Estimating Network Flow Length Distributions via Bayesian Nonnegative Tensor Factorization

Barış Kurt*

Ali Taylan Cemgil*

Güneş Karabulut Kurt†

Engin Zeydan‡

September 5, 2019

Abstract

In this paper, we develop a framework to estimate network flow length distributions in terms of the number of packets. We model the network flow length data as a three-way array with day-of-week, hour-of-day, flow-length as entities where we observe a count. In a high-speed network, only a sampled version of such an array can be observed and reconstructing the true flow statistics from fewer observations becomes a computational problem.

We formulate the sampling process as matrix multiplication so that any sampling method can be used in our framework as long as its sampling probabilities are written in matrix form. We demonstrate our framework on a high volume real-world data set collected from a mobile network provider with a random packet sampling and a flow-based packet sampling methods. We show that modeling the network data as a tensor improves estimations of the true flow length histogram in both sampling methods.

Keywords: network flow length distribution, packet sampling, flow sampling, nonnegative tensor factorization

*Department of Computer Engineering, Bogazici University, Istanbul, Turkey

†Istanbul Technical University, Istanbul, Turkey

‡Centre Technologic de Telecomunicacions de Catalunya, Castelldefels, Spain, 08860. Email: engin.zeydan@cttc.cat

1 Introduction

Monitoring network statistics is crucial for the maintenance and infrastructure planning for network service providers. Statistical information about traffic patterns helps a service provider to characterize its network resource usage and user behavior, to infer future traffic demands, to detect traffic and usage anomalies, and to provide insights to improve the performance of the network [1]. However, network monitoring has become a difficult task due to increasingly high-volume and high-speed data over modern networks and in most cases, it requires special hardware. For this reason, sampling [2] becomes a viable approach for extracting statistics from such high-speed networks. In this work, we are interested in one of the most important network statistics, the flow length distribution (FLD).

A network flow is defined as a set of internet protocol (IP) packets with the same signature observed within a limited time period. The flow signature is composed of the IP and port pairs of both source and destination nodes together with level-3 protocol types such as transport control protocol (TCP) or user datagram protocol (UDP). A flow starts with the arrival of the first packet and terminated when the inter-packet timeout is exceeded. The total number of packets in a flow is referred to as the flow length and the length distribution of a set of flows that are terminated in a time window is called flow length distribution.

In this work, we are using one of the most popular methods for collecting per-flow information, i.e., passive measurement. In this method, network packets are processed as they pass through a passive measurement beacon connected to the network, e.g, router. The beacon keeps a look-up table for flow identification. The beacon processes a packet by searching its corresponding flow inside the look-up table using its signature. If such a flow is found, its statistics are updated. Otherwise, the packet is treated as the first packet of a new flow, and the new flow is inserted into the table. Once a flow is terminated, its statistics are transferred to a storage.

The flow length histogram can be calculated exactly by processing every packet that passes through the measurement beacon. In order to implement such a direct method, the monitoring beacon needs to maintain a table to hold information for all active flows on the network. However, substantial amount of concurrent flows with very short packet inter-arrival times of current high-speed networks (on the order of 10 Gbps to 100 Gbps inside carrier's network today) make this brute-force counting method very costly to implement. First of all, this method would require a large amount of memory to record the flow table. Secondly, in a high-speed link, the inter-arrival times between packets, which may be as small as 8 nanoseconds in an OC-768 link,

may be smaller than the time required to process flow hash operations such as identifying the packet and updating the flow statistics.

The characteristics of the network traffic data inevitably lead to the development of alternative methods for measurement such as random sampling, where a fraction of the network traffic is randomly selected and processed. The simplest sampling method is the uniform packet sampling [3, 4, 5, 6], used in commercial systems [7] and [8]. In uniform sampling, each packet is selected with a predefined constant probability. This approach is easy to implement since it does not require the flow identification of each packet. However, recovering the true flow length distribution from the random packet sampled traffic is a challenging problem. The unbiased estimator of the original flow length n for sampling probability is $\hat{n}(m) = m/$, where m is observed flow length. The relative error of this estimator, calculated as $\sqrt{1/(-1)n}$ [3], grows unboundedly for short flows as the sampling rate gets smaller. The high error on the small flow lengths comes from the fact that most of the samples are collected from longer flows.

Flow-based adaptive sampling methods [9, 10, 11, 12, 13, 14] were proposed for more accurate flow length estimation. In these methods, each incoming packet is processed and then sampled with a probability that is a function of the current sampled length of the flow that the packet belongs to. Here, the main idea is to use a smaller memory by compressing the flow statistics counters in the router. However, these methods need to be implemented on specialized -and expensive- hardware due to the mandatory packet identification and lookup step.

Both packet-based and flow-based adaptive sampling methods rely on numerical methods to recover the true FLD. In this work, we propose a framework that can be used to recover the true FLD from the sampled observation obtained by any sampling method. This framework uses a variant of the nonnegative tensor factorization (NTF) model, which we call the thin nonnegative tensor factorization (ThinNTF), where the "thin" prefix emphasizes that the factorization is applied directly to the sampled, or namely "thinned" data.

In our framework, the network traffic data is modeled as a 3-way array, containing the number of flow length observations, with dimensions interpreted as 1) flow length, 2) hour-of-day and 3) day-of-week to capture the hourly and daily periodicity in the data. The nonnegative factorization of this tensor basically gives us estimates in the form of a non-parametric mixture model. Therefore our model is an improvement of the non-parametric flow length models in [3] and [6] by having the capability of modeling data with an arbitrary amount of mixture components and use the periodicity.

While the ordinary NTF model [15] factorizes an observation tensor, the

ThinNTF directly factorizes its sampled version and recovers the original tensor from the estimated factors. We take a fully Bayesian approach here and provide a generative model for the ThinNTF and a variational Bayes algorithm for inference. The contributions in this paper can be listed as:

- We model one week of flow length observations as a 3-dimensional tensor and observe the periodic behavior.
- We propose a novel tensor factorization scheme, ThinNTF, which is able to find the factors of a latent tensor from its sampled counterpart. By doing so, we also solve the reconstruction problem.
- We apply ThinNTF to real-world data sampled with two different sampling methods: uniform random packet sampling and flow-based adaptive sampling.

The structure of the paper is as follows. In Section 2, we provide the related works on network sampling and tensor factorization. In Section 3, we describe our real-world data and how we visualize it as a tensor. Additionally, we describe the sampling methods that we used to sample the data. In Section 4, we describe our ThinNTF model and the variational Bayes algorithm for estimating the factors. In Section 5, we describe our real-world data collection architecture. In section 6, we present our synthetic and real-world experiments and results. Finally, in Section 7, we draw our conclusions.

2 Related Work

Sampling methods have long been applied to network traffic monitoring. A survey on fundamental network sampling strategies is given in [2]. Uniform packet sampling is extensively studied by various authors. Duffield et al. [3] propose the first non-parametric model for flow length distribution and provides a maximum likelihood estimation to recover the flow lengths. Riberio et al. [4] show that using protocol specific information gives better flow length distribution estimates in TCP flows. Yang et al. [6] adopt the maximum likelihood approach to estimate both flow length and flow volume (number of bytes in a flow) distributions. Additionally, they model the data with a non-parametric mixture model of two components, where the first component models small flows and the second models large ones.

Flow-based sampling methods are proposed as alternatives to the uniform packet sampling since packet sampling has theoretical limitations when recovering true flow statistics [5]. These methods process every incoming

packet and apply sampling conditionally. Kumar et al. [16, 13] propose two different algorithms where the flow size counters are compressed statistically. They also propose a non-uniform packet sampling algorithm based on sketch counting [12]. Hu et al. [10, 14] propose another non-uniform packet sampling algorithm, called adaptive nonlinear sampling (ANLS) for estimating flow lengths per each flow, and then adopts this method to flow volume [11]. In our experiments, we are going to use ANLS as an example of flow-based sampling methods since it is the current state-of-the-art non-uniform sampling method.

Nonnegative Tensor Factorization is the generalization of the nonnegative Matrix factorization (NMF) [17] to multiway arrays. In NMF, a nonnegative matrix is approximated with a multiplication of two nonnegative matrices. Minimizing the Kullback-Leibler divergence between the initial matrix and multiplied factors is a popular formulation of this method and can be solved with fixed-point iterations [18] or full Bayesian methods [19]. NMF has been used in many applications such as spectral data analysis [20], face recognition [17] and document clustering [21].

Modeling the flow length distribution as a mixture of distributions is first proposed by [6]. However, according to our best knowledge, there is no previous work that models a large volume of flow length data as a tensor. This work fills a gap in the literature by introducing tensor factorization methodology to network monitoring.

3 Problem Description

We describe our problem as a tensor thinning problem, where the count entities of the original flow lengths are stored in a tensor. We formulate the sampling process as a matrix multiplication operated on this data tensor. In order to do that, each sampling model should be represented as a matrix that transforms the original data tensor to a sampled one. We provide matrices for two sampling models: uniform packet sampling and ANLS flow-based packet sampling.

3.1 Notation and Indexes

For a clear notation, the scalar values are denoted by lightface letters, such as the index variable j and its maximum value J. The vectors are represented by boldface lower case letters, such as vector $\mathbf{x}$. Boldface upper case letters represent matrices, such as $\mathbf{F}$, $\mathbf{H}$ and $\mathbf{D}$, and the tensors are represented with calligraphic upper case letters i.e $\mathcal{X}$. The individual entries in matrices and

Index	Range	Description
i	$[1; I]$	Original flow lengths
	$[0; I]$	Sampled flow lengths
j	$[1; J]$	Hours of day
k	$[1; K]$	Days
r	$[1; R]$	Components

Table 1: Indexes in the model.

tensors are written like scalars, i.e. $f_{i;r}$ and $x_{i;j;k}$. The index : denotes all the entries in the given dimension. For example $s_{i;:}$ is the i^{th} row of the $\mathbf{S}$ matrix and $\mathcal{X}_{i;:;:}$ is the i^{th} slice of the tensor $\mathcal{X}$ in the first dimension.

