathbb{E}[\theta_{ik}]^2\,\mathbb{V}[\beta_{jk}] + \mathbb{E}[\beta_{jk}]^2\,\mathbb{V}[\theta_{ik}] + \mathbb{V}[\theta_{ik}]\,\mathbb{V}[\beta_{jk}] \\ &= \mu_\theta^2\sigma_\beta^2 + \mu_\beta^2\sigma_\theta^2 + \sigma_\beta^2\sigma_\theta^2 \\ \Rightarrow \mathbb{V}[\sum_k \theta_{ik}\beta_{jk}] &= K[(\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2]\end{aligned}$$

3. For the last equation we start with the definition of covariance:

$$\begin{aligned}\mathrm{Cov}[\sum_k \theta_{ik}\beta_{jk}, \sum_k \theta_{tk}\beta_{lk}] &= \mathbb{E}[\sum_{k,k'} \theta_{ik}\beta_{jk}\theta_{tk'}\beta_{lk'}] - \mathbb{E}[\sum_k \theta_{ik}\beta_{jk}]\,\mathbb{E}[\sum_{k'} \theta_{tk'}\beta_{lk'}] \\ &= \sum_{k,k'} \underbrace{\mathbb{E}[\theta_{ik}\beta_{jk}\theta_{tk'}\beta_{lk'}] - \mathbb{E}[\theta_{ik}]\,\mathbb{E}[\beta_{jk}]\,\mathbb{E}[\theta_{tk'}]\,\mathbb{E}[\beta_{lk'}]}_{\mathrm{Cov}[\theta_{ik}\beta_{jk},\theta_{tk'}\beta_{lk'}]}\end{aligned}$$

Considering Proposition 6.3, we know that only the shared indices k are non zero, thus simplyfiying to:

$$\begin{aligned}\mathrm{Cov}[\sum_k \theta_{ik}\beta_{jk}, \sum_k \theta_{tk}\beta_{lk}] &= \sum_k \mathbb{E}[\theta_{ik}\beta_{jk}\theta_{tk}\beta_{lk}] - \mathbb{E}[\theta_{ik}]\,\mathbb{E}[\beta_{jk}]\,\mathbb{E}[\theta_{tk}]\,\mathbb{E}[\beta_{lk}] \\ &= \sum_k \mathrm{Cov}[\theta_{ik}\beta_{jk}, \theta_{tk}\beta_{lk}] \qquad (6.21)\end{aligned}$$

Now we can calculate $\mathrm{Cov}[\theta_{ik}\beta_{jk}, \theta_{tk}\beta_{lk}]$ for four different cases:

a) if $i \neq t$ & $j \neq l$: because of independence of all variables, we obtain $\mathrm{Cov}[\theta_{ik}\beta_{jk}, \theta_{tk}\beta_{lk}] = 0$

b) if $i = t$ & $j \neq l$:

$$\begin{aligned}\text{Cov}[\theta_{ik}\beta_{jk}, \theta_{ik}\beta_{lk}] &= \mathbb{E}[\theta_{ik}^2\beta_{jk}\beta_{lk}] - \mathbb{E}[\theta_{ik}]^2\,\mathbb{E}[\beta_{jk}]\,\mathbb{E}[\beta_{lk}] \\ &= \mathbb{E}[\theta_{ik}^2]\,\mathbb{E}[\beta_{jk}]E[\beta_{lk}] - \mathbb{E}[\theta_{ik}]^2\,\mathbb{E}[\beta_{jk}]\,\mathbb{E}[\beta_{lk}] \\ &= \mathbb{E}[\beta_{jk}]E[\beta_{lk}](\mathbb{E}[\theta_{ik}^2] - \mathbb{E}[\theta_{ik}]^2) \\ &= \mu_\beta^2\sigma_\theta^2\end{aligned}$$

c) if $i \neq t$ & $j = l$:

$$\begin{aligned}\text{Cov}[\theta_{ik}\beta_{jk}, \theta_{tk}\beta_{jk}] &= \mathbb{E}[\beta_{jk}^2]\,\mathbb{E}[\theta_{ik}]E[\theta_{tk}] - \mathbb{E}[\beta_{jk}]^2\,\mathbb{E}[\theta_{ik}]E[\theta_{tk}] \\ &= \mu_\theta^2\sigma_\beta^2\end{aligned}$$

d) if $i = t$ & $j = l$:

$$\begin{aligned}\text{Cov}[\theta_{ik}\beta_{jk}, \theta_{ik}\beta_{jk}] &= \mathbb{V}[\theta_{ik}\beta_{jk}] \\ &= (\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2\end{aligned}$$

Putting all together using Kronecker delta for the indices in the different cases we obtain

$$\text{Cov}[\theta_{ik}\beta_{jk}, \theta_{tk}\beta_{lk}] = \delta_{it}(\mu_\beta\sigma_\theta)^2 + \delta_{jl}(\mu_\theta\sigma_\beta)^2 + \delta_{it}\delta_{jl}(\sigma_\theta\sigma_\beta)^2) \tag{6.22}$$

We obtain the final results combining Equation 6.21 and Equation 6.22:

$$\Rightarrow \text{Cov}[\sum_k \theta_{ik}\beta_{jk}, \sum_k \theta_{tk}\beta_{lk}] = K[\delta_{it}(\mu_\beta\sigma_\theta)^2 + \delta_{jl}(\mu_\theta\sigma_\beta)^2 + \delta_{it}\delta_{jl}(\sigma_\theta\sigma_\beta)^2)]$$

□

6.6.3 Expected values and variance

Let us now proceed to compute the prior predictive expected value and variance.

Proposition 6.5. *For any combination of valid value for the indexes i,j and the following equations hold:*

1. $\mathbb{E}[Y_{ij}] = K\mu_\theta\mu_\beta$
2. $\mathbb{V}[Y_{ij}] = K[\mu_\theta\mu_\beta + (\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2]$

Proof. By the law of total expectation, $\mathbb{E}[Y_{ij}] = \mathbb{E}[\mathbb{E}[Y_{ij}|\sum_k \theta_{ik}\beta_{jk}]] = \mathbb{E}[\sum_k \theta_{ik}\beta_{jk}]$, already calculated in Proposition 6.4

For the second equation we use the law of total variance (Equation 6.15) $\mathbb{V}[Y_{ij}] = \mathbb{E}[\mathbb{V}[Y_{ij}|\sum_k \theta_{ik}\beta_{jk}]] + \mathbb{V}[\mathbb{E}[Y_{ij}|\sum_k \theta_{ik}\beta_{jk}]]$, because we have a Poisson likelihood we know that $\mathbb{V}[Y_{ij}|\sum_k \theta_{ik}\beta_{jk}] = \mathbb{E}[Y_{ij}|\sum_k \theta_{ik}\beta_{jk}] = \sum_k \theta_{ik}\beta_{jk}$. Now putting both together and using Proposition 6.4 we obtain:

$$\begin{aligned}\mathbb{V}[Y_{ij}] &= \mathbb{E}[\sum_k \theta_{ik}\beta_{jk}] + \mathbb{V}[\sum_k \theta_{ik}\beta_{jk}]\\ &= K[\mu_\theta\mu_\beta + (\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2]\end{aligned}$$

□

6.6.4 Covariance and correlation

Finally, combining the previous results we can obtain the covariance and correlation given by the prior predictive distribution

Proposition 6.6. *The prior predictive covariance is given by*

$$Cov[Y_{ij}, Y_{tl}] = K[\delta_{it}(\mu_\beta\sigma_\theta)^2 + \delta_{jl}(\mu_\theta\sigma_\beta)^2 + \delta_{it}\delta_{jl}(\mu_\theta\mu_\beta + (\sigma_\theta\sigma_\beta)^2)]$$

Proof. Using the law of total covariance (Equation 6.16):

$$\begin{aligned}\mathrm{Cov}[Y_{ij}, Y_{tl}] &= \mathbb{E}[\mathrm{Cov}[Y_{ij}, Y_{tl}|\theta_{i.}, \beta_{j.}, \theta_{t.}, \beta_{l.}]] + \mathrm{Cov}[\mathbb{E}[Y_{ij}|\theta_{i.}, \beta_{j.}], \mathbb{E}[Y_{tl}|\theta_{t.}, \beta_{l.}]]\\ &= \mathbb{E}[\delta_{it}\delta_{jl}\,\mathbb{V}[Y_{ij}|\theta_{i.}, \beta_{j.}]] + \mathrm{Cov}[\sum_k \theta_{ik}\beta_{jk}, \sum_k \theta_{tk}\beta_{lk}]\\ &= \mathbb{E}[\delta_{it}\delta_{jl}\sum_k \theta_{ik}\beta_{jk}] + K[\delta_{it}(\mu_\beta\sigma_\theta)^2 + \delta_{jl}(\mu_\theta\sigma_\beta)^2 + \delta_{it}\delta_{jl}(\sigma_\theta\sigma_\beta)^2)]\\ &= K[\delta_{it}(\mu_\beta\sigma_\theta)^2 + \delta_{jl}(\mu_\theta\sigma_\beta)^2 + \delta_{it}\delta_{jl}(\mu_\theta\mu_\beta + (\sigma_\theta\sigma_\beta)^2)]\end{aligned}$$

