UNIVERSITY OF
CAMBRIDGE

# Relevance Feedback and Novelty Detection under the Bayesian Sets Framework


Joshua Thomas Abbott

Darwin College
University of Cambridge


Supervisors:
Prof. Zoubin Ghahramani
Dr. Stephen Clark

A thesis submitted for the degree of

*Master of Philosophy*
*Computer Speech, Text, and Internet Technology*

June 2010

I, *Joshua Thomas Abbott of Darwin College*, being a candidate for M.Phil in Computer Speech, Text, and Internet Technology, hereby declare that this dissertation and the work described in it are my own work, unaided except as may be specified below, and that the dissertation does not contain material that has already been used to any substantial extent for a comparable purpose.

Signed:

Date:

Number of Words:

# Abstract

The research presented in this thesis focuses on extending the Bayesian Sets Framework for other common methods utilised in information retrieval. In particular, we develop computationally efficient algorithms for novelty detection and relevance feedback. We test these algorithms against a baseline evaluation system comprising a Bayesian framework for content-based image retrieval. We show that our novelty dection algorithm successfully finds outliers in labelled training sets and our relevance feedback algorithm returns more relevant images than the baseline system while also retrieving more relevant images at higher ranks.

# Acknowledgements

I would like to thank my supervisors, Prof. Zoubin Ghahramani, for offering me the opportunity to extend the Bayesian Sets framework and his guidance throughout the project; and Dr. Stephen Clark, for sponsoring the project inside the CSTIT course and his helpful feedback.

I am also grateful for the support from the members of the Machine Learning group of the Cambridge Computational and Biological Learning Lab; in particular Postdoctoral Research Fellow Katherine Heller, and PhD candidates Jurgen Van Gael and Alex Ksikes.

Thanks also to the students, lecturers, and staff of the CSTIT course, particularly Student Administration Manager Lise Gough, for her kindness and help in making sure all adminstrative requirements were met with ease; and fellow coursemate William Montgomery, for his assistance in writing a parser for the Mechanical Turk output.

Finally, I extend the deepest gratitude to my family for their endless support through all these years.

# Contents

**1 Introduction** **1**

**2 Bayesian Sets** **5**

2.1 Background . . . . . 5

2.2 Bayesian Sets Algorithm . . . . . 6

2.3 Bayesian Sets on Sparse Binary Data . . . . . 8

2.4 Summary . . . . . 11

**3 Baseline Evaluation System** **12**

3.1 Bayesian Sets for CBIR . . . . . 12

3.2 Dataset . . . . . 15

3.2.1 Features . . . . . 15

3.3 Evaluation Measures . . . . . 16

3.3.1 Precision at Position $n$ . . . . . 16

3.3.2 Normalized Discount Cumulative Gain . . . . . 16

3.3.3 Amazon Mechanical Turk . . . . . 17

3.4 Results . . . . . 19

3.5 Discussion . . . . . 20

**4 Novelty Detection** **22**

4.1 Novelty Detection in Bayesian Sets . . . . . 22

4.2 Results . . . . . 23

4.2.1 BND on the Corel Training Sets . . . . . 23

4.2.2 BND within CBIR . . . . . 29

4.3 Discussion . . . . . 30

**5 Relevance Feedback** **32**
5.1 Relevance Feedback in Bayesian Sets . . . . . 32
5.2 Results . . . . . 38
5.3 Discussion . . . . . 42

**6 Summary and Future Work** **43**

**References** **44**

**A Bayesian Sets Derivations** **47**
A.1 Bayesian Sets for CBIR . . . . . 47
A.2 Derivations of Marginal Likelihoods . . . . . 48
A.2.1 Computing the marginal likelihood $p(\mathbf{x}^*)$ . . . . . 48
A.2.2 Computing the marginal likelihood $p(\mathcal{D}_q)$ . . . . . 49
A.2.3 Computing the marginal likelihood $p(\mathbf{x}^*, \mathcal{D}_q)$ . . . . . 50
A.3 Efficient Computation of score($\mathbf{x}^*$) . . . . . 51