The index parameters are also fixed for clarity. The list of indexes, their ranges and semantic descriptions are given in Table 1. For example, the i index always represents an original flow length, while presents a sampled flow length. The range of starts from 0, since all of the packets of a flow may be discarded during the sampling process yielding a zero-length sampled flow, which is never observed.

3.2 Data Tensor

The original flow length data is represented in an $I \times J \times K$ tensor $\mathcal{X}$, with individual elements $x_{i;j;k}$, regarded as the number of flows that has length i measured at the hour j of the day k. In this setup, I is the maximum flow length value, J is the hours of the day and K is the days of the week. For our real-world data, collected continuously for 1 week, these values are $I = 2000000; J = 24$ and $K = 7$.

Working with large maximum flow size is not feasible for two reasons. First, learning a mixture model where each flow component has 2 million parameters is not a good formulation for this problem. Secondly, 99.9% of flows in our data have less than 100 packets, which means the tensor $\mathcal{X}$ will be very sparse for $i > 100$. The clamping process can be defined as

$$\bar{\mathcal{X}}_{i;j;k} = \begin{cases} \mathcal{X}_{i;j;k} & \text{for } i < I_{max} \\ \sum_{I \; I_{max}} \mathcal{X}_{I;j;k} & \text{for } i = I_{max} \end{cases} \tag{1}$$

where $\bar{\mathcal{X}}$ is the clamped tensor. The clamping does not require any change in the model and inference equations that are given in Section 4. Therefore, for notational clarity we only use $\mathcal{X}$ as the generic original data tensor.

Figure 1 shows the vertical slices of our unsampled real-world data tensor $\mathcal{X}$, collected at the backbone of a mobile operator during a one week period,

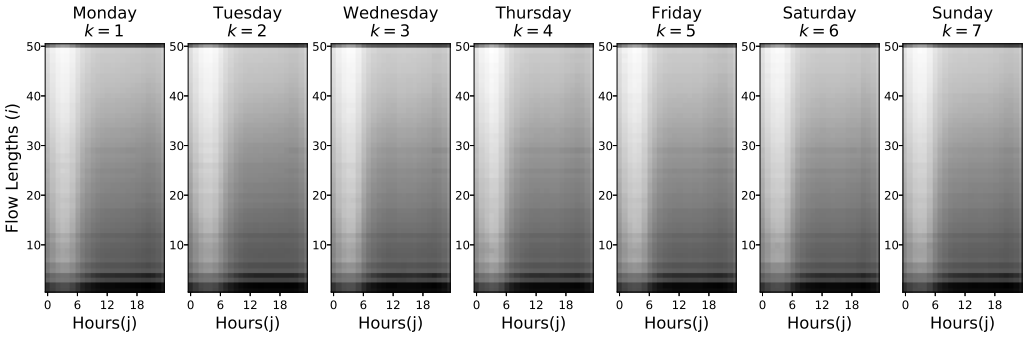


Figure 1: Slices of the original flow length tensor $\mathcal{X}$.

from Monday to Sunday, and clamped at $I_{max} = 50$. The intensity images are generated from the logarithm of the flow length counts. The daily and hourly patterns are easily recognizable in the original FLD data.

3.3 Sampling Methods

Independent of the sampling method, we can define an $I \times (I + 1)$ size $\mathbf{S}$ matrix, where I is the maximum flow length with entries $s_{i;}$ interpreted as the probability of sampling packets from an original flow of length i. Naturally, $\mathbf{S}$ is a lower diagonal matrix of the form

$$\mathbf{S} = \begin{bmatrix} s_{1;0} & s_{1;1} & 0 & 0 & \cdots & 0 \\ s_{2;0} & s_{2;1} & s_{2;2} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ s_{I;0} & s_{I;1} & s_{I;2} & s_{I;3} & \cdots & s_{I;I} \end{bmatrix} \tag{2}$$

where its I^{th} row defines a probability distribution for the sampled flow length of a flow of size I. Given a flow size distribution $\mathbf{x} \in \mathbb{Z}^{I}$, where $\mathbb{Z}$ is the set of nonnegative integers, the expected sampled flow length distribution would be given by $\hat{\mathbf{y}} = \mathbf{S}^T\mathbf{x}$. It immediately follows that the sampled flow size distribution y has length $I+1$, with y_0 is the proportion of sampled flows with none of their packets sampled. During the sampling process, this value will never be observed naturally since the flow identification is performed only on selected packets. In all experiments throughout this paper, the $\mathbf{y}$ vector (or $\mathcal{Y}$ tensor, which will be described later on) will be element-wise multiplied with a binary mask vector $\mathbf{m}$ (or binary mask tensor $\mathcal{M}$), whose entries are set to 1 except the ones corresponding to zero sampled flow lengths, in order to simulate the real life scenario.

Algorithm 1 Uniform Packet Sampling Algorithm

```
1: function SampleUniformlyRandom( , flow_table, packet)
2:     if    > rand_double(0, 1) then
3:         flow = flow_table.lookup(packet)
4:         if flow is null then
5:             flow = new Flow(packet)
6:         else
7:             flow.length += 1
8:         flow_table.insert_or_update(flow)
```

For any given sampling method, we can calculate the $\mathbf{S}$ matrix directly if a closed-form expression is available. Otherwise, it can be approximated by simulating the sampling process and counting the sampling statistics. In this paper, both uniform random sampling and the ANLS provide closed-form expressions for the calculation of $\mathbf{S}$ matrix.

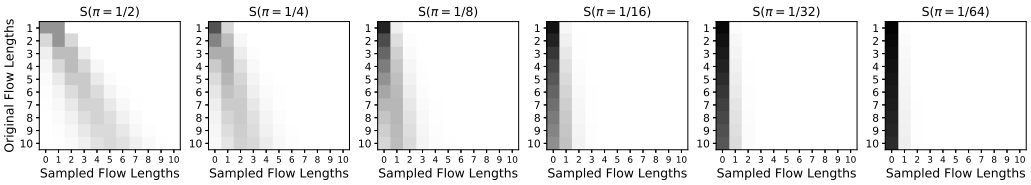
An important practical issue is that, if the original tensor $\mathcal{X}$ is clamped at I_{max}, the $\mathbf{S}$ matrix must also be clamped. In that case a last row entry $s_{I_{max},}$ must present the probability of selecting packets from a flow of length greater or equal to I_{max}. This clamping operation can be done by calculating a full size $\mathbf{S}$ matrix first, and setting $s_{I_{max},} \propto \sum_{i=I_{max}}^{I} s_{i,}$ with a naive assumption that after I_{max} the original flow sizes are uniformly distributed.

3.3.1 Uniform Sampling Method

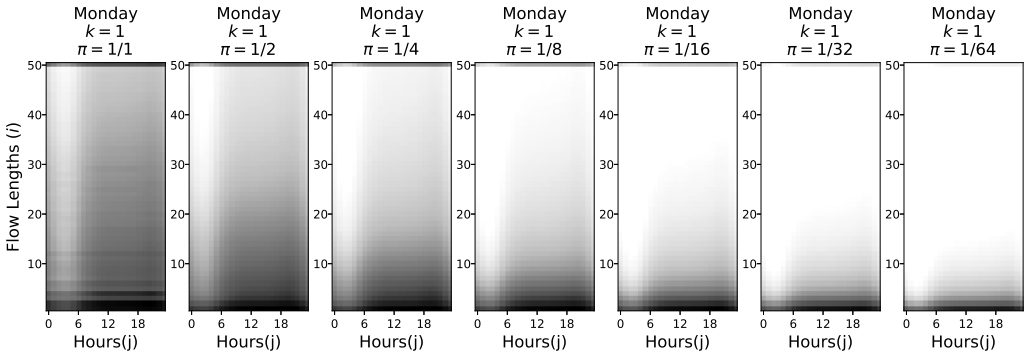
In uniform sampling, each packet is processed with a fixed probability of , irrespective of the flow it belongs to. In this method, the sampling matrix entries $s_{i,}$ are calculated directly through Binomial distribution with i trials and success probability, i.e.,

$$s_{i,} = \begin{cases} \binom{i}{} \quad (1 - \quad)^{i} & \text{for} \quad \leq i \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Algorithm 1 describes how the flow table is updated with uniform sampling upon the arrival of a new packet. The algorithm uniformly draws a random number in interval $[0; 1]$ and if it is less than , processes the packet, otherwise the packet is discarded. For processing the packet, a look-up operation is performed on the flow table to find and update the flow that the packet belongs to. If no such flow is found, a new flow is created using the packet's signature.



(a) Uniform sampling matrices.



(b) Uniformly sampled Monday data with varying sampling probabilities.

Figure 2: Sampling matrices for two different sampling schemes.

Figure 2a shows the top 10×11 entries of the lower diagonal $\mathbf{S}$ matrices with different sampling probabilities. As the sampling probability gets smaller, fewer packets from a flow gets observed, and the flow may even be missed when none of its packets are observed. The rightmost sampling matrix shows the case when = 1=64, where the matrix has a very high concentration of zero-length sampled flows.

Figure 2b shows the original Monday data (the leftmost matrix) and its sampled versions under uniform sampling with the probabilities shown on the top sampling matrices. Here we see that for = 1=64, the observed flow lengths are mostly less than 10, while the majority are not observed at all.

3.3.2 ANLS Sampling Method

The ANLS [10] will be used as the representative of the flow-based adaptive sampling methods. In ANLS, each packet is sampled according to the number of packets previously sampled from its corresponding flow. If a sampled flow has length x, the probability of its next packet to be sampled ($p(x; u)$) is calculated as

Algorithm 2 ANLS Sampling Algorithm

```
1: function SetupANLS(u)
2:     f[0] = 0
3:     for i ∈ [1; I] do
4:         f[i] = ((1 + u)^i − 1)=u
5:         p[i] = 1=(f[i − 1] − f[i])
6:     return f, p
7: function SampleWithANLS(p, flow_table, packet)
8:     flow = flow_table.lookup(packet)
9:     if flow is null then
10:        flow = new Flow(packet)
11:    else if p[flow.length] > rand_double(0, 1) then
12:        flow.length += 1
13:    flow_table.insert_or_update(flow)
```

$$f(x;u) = [(1+u)^x - 1]=u \tag{4}$$

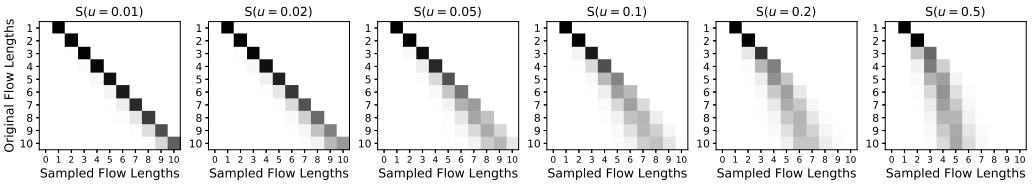
$$p(x;u) = 1=[f(x-1;u) - f(x;u)] \tag{5}$$

Here, $f(x;u)$ is a monotonically increasing function of flow length x, parametrized with u, which makes $p(x;u)$ monotonically decreasing.The u parameter determines the tendency of sampling packets. As u gets smaller, more packets are sampled and estimating original flow lengths gets easier.