□

Proposition 6.7. *The prior predictive correlation is given by*

$$\rho[Y_{ij}, Y_{tl}] = \frac{\delta_{it}(\mu_\beta\sigma_\theta)^2 + \delta_{jl}(\mu_\theta\sigma_\beta)^2 + \delta_{it}\delta_{jl}(\mu_\theta\mu_\beta + (\sigma_\theta\sigma_\beta)^2)}{\mu_\theta\mu_\beta + (\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2}$$

Or alternatevily

$$\rho[Y_{ij}, Y_{tl}] = \begin{cases} 0, \textit{if } i \neq t \;\; \& \;\; j \neq l \\ 1, \textit{if } i = t \;\; \& \;\; j = l \\ \rho_1 = \frac{(\mu_\beta \sigma_\theta)^2}{\mu_\theta \mu_\beta + (\mu_\beta \sigma_\theta)^2 + (\mu_\theta \sigma_\beta)^2 + (\sigma_\theta \sigma_\beta)^2}, \textit{if } i = t \;\; \& \;\; j \neq l \\ \rho_2 = \frac{(\mu_\theta \sigma_\beta)^2}{\mu_\theta \mu_\beta + (\mu_\beta \sigma_\theta)^2 + (\mu_\theta \sigma_\beta)^2 + (\sigma_\theta \sigma_\beta)^2}, \textit{if } i \neq t \;\; \& \;\; j = l \end{cases}$$

Proof. From the definition of correlation we have:

$$\begin{aligned} \rho[Y_{ij}, Y_{tl}] &= \frac{\mathrm{Cov}[Y_{ij}, Y_{tl}]}{\sqrt{\mathbb{V}[Y_{ij}]\,\mathbb{V}[y_{tlj}]}} \\ &= \frac{\mathrm{Cov}[Y_{ij}, Y_{tl}]}{\sqrt{\mathbb{V}[Y_{ij}]^2}} \\ &= \frac{\delta_{it}(\mu_\beta \sigma_\theta)^2 + \delta_{jl}(\mu_\theta \sigma_\beta)^2 + \delta_{it}\delta_{jl}(\mu_\theta \mu_\beta + (\sigma_\theta \sigma_\beta)^2)}{\mu_\theta \mu_\beta + (\mu_\beta \sigma_\theta)^2 + (\mu_\theta \sigma_\beta)^2 + (\sigma_\theta \sigma_\beta)^2} \end{aligned}$$

□

6.6.5 Finding the hyperparameters given the moments

Proposition 6.8. *Given that we know K, $\mathbb{E}[Y_{ij}]$, $\mathbb{V}[Y_{ij}]$, ρ_1 and ρ_2 the following equations are valid:*

$$\sigma_\theta \sigma_\beta = \frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]} \sqrt{\rho_1 \rho_2} \tag{6.23}$$

$$\left(\frac{\sigma_\beta}{\mu_\beta}\right)^2 = K \frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]^2} \rho_2 \tag{6.24}$$

$$\left(\frac{\sigma_\theta}{\mu_\theta}\right)^2 = K \frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]^2} \rho_1 \tag{6.25}$$

$$\rho_1 \left(\frac{\sigma_\beta}{\mu_\beta}\right)^2 = \rho_2 \left(\frac{\sigma_\theta}{\mu_\theta}\right)^2 \tag{6.26}$$

Proof. We can rewrite the columns correlation ρ_1 and row correlation ρ_2 equations from Proposition 6.7 as:

$$\rho_1 \frac{\mathbb{V}[Y_{ij}]}{K} = (\mu_\beta \sigma_\theta)^2 \tag{6.27}$$

$$\rho_2 \frac{\mathbb{V}[Y_{ij}]}{K} = (\mu_\theta \sigma_\beta)^2 \tag{6.28}$$

Multypling them together we obtain

$$\rho_1\rho_2\left(\frac{\mathbb{V}[Y_{ij}]}{K}\right)^2 = \underbrace{(\mu_\beta\mu_\theta)}_{\frac{\mathbb{E}[Y_{ij}]}{K}}{}^2(\sigma_\beta\sigma_\theta)^2$$

$$\implies \rho_1\rho_2\left(\frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]}\right)^2 = (\sigma_\beta\sigma_\theta)^2 \tag{6.29}$$

Taking the root of Equation 6.29 completes the proof for Equation 6.23.

Now, using Equation 6.29, Equation 6.27 and Equation 6.28, we will obtain the value of $\frac{(\sigma_\beta\sigma_\theta)^2}{(\mu_\beta\sigma_\theta)^2}$ and $\frac{(\sigma_\beta\sigma_\theta)^2}{(\mu_\theta\sigma_\beta)^2}$

$$\frac{(\sigma_\beta\sigma_\theta)^2}{(\mu_\beta\sigma_\theta)^2} = \frac{\sigma_\beta^2}{\mu_\beta^2} = \rho_1\rho_2\left(\frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]}\right)^2\frac{K}{\rho_1\,\mathbb{V}[Y_{ij}]} = \rho_2 K\frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]^2} \tag{6.30}$$

$$\frac{(\sigma_\beta\sigma_\theta)^2}{(\mu_\theta\sigma_\beta)^2} = \frac{\sigma_\theta^2}{\mu_\theta^2} = \rho_1\rho_2\left(\frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]}\right)^2\frac{K}{\rho_2\,\mathbb{V}[Y_{ij}]} = \rho_1 K\frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]^2} \tag{6.31}$$

Finally dividing Equation 6.27 for Equation 6.27 we obtain the last result that completes the proof. □

Proposition 6.9. *Given that we know $\mathbb{E}[Y_{ij}]$, $\mathbb{V}[Y_{ij}]$, ρ_1 and ρ_2, we can obtain the number of latent factors K and coefficient of variation ($\frac{\sigma}{\mu}$) of the priors of the Poisson factorization model that would generate data to match those moments.*

$$K = \frac{(1-(\rho_1+\rho_2))\,\mathbb{V}[Y_{ij}] - \mathbb{E}[Y_{ij}]}{\rho_1\rho_2}\left(\frac{\mathbb{E}[Y_{ij}]}{\mathbb{V}[Y_{ij}]}\right)^2 \tag{6.32}$$

$$\left(\frac{\sigma_\theta}{\mu_\theta}\right)^2 = \frac{(1-(\rho_1+\rho_2))\,\mathbb{V}[Y_{ij}] - \mathbb{E}[Y_{ij}]}{\rho_2\,\mathbb{V}[Y_{ij}]} \tag{6.33}$$

$$\left(\frac{\sigma_\beta}{\mu_\beta}\right)^2 = \frac{(1-(\rho_1+\rho_2))\,\mathbb{V}[Y_{ij}] - \mathbb{E}[Y_{ij}]}{\rho_1\,\mathbb{V}[Y_{ij}]} \tag{6.34}$$

Proof. From Proposition 6.5, we can rewrite the expression for the variance as:

$$\mathbb{V}[Y_{ij}] = \mathbb{E}[Y_{ij}] + K\underbrace{(\mu_\theta\sigma_\beta)}_{\rho_2\frac{\mathbb{V}[Y_{ij}]}{K}}{}^2 + K\underbrace{(\mu_\beta\sigma_\theta)}_{\rho_1\frac{\mathbb{V}[Y_{ij}]}{K}}{}^2 + K(\sigma_\theta\sigma_\beta)^2$$

Now, using Equation 6.27 and Equation 6.28 to substitute in the previous equation we obtain:

$$\mathbb{V}[Y_{ij}] = \mathbb{E}[Y_{ij}] + (\rho_1+\rho_2)\,\mathbb{V}[Y_{ij}] + K(\sigma_\theta\sigma_\beta)^2$$

Using the squared version of Equation 6.23 from Proposition 6.8, we know that $K(\sigma_\theta\sigma_\beta)^2 = K\left(\frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]}\right)^2 \rho_1\rho_2$. This results in

$$K\left(\frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]}\right)^2 \rho_1\rho_2 = (1-(\rho_1+\rho_2))\,\mathbb{V}[Y_{ij}] - \mathbb{E}[Y_{ij}]$$

$$\implies K = \frac{(1-(\rho_1+\rho_2))\,\mathbb{V}[Y_{ij}] - \mathbb{E}[Y_{ij}]}{\rho_1\rho_2}\left(\frac{\mathbb{E}[Y_{ij}]}{\mathbb{V}[Y_{ij}]}\right)^2 \tag{6.35}$$

The remaining results are obtained by substituting Equation 6.35 in Equation 6.27 and Equations 6.28. □

6.6.5.1 Gamma priors

For gamma priors parameterized with shape (a,c) and rate (b,d) we have:

$$\mu_\theta = \frac{a}{b}; \sigma_\theta^2 = \frac{a}{b^2}; \mu_\beta = \frac{c}{d}; \sigma_\beta^2 = \frac{c}{d^2}$$

and cofficient of variation given by:

$$\frac{\sigma_\theta^2}{\mu_\theta^2} = \frac{a}{b^2}\frac{b^2}{a^2} = \frac{1}{a} \tag{6.36}$$

$$\frac{\sigma_\beta^2}{\mu_\beta^2} = \frac{d^2}{c^2}\frac{c}{d^2} = \frac{1}{c} \tag{6.37}$$

Thus, Equation 6.36 and Equation 6.37 establishes a close form relationship between shape hyperparameters of Gamma distributed latent variables in Poisson MF and moments of the marginal distribution of the data. This means that any assumption that the expert might have about those moments on the data, can be readily translated into appropriate values in the prior specification.