**B Relevance Feedback Derivations** **54**
B.1 Relevance Feedback in Bayesian Sets . . . . . 54
B.2 Derivations of Marginal Likelihoods . . . . . 56
B.2.1 Computing the marginal likelihood $p(\mathbf{x}^{**})$ . . . . . 56
B.2.2 Computing the marginal likelihood $p(\mathbf{x}_i^*)$ . . . . . 56
B.2.3 Computing the marginal likelihood $p(\mathcal{D}_p)$ . . . . . 57
B.2.4 Computing the marginal likelihood $p(\mathbf{x}^{**}, \mathbf{x}_i^*)$ . . . . . 57
B.2.5 Computing the marginal likelihood $p(\mathbf{x}^{**}, \mathcal{D}_p)$ . . . . . 58
B.3 Efficient Computation of rescore($\mathbf{x}^{**}$) . . . . . 58

# List of Figures


1.1 Example Data Collection $\mathcal{D}$ . . . . . 2
1.2 Example query on $\mathcal{D}$ . . . . . 2
1.3 Example query on $\mathcal{D}$ . . . . . 2

2.1 Graphical Model for score($\mathbf{x}$) . . . . . 8

3.1 CBIR Results for Query: Sunset . . . . . 14
3.2 CBIR Results for Query: Desert . . . . . 14
3.3 MTurk Example . . . . . 18
3.4 CBIR Results for Query: Tower . . . . . 20

4.1 Top 9 images from Coast training set . . . . . 24
4.2 Bottom 9 images from Coast training set . . . . . 24
4.3 Top 9 images from Mountains training set . . . . . 25
4.4 Bottom 9 images from Mountains training set . . . . . 25
4.5 Top and Bottom Results Plot . . . . . 27
4.6 Bottom 9 images from Fireworks training set . . . . . 28
4.7 Bottom 9 images from Fireworks training set with outliers . . . . 28
4.8 BND scores plotted against no. of training examples for Coast . . 30
4.9 BND scores plotted against no. of training examples for Mountains 31

5.1 Graphical Model for rescore($\mathbf{x}^{**}$) . . . . . 33
5.2 Baseline Results for Query: Aerial . . . . . 39
5.3 RF Results for Query: Aerial . . . . . 39
5.4 Baseline Results for Query: Penguins . . . . . 40
5.5 RF Results for Query: Penguins . . . . . 40

5.6 Top 9 Results for 'cell' in Baseline and RF . . . . . 42

# List of Tables

3.1 CBIR Baseline Precision for 50 queries . . . . . 19
3.2 Baseline evaluation over 50 queries . . . . . 20

4.1 Average Precision of top 9 and bottom 9 sets over 50 queries . . . 26
4.2 Evaluation over 50 queries on Labels . . . . . 29
4.3 Evaluation over 50 queries on MTurk . . . . . 29

5.1 Baseline and RF results over 50 queries . . . . . 41
5.2 Evaluation of RF over 50 queries on MTurk . . . . . 41

# List of Algorithms


1 Bayesian Sets . . . . . 11
2 CBIR in Bayesian Sets . . . . . 13
3 Novelty Detection in Bayesian Sets . . . . . 23
4 Relevance Feedback in Bayesian Sets . . . . . 37

# Chapter 1

# Introduction

The research presented in this thesis extends an existing framework for information retrieval called Bayesian Sets (Ghahramani & Heller, 2005). In the past few decades, the growth of the internet has provided a vast resource of information available to people. However, with this growth comes a need for better tools capable of efficiently searching within a data collection for material that satisfies a user's information need (Manning *et al.*, 2008). Fundamentally, this is an inference problem - finding the user's intended target given the relatively small amount of data from the query input. In the application of content-based image retrieval, which we describe in later chapters, this problem is deemed the "semantic gap". As images may be composed of multiple objects and scenes, it becomes quite difficult to infer which objects play a central role in the composition and are considered relevant to the user (Howarth, 2007; Vasconcelos, 2007).

The Bayesian Sets approach to this inference problem considers the user's query as a set of items representing some concept or cluster in the data collection. The goal of the algorithm then is to search through the data collection for other items that would fit into this query set. Though most clustering algorithms are unsupervised, here the user's query set provides supervised hints as to the membership constraints of a particular cluster. To briefly illustrate the Bayesian Sets method of search, consider the following example queries on some data collection $\mathcal{D}$, as depicted in figure 1.1.


Figure 1.1: Example Data Collection $\mathcal{D}$

If we use two elements from $\mathcal{D}$ to partially represent some cluster or some "concept of sports car", the Bayesian Sets search algorithm attempts to complete this cluster and retrieves other sports cars from $\mathcal{D}$, as depicted in figure 1.2.

