ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Learning &
Adaptive Systems

Master Thesis

Spatial Summarization of Image Collections

Spring Term 2016

This dissertation is submitted for the degree of *Master of Science ETH in Computer Science*

Supervised by:
Prof. Dr. Andreas Krause
Dr. Sebastian Tschiatschek
Alkis Gotovos, M.Sc.

Author:
Diego Alfonso Ballesteros Villamizar

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Authored by (in block letters):
For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.

Contents

List of Figures

List of Tables

Preface

Acknowledgments and such …

Abstract

The Abstract …

Symbols

Symbols

η	Learning rate
κ	Largest subset size
ν	Noise-to-data ratio
θ	Model parameters
σ	Gaussian distribution's standard deviation
$\mathcal{D}$	Set of data samples
K	Latent coherence dimensions
L	Latent diversity dimensions
M	Number of features
$\mathcal{N}$	Set of noise samples
$\mathbb{R}$	Real numbers
S, T	Subset
V	Ground set
Z	Normalization constant

Matrices

$\mathbf{B}$	Feature diversity weights
$\mathbf{E}$	Feature coherence weights
$\mathcal{I}$	Identity
$\mathbf{W}^b$	Item coherence weights
$\mathbf{W}^e$	Item diversity weights
$\mathbf{X}$	Item features

Vectors

$\mathbf{a}$	Utility weights for features
$\mathbf{u}$	Item utilities

Indices

i, j	Item

k	Feature
c	Coherence dimension
d	Diversity dimension

Acronyms and Abbreviations

ETH	Eidgenössische Technische Hochschule
FLID	Facility Location Diversity
FLDC	Facility Location Diversity and Coherence
FFLDC	Featurized Facility Location Diversity and Coherence
MLE	Maximum Likelihood Estimation
NCE	Noise Contrastive Estimation
LAS	Learning & Adaptive Systems
SGD	Stochastic Gradient Descent

Chapter 1

Introduction

Motivation (tourist routes and summarization), and quick idea of the methodology.

1.1 Contributions

Summarization of contributions.

Chapter 2

Related Work

2.1 Probabilistic Sub/super-modular models

Talk about the importance of these models and the recent interest in them, e.g. DPPs, Sampling.

2.2 Image collection summarization

Generally about the problem and the different approaches.

2.3 Tourist routes recommendation

The initial papers I found on this, basically Markov chains and clustering.

Chapter 3

Models

This chapter explores different probabilistic models for the problem, explains how they can be efficiently learned from data and presents the learning algorithm's performance on synthetic data.

3.1 Submodular model: FLID

The Facility Location Diversity (FLID) model was first proposed by Tschiatschek et al. (2016) and belongs to the class of log-submodular probability distributions over sets.

Definition 3.1.1. A distribution over sets $S \subseteq V$, where w.l.o.g $V = \{1, \ldots, |V|\}$, of the form $P(S) = \frac{1}{Z}\exp(F(S))$ is called a log-submodular probability distribution, if $F(S)$ is a submodular function (Djolonga and Krause, 2014).

Definition 3.1.2. A function $F : 2^V \rightarrow \mathbb{R}$ is submodular if it exhibits a "diminishing returns" property (Krause and Golovin, 2014), namely:

$$\forall S, T \subseteq V : S \subseteq T, i \notin T \mid F(S \cup i) - F(S) \geq F(T \cup i) - F(T)$$

Submodular functions intuitively indicate that adding an item to a smaller set results in a larger gain than adding it to a larger one. This is a natural property in the context of summarization where adding more information to a large summary is less effective than adding it to a smaller one.
The FLID model consists of two terms, a modular one which considers the individual utility or relevance of the items in the set, namely:

$$P(S) \propto \exp\left(\sum_{i \in S} u_i\right) \tag{3.1}$$

Where $u_i \in \mathbb{R}$ quantifies the utility of item i. In the context of location summarization, this utility could be proportional to the popularity of a place or how many times it has been photographed.
The diversity term is based on the idea of a latent concept space of dimension L where each item can be represented with a vector $\mathbf{w}_i^b \in \mathbb{R}^L_{\geq 0}$. The representation in this space allows the model to identify items that are similar and penalize sets that include them together. Formally, the diversity term for a set $S \subseteq V$ is:

$$\text{div}(S) = \sum_{d=1}^{L} \left(\max_{i \in S} w_{i,d}^b - \sum_{i \in S} w_{i,d}^b \right) \tag{3.2}$$

Putting this two terms together results in the FLID probability model proposed by Tschiatschek et al. (2016).

Table 3.1: FLID probability distribution for the scenario in example 3.1.2

S	$P(S)$
$\{h,s\},\{h,f\}$	≈ 0.41
$\{h\},\{s\},\{f\}$	≈ 0.06
$\{\},\{s,f\},\{h,s,f\}$	≈ 0.00

$$P(S) = \frac{1}{Z}\exp\left(\sum_{i\in S} u_i + \sum_{d=1}^{L}\left(\max_{i\in S} w_{i,d}^b - \sum_{i\in S} w_{i,d}^b\right)\right) \tag{FLID}$$

In this model, $\mathbf{u} \in \mathbb{R}^{|V|}$ is the vector of utilities and $\mathbf{W}^b \in \mathbb{R}_{\geq 0}^{|V|\times L}$ is the diversity weight matrix where each row is the aforementioned $\mathbf{w}_i^b$ vector. u_i is the i-th entry of $\mathbf{u}$ and $w_{i,d}^b$ is the (i,d)-th entry of $\mathbf{W}^b$. The positivity constraint on $\mathbf{W}^b$ is required to ensure that the model is log-submodular.

3.1.1 Partition function

In log-submodular probabilistic models, the normalization constant Z is known as the *partition function* (Djolonga and Krause, 2014) and its exact computation is known to be #P-complete (Jerrum and Sinclair, 1990). However, it has been proven that for FLID the partition function can be computed exactly in $O(|V|^{L+1})$ time, which can be efficient for $L \ll |V|$ (Tschiatschek et al., 2016). This is an important property because the partition function is necessary to compute marginal probabilities and other quantities from the model.