In conclusion, given the chosen moments, the prior especification of Gamma-Poisson MF model reduces to one degree of freedom, given that the latent dimensionality, and shape hyperparameters are determined. The only two hyperparameters left are the rate/scale, although they would be restriced to be obey a relationship with functional form

$$b \propto \frac{1}{d}$$

.

Proposition 6.10. *Given that we know the moments $\mathbb{E}[Y_{ij}]$, $\mathbb{V}[Y_{ij}]$, ρ_1 and ρ_2, we can obtain the scale parameters of the Gamma priors speficied as $f(\mu_\theta, \sigma_\theta^2) =$*

Gamma(a,b) *and* $f(\mu_\beta,\sigma^2_\beta) =$ *Gamma*(c,d) *in the Gamma-Poisson factorization model such that the prior predictive moments would match those given moments.*

$$\frac{1}{a} = \frac{(1-(\rho_1+\rho_2))\,\mathbb{V}[Y_{ij}] - \mathbb{E}[Y_{ij}]}{\rho_2\,\mathbb{V}[Y_{ij}]} \tag{6.38}$$

$$\frac{1}{c} = \frac{(1-(\rho_1+\rho_2))\,\mathbb{V}[Y_{ij}] - \mathbb{E}[Y_{ij}]}{\rho_1\,\mathbb{V}[Y_{ij}]} \tag{6.39}$$

Proof. Immediate from Proposition 6.9 and the parameterization of Gamma distribution discussed. □

Also we can rewrite Eq. 6.23 with Gamma parameterization to obtain

$$\begin{aligned}\frac{a}{b^2}\frac{c}{d^2} = \sigma^2_\theta\sigma^2_\theta &= \left(\frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]}\right)^2 \rho_1\rho_2 \\ \implies (bd)^2 &= \left(\frac{\mathbb{E}[Y_{ij}]}{\mathbb{V}[Y_{ij}]}\right)^2 \frac{ac}{\rho_1\rho_2} \\ \implies bd &= \frac{\mathbb{E}[Y_{ij}]}{\mathbb{V}[Y_{ij}]}\sqrt{\frac{ac}{\rho_1\rho_2}} \end{aligned} \tag{6.40}$$

6.6.6 Derivation of Analytic Solution for Compound Poisson Matrix Factorization

We will work with the Exponential Dispersion models (EDM) family of observation that makes Compound Poisson matrix factorization models. Keeping the same notation of the previous section, but adding variable N_{ui} as a Poisson distributed latent count factor of the ED model. With abuse of notation, for example this model allow for observations of the type $Y_{ij} = \sum_{i=1}^{N_{ij}} \mathcal{N}(1,1)$, where N_{ij} is a Poisson random variable, extending Poisson factorization to the domain of real valued observations. Also, from the additive properties Jorgensen (1987); Basbug and Engelhardt (2016) of EDM models, $Y_{ij} = \sum_{i=1}^{N_{ij}} Y_{ijk}$ with $Y_{ijk} \overset{\text{iid}}{\sim} \text{ED}(w,\kappa)$ is equivalent to $Y_{ij} \sim \text{ED}(w, \kappa N_{ij})$. Thus, the Compound Poisson Matrix Factorization (CPMF) model we use is defined as

$$\begin{aligned}
Y_{ij} &\sim \text{ED}(w, \kappa N_{ij}) \\
N_{ij} &\sim \text{Poisson}(\sum_k \theta_{ik}\beta_{jk}) \\
\theta_{ik} &\sim f(\mu_\theta, \sigma_\theta^2) \\
\beta_{jk} &\sim f(\mu_\beta, \sigma_\beta^2)
\end{aligned}$$

where $p(Y_{ij}; w, \kappa N_{ij}) = \exp(Y_{ij}w - \kappa N_{ij}\psi(w))h(Y_{ij}, \kappa N_{ij})$, $\mathbb{E}[Y_{ij}; w, \kappa N_{ij}] = \kappa N_{ij}\psi'(w)$ and $\mathbb{V}[Y_{ij}; w, \kappa N_{ij}] = \kappa N_{ij}\psi''(w)$.

6.6.6.1 Mean, variance, covariance and correlation

Proposition 6.11. *For any combination of valid value for the indexes i,j and the following equations hold:*

1. $\mathbb{E}[Y_{ij}] = \kappa\psi'(w)K\mu_\theta\mu_\beta$
2. $\mathbb{V}[Y_{ij}] = \kappa\psi''(w)K\mu_\theta\mu_\beta + (\kappa\psi'(w))^2K[\mu_\theta\mu_\beta + (\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2]$

Proof. By the law of total expectation and the properties of the mean of ED family, $\mathbb{E}[Y_{ij}] = \mathbb{E}[\kappa\psi'(w)N_{ij}]$, which simplifies to $\mathbb{E}[Y_{ij}] = \kappa\psi'(w)\,\mathbb{E}[N_{ij}]$ and from Proposition 6.5 we know the expected value of the latent Poisson count N_{ij}, concluding that $\mathbb{E}[Y_{ij}] = \kappa\psi'(w)K\mu_\theta\mu_\beta$.

Using the law of total variance $\mathbb{V}[Y_{ij}] = \mathbb{E}[\mathbb{V}[Y_{ij}|N_{ij}]] + \mathbb{V}[\mathbb{E}[Y_{ij}|N_{ij}]$, that simplifies to

$$\mathbb{V}[Y_{ij}] = \kappa\psi''(w)\,\mathbb{E}[N_{ij}] + [\kappa\psi'(w)]^2\,\mathbb{V}[N_{ij}]$$

, again substituting Proposition 6.5 completes the proof. □

Proposition 6.12. *The prior predictive correlation is given by:*

$$\rho[Y_{ij}, Y_{tl}] = \begin{cases} 0, \textit{if } i \neq t \ \ \& \ \ j \neq l \\ 1, \textit{if } i = t \ \ \& \ \ j = l \\ \rho_1, \textit{if } i = t \ \ \& \ \ j \neq l \\ \rho_2, \textit{if } i \neq t \ \ \& \ \ j = l \end{cases}$$

, with

$$\begin{aligned}
\rho_1 &= \frac{K[\kappa\psi'(w)]^2}{\mathbb{V}[Y_{ij}]}(\mu_\beta\sigma_\theta)^2 \\
\rho_2 &= \frac{K[\kappa\psi'(w)]^2}{\mathbb{V}[Y_{ij}]}(\mu_\theta\sigma_\beta)^2
\end{aligned}$$

Proof. Starting with the covariance and apply the law of total covariance we obtain:

$$\mathrm{Cov}[Y_{ij}, Y_{tl}] = \delta_{it}\delta_{jl}\kappa\psi''(w)\,\mathbb{E}[N_{ij}] + [\kappa\psi'(w)]^2\,\mathrm{Cov}[N_{ij}, N_{tl}] \tag{6.41}$$

When all the indices coincide this will be equal to the variance, thus leading to a correlation of 1, when all the indices are different this will lead to correlation of zero. This means that the main difference between the prior predictive correlation structure of Compound Poisson Matrix Factorization Model and Poisson Factorization model will be in the rows and columns correlation, that we will be able to calculate because we know the covariance $\mathrm{Cov}[N_{ij}, N_{tl}]$ from Proposition 6.6.

$$\begin{aligned}
\rho_1 &= \frac{[\kappa\psi'(w)]^2\,\mathrm{Cov}[N_{ij}, N_{il}]}{\mathbb{V}[Y_{i,j}]} \\
&= \frac{K[\kappa\psi'(w)]^2(\mu_\beta\sigma_\theta^2)}{\mathbb{V}[Y_{i,j}]} \\
\rho_2 &= \frac{[\kappa\psi'(w)]^2\,\mathrm{Cov}[N_{ij}, N_{tj}]}{\mathbb{V}[Y_{i,j}]} \\
&= \frac{K[\kappa\psi'(w)]^2(\mu_\theta\sigma_\beta^2)}{\mathbb{V}[Y_{i,j}]}
\end{aligned}$$