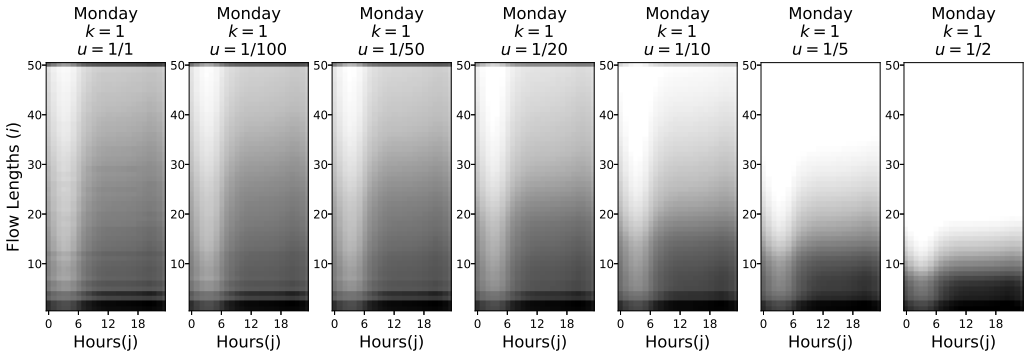
The ANLS method is described in details in Algorithm 2. Prior to the sampling, the **f** and **p** vectors are calculated in the SetupANLS function, according to equations 4 and 5. During sampling, for each incoming packet a look-up operation is performed unconditionally. If the corresponding flow is found, it is updated with probability relative to its current observed size. Otherwise, a new flow is created, ensuring that every flow is observed with at least one packet.

We calculate the sampling matrix **S** recursively for ANLS. In this method, the first packet is always sampled since $p(1;u) = 1$ independent of u. We start by assigning all zero sampling probabilities as $s_{:;0} = 0$ and $s_{1;1} = 1$. The recursive equation for calculating the sampling matrix can be deduced as

$$s_{i;} \propto s_{i\ 1;}\ \ {}_1p(i-1;u) + s_{i\ 1;}\ (1 - p(i-1;u)) \tag{6}$$



(a) ANLS sampling matrices.



(b) ANLS sampling with varying u.

Figure 3: The visualization of the Monday slice with uniform and ANLS sampling methods.

Figure 3 shows the ANLS sampling matrices for first 10 flow lengths and the Monday data sampled with them respectively, similarly to Figure 2. First, we can see that when u is small, the $\mathbf{S}$ matrix looks like identity and as u gets larger, the sampling probability of large flows decreases. Secondly, compared to uniform sampling, the ANLS method has much higher sampling ratios than uniform sampling. However, operating with such high sampling ratios would require specialized hardware in real time.

4 Methodology

Our methodology is based on the nonnegative factorization of the data tensor. Our model, which we call ThinNTF, introduces the sampling matrix as a constant factor to the original NTF with the Poisson-Gamma observation model. The rationale for using factorization for recovering true flow sizes is that the flow size distributions have daily periodic behavior, as we show in Section 3. Inferring the factors, instead of the original matrix, requires less parameter estimation and results in smoother estimates compared to the standard maximum likelihood estimation described in [3]. In this section, we first describe the original tensor factorization model and we provide our novel

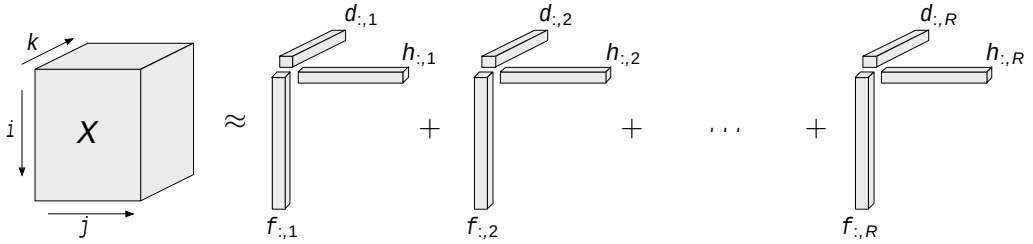


Figure 4: PARAFAC Factorization

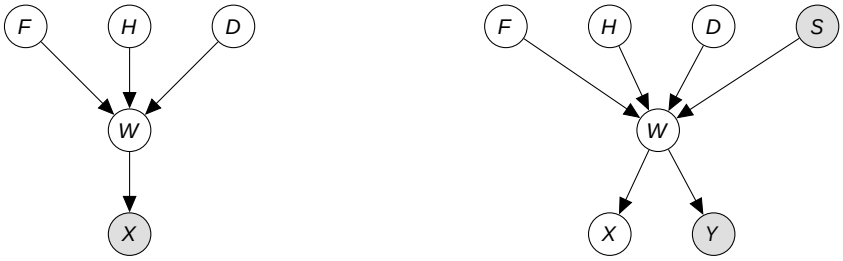
version: the ThinNTF. At the end of the section, we present the full-Bayesian variational Bayes algorithm for the inference.

4.1 Nonnegative Tensor Factorization

NTF is the generalization of the 2-dimensional NMF model to multiple dimensions. In NTF, an N-dimensional tensor is approximated by the multiplication of lower dimensional factors. Unlike NMF, tensor factorization can be done in multiple ways. In this work, we are going to use the PARAFAC [22, 23, 24] factorization scheme. In PARAFAC, an $I_1 \times I_2 \times \cdots \times I_N$ tensor is approximated by $I_n \times R$ matrices for $n \in [1; N]$. Here, R is the number of components, i.e. the number of clusters in the data. Figure 4 shows the PARAFAC factorization of our FLD tensor $\mathcal{X}$, into 3 factors: an $I \times R$ factor $\mathbf{F}$ for representing the flow length clusters, a $J \times R$ factor $\mathbf{H}$ for representing hourly behavior and a $K \times R$ factor $\mathbf{D}$ for representing the daily behavior of the data. Every single entry of the $\mathcal{X}$ tensor is approximated by

$$x_{i;j;k} \approx \hat{x}_{i;j;k} = \sum_r f_{i;r} h_{j;r} d_{k;r} \tag{7}$$

Bro [25] explains that the PARAFAC factorization is unique under certain circumstances, where uniqueness is defined as begin unable to rotate the factorization without loss of fit. NMF and NTF are statistical models that imposes nonnegativity constraint without uniqueness property. The uniqueness may be important if individual factors are of special interest. In our case, we are concerned with the estimation of the original data tensor $\mathcal{X}$ from sampled tensor $\mathcal{Y}$, but not the individual factors for any interpretation. Our problem is more close to a missing value imputation problem, hence uniqueness is not a requirement.



(a) NTF (b) ThinNTF

Figure 5: Graphical models representing the dependency structure of NTF and ThinNTF models in PARAFAC scheme.

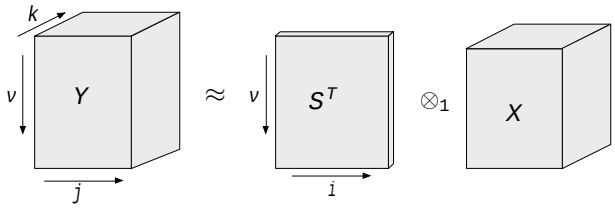


Figure 6: ThinNTF Model

4.2 Thin Nonnegative Tensor Factorization

ThinNTF is basically an NTF with an additional constant factor, which in our case is the sampling matrix S. Figure 5 shows the graphical models of the NTF and the ThinNTF models for factorizing original and sampled flow length observations. In the graphical models, the shaded nodes are the observed entities, and the unshaded ones are the latent entities.

In Section 3.3, we have described the sampling process as a matrix multiplication operation with a sampling matrix $\mathbf{S}$. In ThinNTF, this sampling matrix operates on the original tensor $\mathcal{X}$ and creates a thinned version of it, which we call $\mathcal{Y}$, by down-sampling its entries according to a sampling scheme, as shown in Figure 6. The entries of $\mathcal{Y}$ tensor $y_{;j;k}$ presents the number of flows of sampled-length , at hour j at day k. The $\otimes_1$ operation denotes the 1-mode product of matrix $\mathbf{S}^T$ and tensor $\mathcal{X}$, which corresponds to the set of matrix multiplications $\mathcal{Y}_{:,:,k} = \mathbf{S}^T \mathcal{X}_{:,:,k}$ for $k \in [1; K]$.

In this scheme, one can immediately suspect that $\mathcal{X}$ can be estimated by $(\mathbf{S}^T)^{\ 1} \otimes_1 \mathcal{Y}$. However, this solution is not feasible for several reasons. First, the $\mathbf{S}$ matrix is not square, hence not invertible. Instead its pseudo-inverse can be calculated but this does not impose nonnegativity. Moreover, the top slice of the $\mathcal{Y}$ tensor, which stores the number of flows with zero-sampled size is never observed, hence must be estimated. Therefore we need a solid

Tensor	Index Set	Description
$\mathcal{X}$	$i;j;k$	Original flow length tensor
$\mathcal{Y}$	$;j;k$	Sampled flow length tensor
$\mathcal{M}$	$;j;k$	Mask tensor
$\mathcal{W}$	$;i;j;k;r$	Latent variable tensor
$\mathbf{F}$	$i;r$	Flow length factor
$\mathbf{H}$	$j;r$	Hour of day factor
$\mathbf{D}$	$k;r$	Day of week factor
$\mathbf{S}$	$i;$	Sampling matrix
$\mathbf{A}^F;\mathbf{B}^F$	$i;r$	Gamma priors for $\mathbf{F}$
$\mathbf{A}^H;\mathbf{B}^H$	$j;r$	Gamma priors for $\mathbf{H}$
$\mathbf{A}^D;\mathbf{B}^D$	$k;r$	Gamma priors for $\mathbf{D}$

Table 2: Tensors in the model and their corresponding index sets.

statistical model and an inference method to estimate $\mathcal{X}$ under this model.