Figure 1.2: Example query on $\mathcal{D}$

If we use some different elements of $\mathcal{D}$ like a red sports car and a tomato, the algorithm should retrieve other elements of $\mathcal{D}$ in this cluster or concept of "things that are red" as illustrated in figure 1.3.

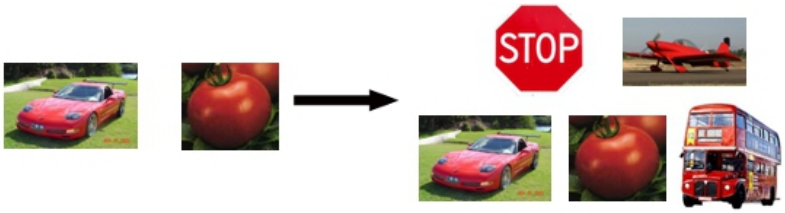


Figure 1.3: Example query on $\mathcal{D}$

The Bayesian Sets approach has been tested with success on numerous datasets, over various applications including searching through an image database for particular image categories (Heller & Ghahramani, 2006), performing analogical reasoning with relational data (Silva *et al.*, 2007), searching scientific literature for groupings of similar papers (Heller, 2008), and searching through the Internet Movie DataBase (Ksikes, 2010).

The main contributions of this thesis will be extending the Bayesian Sets framework to perform novelty detection and relevance feedback. Novelty detection is the identification of unknown or noisy data, generally used to clean a corpus used for training learning algorithms. We show how Bayesian Sets can be extended to find both prototypical and outlier elements of a categorized data collection. Relevance feedback is a process where the user performs an initial query and then informs the search system which of the results were relevant. The retrieval system uses these relevance markings to perform a revised search with a better focus on the user's target information need. We show how incorporating user-defined positive and negative examples increases the performance of Bayesian Sets in the task of content-based image retrieval.

The thesis proceeds as follows:

- In chapter 2, we introduce the motivation for Bayesian Sets and derive a computationally efficient ranking algorithm for information retrieval on sparse binary datasets.
- In chapter 3, we outline a Bayesian Sets system for use with content-based image retrieval (CBIR) which will be used as a baseline evaluation. We also describe the experimental setup for testing our algorithms, consisting of the dataset and evaluation measures we use.
- In chapter 4, we review the motivation for novelty detection and develop a system within the Bayesian Set framework to find outliers in sets of labelled training data. We test our system on sets of images from the Corel dataset and investigate the efficacy of incorporating novelty detection in the Bayesian CBIR system.

- In chapter 5, we review the concepts behind relevance feedback and develop a system within the Bayesian Set framework that incorporates user-defined positive and negative examples to increase the performance of an information retrieval task. We test our relevance feedback algorithm by incorporating it within the Bayesian CBIR system.

- Lastly, in chapter 6, we review the main contributions of this thesis, and discuss future work.

# Chapter 2

# Bayesian Sets

One of the primary issues in information retrieval is determining the user's intended target given the limited amount of data from the query input. This problem has an analogue in the cognitive sciences, namely that of human generalisation and inductive inference. In this chapter, we develop the ranking criterion used in Bayesian Sets by exploring a recent probabilistic framework for human concept learning on sets of novel stimuli. Based on this criterion, we then derive a computationally efficient algorithm for information retrieval on sparse binary datasets.

## 2.1 Background

Humans have an uncanny ability to infer structure and learn new concepts from a relatively small set of examples. The famed cognitive psychologist Roger Shepard proposed a universal law governing this generalisation from a single stimulus within a metric of psychological space (Shepard, 1987). For example, suppose a doctor can quantify the skin coloration of a mole with some arbitrary measure between 0 and 100 and has determined the pigmentation level of a mole on a healthy patient is 60. What other pigmentation levels of moles are considered healthy? Shepard formulated this problem of generalisation as follows: Given a single example $x$ from some consequential region $C$, if we encounter a new object $y$, how likely is it that $y$ is also a member of $C$? In a probabilistic interpretation, this question can be formalised as the expression $p(y \in C|x)$; the conditional

probability that $y$ is a member of $C$ given we have observed the example $x$ as a member of $C$.