□

6.6.6.2 Finding the hyperparameters given the moments

Proposition 6.13. *For Compound Poisson MF, given that we know $\mathbb{E}[Y_{ij}]$, $\mathbb{V}[Y_{ij}]$, ρ_1 and ρ_2, we can obtain the number of latent factors K of model that would generate data to match those moments.*

$$K = \frac{(1-(\rho_1+\rho_2))\,\mathbb{V}[Y_{ij}] - \left(\kappa\psi'(w) + \frac{\psi''(w)}{\psi'(w)}\right)\mathbb{E}[Y_{ij}]}{\rho_1\rho_2}\left(\frac{\mathbb{E}[Y_{ij}]}{\mathbb{V}[Y_{ij}]}\right)^2 \tag{6.42}$$

Proof. We will start by showing that:

$$\sigma_\theta\sigma_\beta = \frac{\mathbb{V}[Y_{ij}]}{\mathbb{E}[Y_{ij}]\kappa\psi'(w)}\sqrt{\rho_1\rho_2} \tag{6.43}$$

Take ρ_1 and ρ_2 and multiply them to obtain:

$$\rho_1\rho_2 = \left(\frac{K[\kappa\psi'(w)]^2}{\mathbb{V}[Y_{ij}]}\right)^2(\mu_\theta\mu_\beta)^2(\sigma_\theta\sigma_\beta)^2$$

From Proposition 6.11, we know $K\mu_\theta\mu_\beta = \frac{\mathbb{E}[Y_{ij}]}{\kappa\psi'(w)}$, so we can substitute that on the previous equation obtaining:

$$\rho_1\rho_2 = \left(\frac{[\kappa\psi'(w)]^2}{\mathbb{V}[Y_{ij}]}\right)^2 \left(\frac{\mathbb{E}[Y_{ij}]}{\kappa\psi'(w)}\right)^2 (\sigma_\theta\sigma_\beta)^2$$
$$\rho_1\rho_2 = [\kappa\psi'(w)]^2 \left(\frac{\mathbb{E}[Y_{ij}]}{\mathbb{V}[Y_{ij}]}\right)^2 (\sigma_\theta\sigma_\beta)^2$$

Now let us turn our attention to $\mathbb{V}[Y_{ij}]$ and re-write it using the previous results together with Proposition 6.12 for the correlations, and Proposition 6.11 for the mean:

$$\begin{aligned}
\mathbb{V}[Y_{ij}] &= \psi''(w)\underbrace{\kappa K\mu_\theta\mu_\beta}_{\frac{\mathbb{E}[Y_{ij}]}{\psi'(w)}} + \underbrace{K[\kappa\psi'(w)]^2\mu_\theta\mu_\beta}_{\kappa\psi'(w)\,\mathbb{E}[Y_{ij}]} + K[\kappa\psi'(w)]^2[(\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2] \\
&= \left(\frac{\psi''(w)}{\psi'(w)} + \kappa\psi'(w)\right)\mathbb{E}[Y_{ij}] + \underbrace{K[\kappa\psi'(w)]^2(\mu_\beta\sigma_\theta)^2}_{\rho_1\,\mathbb{V}[Y_{ij}]} + \underbrace{K[\kappa\psi'(w)]^2(\mu_\theta\sigma_\beta)^2}_{\rho_2\,\mathbb{V}[Y_{ij}]} + K[\kappa\psi'(w)]^2(\sigma_\theta\sigma_\beta)^2 \\
&= \left(\frac{\psi''(w)}{\psi'(w)} + \kappa\psi'(w)\right)\mathbb{E}[Y_{ij}] + (\rho_1+\rho_2)\,\mathbb{V}[Y_{ij}] + K\underbrace{[\kappa\psi'(w)]^2(\sigma_\theta\sigma_\beta)^2}_{\rho_1\rho_2\left(\frac{\mathbb{V}[Y_{ij}]]}{\mathbb{E}[Y_{ij}}\right)^2} \\
&= \left(\frac{\psi''(w)}{\psi'(w)} + \kappa\psi'(w)\right)\mathbb{E}[Y_{ij}] + (\rho_1+\rho_2)\,\mathbb{V}[Y_{ij}] + K\rho_1\rho_2\left(\frac{\mathbb{V}[Y_{ij}]]}{\mathbb{E}[Y_{ij}}\right)^2
\end{aligned}$$

Reorganizing the terms and isolating K we obtain the final formula desired.

$$K = \frac{(1-(\rho_1+\rho_2))\,\mathbb{V}[Y_{ij}] - \left(\kappa\psi'(w) + \frac{\psi''(w)}{\psi'(w)}\right)\mathbb{E}[Y_{ij}]}{\rho_1\rho_2}\left(\frac{\mathbb{E}[Y_{ij}]}{\mathbb{V}[Y_{ij}]}\right)^2$$

□

6.6.7 Generalizing the results to other likelihoods

We will make our derivations using a general form for PMF

$$
\begin{aligned}
\theta_{ik} &\sim f_\theta(\mu_\theta, \sigma_\theta) && (6.44)\\
\beta_{jk} &\sim f_\beta(\mu_\beta, \sigma_\beta) && (6.45)\\
Y_{ij} &\sim f_Y(\sum_{k=1}^{K} \theta_{ik}\beta_{jk}) && (6.46)\\
\text{with } \mathbb{E}[Y_{ij}|\boldsymbol{\theta}, \boldsymbol{\beta}] &= \sum_{k=1}^{K} \theta_{ik}\beta_{jk}
\end{aligned}
$$

Proposition 6.14. *For any entry of the matrix $Y = \{Y_{ij}\} \in \mathbb{R}^{N\times M}$, the mean and variance is given by:*

$$
\begin{aligned}
\mathbb{E}[Y_{ij}] &= K\mu_\theta\mu_\beta && (6.47)\\
\mathbb{V}[Y_{ij}] &= \mathbb{E}[\mathbb{V}(Y_{ij}|\boldsymbol{\theta}, \boldsymbol{\beta})]\\
&\quad + K[(\mu_\beta\sigma_\theta)^2 + (\mu_\theta\sigma_\beta)^2 + (\sigma_\theta\sigma_\beta)^2] && (6.48)
\end{aligned}
$$

Proposition 6.15. *For any pair of entries Y_{ij} and Y_{tl} of the matrix Y, their correlation is given by:*

$$
\rho[Y_{ij}, Y_{tl}] = \begin{cases} 0, \textit{if } i \neq t \;\&\; j \neq l \\ 1, \textit{if } i = t \;\&\; j = l \\ \rho_1, \textit{if } i = t \;\&\; j \neq l \\ \rho_2, \textit{if } i \neq t \;\&\; j = l \end{cases} \qquad (6.49)
$$

$$
\begin{aligned}
\rho_1 &= \frac{K(\mu_\beta\sigma_\theta)^2}{\mathbb{V}[Y_{ij}]}\\
\rho_2 &= \frac{K(\mu_\theta\sigma_\beta)^2}{\mathbb{V}[Y_{ij}]}
\end{aligned}
$$

The proof of propositions 6.14 and 6.15 are the same calculation developed before for the Poisson and compound Poisson factorization models, with the difference that $\mathbb{E}[\mathbb{V}(Y_{ij}|\boldsymbol{\theta}, \boldsymbol{\beta})]$ will depend on the specific choice of distribution f_Y. Given Propositions 6.14 and 6.15 and some target values for the moments, we can directly solve e.g. for the number of latent factors K. Denoting $\tau = 1 - (\rho_1 + \rho_2)$, we obtain our main result in Theorem 6.1.

Theorem 6.1. *Given that we know the observation mean $\mathbb{E}[Y_{ij}]$, variance $\mathbb{V}[Y_{ij}]$, correlations ρ_1 and ρ_2, and the expected conditional variance (model dependent) $\mathbb{E}[\mathbb{V}(Y_{ij}|\boldsymbol{\theta}, \boldsymbol{\beta})]$, we can obtain the number of latent factors K to match those quantities using the formula:*

$$K = \frac{\tau\, \mathbb{V}[Y_{ij}] - \mathbb{E}[\mathbb{V}(Y_{ij}|\boldsymbol{\theta}, \boldsymbol{\beta})]}{\rho_1 \rho_2} \left(\frac{\mathbb{E}[Y_{ij}]}{\mathbb{V}[Y_{ij}]} \right)^2 . \tag{6.50}$$

The term $\mathbb{E}[\mathbb{V}(Y_{ij}|\boldsymbol{\theta}, \boldsymbol{\beta})]$ is model dependent, for example in the traditional Probabilistic Matrix Factorization we use a Gaussian observation model given by $Y_{ij} \sim \mathcal{N}(\sum_{k=1}^{K} \theta_{ik}\beta_{jk}, \sigma_Y^{-1})$, and obtain $\mathbb{E}[\mathbb{V}(Y_{ij}|\boldsymbol{\theta}, \boldsymbol{\beta})] = \sigma_Y^2$, while on Poisson Matrix Factorization $Y_{ij} \sim \text{Poisson}(\sum_{k=1}^{K} \theta_{ik}\beta_{jk})$, and $\mathbb{E}[\mathbb{V}(Y_{ij}|\boldsymbol{\theta}, \boldsymbol{\beta})] = \mathbb{E}[Y_{ij}]$

Conclusion 7

"No one can deny that a network (a world network) of economic and psychic affiliations is being woven at ever-increasing speed which envelops and constantly penetrates more deeply within each of us. With every day that passes it becomes a little more impossible for us to act or think otherwise than collectively."