In ThinNTF, we observe the $\mathcal{Y}$ tensor, but try to factorize the $\mathcal{X}$ tensor, which is latent (Figure 5b). In the end, the factors of $\mathcal{X}$ are going to provide us an approximation $\hat{\mathcal{X}}$ which solves the original flow length distribution reconstruction problem. We mathematically express this approximation as

$$y_{;j;k} \approx \hat{y}_{;j;k} = \sum_{i;r} s_{;i} f_{i;r} h_{j;r} d_{k;r} \tag{8}$$

where $\mathbf{F};\mathbf{H}$ and $\mathbf{D}$ are described in exactly the same way in the original NTF case. In subsections 3.3.1 and 3.3.2, we described two different $\mathbf{S}$ matrices for two different schemes. ThinNTF model can be employed with any sampling method as long as it is described with a sampling matrix.

4.3 Generative Model

Taking Bayesian approach, we first provide a generative model for the ThinNTF, then describe how we can estimate the posterior probabilities of model parameters (in this case, the factor matrices) conditioned on the sampled flow length observations $\mathcal{Y}$ and the sampling matrix $\mathbf{S}$ using the well known Bayes rule. Table 2 contains all tensors and matrices used in the model together with their index sets.

The original and latent data tensor $\mathcal{X}$, and the sampled and observed data tensor Y have nonnegative integer entries. The natural probability distribution for this type of count data is the Poisson distribution. We assume

that each entry of a latent 5-dimensional tensor $\mathcal{W}$ is drawn from a Poisson distribution whose parameters are functions of sampling matrix $\mathbf{S}$ and factors $\mathbf{F}$; $\mathbf{H}$ and, $\mathbf{D}$, such as

$$w_{;i;j;k;r} \sim \mathcal{PO}(w_{;i;j;k;r}; s_{;i} f_{i;r} h_{j;r} d_{k;r}) \tag{9}$$

where the Poisson distribution is defined as

$$\mathcal{PO}(w; \quad) = \exp(w \log \quad - \quad - \log \Gamma(w+1)) \tag{10}$$

We choose the prior distributions for the factor entries as the Gamma distribution since it is the conjugate prior of Poisson distribution [26]. For each entry of factor $\mathbf{F}$, we write

$$f_{i;r} \sim \mathcal{G}\left(f_{i;r}; a^f_{i;r}; \frac{b^f_{i;r}}{a^f_{i;r}}\right) \tag{11}$$

where the Gamma distribution is described as

$$\mathcal{G}(f; \; ; \Theta) = \exp((\quad - 1) \log f - \frac{f}{\Theta} - \quad \log \Theta - \log \Gamma(\quad)) \tag{12}$$

with shape parameter and scale parameter . In our generative model, the parameters for Gamma distributions are $= a^f_{i;r}$ and $\Theta = b^f_{i;r} / a^f_{i;r}$ respectively. This means that the mean of $f_{i;r}$ is $\Theta = b^f_{i;r}$, which is independent of $a^f_{i;r}$. The variance of $f_{i;r}$ becomes $\Theta^2 = (b^f_{i;r})^2 / a^f_{i;r}$, which means that as $a^f_{i;r}$ gets smaller, the factors gets sparser.

In order to avoid repetition, we are going to omit the equations regarding the factors $\mathbf{H}$ and $\mathbf{D}$ throughout the paper. These factors behave exactly like factor $\mathbf{F}$ and it's easy to derive equations related to these factors once their corresponding equation for $\mathbf{F}$ is given.

Finally, we generate $\mathcal{X}$ and $\mathcal{Y}$ tensor from $\mathcal{W}$. Each entry $w_{;i;j;k;r}$ of $\mathcal{W}$ can be interpreted as the number of original flows of length i, generated on hour j, day k, by cluster r and observed as length . By summing $\mathcal{W}$ over dimensions cluster (r) and original lengths (i), we get the sampled observations tensor $\mathcal{Y}$. Similarly, by summing $\mathcal{W}$ over dimensions cluster (r) and sampled lengths ( ), we get the original flow length tensor $\mathcal{X}$. The whole generative process is summarized in Algorithm 3. The set of all indexes and tensors in the model are summarized in Tables 1 and 2 respectively.

Algorithm 3 ThinNTF Generative Model

function RANDINIT($\mathbf{S}; \mathbf{A}^F; \mathbf{B}^F; \mathbf{A}^H; \mathbf{B}^H; \mathbf{A}^D; \mathbf{B}^D$)

// Sample factor $\mathbf{F}$ from Gamma($\mathbf{A}^F; \mathbf{B}^F$)

for all $i \in [1; I];\ r \in [1; R]$ **do**

$f_{i;r} \sim \mathcal{G}\left(f_{i;r}; a^f_{i;r}; \frac{b^f_{i;r}}{a^f_{i;r}}\right)$

// Sample factor $\mathbf{H}$ from Gamma($\mathbf{A}^H; \mathbf{B}^H$)

for all $k \in [1; K];\ r \in [1; R]$ **do**

$h_{j;r} \sim \mathcal{G}\left(h_{j;r}; a^h_{j;r}; \frac{b^h_{j;r}}{a^h_{j;r}}\right)$

// Sample factor $\mathbf{D}$ from Gamma($\mathbf{A}^D; \mathbf{B}^D$)

for all $j \in [1; J];\ r \in [1; R]$ **do**

$d_{k;r} \sim \mathcal{G}\left(d_{k;r}; a^d_{k;r}; \frac{b^d_{k;r}}{a^d_{k;r}}\right)$

// Sample latent tensor $\mathcal{W}$ from Poisson distributions

for all $\in [1; I+1];\ i \in [1; I];\ j \in [1; J];\ k \in [1; K];\ r \in [1; R]$ **do**

$w_{;i;j;k;r} \sim \mathcal{PO}(w_{;i;j;k;r}; s_{;i} f_{i;r} h_{j;r} d_{k;r})$

return $\{\mathbf{F}, \mathbf{H}, \mathbf{D}, \mathcal{W}\}$

function GENERATEDATA($\mathbf{S}; \mathbf{A}^F; \mathbf{B}^F; \mathbf{A}^H; \mathbf{B}^H; \mathbf{A}^D; \mathbf{B}^D$)

// Randomly initialize factors and latent tensor

$\{\mathbf{F}; \mathbf{H}; \mathbf{D}; \mathcal{W}\} \leftarrow$ RANDINIT($\mathbf{S}; \mathbf{A}^F; \mathbf{B}^F; \mathbf{A}^H; \mathbf{B}^H; \mathbf{A}^D; \mathbf{B}^D$)

// Generate original tensor $\mathcal{X}$

for all $i \in [1; I];\ j \in [1; J];\ k \in [1; K]$ **do**

$x_{i;j;k} = \sum_{;r} w_{;i;j;k;r}$

// Generate sampled tensor $\mathcal{Y}$

for all $\in [1; I+1];\ j \in [1; J];\ k \in [1; K]$ **do**

$y_{;j;k} = \sum_{i;r} w_{;i;j;k;r}$

return $\{\mathbf{F}, \mathbf{H}, \mathbf{D}, \mathcal{W}, \mathcal{X}, \mathcal{Y}\}$

4.4 Variational Bayes

After defining the generative model, we can inter the factors $\mathbf{F}; \mathbf{H}$, and $\mathbf{D}$ of a sampled flow length observation tensor $\mathcal{Y}$. In the original NMF paper, Lee and Seung [17] provide fixed-point update equations for inferring the factors. Bro [25] gives similar fixed-point equations for updating the factors in PARAFAC factorization. Cemgil [19] shows that these updates correspond to

the Kullback-Leibler minimization between the original matrix (or tensor $\mathcal{X}$) and the approximated one ($\hat{\mathcal{X}}$), and also provides a full Bayesian variational algorithm for the matrix factorization. Ermis et. al. [15] provide a similar variational algorithm for the Gamma-Poisson tensor factorization.

We start our Bayesian inference by calculating the posterior distributions over the factors $\mathbf{F}; \mathbf{H}$ and $\mathbf{D}$ conditioned on observed tensor $\mathcal{Y}$. For notational clarity, we introduce $= (\mathbf{A}^F; \mathbf{B}^F; \mathbf{A}^H; \mathbf{B}^H; \mathbf{A}^D; \mathbf{B}^D)$ as the list of model hyper-parameters. The log-likelihood observing $\mathcal{Y}$ under the model parameters is written as

$$\log p(\mathcal{Y}| \ ; \mathbf{S}) = \log \int_{\mathbf{F};\mathbf{H};\mathbf{D}} d\mathbf{F} \ d\mathbf{H} \ d\mathbf{D} \sum_{\mathcal{W}} p(\mathcal{Y}; \mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D}| \ ; \mathbf{S}) \tag{13}$$

This log-likelihood is intractable due to the integration over the latent factors, but it is lower bounded as

$$\begin{aligned} \log p(\mathcal{Y}| \ ; \mathbf{S}) &\leq \mathcal{L} && (14) \\ &= \langle \log p(\mathcal{Y}; \mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D}| \ ; \mathbf{S}) \rangle q(\mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D}) + \mathcal{H}_{q(\mathcal{W};\mathbf{F};\mathbf{H};\mathbf{D})} && (15) \end{aligned}$$

where q is an auxiliary joint distribution of latent factors. This bound is tight when $q(\mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D}) = p(\mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D}|\mathcal{Y}; \ ; \mathbf{S})$. However, this is also intractable to calculate. Instead, we use a variational approximation [27] for q such that

$$q(\mathcal{W}) \propto \exp\left(\langle \log p(\mathcal{Y}; \mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D}| \ ) \rangle_{q(\mathbf{F};\mathbf{H};\mathbf{D})} \right) \tag{16}$$

$$q(\mathbf{F}) \propto \exp\left(\langle \log p(\mathcal{Y}; \mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D}| \ ) \rangle_{q(\mathcal{W};\mathbf{H};\mathbf{D})} \right) \tag{17}$$

$$q(\mathbf{H}) \propto \exp\left(\langle \log p(\mathcal{Y}; \mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D}| \ ) \rangle_{q(\mathcal{W};\mathbf{F};\mathbf{D})} \right) \tag{18}$$

$$q(\mathbf{D}) \propto \exp\left(\langle \log p(\mathcal{Y}; \mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D}| \ ) \rangle_{q(\mathcal{W};\mathbf{F};\mathbf{H})} \right) \tag{19}$$

where we iteratively update the posterior distribution of each factor by calculating the expectation of the logarithm of the full joint likelihood $p(\mathcal{Y}; \mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D})$ under the posteriors of all other latent factors.