3.1.2 Example: Two landmarks

In order to illustrate the model, consider a town with 3 popular locations: A town hall (h), a statue (s) and a fountain (f). Data shows that visitors only take photos at either the town hall and the statue, or at the town hall and the fountain. This can be modeled with FLID by introducing a latent concept that discourages taking photos at both the fountain and statue.
Concretely, let $V = \{h,s,f\}$ and $\mathbf{u} = (2,2,2)^\intercal$, indicating that all locations are equally popular. A suitable diversity weight vector would then be $\mathbf{W}^b = (0,20,20)^\intercal$. Table 3.1 shows the resulting probabilities of the subsets, accurately representing the aforementioned description of the problem. Note that the probabilities are not exactly $0.5/0.5$ for the sets of interest but this can be easily fixed by increasing the utilities to ensure that sets of size 2 have a larger unnormalized magnitude compared to individual ones.

3.2 Mixed model: FLDC

Diversity is an important property in the context of summarization (Tschiatschek et al., 2016), however coherence is also a desired property of summaries (Yan et al., 2011). Balancing coherence and diversity is considered a challenge because maxiziming only one of these properties may lead to poor results on the other one (Shahaf et al., 2012).
In order to model coherence in this model, the addition of a log-supermodular term analogous to the diversity term 3.2 is proposed.

Definition 3.2.1. A function $F : 2^V \to \mathbb{R}$ is supermodular iff $-F(S)$ is submodular.

Table 3.2: Probability distribution for example 3.2.1

S	$P(S)$
$\{1,2\},\{3,4\}$	0.5
$2^V \setminus \{\{1,2\},\{3,4\}\}$	0.0

The supermodular term encodes the items into another latent concept space, of dimension K, where sets containing items with high values in some latent dimension are rewarded. Hence modeling complementarity between items.
For example, consider a spatial summary of a city where people tend to stay close to the city center. A possible latent dimension could encode the distance to the center, and rewarding coherence on this dimension would create summarizes where all locations are close together which is the modeled behavior.
The extended model will be reffered to as the Facility Location Diversity and Coherence (FLDC) model and its probability distribution is given by:

$$P(S) = \frac{1}{Z} \exp\left(\sum_{i \in S} u_i + \operatorname{div}(S) + \sum_{c=1}^{K} \left(\sum_{i \in S} w^e_{i,c} - \max_{i \in S} w^e_{i,c} \right)\right) \tag{FLDC}$$

Where $w^e_{i,c}$ is the (i,c)-th entry of $\mathbf{W}^e$. Each row of $\mathbf{W}^e$ encodes the representation of an item i in the newly introduced concept space of dimension K, hence $\mathbf{W}^e \in \mathbb{R}^{|V| \times K}_{\geq 0}$. Once again a positivity constraint is introduced, in this case to ensure the term is a supermodular function.
It should be noted that the FLDC model is neither log-submodular or log-supermodular unless $K = 0$ or $L = 0$, respectively. However, it can be used and learned in a similar fashion as the FLID model.

3.2.1 Example: Disjoint pairs

As an example of the extended model, consider the distribution presented in table 3.2 for $V = \{1,2,3,4\}$. It represents a set of two disjoint pairs, which indicates there exists a diversity component between the two pairs whilist having a coherence component between the items contained in each pair.
Concretely, the weight matrices $\mathbf{W}^b, \mathbf{W}^e$ in figure 3.1 illustrate one possible instance of the model. The corresponding utility vector is $\mathbf{u} = \overrightarrow{0}$, because there is no indication that individual items are favored over the pairs. Note that this model is easily interpretable and accurately realizes the distribution.

3.3 Featurized model: FFLDC

An important characteristic of the FLDC and FLID models is that they are agnostic to the type of items in the ground set, this allows its application to a wide range of problems without prior knowledge. However the downside is that the model has no capability to make use of information about the items, if available, to improve the modeling of the data. Moreover, if a new item is added to the set there is no way to generalize the existing knowledge about similar items to it.
In order to solve these problems, a further extension to the model is proposed. Firstly, the information about each item ($i \in V$) is represented as a vector $\mathbf{x}_i \in \mathbb{R}^M$ where each entry is a feature, e.g. for venues one feature could be its aggregated rating while another indicates whether it is indoors or outdoors.

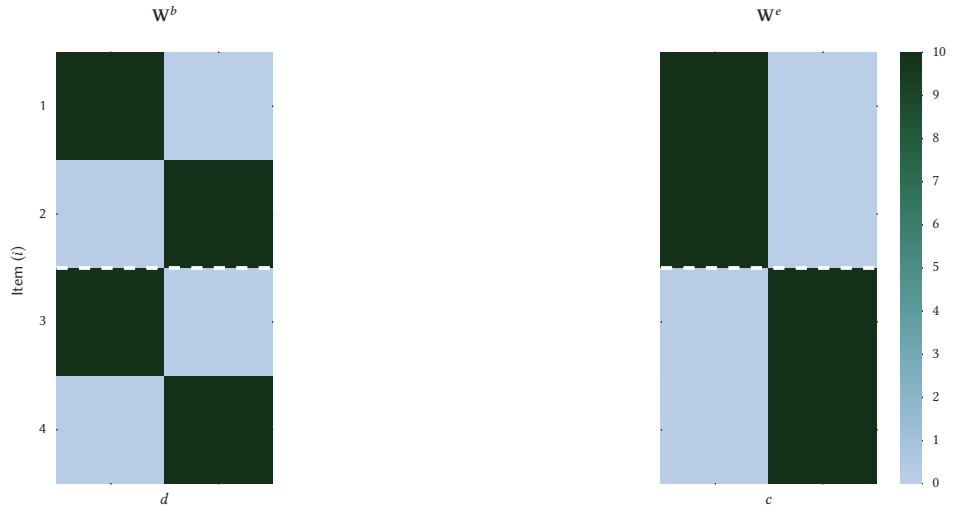

Figure 3.1: Diversity and coherence weights for FLDC model in example 3.2.1. The dotted line divides the paired items.