— Pierre Teilhard de Chardin, *The Future of Man*

This thesis is focused on the design of probabilistic models for recommender systems and collaborative filtering. We extend and create new models to include rich contextual and content information (item textual content, user social network, location, time, etc.), and we develop scalable approximate inference algorithms for these models. The working hypothesis is that multi-relational data can be integrated in a joint probabilistic factorization model, allowing for the utilization of various data sources combined via latent variables with the flexibility to predict users' interaction with the items, and improve inferences about the user behaviour and recommendations. The work has branched into the following challenges: (1) modeling contextual information into probabilistic factorization models for recommender system; (2) modeling temporal dynamics using factorization techniques and temporal point processes; (3) analysis of existing models using prior predictive techniques to determine the hyperparameters of the model. One overarching direction has been the use of shared latent variables as anchoring points for combining different aspects of the data in a modular fashion, yielding recommendation models utilizing multi-source data. This has led to the proposal of a generic model design where relations expressed in terms of matrices and tensors are combined and leveraged into a single joint model. Furthermore, the use of shared latent variables has been applied to assemble a temporal point process model with a RNN session-based recommendation model. Finally, we developed an analytical tool to facilitate the choice of hyperparameters in Bayesian recommender system models, demonstrating in the case of Poisson factorization that the technique can be utilized to aprioristically determine the dimensionality of the latent space, as well as specifying most of the hyperparameters for the Gamma priors. In general, we position our work as a contribution to model design and analysis in the context of recommender system utilizing multi-relational data as a signal for contextual information.

Research questions

In the course of this work we explored several techniques seeking to answer the following research questions:

- **RQ1** Is there an overarching strategy for incorporating contextual information into factorization models for recommender system? What improvements are observed by adding contextual information such as social networks and item textual content in a joint model for recommendation?
- **RQ2** How to incorporate implicit feedback using count data models in factorization models for recommender system and what are the advantages of doing so?
- **RQ3** How can we include periodic time information into matrix and tensor factorization models for recommender system and what are the observable gains from doing so?
- **RQ4** What is the effect of adding a temporal point process model in a sequential multi-session recommendation model?
- **RQ5** How to analyze the properties of Bayesian factorization models for recommender system in order to specify the hyperparameters of the model?

In order to investigate **RQ1** we have engaged with the idea of latent space modeling using shared latent variables to couple different aspects of a model. We developed three models in this thesis exploiting this idea. In Chapter 3 we describe PoissonMF-CS, a Poisson factorization model that includes item textual information and users social networks in a joint model, where shared Gamma distributed latent variables are used to couple the user–item matrix, with user–user social network and item–term matrix. In Chapter 4 we describe Temporal Poisson Tensor Factorization (TPTF) and Collective Temporal Poisson Tensor Factorization (TPTF-C) extending Poisson factorization to the time domain, using also shared-latent variables to connect consecutive time periods and including contextual data for the items. We found in both cases that adding contextual information improved the quality of the recommendations measured in terms of different metrics. Chapter 5 explored the combination of Hierarchical RNN with temporal point processes (TPP) using a shared latent space to couple the two models and creating a model for multi-session recommendation capable of performing both next-item recommendation and time prediction of next session. This setting proved to achieve better results compared to baseline in the joint task of recommendation and time prediction. These three chapters in conjunction demonstrated the potential improvements of modeling context via shared latent variables across different recommendation tasks.

Implicit feedback using count models (**RQ2**) was incorporated using Poisson likelihood models in Chapter 3 and Chapter 4. The evaluation of the impact of this model choice was done by comparing the proposed models with more traditional models that either implicit (in tha case of matrix factorization) or explicitly (in the case of probabilistic matrix factorization) use a Gaussian likelihood model, demonstrating improvement with the use of Poisson likelihood across different tasks and models. The advantage of this choice stems to the fact that typically implicit feedback appear in a form of count data. The question of periodic time model posed in **RQ3** was investigated in Chapter 4 with a proposal of a Gamma distributed latent variable cyclic chain, where each consecutive time period is interlinked in a cyclical way to fit the periodic pattern in consideration. This construction was crafted in a way that the variational inference would still be possible and efficient, thus conserving the conditional conjugacy property of the model. The empirical evaluation was performed in a task of item recommendation using spatio-temporal data.

The combination of a temporal point process and a hierarchical RNN studied in Chapter 5 also allowed us to study the issues raised by **RQ4**. In general we observed that we could obtain time-prediction results while obtaining good recommendation performance, which indicates that the RNN hidden states were encoding sufficient information for the time model. We also observed that the time prediction model could be modulated to target different time horizons because the time model included terms that weighted short and long term predictions differently. This motivated an adjustment to the time loss, based on modulating the way that a given time interval affects the overal time model in terms of short-term or long-term predictions. The empirical evaluation indicates that the proposed mechanism is able to achieve this modulation between short-term and long-term predictions, while maintaining an overall prediction accuracy.

Finally, in Chapter 6 we focus on the problem of hyperparameter selection in Bayesian factorization models for recommender system, raised by **RQ5**. We focus our analysis on the Poisson and compound Poisson matrix factorization models. First, we notice that many methods for hyperparameter selection relies on fitting the model parameters to the data multiple times, usually optimizing some criteria for quality. This creates an extra computational cost, that sometimes can be prohibitive depending on the available resources. With that in mind, we studied the technique of prior predictive checks, that utilizes the prior predictive distribution (PPD) to analyze the effect of the hyperparameters on the model predictive distribution before fitting any data. Furthermore, we develop this idea into a tool for automatic hyperparameter selection based on matching *virtual statistics* of the PPD and statistics of the data. The idea is that sample data from the PPD generates the *virtual statistics*[1], and by comparing those with data statistics (obtained either using estimated values or prior expert knowledge), we can optimize the hyperparameters to increase the match between the two. When the true data generating process is within the assumed model family, the approach can recover the true hyperparameters of the model, and we show empirically that the method is robust for small model misspecification. If the data fits poorly and generate poor predictions, which can be interpreted as a sign of model mismatch and need for model refinement. This approach can be used as a first attempt for hyperparameter selection, before trying more expensive methods. This approach proved useful in finding analytical equations for prior specification when applied to Poisson and compound Poisson factorization, particularly we demonstrated a novel (and to our knowledge, unique) equation for the dimensionality of the latent space of those models. An alternative gradient-based optimization approach was developed and applied in the same model class, showing promising results as well. The optimization based approach is sensible for model parameterization, which indicates possible avenues of more research into how to solve a prior predictive hyperparameter optimization problem with different parameterizations.

Future directions

The models presented in Chapter 3 and Chapter 4 reveal a model design that can be further generalized in two main directions: generalized matrix-tensor factorization and utilization of more generic likelihood models. First, we notice that there is not limitation on the number of matrices or tensors that we can jointly factorize. The only necessary step is to specify how the to connect the different matrices or tensors, which has implication on which parts of the model will have shared latent variables. Second, we can generalize the observation model to a compound Poisson model, which would allow us to propose a generic joint factorization model for multiple data types and data sources, both in tensor and matrix form. The idea of exponential family embeddings (Rudolph et al., 2016) can be leveraged as well for generalization of the observation model. We can apply this generic model in joint probabilistic factorization models for recommendations with language models based on RNNs, in

[1]The term virtual statistics is to indicate that these are statistics calculated over simulated data, not from the observations.

a similar fashion as the models developed in Dieng et al. (2017), Ailem et al. (2017) and Dieng et al. (2020), leading to a recommendations with a full language model as context. Furthermore, this idea can be explored for joint factorization models with network/graph embeddings, motivated by the observation that graph embeddings can be posed as a matrix factorization (Qiu et al., 2018) and generalized models of graph embeddings using exponential family conditional distributions (Çelikkanat and Malliaros, 2020). All these considerations are pointing towards future research of models with the capabilities for integrating even more complex data sources as contextual information into a joint recommendation model.

Furthermore, recent works have explored the role of the attention mechanism (Vaswani et al., 2017) in capturing long-range dependencies in sequential models, in particular we highlight the works combining attention and self-attention mechanisms with Hawkes processes (Zhang et al., 2020; Zuo et al., 2020). In Chapter 5 we developed a custom mechanism for modulating short-range and long-range time prediction, which leads to questions about more generic mechanism for such tasks. One possibility would be extending the existing time model with the attention mechanism for the inter-session recommendation task, either combining with the aforementioned models, or designing new models. With that regards, it is necessary as well to investigate at which level to integrate the attention mechanism, since one could have it both at inter-session level, as well as intra-session.