4.5 Update Equations

Here we provide the update equations for $q(\mathcal{W})$ and $q(\mathbf{F})$. The updates of $q(\mathbf{H})$ and $q(\mathbf{D})$ can be easily deduced from the update equations of $q(\mathbf{F})$.

The full joint likelihood whose expectation will be calculated at each step is

$$\begin{aligned}\mathcal{J} &= \log p(\mathcal{Y}; \mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D}|\ ) && (20)\\ &= \log p(\mathcal{Y}|\mathcal{W}) + \log p(\mathcal{W}|\mathbf{F}; \mathbf{H}; \mathbf{D}) \\ &\quad + \log p(\mathbf{F}|\ ) + \log p(\mathbf{H}|\ ) + \log p(\mathbf{D}|\ ) && (21)\end{aligned}$$

where $\mathcal{Y}|\mathcal{W}$ is a degenerate distribution (()) to make sure the summation of $\sum_{i;r} \mathcal{W}$ equals $\mathcal{Y}$. By inserting the necessary Poisson and Gamma distributions given in the generative model into the Equation 21 we get the following expression

$$\begin{aligned}\mathcal{J} &= \sum_{;j;k} m_{;j;k} \log \left(y_{;j;k} - \sum_{i;r} w_{;i;j;k;r} \right) \\ &\quad + \sum_{i;r} \sum_{;j;k} m_{;j;k} \Big(w_{;i;j;k;r} \log s_{;i} f_{i;r} h_{j;r} d_{k;r} \\ &\qquad - s_{;i} f_{i;r} h_{j;r} d_{k;r} - \log \Gamma(w_{;i;j;k;r} + 1) \Big) \\ &\quad + \sum_{i;r} \left((a^f_{i;r} - 1) \log f_{i;r} - f_{i;r} \frac{a^f_{i;r}}{b^f_{i;r}} - a^f_{i;r} \log \frac{b^f_{i;r}}{a^f_{i;r}} - \log \Gamma(a^f_{i;r}) \right) \\ &\quad + \sum_{j;r} \left((a^h_{j;r} - 1) \log h_{j;r} - h_{j;r} \frac{a^h_{j;r}}{b^h_{j;r}} - a^h_{j;r} \log \frac{b^h_{j;r}}{a^h_{j;r}} - \log \Gamma(a^h_{j;r}) \right) \\ &\quad + \sum_{k;r} \left((a^d_{k;r} - 1) \log d_{k;r} - d_{k;r} \frac{a^d_{k;r}}{b^d_{k;r}} - a^d_{k;r} \log \frac{b^d_{k;r}}{a^d_{k;r}} - \log \Gamma(a^d_{k;r}) \right) \quad (22)\end{aligned}$$

4.5.1 Update Rule for $q(\mathcal{W})$

Considering the terms in the log-likelihood expression in Equation 22, that only includes $w_{;i;j;k;r}$, we find that

$$\begin{aligned}q(w_{v;:;j;k;:}) &\propto \exp \left(m_{;j;k} \log \left(y_{;j;k} - \sum_{i;r} w_{;i;j;k;r} \right) \right. \\ &\qquad \left. + \sum_{i;r} m_{;j;k} \Big(w_{;i;j;k;r} \log s_{;i} f_{i;r} h_{j;r} d_{k;r} - \log \Gamma(w_{;i;j;k;r} + 1) \Big) \right) \\ &\propto \text{Multinomial}(w_{;j;k;:}; x_{i;j;k}; p_{;i;j;k;r})^{m_{;j;k}} \quad (23)\end{aligned}$$

where, $w_{:j:k:::}$ becomes multinomial distributed. The expectation of $\mathcal{W}$ is calculated as

$$p_{:i:j:k:r} = \frac{\exp\left(s_{:i} + \langle \log f_{i:r} \rangle + \langle \log h_{j:r} \rangle + \langle \log d_{k:r} \rangle\right)}{\sum_{i:r} \exp\left(s_{:i} + \langle \log f_{i:r} \rangle + \langle \log h_{j:r} \rangle + \langle \log d_{k:r} \rangle\right)} \tag{24}$$

$$\langle w_{:i:j:k:r} \rangle = y_{:j:k} p_{:i:j:k:r} \tag{25}$$

4.5.2 Update Rule for $q(\mathbf{F})$

Similarly, considering the terms in log-likelihood Equation 22 that only includes $f_{i:r}$, we find that

$$\begin{aligned} q(f_{i:r}) \propto & \left(\sum_{:j:k} m_{:j:k} \langle w_{:i:j:k:r} \rangle + a^f_{i:r} - 1 \right) \log f_{i:r} \\ & - \left(\sum_{:j:k} m_{:j:k} s_{:i} \langle h_{j:r} \rangle \langle d_{k:r} \rangle + \frac{a^f_{i:r}}{b^f_{i:r}} \right) f_{i:r} \qquad (26) \\ \propto & \text{ Gamma}(f_{i:r};\ {}^{f}_{i:r};\ {}^{f}_{i:r}) \qquad (27) \end{aligned}$$

where $f_{i:r}$ becomes Gamma distributed with shape and scale parameters

$${}^{f}_{i:r} = a^f_{i:r} + \sum_{:j:k} m_{:j:k} \langle w_{:i:j:k:r} \rangle \tag{28}$$

$${}^{f}_{i:r} = \left(\frac{a^f_{i:r}}{b^f_{i:r}} + \sum_{:j:k} m_{:j:k} s_{:i} \langle h_{j:r} \rangle \langle d_{k:r} \rangle \right)^{1} \tag{29}$$

We calculate the expectation of $f_{i:r}$ and the logarithm of $f_{i:r}$ as

$$\langle f_{i:r} \rangle = {}^{f}_{i:r}\ {}^{f}_{i:r} \tag{30}$$

$$\langle \log f_{i:r} \rangle = \Psi({}^{f}_{i:r}) + \log {}^{f}_{i:r} \tag{31}$$

The variational Bayes algorithm that uses the above equations is given in Algorithm 4. The calculation of the lower bound is given in Appendix 1. The exact derivations of all equations can be found in [28].

4.6 Computational Complexity

The nonnegative tensor factorization is an NP-hard problem [29]. The variational Bayes algorithm we introduced in Algorithm 4 is an iterative solution that converges to a local maximum solution. The complexity of each iteration is determined by the leading term, which is the Equation 25. In general,

Algorithm 4 Variational Bayes Algorithm

function ThinNTF_VB($\mathcal{Y}$, $\mathbf{S}$, $\mathbf{A}^F; \mathbf{B}^F; \mathbf{A}^H; \mathbf{B}^H; \mathbf{A}^D; \mathbf{B}^D$)
  // Randomly initialize factors and latent tensor
  $\{\mathbf{F}; \mathbf{H}; \mathbf{D}; \mathcal{W}\} \leftarrow$ RandInit($\mathbf{S}; \mathbf{A}^F; \mathbf{B}^F; \mathbf{A}^H; \mathbf{B}^H; \mathbf{A}^D; \mathbf{B}^D$)
  repeat
    Calculate $\tilde{f}_{i;r}$; $\hat{f}_{i;r}$ and $\langle f_{i;r} \rangle$ as in Equations 28, 29 and 30.
    Calculate $\tilde{h}_{j;r}$; $\hat{h}_{k;r}$ and $\langle h_{j;r} \rangle$ similarly.
    Calculate $\tilde{d}_{k;r}$; $\hat{d}_{k;r}$ and $\langle d_{k;r} \rangle$ similarly.
    Calculate $\langle \log f_{i;r} \rangle$ as in Equation 31.
    Calculate $\langle \log f_{i;r} \rangle$ similarly.
    Calculate $\langle \log f_{i;r} \rangle$ similarly.
    Calculate $\langle w_{;i;j;k;r} \rangle$ as in Equation 25.
    Calculate lower bound
  until Max iterations are reached or lower bound converged
  return $\mathbf{F}$, $\mathbf{H}$, $\mathbf{D}$, $\mathcal{X}$

calculating a ThinNTF model with R components for a dimensional tensor with all dimensions of length N has $O(\ N^{(\ +1)}R)$ complexity for a single iteration.

5 Data Collection

Applying a tensor model to the flow length estimation problem requires high volume data collected over a long period, to capture the timely behavior of the network. The already available online data sets do not fulfill this requirement. Therefore, we collected our own real-world data from a mobile network service provider in Turkey [30]. The architecture of our system and the description of the data we collected as presented as follows.