Then, the utility vector $\mathbf{u}$ and weight matrices $\mathbf{W}^b, \mathbf{W}^e$ are factorized to include the feature matrix $\mathbf{X} \in \mathbb{R}^{|V| \times M}$ as follows:

$$\mathbf{u} = \mathbf{X}\mathbf{a} \tag{3.3}$$

$$\mathbf{W}^b = \mathbf{X}\mathbf{B} \tag{3.4}$$

$$\mathbf{W}^e = \mathbf{X}\mathbf{E} \tag{3.5}$$

Where $a \in \mathbb{R}^M$ represents the contribution of each feature to the total utility of an item, whilist $\mathbf{B} \in \mathbb{R}^{M \times L}$ and $\mathbf{E} \in \mathbb{R}^{M \times K}$ encode the contribution of each feature to each latent diversity and coherence dimension, respectively. The intuition behind this factorization is that the information about the items can enhance the latent representations, hence producing a richer model.
The extended model will be reffered to as the Featurized Facility Location Diversity and Coherence (FFLDC) model and its probability distribution is given by:

$$P(S) = \frac{1}{Z} \exp\left(\sum_{i \in S} \sum_{k=1}^{M} x_{i,k} a_k + fdiv(S) + fcoh(S) \right) \tag{FFLDC}$$

$$fdiv(S) = \sum_{d=1}^{L} \left(\max_{i \in S} \sum_{k=1}^{M} x_{i,k} b_{k,d} - \sum_{i \in S} \sum_{k=1}^{M} x_{i,k} b_{k,d} \right) \tag{3.6}$$

$$fcoh(S) = \sum_{c=1}^{K} \left(\sum_{i \in S} \sum_{k=1}^{M} x_{i,k} e_{k,c} - \max_{i \in S} \sum_{k=1}^{M} x_{i,k} e_{k,c} \right) \tag{3.7}$$

Where a_i is the i-th entry of $\mathbf{a}$, $b_{k,d}$ is the (k,d)-th entry of $\mathbf{B}$, $e_{k,d}$ is the (k,e)-th entry of $\mathbf{E}$ and $x_{i,k}$ is the (i,k)-th entry of X.

Remark. If $\mathbf{X} = \mathcal{I}$, then FFLDC is equivalent to FLDC, with $\mathbf{a} = \mathbf{u}$, $\mathbf{W}^b = \mathbf{B}$ and $\mathbf{W}^e = \mathbf{E}$.

The use of features also allows the application of the model to previously unknown items, hence solving the aforementioned problem of generalization. This is because the parameters of the FFLDC model, i.e. $\mathbf{a}, \mathbf{B}, \mathbf{E}$, do not depend on the ground set V but rather on the

Table 3.3: Synthetic data for example 3.3.1

Locations visited (S)	$P(S)$
$\{1,3\}$	0.30
$\{3,4\}$	0.25
$\{3,6\}$	0.15
$\{2\}$	0.10
$\{1\}$	0.06
$\{3\},\{4\}$	0.04
$\{5\},\{6\}$	0.03

space of features $\mathbb{R}^M$. If an item $j \notin V$ is considered, a model learned on only items in V can immediately be applied to the new set $V \cup \{j\}$ using only its feature representation, contrary to the case of FLID or FLDC where it would require adding a new row to the weight matrices and learning it.

3.3.1 Example: Rated locations

A simple town has 6 popular locations, each of them has been rated from 0 (terrible) to 5 (excellent). It is known that a typical visitor cares about these ratings but also is interested in visiting places that are outdoors and/or serve food. Given data from previous visitors, shown in table 3.3, the task is to model this behavior using the FFLDC model.
In this example there is knowledge about the items and what features are relevant for the data. These features are summarized in equation 3.8. The first column corresponds to the aforementioned rating, the second and third are binary features indicating whether the location is an outdoor one and whether it serves food, respectively.

$$\mathbf{X} = \begin{pmatrix} 4 & 1 & 0 \\ 4 & 1 & 1 \\ 3 & 0 & 1 \\ 3 & 1 & 0 \\ 2 & 1 & 1 \\ 2 & 1 & 0 \end{pmatrix} \tag{3.8}$$

Looking at the data and features, it is possible to draw a FFLDC model that encourages diversity on the second a third feature while assigning a positive utility and coherence values to the first feature. One such model is presented in figure 3.2, this approximates distribution from table 3.3 and illustrates the type of model that is useful in this example. With this model, if a new location j is considered then it is straightforward to estimate what its probability of being visited $P(j \in S)$ would be, only knowing its feature representation.

3.4 Learning from data

As noted by Tschiatschek et al. (2016), Maximum Likelihood Estimation (MLE) is generally intractable in FLID due to the complexity of computing the partition function for large L. This result can be extended to the FLDC model utilizing the fact that the algorithm presented by Tschiatschek et al. (2016) for computing Z doesn't require the diversity weights $w^b_{i,d}$ to be positive, therefore it can be easily extended to included the coherence weights as negative diversity ones.

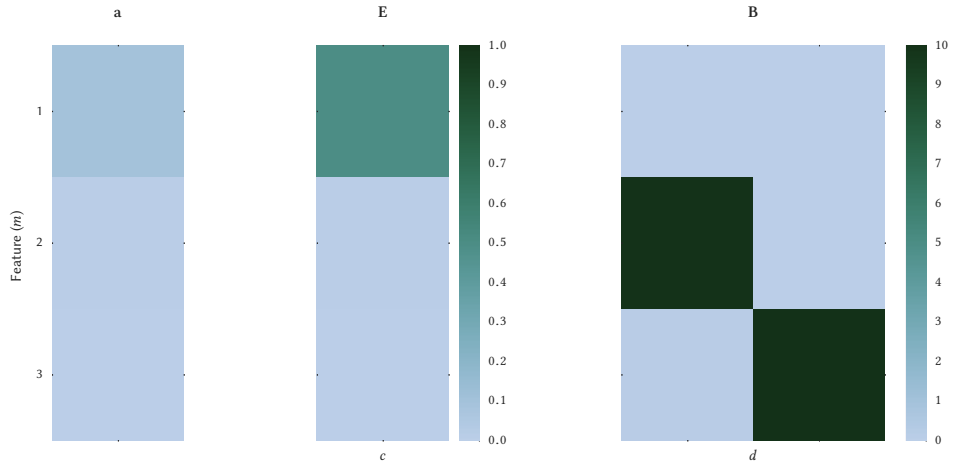

Figure 3.2: FFLDC sample model for example 3.3.1.

Proposition 3.4.1. *The time complexity of calculating the partition function for the FLDC model is* $O(|V|^{L+K+1})$*, using a modified version of the algorithm proposed by (Tschiatschek et al., 2016).*

Additionally, this result can be extended to the FFLDC case because a FFLDC model can always be reduced to an equivalent FLDC model through the factorization in equations 3.3-3.5. This reduction has a time complexity of $O(M|V|(L+K))$, corresponding to the matrix multiplications, which is considerably smaller than the complexity of the partition function computation for FLDC.
Fortunately, there exists an alternative method for estimating unnormalized probabilistic models from observed data, i.e. without computing the partition function. This method is called Noise Constrastive Estimation (NCE) (Gutmann and Hyvärinen, 2012) and it will be described in more detail in the following section.