Finally, the tool set developed in Chapter 6 for hyperparameter selection can be further investigated in more families of models and integrated into a Bayesian workflow for model development (Gelman et al., 2020). The problem of hyperparameter selection in Bayesian factorization models is complex and widespread, and the solution developed in this thesis has been tested in the limited context of Poisson and compound Poisson factorization models. For the generic gradient-based solution there are open questions related to how the parameterization of the priors affect the optimization landscape, which could be solved with techniques such as the natural gradient using the Fisher information metric, similar to the work of Tang and Ranganath (2019). Other improvements to the method include exploring better empirical estimators for the gradients (Mohamed et al., 2020) to reduce optimization noise, and support for discrete hyperparameter

Bibliography

Odd O. Aalen, Ørnulf Borgan, and Håkon K. Gjessing. *Survival and Event History Analysis: A Process Point of View.* Springer New York, 2008. doi: 10.1007/978-0-387-68560-1. URL `https://doi.org/10.1007/978-0-387-68560-1`.

Ayan Acharya, Dean Teffer, Jette Henderson, Marcus Tyler, Mingyuan Zhou, and Joydeep Ghosh. Gamma process poisson factorization for joint modeling of network and documents. In *ECML/PKDD (1)*, volume 9284 of *Lecture Notes in Computer Science*, pages 283–299. Springer, 2015.

Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.*, 17(6):734–749, 2005.

Amr Ahmed, Mohamed Aly, Joseph Gonzalez, Shravan M. Narayanamurthy, and Alexander J. Smola. Scalable inference in latent variable models. In *WSDM*, pages 123–132. ACM, 2012.

Melissa Ailem, Aghiles Salah, and Mohamed Nadif. Non-negative matrix factorization meets word embedding. In *SIGIR*, pages 1081–1084. ACM, 2017.

A. Akbarov. *Probability elicitation: Predictive approach.* PhD thesis, University of Salford, 2009.

Francis R Bach and Michael I Jordan. A probabilistic interpretation of canonical correlation analysis. Technical report, Department of Statistics, University of California, Berkeley, 2005.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

Lu Bai, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. Group sparse topical coding: from code to topic. In *WSDM*, pages 315–324. ACM, 2013.

David J. Bartholomew, Martin Knott, and Irini Moustaki. *Latent Variable Models and Factor Analysis: A Unified Approach*, volume 904. John Wiley & Sons, Chichester, [England] :, 3rd ed. edition, 2011. ISBN 1119973708.

Mehmet Emin Basbug and Barbara E. Engelhardt. Hierarchical compound poisson factorization. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1795–1803. JMLR.org, 2016.

Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *J. Mach. Learn. Res.*, 18:153:1–153:43, 2017.

Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8): 1798–1828, 2013.

J. M. Bernardo and A. F. M. Smith. *Bayesian Theory.* John Wiley & Sons, New York, 1994.

Michael W. Berry, Susan T. Dumais, and Gavin W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Rev.*, 37(4):573–595, 1995. doi: 10.1137/1037127. URL `https://doi.org/10.1137/1037127`.

Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul A. Szerlip, Paul Horsfall, and Noah D. Goodman. Pyro: Deep universal probabilistic programming. *J. Mach. Learn. Res.*, 20:28:1–28:6, 2019.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518): 859–877, 2017.

Sergey G. Bobkov, Gennadiy P. Chistyakov, and Friedrich Götze. Non-uniform bounds in the poisson approximation with applications to informational distances I. *IEEE Trans. Inf. Theory*, 65(9):5283–5293, 2019.

Kenneth A. Bollen. Latent variables in psychology and the social sciences. *Annual Review of Psychology*, 53(1):605–634, 2002. doi: 10.1146/annurev.psych.53.100901.135239. URL `https://doi.org/10.1146/annurev.psych.53.100901.135239`.

Charles Bouveyron, Pierre Latouche, and Pierre-Alexandre Mattei. Exact dimensionality selection for Bayesian PCA. *Scandinavian Journal of Statistics*, 2019.

Thomas Brouwer and Pietro Lio. Prior and likelihood choices for Bayesian matrix factorisation on small datasets. *arXiv preprint arXiv:1712.00288*, 2017.

Wray L. Buntine and Aleks Jakulin. Discrete component analysis. In *SLSFS*, volume 3940 of *Lecture Notes in Computer Science*, pages 1–33. Springer, 2005.

John F. Canny. Gap: a factor model for discrete data. In *SIGIR*, pages 122–129. ACM, 2004.

Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. Second workshop on information heterogeneity and fusion in recommender systems (hetrec2011). In *RecSys*, pages 387–388. ACM, 2011.

Bob Carpenter, Andrew Gelman, Matthew D. Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of Statistical Software, Articles*, 76(1):1–32, 2017. ISSN 1548-7660. doi: 10.18637/jss.v076.i01. URL `https://www.jstatsoft.org/v076/i01`.

George Casella. An introduction to empirical bayes data analysis. *The American Statistician*, 39:83–87, 1985.

George Casella and Roger Berger. *Statistical Inference.* Duxbury Resource Center, June 2001. ISBN 0534243126.

Abdulkadir Çelikkanat and Fragkiskos D. Malliaros. Exponential family graph embeddings. In *AAAI*, pages 3357–3364. AAAI Press, 2020.

Ali Taylan Cemgil. Bayesian inference for nonnegative matrix factorisation models. *Comp. Int. and Neurosc.*, 2009:785152:1–785152:17, 2009.

Allison June-Barlow Chaney, David M. Blei, and Tina Eliassi-Rad. A probabilistic model for using social networks in personalized item recommendation. In *RecSys*, pages 43–50. ACM, 2015.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, 2014. URL `http://arxiv.org/abs/1412.3555`.

Gregory F. Cooper. The computational complexity of probabilistic inference using bayesian belief networks. *Artif. Intell.*, 42(2-3):393–405, 1990.

Eliezer de Souza da Silva. New probabilistic models for recommender systems with rich contextual and content information. In *WSDM*, page 839. ACM, 2017.

Eliezer de Souza da Silva and Dirk Ahlers. Poisson factorization models for spatiotemporal retrieval. In *GIR*, pages 3:1–3:2. ACM, 2017.

Eliezer de Souza da Silva, Helge Langseth, and Heri Ramampiaro. Content-based social recommendation with poisson matrix factorization. In *ECML/PKDD (1)*, volume 10534 of *Lecture Notes in Computer Science*, pages 530–546. Springer, 2017.

Eliezer de Souza da Silva, Tomasz Kuśmierczyk, Marcelo Hartmann, and Arto Klami. Prior specification via prior predictive matching: Poisson matrix factorization and beyond. *CoRR*, 2019. URL `http://arxiv.org/abs/1910.12263`.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *J. Am. Soc. Inf. Sci.*, 41(6):391–407, 1990.

Adji B. Dieng, Chong Wang, Jianfeng Gao, and John W. Paisley. Topicrnn: A recurrent neural network with long-range semantic dependency. In *ICLR (Poster)*. OpenReview.net, 2017.

Adji Bousso Dieng, Francisco J. R. Ruiz, and David M. Blei. Topic modeling in embedding spaces. *Trans. Assoc. Comput. Linguistics*, 8:439–453, 2020.

Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *KDD*, pages 1555–1564. ACM, 2016a.

Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *KDD*, pages 1555–1564. ACM, 2016b.

Bing Fang, Shaoyi Liao, Kaiquan Xu, Hao Cheng, Chen Zhu, and Huapin Chen. A novel mobile recommender system for indoor shopping. *Expert Syst. Appl.*, 39 (15):11992–12000, 2012.

Mehrdad Farajtabar. *Point process modeling and optimization of social networks.* PhD thesis, Georgia Institute of Technology, 2018. URL `http://hdl.handle.net/1853/59858`.

Mikhail Figurnov, Shakir Mohamed, and Andriy Mnih. Implicit reparameterization gradients. In *Advances in Neural Information Processing Systems*, NIPS 2018, 2018.

Jonah Gabry, Daniel Simpson, Aki Vehtari, Michael Betancourt, and Andrew Gelman. Visualization in Bayesian workflow. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 182(2):389–402, 2019.

Bruno Giovanni Galuzzi, Ilaria Giordani, Antonio Candelieri, Riccardo Perego, and Francesco Archetti. Bayesian optimization for recommender system. In *Optimization of Complex Systems: Theory, Models, Algorithms and Applications*, pages 751–760, 2019.

Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis.* CRC press, 2013.

Andrew Gelman, Daniel Simpson, and Michael Betancourt. The prior can often only be understood in the context of the likelihood. *Entropy*, 19(10):555, 2017. doi: 10.3390/e19100555. URL `https://doi.org/10.3390/e19100555`.