5.1 System Architecture

The system architecture of a mobile operator's general packet radio service (GPRS) network infrastructure, including radio access and core network elements, is illustrated in Figure 7. IP traffic generated or received by mobile devices between mobile station (MS) and packet data network (PDN), e.g. IP Multimedia Subsystem (IMS), is tunneled in the core network of mobile operators through serving GPRS support node (SGSN) and gateway GPRS support node (GGSN) via the user data part of the GPRS tunneling protocol (GTP) [31]. The GTP message exchanges include information such as the

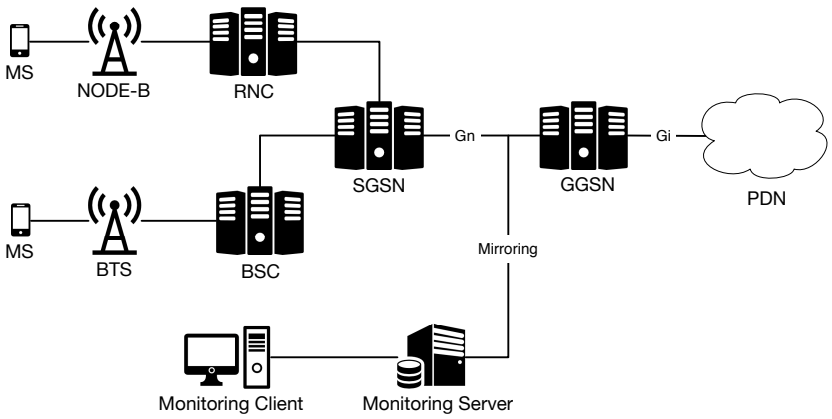


Figure 7: The placement of our monitoring server inside the premises of the mobile operator running a commercial cellular network.

size of the traffic, IP session start and end time, device and user identifiers.

The Gn interface [1] carries user packets to be transferred between the mobile users and the internet together with control packets necessary for the universal mobile telecommunications service (UMTS) core network [32]. All packets in this channel are carried by the GTP, which is an IP based protocol for carrying GPRS data within UMTS networks, used for data encapsulation in order to keep the core network independent of the protocols that are being used between MS and the packet-switched network.

The Gn interface carries mainly two types of GTP message structures or packets: GTP-C and GTP-U. GTP-C is used for signaling between SGSN and GGSN in core network which carries packet data protocol (PDP) Context messages such as activating and deactivating user session, configuring service parameters or updating the session. GTP-U is used for transmitting user data between the radio access network and core network. In our experiments, we filtered out GTP-C packets (since they are not considered to be part of a flow due to flow definition), which makes 10% of the total Gn traffic. Therefore, the sampling is applied to GTP-U packets only. GTP is carried mainly over UDP.

5.2 Data Extraction process

The mobile operators network consists of several districts with more than 10 regional core areas through-out Turkey. The average total traffic in all regional areas consists of approximately over 15 billion packets in the uplink

[1] Gn is an interface between SGSN and GGSN where GTP is the main protocol for network packets flowing through.

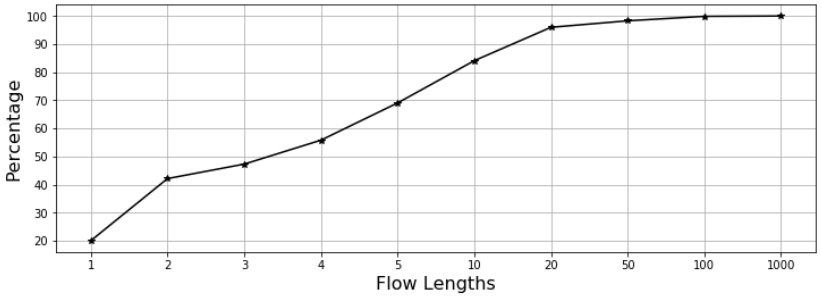


Figure 8: Cumulative flow lengths in the real world data.

direction and over 20 billion packets in the downlink direction daily. This corresponds to approximately 80 terabytes of total data flowing in uplink and downlink daily inside the mobile operators core network as a whole. In this work, the Gn interface which connects the SGSN and GGSN nodes are mirrored and the network traffic is transferred into a FLD server located at mobile operator's technology center in the core network. A speed of 200Mbit/sec at peak hours can be observed through one of the mirrored interfaces in the core network.

We monitored the network traffic in one of the servers of a mobile operator continuously for 10 days. We developed a packet extraction tool inside the monitoring server shown in Figure 7 which parses each GTP-U packet and stores the packet signature together with packet length and arrival time, discarding their payload. The stored network data is processed offline for extracting the true flow lengths.

After the data collection and flow extraction, the total number of packets collected is found to be 4×10^{11}, which makes up around 2.5×10^{10} flows. Figure 8 shows the cumulative flow length distribution of the data. We see that most of the flows have less than 5 packets, and 99.9% of them have less than 100 packets.

6 Experiments and Results

We designed two sets of experiments in order to verify our model: synthetic and real-world experiments. In each set, we sampled the original data with both uniform and ANLS models with different sampling parameters. Then we tried to recover the original tensor with ThinNTF models. The ThinNTF model takes a single parameter R which is the number of components in each factor. Additionally, we also represented data as $I \times JK$ matrix by unfolding the $\mathcal{X}$ tensor in the first dimension as described in [28], and applied the

2-dimensional version of ThinNTF, which we simply call thin nonnegative matrix factorization (ThinNMF). For Uniform sampling, we used the maximum likelihood estimation (MLE) defined in [3] as the baseline. For ANLS, we used both MLE and its own unbiased estimator of the model as baselines.

Both ThinNMF and ThinNTF models explain the data as a linear combination of R flow length distributions, stored in the columns of $\mathbf{F}$ matrix. In the ThinNMF model, we have JK coefficient sets for this combination. On the other hand, in ThinNTF, we have J coefficients for hour-of-day and K coefficients for day-of-week. The Cartesian product of these coefficient sets make a total of JK coefficients and creates a dependency between the hour and day attributes. Therefore, we expect that ThinNTF captures the periodicity and give better estimates.

During the experiments, we always run the stochastic algorithms, i.e. ThinNMF, ThinNTF, and MLE, for 10 times and keep the parameters of the model with the highest lower bound value. Then we reported the success of our algorithm with the weighted mean relative distance (WMRD) metric as this was used in all previous flow size estimation works. The WMRD is a metric which gives more weights to the relative differences that occur with larger frequency. It is formulated as

$$\mathrm{wmrd}(\mathbf{x}; \hat{\mathbf{x}}) = \frac{\sum_i |x_i - \hat{x}_i|}{\sum_i (x_i + \hat{x}_i) / 2} \tag{32}$$

where $\mathbf{x}$ is an original flow size distribution measured at the end of the hour and $\hat{\mathbf{x}}$ is its estimated version. For the whole tensor $\mathcal{X}$, we calculate the average WMRD value.

Additionally, we report the Kullback-Leibler divergence between the original and the estimated tensors, since this is the metric minimized during the variational Bayes algorithm. The KL divergence between two distributions $\mathbf{x}$ and $\hat{\mathbf{x}}$ is calculated as

$$\mathrm{KL}(\mathbf{x}; \hat{\mathbf{x}}) = \sum_i x_i \log\left(\frac{\hat{x}_i}{x_i}\right) \tag{33}$$

6.1 Experiments on Synthetic Data

We prepared our synthetic experiments to test the validity of our models. In this experiment set, we used the generative model of the ThinNTF model as described in Algorithm 3 to generate a small network with maximum flow size $I = 10; J = 24$ and $K = 7$. The original synthetic flow length distribution $\mathcal{X}$ is generated by a generative model with 3 components, where each

Period	ThinNMF-R3	ThinNTF-R3	MLE
2	0.53	0.49	0.88
4	0.63	0.59	1.20
8	0.65	0.61	1.29
16	0.68	0.61	1.41
32	0.74	0.61	1.37
64	0.85	0.61	1.50

Table 3: Uniform Sampling Results on Synthetic Data

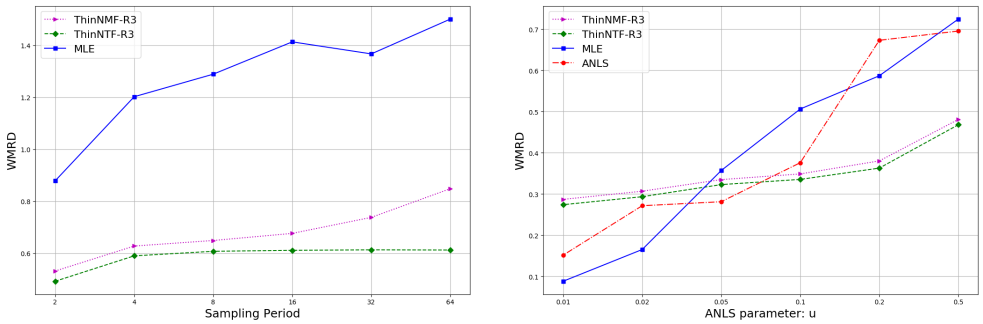
U	ThinNMF-R3	ThinNTF-R3	MLE	ANLS
0.01	0.29	0.27	0.09	0.15
0.02	0.31	0.29	0.17	0.27
0.05	0.33	0.32	0.36	0.28
0.1	0.35	0.34	0.51	0.38
0.2	0.38	0.36	0.59	0.67
0.5	0.48	0.47	0.72	0.70

Table 4: ANLS Sampling Results on Synthetic Data

component is a column in the **F** factor. We selected these 3 components as exponential, inverted exponential, and uniform random distributions. Therefore, in experiments, we used ThinNMF-R3 and ThinNTF-R3 models, where the suffix R3 shows that the model has 3 components.

We sampled the synthetic data with uniform and ANLS sampling methods with different sampling parameters. The sampling was done simply by randomly drawing a sampled size for each flow according to the sampling probabilities in the **S** matrix. By this way, we ignored the flow splitting problem and this gave us an ideal data for the ThinNTF model. We report and compare the mean standard deviation of these WMRD values for all experiments.

The ThinNTF model always performed best with the uniform sampling model, as shown in Table 3 as expected. On ANLS sampling, the MLE and the ANLS estimators performed better with high sampling probabilities, when $u \in (0.01; 0.02; 0.05)$, as shown in Table 4. On the other hand, when the sampling probability of the ANLS model decreases, the ThinNMF helped with better estimations. From the initial results, we conclude that the factorization is definitely helpful for more difficult uniform sampling method and helps lower the sampling probabilities in flow-based packet sampling. The results are also visible in Figure 9.



(a) Synthetic Data with Uniform Sampling

(b) Synthetic Data with ANLS Sampling

Figure 9: Synthetic experiment results.