3.4.1 NCE Learning

The idea behind NCE is to transform the unsupervised learning task of estimating a probability density from data into a supervised classification task. In order to do this, the observed data $\mathcal{D}$, assumed to be drawn from an unknown distribution P_d, is compared to an artificially generated set of noise samples $\mathcal{N}$ drawn from a known distribution P_n. The classification task is then setup to optimize the likelihood of correctly discriminating each sample as either observed data or artificial noise.
Formally, denote $\mathcal{A}$ as the complete set of labeled samples, i.e. $\mathcal{A} = \{(S, Y_s) : S \in \mathcal{D} \cup \mathcal{N}\}$ where $Y_s = 1 \equiv S \in \mathcal{D}$ and $Y_s = 0 \equiv S \in \mathcal{N}$. Additionally, let ν be the noise-to-data ratio, i.e. $\nu = {}^{|\mathcal{N}|}/_{|\mathcal{D}|}$.
The goal is to estimate the posterior probabilities $P(Y_s = 1 \mid S; \theta)$ and $P(Y_s = 0 \mid S; \theta)$, in order to discriminate noise from data samples. These probabilities are given by equations 3.9 and 3.10.

$$P(Y_s = 1 \mid S; \theta) = \frac{\hat{P}_d(S; \theta)}{\hat{P}_d(S; \theta) + \nu P_n(S)} \tag{3.9}$$

$$P(Y_s = 0 \mid S; \theta) = \frac{\nu P_n(S)}{\hat{P}_d(S; \theta) + \nu P_n(S)} \tag{3.10}$$

It is worth noting that a estimated distribution $\hat{P}_d$ is used instead of P_d, because the real density is not known. As indicated by Gutmann and Hyvärinen (2012), $\hat{P}_d$ can be an

unnormalized distribution for NCE where the partition function Z is included in the set of parameters θ as $\hat{Z}$, hence resulting in an approximately normalized distribution after the optimization.
Estimating the posterior probabilities is equivalent to maximizing the conditional log-likelihood objective in 3.11 (Gutmann and Hyvärinen, 2012).

$$g(\theta) = \sum_{S \in \mathcal{D}} \log P(Y_s = 1 \mid S; \theta) + \sum_{S \in \mathcal{N}} \log P(Y_s = 0 \mid S; \theta) \tag{3.11}$$

For the models previously introduced, the set of parameters θ to adjust in order to maximize the objective are:

- FLID (Tschiatschek et al., 2016): $\theta = [\hat{Z}, \mathbf{u}, \mathbf{W}^b]$.
- FLDC: $\theta = [\hat{Z}, \mathbf{u}, \mathbf{W}^b, \mathbf{W}^e]$.
- FFLDC: $\theta = [\hat{Z}, \mathbf{a}, \mathbf{B}, \mathbf{E}]$.

Lastly, a couple of conditions must be taken into consideration for NCE to work (Gutmann and Hyvärinen, 2012):

1. The parameterized probability function $\hat{P}_d(S; \theta)$ must be of the same family as the real distribution P_d, i.e. $\exists \theta^* \mid \hat{P}_d(S; \theta^*) = P_d$.
2. The noise distribution P_n is nonzero whenever P_d is nonzero.

Stochastic Gradient Descent (SGD) will be used for the training of the proposed models, this is the same method used by Tschiatschek et al. (2016) for the FLID model. SGD is a gradient-based method that has proven effective in large-scale learning tasks due to its efficiency when the computation time is a limiting factor (Bottou, 2010; Zhang, 2004), hence making it appropriate for the scale of data sourced from the internet, such as public user photos in Flickr.
In each iteration, the sub-gradient $\nabla \log P(Y_s = y \mid S; \theta)$ must be computed. The following sections define the sub-gradient for each model.

FLID

For FLID, $\nabla \log P(Y_s = y \mid S; \theta)$ is given by the following equations (Tschiatschek et al., 2016):

$$\nabla \log P(Y_s = y \mid S; \theta) = \left(y - \frac{1}{1 + \nu \frac{P_n(S)}{\hat{P}_d(S;\theta)}} \right) \nabla \log \hat{P}_d(S; \theta) \tag{3.12}$$

$$\hat{P}_d(S; \theta) = \frac{1}{\hat{Z}} P(S; \mathbf{u}, \mathbf{W}^b) \tag{3.13}$$

$$\left(\nabla_{\mathbf{u}} \log \hat{P}_d(S; \hat{Z}, \mathbf{u}, \mathbf{W}^b) \right)_i = \begin{cases} 1 & \text{if } i \in S \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

$$\left(\nabla_{\mathbf{W}^b} \log \hat{P}_d(S; \hat{Z}, \mathbf{u}, \mathbf{W}^b) \right)_{i,d} = \begin{cases} -1 & \text{if } i \in S \text{ and } i \neq \operatorname{argmax}_{j \in S} w^b_{j,d} \\ 0 & \text{otherwise} \end{cases} \tag{3.15}$$

$$\nabla_{\hat{Z}} \log \hat{P}_d(S; \hat{Z}, \mathbf{u}, \mathbf{W}^b) = \frac{-1}{\hat{Z}} \tag{3.16}$$

Where $P(S; \mathbf{u}, \mathbf{W}^b)$ is the unnormalized FLID equation, $(\nabla_{\mathbf{u}} \cdot)_i$ represents the i-th entry of the sub-gradient with respect to $\mathbf{u}$ and $(\nabla_{\mathbf{W}^b} \cdot)_{i,d}$ represents the (i,d)-th entry of the sub-gradient with respect to $\mathbf{W}^b$.

FLDC

For FLDC, the sub-gradient needs to be expanded to include the coherence term, while the other terms remain the same. This is given by:

$$\left(\nabla_{\mathbf{W}^e} \log \hat{P}_d(S; \hat{Z}, \mathbf{u}, \mathbf{W}^e, \mathbf{W}^e)\right)_{i,c} = \begin{cases} 1 & \text{if } i \in S \text{ and } i \neq \text{argmax}_{j \in S}\, w^e_{j,c} \\ 0 & \text{otherwise} \end{cases} \tag{3.17}$$

Where $(\nabla_{\mathbf{W}^e \cdot})_{i,c}$ represents the (i,c)-th entry in the sub-gradient with respect to $\mathbf{W}^e$.