Andrew Gelman, Aki Vehtari, Daniel Simpson, Charles C. Margossian, Bob Carpenter, Yuling Yao, Lauren Kennedy, Jonah Gabry, Paul-Christian Bürkner, and Martin Modrák. Bayesian workflow, 2020.

Manuel Gomez-Rodriguez, Jure Leskovec, and Bernhard Schölkopf. Modeling information propagation with survival theory. In *ICML (3)*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 666–674. JMLR.org, 2013.

Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. *Deep Learning.* Adaptive computation and machine learning. MIT Press, 2016.

Leo A. Goodman. *Latent Class Analysis: The Empirical Study of Latent Types, Latent Variables, and Latent Structures*, page 3–55. Cambridge University Press, 2002. doi: 10.1017/CBO9780511499531.002.

Prem Gopalan, Laurent Charlin, and David M. Blei. Content-based recommendations with poisson factorization. In *NIPS*, pages 3176–3184, 2014a.

Prem Gopalan, Francisco J. R. Ruiz, Rajesh Ranganath, and David M. Blei. Bayesian nonparametric poisson factorization for recommendation systems. In *AISTATS*, volume 33 of *JMLR Workshop and Conference Proceedings*, pages 275–283. JMLR.org, 2014b.

Prem Gopalan, Jake M. Hofman, and David M. Blei. Scalable recommendation with hierarchical poisson factorization. In *UAI*, pages 326–335. AUAI Press, 2015.

Olivier Gouvert, Thomas Oberlin, and Cédric Févotte. Recommendation from raw data with adaptive compound poisson factorization. In *UAI*, volume 115 of *Proceedings of Machine Learning Research*, pages 91–101. AUAI Press, 2019.

Alex Graves. Practical variational inference for neural networks. In *NIPS*, pages 2348–2356, 2011.

Peter D. Grünwald. *The Minimum Description Length Principle*, volume 1. The MIT Press, September 2007.

Joseph Y. Halpern. *Reasoning about Uncertainty*. The MIT Press, 04 2017.

Marcelo Hartmann, George Agiashvili, Paul Bürkner, and Arto Klami. Prior predictive elicitation via the prior predictive distribution. In *Uncertainty in Artificial intelligence*, 2020.

Alan G Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. 2016a.

Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *RecSys*, pages 241–248. ACM, 2016b.

Joseph M. Hilbe. *Modeling Count Data*. Cambridge University Press, 2014. doi: 10.1017/CBO9781139236065.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997.

Matthew D. Hoffman, David M. Blei, Chong Wang, and John W. Paisley. Stochastic variational inference. *J. Mach. Learn. Res.*, 14(1):1303–1347, 2013.

Thomas Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, pages 50–57. ACM, 1999.

Antti Honkela and Harri Valpola. Variational learning and bits-back coding: an information-theoretic view to bayesian learning. *IEEE Trans. Neural Networks*, 15(4):800–810, 2004.

H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933. doi: 10.1037/h0071325.

Changwei Hu, Piyush Rai, Changyou Chen, Matthew Harding, and Lawrence Carin. Scalable bayesian non-negative tensor factorization for massive count data. In *ECML/PKDD (2)*, volume 9285 of *Lecture Notes in Computer Science*, pages 53–70. Springer, 2015.

Changwei Hu, Piyush Rai, and Lawrence Carin. Non-negative matrix factorization for discrete data with hierarchical side-information. In *AISTATS*, volume 51 of *JMLR Workshop and Conference Proceedings*, pages 1124–1132. JMLR.org, 2016.

Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, pages 263–272. IEEE Computer Society, 2008.

Lawrence Hubert, Jacqueline Meulman, and Willem Heiser. Two purposes for matrix factorization: A historical appraisal. *SIAM Review*, 42(1):68–82, 2000. ISSN 00361445.

Dietmar Jannach and Malte Ludewig. When recurrent neural networks meet the neighborhood for session-based recommendation. In *RecSys*, pages 306–310. ACM, 2017.

E. T. Jaynes. *Probability Theory: The Logic of Science.* Cambridge University Press, 2003.

How Jing and Alexander J. Smola. Neural survival recommender. In *WSDM*, pages 515–524. ACM, 2017.

Bent Jorgensen. Exponential dispersion models. *Journal of the Royal Statistical Society. Series B (Methodological)*, 49(2):127–162, 1987.

Joseph B. Kadane, James M. Dickey, Robert L. Winkler, Wayne S. Smith, and Stephen C. Peters. Interactive elicitation of opinion for a normal linear model. *Journal of the American Statistical Association*, 75:845–854, 1980.

David Kaltenpoth and Jilles Vreeken. We are not your real parents: Telling causal from confounded using MDL. In *SDM*, pages 199–207. SIAM, 2019.

Jeon-Hyung Kang and Kristina Lerman. LA-CTR: A limited attention collaborative topic regression for social media. In *AAAI*. AAAI Press, 2013.

Komal Kapoor, Mingxuan Sun, Jaideep Srivastava, and Tao Ye. A hazard based approach to user return time prediction. In *KDD*, pages 1719–1728. ACM, 2014.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

Arto Klami, Seppo Virtanen, and Samuel Kaski. Bayesian canonical correlation analysis. *J. Mach. Learn. Res.*, 14(1):965–1003, 2013.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques.* MIT Press, 2009.

Yehuda Koren, Robert M. Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M. Blei. Automatic differentiation variational inference. *J. Mach. Learn. Res.*, 18: 14:1–14:45, 2017.

Daniel Kunin, Jonathan M. Bloom, Aleksandrina Goeva, and Cotton Seed. Loss landscapes of regularized linear autoencoders. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 3560–3569. PMLR, 2019.

Thomas K Landauer and Susan T Dumais. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211, 1997.

Helge Langseth and Thomas D. Nielsen. Latent classification models. *Mach. Learn.*, 59(3):237–265, 2005.

Paul F Lazarsfeld. The logical and mathematical foundation of latent structure analysis. *Studies in Social Psychology in World War II Vol. IV: Measurement and Prediction*, pages 362–412, 1950.

Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4):541–551, 1989.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521 (7553):436–444, 2015.

Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In *CIKM*, pages 1419–1428. ACM, 2017.

Daryl Lim, Julian McAuley, and Gert R. G. Lanckriet. Top-n recommendation with missing implicit feedback. In *RecSys*, pages 309–312. ACM, 2015.

Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Comput.*, 7(1):76–80, 2003.

Thomas Josef Liniger. *Multivariate Hawkes processes.* PhD thesis, Swiss Federal Institute of Technology (ETH), 2009. URL `https://doi.org/10.3929/ethz-a-006037599`.

Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. Context-aware sequential recommendation. pages 1053–1058, 2016.

Malte Ludewig and Dietmar Jannach. Evaluation of session-based recommendation algorithms. *User Model. User Adapt. Interact.*, 28(4-5):331–390, 2018.

Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *EMNLP*, pages 1412–1421. The Association for Computational Linguistics, 2015.

Hao Ma, Chao Liu, Irwin King, and Michael R. Lyu. Probabilistic factor models for web site recommendation. In *SIGIR*, pages 265–274. ACM, 2011a.

Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. Recommender systems with social regularization. In *WSDM*, pages 287–296. ACM, 2011b.

David J. C. MacKay. *Information theory, inference, and learning algorithms.* Cambridge University Press, 2003.

Andrés R. Masegosa, Rafael Cabañas, Helge Langseth, Thomas D. Nielsen, and Antonio Salmerón. Probabilistic models with deep neural networks. *Entropy*, 23 (1):117, 2021.

Hongyuan Mei and Jason Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *NIPS*, pages 6757–6767, 2017.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *ICLR (Workshop Poster)*, 2013.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning. *J. Mach. Learn. Res.*, 21:132:1–132:62, 2020. URL `http://jmlr.org/papers/v21/19-346.html`.

Joris M. Mooij, Oliver Stegle, Dominik Janzing, Kun Zhang, and Bernhard Schölkopf. Probabilistic latent variable models for distinguishing between cause and effect. In *NIPS*, pages 1687–1695. Curran Associates, Inc., 2010.

Francesco Orciuoli and Mimmo Parente. An ontology-driven context-aware recommender system for indoor shopping based on cellular automata. *J. Ambient Intell. Humaniz. Comput.*, 8(6):937–955, 2017.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. ISBN 0934613737.

Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901. doi: 10.1080/14786440109462720. URL `https://doi.org/10.1080/14786440109462720`.

Charles S. Peirce. The numerical measure of the success of predictions. *Science*, ns-4(93):453–454, 1884. ISSN 0036-8075. doi: 10.1126/science.ns-4.93.453-a.

Ioannis Psorakis, Stephen Roberts, Mark Ebden, and Ben Sheldon. Overlapping community detection using Bayesian non-negative matrix factorization. *Phys. Rev. E*, 83:066114, Jun 2011.