6.2 Experiments on Real World Data

The original data collected from a mobile network provider as we described in Section 5, is sampled with both sampling methods. However, this time we simulated the real network offline by feeding the packets one by one to the monitoring server, as described in Section 5, This way, we were able to create the actual conditions on a sampler installed at a network provider's backbone. This also created the flow splitting problem, since we applied a 30 seconds time-out in our flow hash. We set the maximum flow length as $I = 100$, meaning that $\mathcal{X}_{100,:,:}$ entries show the count of flows that have more than 99 packets. This clamping decision was made according to the cumulative distribution of flow lengths as shown in Figure 8 We also clamped the sampling matrices $\mathbf{S}$ so that they exactly match the model.

Since the number of components in the original flow distribution is unknown, we run our experiments with $R \in [2; 3; 4]$ components for ThinNMF and ThinNTF. The rest of the experiment is similar to the synthetic one. The sampled Y matrix with shape $100 \times 24 \times 7$ is factorized and $\hat{\mathcal{X}}$ is reconstructed with the estimated factors. We reported and compared the mean and standard deviation of 24×7 WMRD and KL values.

The factorization models, both ThinNMF and ThinNTF helped lower the WMRD score in both uniform and ANLS sampling methods. ThinNTF-R4 model consistently gave lower error than the MLE baseline for uniform model as shown in Table 5 and Figure 10. Indeed, our factorization framework improved results overall for uniform sampling. However, since recovering true estimates in uniform sampling is quite difficult, we see less impact of the factorization as the sampling ratio increases.

Figure 10 also gives the KL values between the true and estimated flow

Period	ThinNMF			ThinNTF			MLE
	R=2	R=3	R=4	R=2	R=3	R=4	
2	0.23	0.24	0.23	0.21	0.25	0.22	0.41
4	0.55	0.52	0.53	0.50	0.48	0.49	0.69
8	0.94	0.93	0.94	0.91	0.90	0.87	0.97
16	1.15	1.11	1.11	1.09	1.05	1.04	1.05
32	1.25	1.24	1.24	1.16	1.13	1.10	1.22
64	1.31	1.29	1.30	1.09	1.06	1.04	1.22

Table 5: Uniform Sampling Results on Real Data

U	ThinNMF			ThinNTF			MLE	ANLS
	R=2	R=3	R=4	R=2	R=3	R=4		
0.01	0.03	0.02	0.01	0.05	0.04	0.03	0.05	0.12
0.02	0.04	0.03	0.02	0.06	0.04	0.03	0.08	0.21
0.05	0.04	0.03	0.02	0.07	0.05	0.04	0.13	0.39
0.1	0.06	0.05	0.04	0.08	0.07	0.05	0.17	0.61
0.2	0.08	0.08	0.08	0.10	0.09	0.07	0.21	0.70
0.5	0.13	0.13	0.11	0.16	0.15	0.13	0.33	0.94

Table 6: ANLS Sampling Results on Real Data

length distributions. While the scale of this metric is different, it gives consistent results with the WMRD. This shows that our model, which minimizes the KL metric also minimizes the commonly used WMRD metric, hence the model is suitable for this problem.

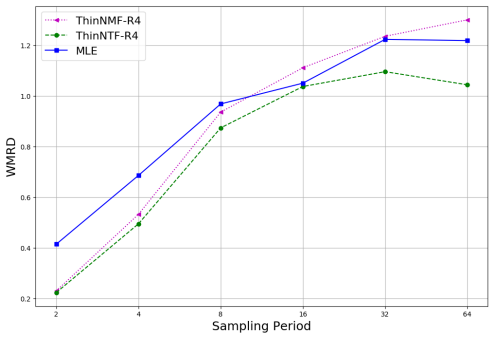
Another important issue is that for uniform sampling, 3-way factorization is more successful than the 2-way factorization. The periodicity information which is captured by the ThinNTF model help improve the estimates and makes it a more successful model for this sampling method.

In ANLS, all our factorization models gave lower error values than the MLE and unbiased estimator of ANLS as shown in Table 6 and Figure 11. Since ANLS is a more powerful sampling method than uniform sampling, our the effect framework is slightly less for small sampling parameter u. However, both ThinNMF and ThinNTF gave better result while sampling smaller number of packets (when u is large). Furthermore, since we are trying to recover the same original data in both experiments, we can compare our ThinNMF and ThinNTF models under two sampling methods. We see that in both methods as the number of components increases, the models gave lower error rates. However, with the uniform sampling method, 3-dimensional methods give better results, while with ANLS, 2-dimensional models perform slightly better.

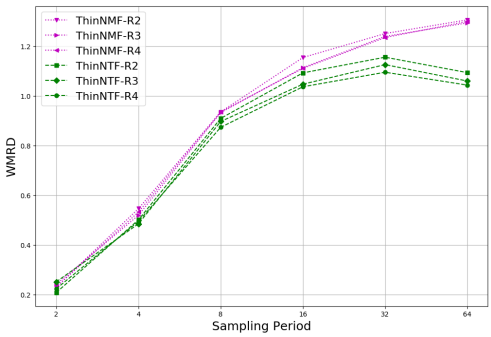
6.3 Effect of Clamping

The choice of where to clamp the data can be given by multiple factors. First of all, one can set the clamping value I_{max} according to a value of special interest. Otherwise, we would like to choose a small I_{max} so that we deal with a dense tensor and we deal with less parameters. On the other hand, we would like to set I_{max} as high as possible so that the clamped portion of the data is as small as possible.

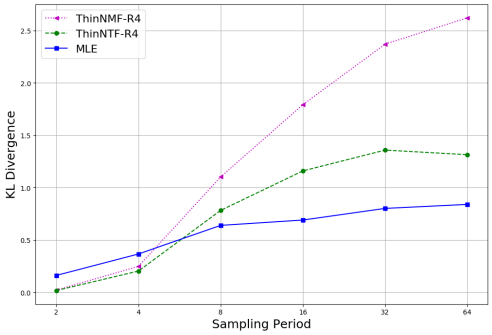
We run the best algorithms found in previous section for uniform and ANLS sampling methods with $I_{max} \in \{25; 50; 75; 100\}$. The WMRD values are given in Figure 12. In both methods, $I_{max} = 25$ gave relatively poor performance and $I_{max} = 100$ was generally the best choice. Also the results with $I_{max} \geq 50$ are closer to each other. This is consistent with the graph in Figure 8, where the cumulative flow lengths do not change much after $I_{max} = 50$. A final remark from this experiment is that as the clamping value increases estimation becomes harder with small sampling rates. This explains the results in uniform sampling with sampling rate 1/64.



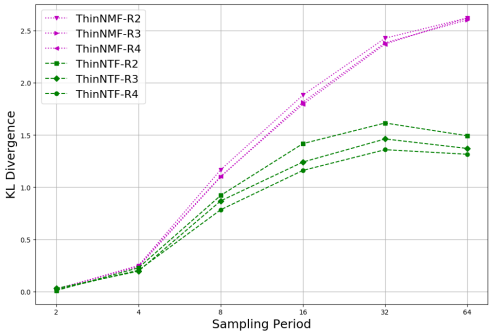
(a) WMRD metrics.



(b) WMRD metrics wrt. R.



(c) KL metrics



(d) KL metrics wrt. R.

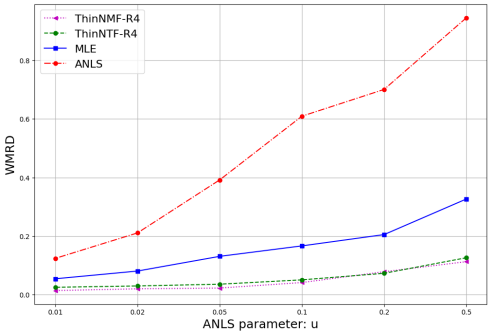
Figure 10: Real world data results with Uniform sampler.

7 Conclusions

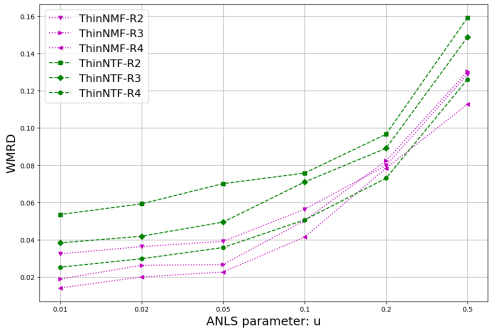
In this work, we introduced a novel nonnegative tensor factorization model called ThinNTF, which extends the classic nonnegative tensor factorization with an additional constant factor that can represent a network packet sampling method. We showed that this model can be employed to improve the current reconstruction algorithms in recovering the original flow length distributions.

We tested our model with two different types of sampling methods: the uniform packet sampling method and a flow-based packet sampling method, called ANLS. We described how to use these methods by showing how to build their sampling matrices.

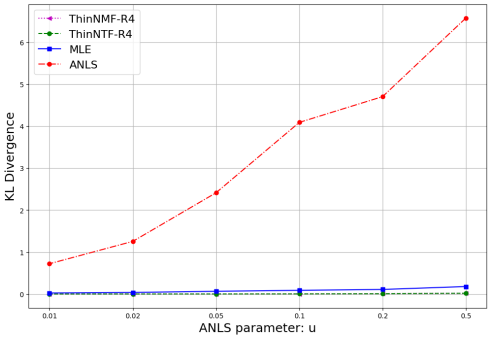
In order to test our model, we collected high-volume data from a mobile network provider for a long period in order to observe the periodical behavior of the flow length distribution. In experiments on synthetic and real-world data, our models gave promising results by lowering the estimation errors compared to the baselines of each sampling method. We conclude that our



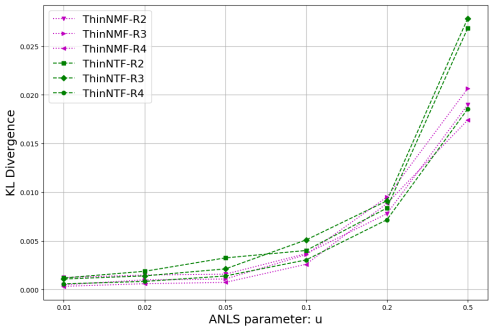
(a) WMRD metrics.