FFLDC

For FFLDC, the sub-gradient must be defined for $\mathbf{a}, \mathbf{B}$ and $\mathbf{E}$ instead of the utility vector and $\mathbf{W}$ matrices. These are given by:

$$\left(\nabla_{\mathbf{a}} \log \hat{P}_d(S; \hat{Z}, \mathbf{a}, \mathbf{B}, \mathbf{E})\right)_m = \sum_{i \in S} x_{i,m} \tag{3.18}$$

$$\left(\nabla_{\mathbf{B}} \log \hat{P}_d(S; \hat{Z}, \mathbf{a}, \mathbf{B}, \mathbf{E})\right)_{m,d} = x_{i^d,m} - \sum_{i \in S} x_{i,m} \tag{3.19}$$

$$\left(\nabla_{\mathbf{E}} \log \hat{P}_d(S; \hat{Z}, \mathbf{a}, \mathbf{B}, \mathbf{E})\right)_{m,c} = x_{i^e,m} - \sum_{i \in S} x_{i,m} \tag{3.20}$$

Where $i^d = \text{argmax}_{i \in S} \sum_{k=1}^{M} x_{i,k} b_{k,d}$ and $i^c = \text{argmax}_{i \in S} \sum_{k=1}^{M} x_{i,k} e_{k,c}$. $(\nabla_{\mathbf{a}} \cdot)_m$ represents the m-th entry of the sub-gradient with respect to a, $(\nabla_{\mathbf{B}} \cdot)_{m,d}$ represents the (m,d)-th entry of the sub-gradient with respect to B and $(\nabla_{\mathbf{E}} \cdot)_{m,c}$ represents the (m,c)-th entry of the sub-gradient with respect to E.

Remark. For $\mathbf{X} = \mathcal{I}$, the sub-gradients in 3.18-3.20 are equivalent to the FLDC sub-gradients. Because $x_{i,m} = 1$ only when $i = m$.

3.4.2 Computational Complexity

As described by Tschiatschek et al. (2016), the running time requirement for calculating the sub-gradient for FLID is $O(L|S|)$, and for a complete pass over data and noise samples is then $O(|\mathcal{D} \cup \mathcal{N}|\kappa L)$ where $\kappa = \max_{S \in \mathcal{D} \cup \mathcal{N}} |S|$.
Extending this for FLDC is straightforward, the calculation of the additional sub-gradient component in equation 3.17 is $O(L|S|)$. Therefore, the running time for a single pass is $O(|\mathcal{D} \cup \mathcal{N}|\kappa(L + K))$.
For FFLDC, the sub-gradients in equations 3.19,3.20 require the calculation of i^d and i^e which takes $O(M|S|)$ for each L and K dimension, respectively. Hence the complexity of each sub-gradient computation is $O(M(L + K)|S|)$. Overall the computation of one pass over the samples requires $O(|\mathcal{D} \cup \mathcal{N}|M(L + K)\kappa)$ time.

3.4.3 Example datasets

This section presents the results obtained after learning the datasets introduced in the examples for each model. In every instance 1000 samples were generated from the specific distribution, along with 2000 noise samples ($\nu = 2$). This shows the ability of NCE to learn models that incorporate diversity and coherence between items and features.

FLID: Two landmarks

Firstly, recall the following characteristics about the dataset in the example:

- There are 3 items, h, s, f.

Table 3.4: Learned distribution for example 3.3.1

S	$P(S)$
$\{h, s\}$	0.48
$\{h, f\}$	0.47
$\{h\}$	0.03
$\{h, s, f\}$	0.02

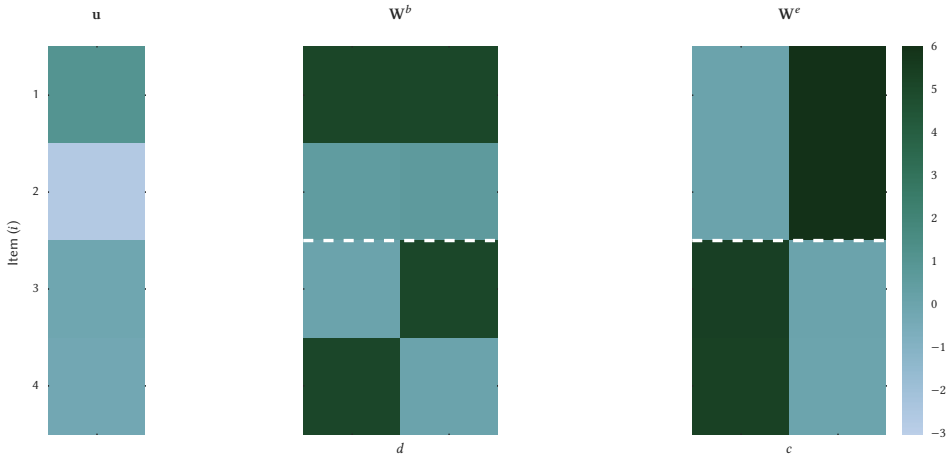

Figure 3.3: Learned model for example 3.2.1. The white line divides the disjoint pairs described in the example.

- There are only 2 possible sets, $\{h, s\}$ and $\{h, f\}$. Both equally likely, i.e. $P(S) = 0.5$.
- The model can be realized with a single diversity dimension, i.e. $L = 1$.

The resulting utility vector and weight matrix are:

$$\mathbf{u} = (5.42, 2.82, 2.79)^\intercal$$
$$\mathbf{W}^b = (0.02, 6.10, 6.10)^\intercal$$

This is a slightly different model from the one suggested in section 3.1.2, however the normalized probability distribution for this model shown in table 3.4 proves that it closely approximates the distribution.

FLDC: Disjoint pairs

Firstly, recall the following characteristics about the dataset in the example:

- There are 4 items, i.e. $V = \{1, 2, 3, 4\}$.
- There are only 2 possible sets, $\{1, 2\}$ and $\{3, 4\}$. Both equally likely, i.e. $P(S) = 0.5$.
- The model can be realized with a 2 diversity and 2 coherence dimensions.

Figure 3.3 shows the resulting utility vector $\mathbf{u}$ and weight matrices $\mathbf{W}^b, \mathbf{W}^e$ after learning. The model shown in the figure displays the desired properties of diversity between the disjoint pairs whilist having coherence between the items in each pair. Despite being different from the proposed model in section 3.2.1 it approximately realizes the desired distribution.