Sanjay Purushotham and Yan Liu. Collaborative topic regression with social matrix factorization for recommendation systems. In *ICML*. icml.cc / Omnipress, 2012.

Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *WSDM*, pages 459–467. ACM, 2018.

Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *RecSys*, pages 130–137. ACM, 2017.

D. Raikov. On the decomposition of Gauss and Poisson laws. *Izv. Akad. Nauk SSSR Ser. Mat.*, 2(1):91–124, 1938.

Rajesh Ranganath, Sean Gerrish, and David M. Blei. Black box variational inference. In *AISTATS*, volume 33 of *JMLR Workshop and Conference Proceedings*, pages 814–822. JMLR.org, 2014.

Jakob Gulddahl Rasmussen. Lecture notes: Temporal point processes and the conditional intensity function, 2018.

Yongli Ren, Martin Tomko, Flora Dilys Salim, Kevin Ong, and Mark Sanderson. Analyzing web behavior in indoor retail spaces. *JASIST*, 68(1):62–76, 2017. doi: 10.1002/asi.23587. URL `https://doi.org/10.1002/asi.23587`.

Yongli Ren, Martin Tomko, Flora Dilys Salim, Jeffrey Chan, Charles L. A. Clarke, and Mark Sanderson. A location-query-browse graph for contextual recommendation. *IEEE Trans. Knowl. Data Eng.*, 30(2):204–218, 2018.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461. AUAI Press, 2009a.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. pages 452–461, 2009b.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1278–1286. JMLR.org, 2014.

Jorma Rissanen. Fisher information and stochastic complexity. *IEEE Trans. Inf. Theory*, 42(1):40–47, 1996.

Dan Roth. On the hardness of approximate reasoning. *Artif. Intell.*, 82(1-2): 273–302, 1996.

Maja R. Rudolph, Francisco J. R. Ruiz, Stephan Mandt, and David M. Blei. Exponential family embeddings. In *NIPS*, pages 478–486, 2016.

Massimiliano Ruocco, Ole Steinar Lillestøl Skrede, and Helge Langseth. Inter-session modeling for session-based recommendation. In *DLRS@RecSys*, pages 24–31. ACM, 2017.

Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, NIPS 2007, page 1257–1264, 2007.

John Salvatier, Thomas V. Wiecki, and Christopher Fonnesbeck. Probabilistic programming in python using pymc3. *PeerJ Comput. Sci.*, 2:e55, 2016.

Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295. ACM, 2001.

Daniel J. Schad, Michael Betancourt, and Shravan Vasishth. Toward a principled Bayesian workflow in cognitive science. *arXiv preprint arXiv:1904.12765*, 2019.

Aaron Schein. *Allocative Poisson Factorization for Computational Social Science*. PhD thesis, University of Massachusetts Amherst, 2019. URL `https://scholarworks.umass.edu/dissertations_2/1656/`.

Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

Albert Nikolaevich Shiryaev and R. P. Boas. *Probability (2nd Ed.)*. Springer-Verlag, Berlin, Heidelberg, 1995. ISBN 0387945490.

Umut Simsekli, Ali Taylan Cemgil, and Yusuf Kenan Yilmaz. Learning the beta-divergence in tweedie compound poisson matrix factorization models. In *ICML (3)*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 1409–1417. JMLR.org, 2013.

Anders Skrondal and Sophia Rabe-Hesketh. Latent variable modelling: A survey. *Scandinavian Journal of Statistics*, 34(4):712–745, 2007.

Elena Smirnova and Flavian Vasile. Contextual sequence modeling for recommendation with recurrent neural networks. In *DLRS@RecSys*, pages 2–9. ACM, 2017.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, NIPS 2012, 2012.

C. Spearman. "general intelligence," objectively determined and measured. *The American Journal of Psychology*, 15(2):201–292, 1904. ISSN 00029556. URL `http://www.jstor.org/stable/1412107`.

G. W. Stewart. On the early history of the singular value decomposition. *SIAM Review*, 35(4):551–566, 1993. doi: 10.1137/1035134. URL `https://doi.org/10.1137/1035134`.

Vincent Y. F. Tan and Cedric Fevotte. Automatic relevance determination in nonnegative matrix factorization with the β-divergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(7):1592–1605, 2013.

Yong Kiam Tan, Xinxing Xu, and Yong Liu. Improved recurrent neural networks for session-based recommendations. pages 17–22, 2016.

Da Tang and Rajesh Ranganath. The variational predictive natural gradient. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 6145–6154. PMLR, 2019.

Jiliang Tang, Xia Hu, and Huan Liu. Social recommendation: a review. *Social Network Analysis and Mining*, 3(4):1113–1133, 2013. ISSN 1869-5469. doi: 10.1007/s13278-013-0141-9. URL `http://dx.doi.org/10.1007/s13278-013-0141-9`.

Michael E. Tipping and Christopher M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

Dustin Tran, Matthew D. Hoffman, Rif A. Saurous, Eugene Brevdo, Kevin Murphy, and David M. Blei. Deep probabilistic programming. In *ICLR (Poster)*. OpenReview.net, 2017.

Peter D. Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *J. Artif. Intell. Res.*, 37:141–188, 2010.

Bartlomiej Twardowski. Modelling contextual information in session-aware recommender systems with neural networks. In *RecSys*, pages 273–276. ACM, 2016.

Karen Ullrich. *A coding perspective on deep latent variable models.* PhD thesis, University of Amsterdam, 2020. URL `https://hdl.handle.net/11245.1/2d6e0b96-90d3-4683-bbbe-00d2a7f1dd54`.

Bjørnar Vassøy, Massimiliano Ruocco, Eliezer de Souza da Silva, and Erlend Aune. Time is of the essence: A joint hierarchical RNN and point process model for time and item predictions. In *WSDM*, pages 591–599. ACM, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

Aki Vehtari, Andrew Gelman, and Jonah Gabry. Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*, 27(5):1413–1432, 2017.

Jeroen K. Vermunt and Jay Magidson. Latent class models for classification. *Comput. Stat. Data Anal.*, 41(3-4):531–537, 2003.

Seppo Virtanen, Arto Klami, Suleiman A. Khan, and Samuel Kaski. Bayesian group factor analysis. In *AISTATS*, volume 22 of *JMLR Proceedings*, pages 1269–1277. JMLR.org, 2012.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1(1-2):1–305, 2008.

Chong Wang and David M. Blei. Collaborative topic modeling for recommending scientific articles. In *KDD*, pages 448–456. ACM, 2011.

Chong Wang and David M. Blei. Variational inference in nonconjugate models. *J. Mach. Learn. Res.*, 14(1):1005–1031, 2013.

Hao Wang, Binyi Chen, and Wu-Jun Li. Collaborative topic regression with social regularization for tag recommendation. In *IJCAI*, pages 2719–2725. IJCAI/AAAI, 2013.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Qingyun Wu, Hongning Wang, Liangjie Hong, and Yue Shi. Returning is believing: Optimizing long-term user engagement in recommender systems. In *CIKM*, pages 1927–1936. ACM, 2017.

Shuai Xiao, Junchi Yan, Xiaokang Yang, Hongyuan Zha, and Stephen M. Chu. Modeling the intensity function of point process via recurrent neural networks. In *AAAI*, pages 1597–1603. AAAI Press, 2017.

Teng Xiao, Hui Tian, and Hong Shen. Variational deep collaborative matrix factorization for social recommendation. In *PAKDD (1)*, volume 11439 of *Lecture Notes in Computer Science*, pages 426–437. Springer, 2019.

Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *Proceedings ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, page 267–273, 2003.

Sinan Yildirim, M. Burak Kurutmaz, Melih Barsbey, Umut Simsekli, and A. Taylan Cemgil. Bayesian allocation model: Marginal likelihood-based model selection for count tensors. *IEEE J. Sel. Top. Signal Process.*, 15(3):560–573, 2021.

Quan Yuan, Li Chen, and Shiwan Zhao. Factorization vs. regularization: fusing heterogeneous social relationships in top-n recommendation. In *RecSys*, pages 245–252. ACM, 2011.

Daniel Zelterman. Discrete distributions : applications in the health sciences, 2004.

Qiang Zhang, Aldo Lipani, Ömer Kirnap, and Emine Yilmaz. Self-attentive hawkes process. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 11183–11193. PMLR, 2020.

Mingyuan Zhou, Lauren Hannah, David B. Dunson, and Lawrence Carin. Beta-negative binomial process and poisson factor analysis. In *AISTATS*, volume 22 of *JMLR Proceedings*, pages 1462–1471. JMLR.org, 2012.

Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. What to do next: Modeling user behaviors by time-lstm. In *IJCAI*, pages 3602–3608. ijcai.org, 2017.

Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. Transformer hawkes process. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 11692–11702. PMLR, 2020.

ISBN 978-82-326-5463-5 (printed ver.)
ISBN 978-82-326-5273-0 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

NTNU

Norwegian University of
Science and Technology