(b) WMRD metrics wrt. R.

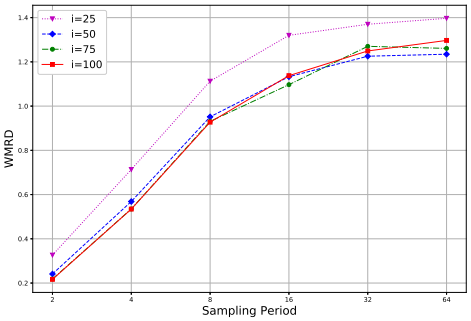


(c) KL metrics

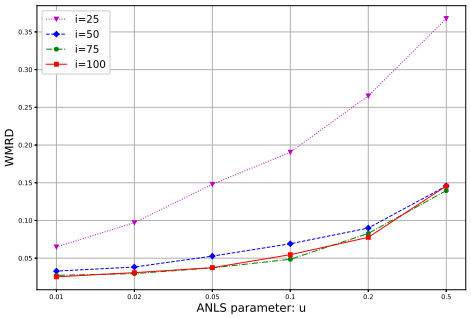


(d) KL metrics wrt. R.

Figure 11: Real world data results with ANLS sampler.



(a) Uniform Sampling ThinNMF-R4



(b) ANLS Sampling ThinNTF-R4

Figure 12: Clamping Experiments.

model can be used to decrease estimation errors or to decrease the sampling probabilities without increasing the estimation error.

An important issue left as future work is the online execution of the ThinNTF model. Theoretically, the ThinNTF model can be used online once sufficient data from the target network is collected and the flow length

distribution components, ie. the **F** factor, are inferred. The power of our model is that this inference can be done directly from the sampled observations. Once the **F** factor is estimated, for each incoming observation the corresponding entries in other factors can be inferred by keeping **F** constant during the inference. Moreover, **F** can be updated periodically, say weekly, in a sliding window fashion and kept up to date with the networks flow length behavior.

8 Conflict of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Appendix A Variational Lower Bound Calculation

The calculation of the lower bound includes a few arithmetic tricks. We provide a Bayesian nonnegative matrix factorization [28] tutorial for the detailed derivation and coding tricks. The final form of the lower bound equation is

$$
\begin{aligned}
\mathcal{L} &= \langle \log p(\mathcal{Y}; \mathcal{W}; \mathbf{F}; \mathbf{H}; \mathbf{D} \mid \;) \rangle_{q(W;\mathbf{F};\mathbf{H};\mathbf{D})} + H_{q(W;\mathbf{F};\mathbf{H};\mathbf{D})} \qquad (34) \\
&= - \sum_{;i;j;k;r} m_{;j;k} \left(s_{;i} \langle f_{i;r} \rangle \langle h_{j;r} \rangle \langle d_{k;r} \rangle \right) \\
&\quad + \sum_{i;r} \left(- \langle f_{i;r} \rangle \frac{a^f_{i;r}}{b^f_{i;r}} - a^f_{i;r} \log \frac{b^f_{i;r}}{a^f_{i;r}} - \log \Gamma(a^f_{i;r}) \right) \\
&\quad + \sum_{j;r} \left(- \langle h_{j;r} \rangle \frac{a^h_{j;r}}{b^h_{j;r}} - a^h_{j;r} \log \frac{b^h_{j;r}}{a^h_{j;r}} - \log \Gamma(a^h_{j;r}) \right) \\
&\quad + \sum_{k;r} \left(- \langle d_{k;r} \rangle \frac{a^d_{k;r}}{b^d_{k;r}} - a^d_{k;r} \log \frac{b^d_{k;r}}{a^d_{k;r}} - \log \Gamma(a^d_{k;r}) \right) \\
&\quad - \sum_{;j;k} m_{;j;k} \log \Gamma(y_{;j;k} + 1) \\
&\quad - \sum_{;i;j;k;r} m_{;j;k} \langle w_{;i;j;k;r} \rangle \log(p_{;i;j;k;r} / s_{;i}) \\
&\quad + \sum_{i;r} \left({}^{f}_{i;r} (\log {}^{f}_{i;r} + 1) + \log \Gamma({}^{f}_{i;r}) \right) \\
&\quad + \sum_{j;r} \left({}^{h}_{j;r} (\log {}^{h}_{k;r} + 1) + \log \Gamma({}^{h}_{j;r}) \right) \\
&\quad + \sum_{k;r} \left({}^{d}_{k;r} (\log {}^{d}_{k;r} + 1) + \log \Gamma({}^{d}_{k;r}) \right) \qquad (35)
\end{aligned}
$$

References

[1] G. Varghese and C. Estan, "The measurement manifesto," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 1, pp. 9–14, 2004.

[2] N. Duffield, "Sampling for passive internet measurement: A review," *Statistical Science*, vol. 19, no. 3, pp. 472–498, 2004.

[3] N. Duffield, C. Lund, and M. Thorup, "Estimating flow distributions from sampled flow statistics," *IEEE/ACM Transactions on Networking*, vol. 13, no. 5, pp. 933–946, 2005.

[4] B. F. Ribeiro, D. F. Towsley, T. Ye, and J. Bolot, "Fisher information of sampled packets: an application to flow size estimation," in *Internet Measurement Conference*, 2006, pp. 15–26.

[5] N. Hohn and D. Veitch, "Inverting sampled traffic," in *IMC '03: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*. ACM Press, 2003, pp. 222–233.

[6] L. Yang and G. Michailidis, "Sampled based estimation of network traffic flow characteristics," in *INFOCOM 2007, 26th IEEE International Conference on Computer Communications*, May 2007, pp. 1775–1783.

[7] Cisco netflow. [Online]. Available: http://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-netflow/index.html

[8] ntop. [Online]. Available: http://www.ntop.org/

[9] A. Kumar, M. Sung, J. J. Xu, and E. W. Zegura, "A data streaming algorithm for estimating subpopulation flow size distribution," *SIGMETRICS Perform. Eval. Rev.*, vol. 33, no. 1, pp. 61–72, 2005.

[10] C. Hu, B. Liu, S. Wang, Y. Cheng, and Y. Chen, "Anls: Adaptive non-linear sampling method for accurate flow size measurement," *IEEE Transactions on Communications*, vol. 60, no. 3, pp. 789–798, 2012.

[11] C. Hu, B. Liu, H. Zhao, K. Chen, Y. Chen, Y. Cheng, and H. Wu, "Discount counting for fast flow statistics on flow size and flow volume," *IEEE/ACM Transactions on Networking*, vol. 22, no. 3, pp. 970–981, 2014.

[12] A. Kumar and J. J. Xu, "Sketch guided sampling-using on-line estimates of flow size for adaptive data collection," in *Proc. 2006 IEEE INFOCOM*, 2006.

[13] A. Kumar, J. Xu, and J. Wang, "Space-code bloom filter for efficient per-flow traffic measurement," *IEEE J.Sel. A. Commun.*, vol. 24, no. 12, pp. 2327–2339, 2006.

[14] C. Hu, S. Wang, J. Tian, B. Liu, Y. Cheng, and C. Yan, "Accurate and efficient traffic monitoring using adaptive nonlinear sampling method," in *Proc 2008 IEEE INFOCOM*, 2009.

[15] B. Ermis and A. T. Cemgil, "A Bayesian Tensor Factorization Model via Variational Inference for Link Prediction." *ArXiv*, 2014.

[16] A. Kumar, M. Sung, J. J. Xu, and J. Wang, "Data streaming algorithms for efficient and accurate estimation of flow size distribution," *SIGMETRICS Perform. Eval. Rev.*, vol. 32, no. 1, pp. 177–188, 2004.

[17] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization." *Nature*, vol. 401, pp. 788–91, 1999.

[18] ——, "Algorithms for Non-negative Matrix Factorization," in *Advances in Neural Information Processing Systems 13*, no. 1, 2001, pp. 556–562.

[19] A. T. Cemgil, "Bayesian inference for nonnegative matrix factorisation models." *Computational intelligence and neuroscience*, vol. 2009, 2009.

[20] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons, "Algorithms and applications for approximate nonnegative matrix factorization," *Computational Statistics & Data Analysis*, vol. 52, no. June 2006, pp. 155–173, 2007.

[21] W. Xu, X. Liu, and Y. Gong, "Document Clustering Based On Non-negative Matrix Factorization," in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, 2003, pp. 267–273.

[22] F. L. Hitchcock, "The expression of a tensor or a polyadic as a sum of products," *J. Math. Phys*, vol. 6, no. 1, pp. 164–189, 1927.

[23] J. Douglas Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition," *Psychometrika*, vol. 35, pp. 283–319, 01 1970.

[24] R. A. Harshman, "Foundations of the parafac procedure: Model and conditions for an"explanatory"multi-mode factor analysis," *UCLA Work. Pap. Phon.*, vol. 16, 11 1969.

[25] R. Bro, "Parafac. tutorial and applications," *Chemometrics and Intelligent Laboratory Systems*, vol. 38, no. 2, pp. 149 – 171, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0169743997000324

[26] N. Johnson, A. Kemp, and S. Kotz, *Univariate Discrete Distributions*, ser. Wiley Series in Probability and Statistics. Wiley, 2005.

[27] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[28] Bayesian nonnegative matrix factorization tutorial. [Online]. Available: https://github.com/bariskurt/bptf

[29] S. Vavasis, "On the complexity of nonnegative matrix factorization," *SIAM Journal on Optimization*, vol. 20, no. 3, pp. 1364–1377, 2010.

[30] B. Kurt, E. Zeydan, U. Yabas, I. A. Karatepe, G. K. Kurt, and A. T. Cemgil, "A network monitoring system for high speed network traffic," in *2016 13th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, June 2016, pp. 1–3.

[31] 3GPP, "3gpp evolved packet system (eps); evolved general packet radio service (gprs) tunnelling protocol for control plane (gtpv2-c); stage 3," 3GPP TS 29.274 13.4.0, Tech. Rep., 2015.

[32] A. Springer and R. Weigel, *The Umts (Universal Mobile Telecom Standard) Physical Layer Basics,Standard,and Frontend Matters*. Berlin, Heidelberg: Springer-Verlag, 2002.