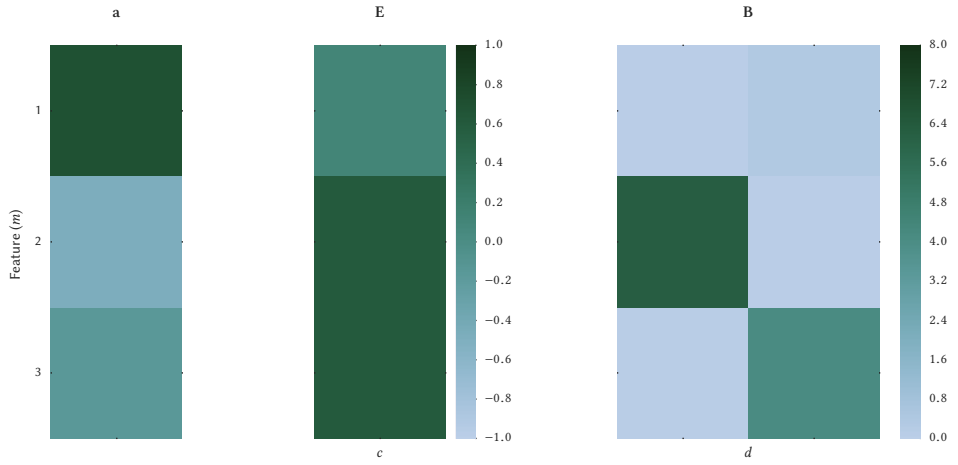

Figure 3.4: Learned model for example 3.3.1

FFLDC: Rated locations

Firstly, recall the following characteristics about the dataset in the example:

- There are 6 items, i.e. $V = \{1, 2, 3, 4, 5, 6\}$.
- The full distribution is related to the features and is given in table 3.3.
- The proposed model has 2 diversity dimensions and one coherence dimension.

Figure 3.4 shows the resulting model after learning. Unfortunately, this model can be not as easily interpreted as the one proposed in figure 3.2, nevertheless it accurately models the distribution from the example. This result demonstrates the ability of the learning algorithm to represent a distribution using features.
It is important to note that the presented results for these experiments correspond to the best of multiple runs. It was observed that the learning algorithm does not always arrive to a good solution. This is explained by the fact that $g(\theta)$ is not a convex function which means that there exists the risk of reaching a local minima in some cases. Due to this, most of the experiments in the following sections are executed over several data folds to account for the randomness.

3.4.4 Adagrad

This section will introduce a modification to the SGD method which was found to be useful when dealing with real data and featurized models.
A commonly used learning rate function for SGD is $\eta(t) = \eta_0 t^{-p}$, which offers the best convergence speed in theory and also does it often in practice (Bottou, 2012), however it can also lead to poor performance due to slow rates of convergence to the solution (Darken and Moody, 1992). On the other hand choosing a larger η_0 may not lead to better results due to the instability in the parameters for small t (Darken and Moody, 1990).
This behavior was observed during the experiments with real data to be presented in later sections, therefore an alternative for the learning rate was sought. In particular, Adaptive Gradient (AdaGrad) was implemented.
AdaGrad was proposed by Duchi et al. (2011), the idea of this method is to adapt the learning rate for each parameter in θ individually in a way that frequently updated parameters have slower learning rates while infrequent parameters have larger ones. The intuition

is that an update to a rare parameter is more informative than one to a parameter that is frequently updated.
Concretely, with AdaGrad the update step for each parameter in θ is given by:

$$\theta_j^{\tau+1} \leftarrow \theta_j^{\tau} - \frac{\eta}{\sqrt{G_{j,j}}} \nabla g(\theta)_j^{\tau} \tag{3.21}$$

Where $G_{j,j}$ contains the information about past gradients for parameter j and is given by:

$$G_{j,j} = \sum_{t=1}^{\tau} \left(\nabla g(\theta)_j^t \right)^2 \tag{3.22}$$

In equation 3.22 $\nabla g(\theta)_j^t$ denotes the j-th entry of the gradient at time step t.
It is worth noting that AdaGrad does not incur in an significant increase in running time or memory requirements for the training which makes it inexpensive to include in the implementation of NCE.

3.4.5 Learning rate sensitivity

This section expands on the discussion about the learning rate and explores its effect on the learning of the synthetic data from the FFLDC example in section 3.3.1, as well as the effect of implementing AdaGrad.
The first experiment considers the stability of the optimization without AdaGrad. In the experiments presented in this section the following configuration of the learning algorithm is used:

- 1000 data samples from the distribution in table 3.3 are used, along with 2000 noise samples, i.e. $\nu = 2$.
- Each run does 100 passes over the data and noise samples.
- The algorithm is executed 8 times to obtain statistics about the results.

Figure 3.5 shows the behavior of $g(\theta)$ during learning, each point corresponds to the average value of $g(\theta)$ after each pass over the data and noise samples with its corresponding 95% confidence interval.
The results in figure 3.5 display what was mentioned in section 3.4.4, they show that for small values of η_0 the learning can be significantly slow and that large values of η_0 may lead to instability. Concretely, the first row displays slow but stable learning rates while the last row displays learning rates that are too high, the middle row displays the ideal learning rates. Fortunately, in the particular case of this synthetic dataset there exists a learning rate for which the objective function converges however, as it will be seen on real data, this is not always the case. An interesting characteristic shown in this figure is the behavior of the confidence intervals for $\eta_0 = 0.005$, even though the learning is considerably slow there is also a large variation, in this instance the initialization has a larger influence because the learned model doesn't change much from it.
In contrast, figure 3.6 shows the objective function for various values of η_0 after incorporating AdaGrad to the implementations. These results show that training with AdaGrad is stable for a larger range of eta_0, although slower for small values of η_0 which were good for the learning without AdaGrad. For large enough η_0 values, it can be clearly seen from the figure that SGD with AdaGrad reaches convergence significantly faster, often in the first few iterations.
Finally, figure 3.7 shows a direct comparison between training with and without AdaGrad using the best values of η_0 for each configuration. In this figure both algorithms converge to an optimal value, however it also shows that training with AdaGrad converges significantly faster and its average appears slightly more stable.
These combined results indicate that using AdaGrad is the best strategy for learning the models.

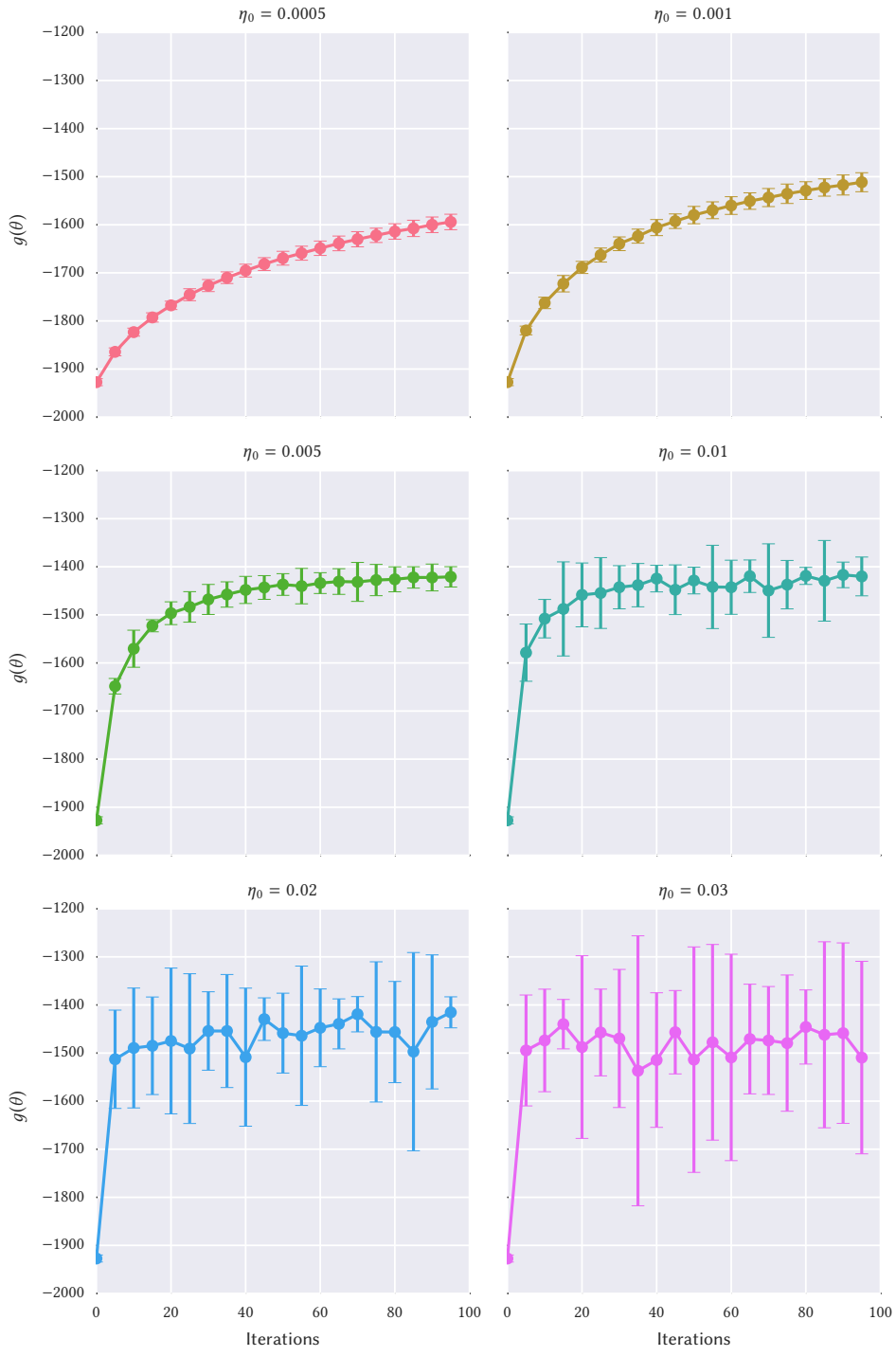

Figure 3.5: Objective function during learning for various values of η_0 without AdaGrad.

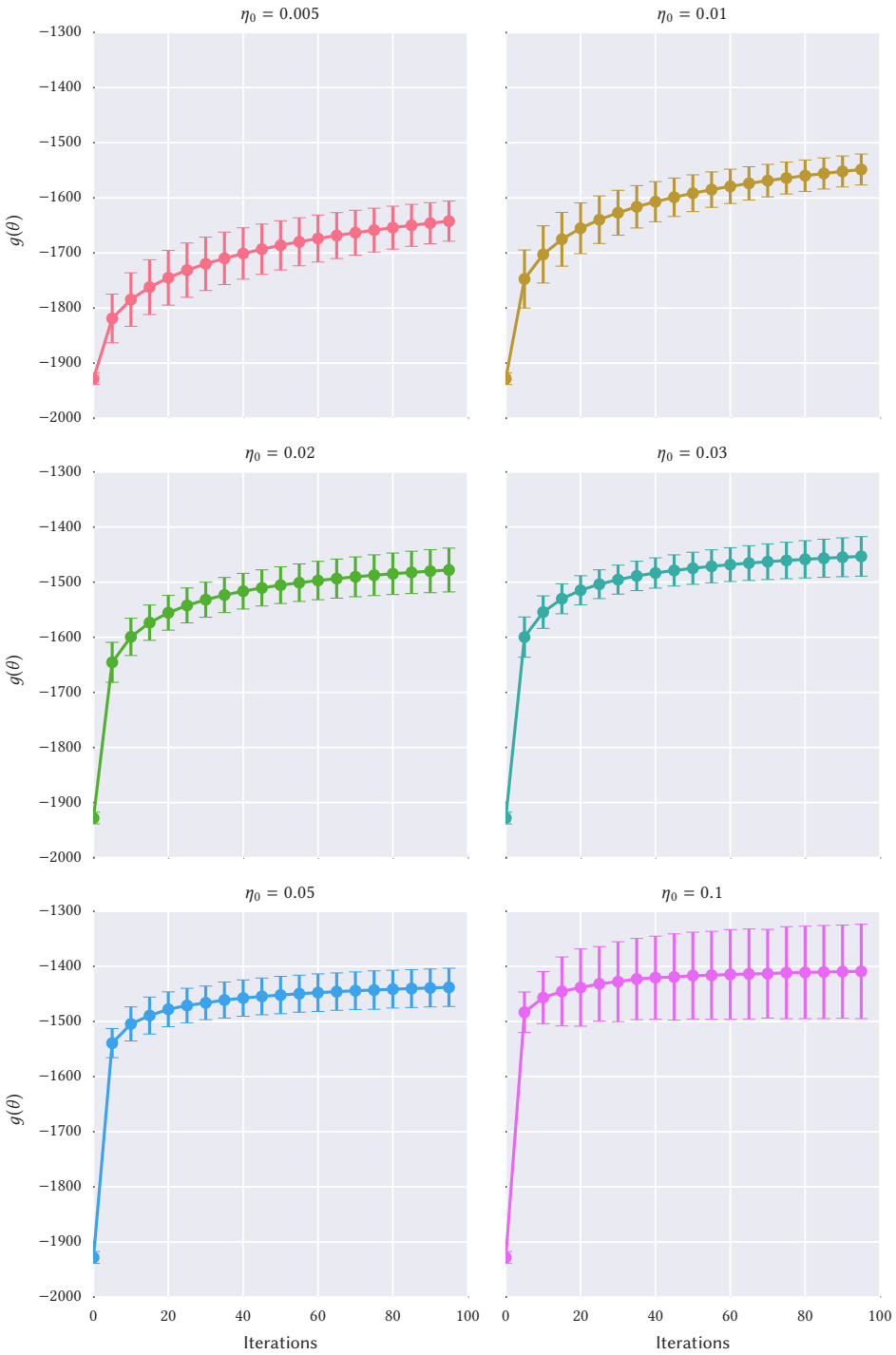

Figure 3.6: Objective function during learning for various values of η_0 with AdaGrad.

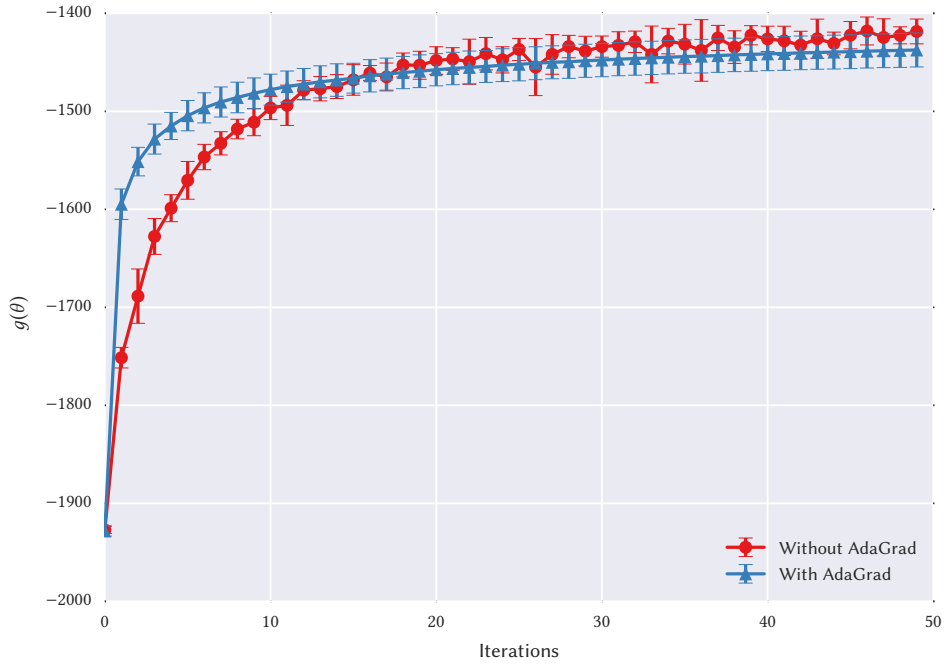

Figure 3.7: Comparison of learning performance with and without AdaGrad. Initial learning rates are $\eta_0 = 0.05$ with AdaGrad and $\eta_0 = 0.005$ without AdaGrad.

3.4.6 Noise-to-data ratio

One final consideration about NCE is how to choose the noise, Gutmann and Hyvärinen (2012) suggest the following about the noise distribution:

- A distribution that can be sampled easily.
- Noise that is as similar as possible to the data, otherwise the classification problem could be too easy.
- A noise sample as large as possible.

Following the first two suggestions, the noise distribution is sampled from a modular model as the one from equation 3.1. This distribution is constructed from estimated marginals, i.e. $\hat{P}(i \in S)$, which are computed using the maximum likelihood estimator, i.e. $\hat{P}(i \in S) = {}^{\#N(i \in S)}/_{|\mathcal{D}|}$. Then the utilities for the modular model are given by:

$$u_i = \log\left(\frac{1}{\hat{P}(i \in S)} - 1\right) \tag{3.23}$$

This model is easy to sample from, hence allowing an efficient generation of large noise samples.

Chapter 4

Experimental Setup

4.1 Flickr Dataset

Dataset description and how it was collected.

4.2 Path finding

How the actual sets (paths) are built from the data, and the 2 sets of data to be explore 10 items (small) and 100 items (big).

4.3 NCE learning

Details on the implementation of NCE, noise generation.

4.4 Baselines

The baseline models (Markov, distance).

4.5 Evaluation

10 fold evaluation, accuracy and MRR.

Chapter 5

Results

5.1 Small dataset

Results on the small dataset. Difference between FLID, FLDC, FFLDC models. Different feature sets.

5.2 Large dataset

Results on the large dataset. Difference between FLID, FLDC, FFLDC models. Different feature sets.

Chapter 6

Conclusion

Bibliography

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th International Conference on Computational Statistics*, pages 177–186.

Bottou, L. (2012). Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*, pages 421–436. Springer.

Darken, C. and Moody, J. (1990). Note on learning rate schedules for stochastic optimization. In *Neural Information Processing Systems*, pages 832–838.

Darken, C. and Moody, J. (1992). Towards faster stochastic gradient search. In *Neural Information Processing Systems*, pages 1009–1016.

Djolonga, J. and Krause, A. (2014). From MAP to marginals: Variational inference in bayesian submodular models. In *Neural Information Processing Systems (NIPS)*.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159.

Gutmann, M. U. and Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13:307–361.

Jerrum, M. and Sinclair, A. (1990). Polynomial-time approximation algorithms for the ising model. In *Automata, Languages and Programming*, pages 462–475. Springer Berlin Heidelberg.

Krause, A. and Golovin, D. (2014). Submodular function maximization. In *Tractability: Practical Approaches to Hard Problems*, chapter 3. Cambridge University Press.

Shahaf, D., Guestrin, C., and Horvitz, E. (2012). Trains of thought: Generating information maps. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 899–908, New York, NY, USA. ACM.

Tschiatschek, S., Djolonga, J., and Krause, A. (2016). Learning probabilistic submodular diversity models via noise contrastive estimation. In *Proc. International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Yan, R., Wan, X., Otterbacher, J., Kong, L., Li, X., and Zhang, Y. (2011). Evolutionary timeline summarization: A balanced optimization framework via iterative substitution. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 745–754, New York, NY, USA. ACM.

Zhang, T. (2004). Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the Twenty-first International Conference on Machine Learning*, pages 116–123, New York, NY, USA. ACM.