More recently, a Bayesian framework has been proposed extending Shepard's formulation for sets of multiple stimuli (Tenenbaum & Griffiths, 2001). For example, if our doctor above saw 3 healthy patients with mole pigmentation levels of 60, 65, and 67, how should the doctor generalise other healthy pigmentation levels? This situation is a more natural occurence in scenarios of human generalisation and inference. If we assume a person's knowledge or hypothesis $h$ about the consequential region $C$ is represented as a probability distribution $p(h)$ over a hypothesis space of possible consequential regions $h \in H$, we can compute the posterior probability of $h$ after observing stimuli $\mathbf{x}$ using Bayes rule:

$$\mathrm{p}(h|\mathbf{x}) = \frac{p(\mathbf{x}|h)p(h)}{p(\mathbf{x})} \tag{2.1}$$

where $p(h|\mathbf{x})$ represents how a person updates their belief in hypothesis $h$ after observing $\mathbf{x}$, and $p(\mathbf{x}|h)$ is the *likelihood* of observing $\mathbf{x}$ assuming $h$ is the true consequential region. Motivated by this work, we formulate the inference problem in information retrieval in a similar fashion as an induction problem on sets of observed examples.

## 2.2 Bayesian Sets Algorithm

Given a data collection $\mathcal{D}$, and a subset of items $\mathcal{D}_q = \{\mathbf{x}_i\} \subset \mathcal{D}$ representing a query concept $Q$, we wish to find other elements in $\mathcal{D}$ that are similar to the concept in question. Thus for each $\mathbf{x} \in \mathcal{D}$ we compute a score that represents the likelihood of $\mathbf{x}$ fitting in $Q$. If we use a model-based probabilistic criterion to measure this, we compute $p(\mathbf{x} \in Q|\mathcal{D}_q)$; having observed $\mathcal{D}_q$ belonging to a concept $Q$, how probable is it that $\mathbf{x}$ also belongs to $Q$. However, since some items $\mathbf{x} \in \mathcal{D}$ might be more probable than other items *a priori*, we normalize this computation by the prior probability of $\mathbf{x}$. One can think of this as similar to the *inverse document frequency* (IDF) component of TF-IDF(Jones, 1993).

Thus the scoring criterion we must compute is:

$$\text{score}(\mathbf{x}) = \frac{p(\mathbf{x} \in Q|\mathcal{D}_q)}{p(\mathbf{x})} \tag{2.2}$$

Using Bayes rule, we can rewrite this score as:

$$\text{score}(\mathbf{x}) = \frac{p(\mathbf{x}, \mathcal{D}_q)}{p(\mathbf{x})p(\mathcal{D}_q)} \tag{2.3}$$

A natural model-based way of defining a concept or cluster is to assume the data points in the cluster all come i.i.d. from some simple parameterized statistical model. We assume the parameterized model is $p(\mathbf{x}|\theta)$, where $\theta$ are the parameters. If the datapoints in $\mathcal{D}_q$ all belong to one cluster, then under this definition they were generated from the same setting of parameters; however, that setting is unknown, so we need to average over possible parameter values weighted by some prior density on parameter values, $p(\theta)$. Each of the three terms in equation 2.3 above are marginal likelihoods and, under these assumptions, can be written as integrals of the following form:

$$p(\mathbf{x}) = \int p(\mathbf{x}|\theta)p(\theta)d\theta \tag{2.4}$$

where $\theta$ are the parameters of some distribution which has been chosen to model the item feature vectors, $p(\theta)$ is the prior over these parameters, and $p(\mathbf{x}|\theta)$ is the likelihood; the probability of observing $\mathbf{x}$ given that our model is parameterized by $\theta$. Integrating over $\theta$ as above corresponds to computing the prior probability of observing $\mathbf{x}$ by averaging over all possible settings of the model parameters. The prior probability of observing the given query set is computed similarly:

$$p(\mathcal{D}_q) = \int \left[\prod_{i=1}^{N} p(\mathbf{x}_i|\theta)\right] p(\theta)d\theta \tag{2.5}$$

The bracketed product means every item $\mathbf{x}_i$ in the query set is assumed to be drawn i.i.d. from our model with unknown, but the *same* parameters $\theta$. Lastly, the numerator of 2.3, representing the joint probability of observing $\mathbf{x}$ *and* $\mathcal{D}_q$, can be computed as:

$$p(\mathbf{x}, \mathcal{D}_q) = \int \left[\prod_{i=1}^{N} p(\mathbf{x}_i|\theta)\right] p(\mathbf{x}|\theta)p(\theta)d\theta \tag{2.6}$$

Here we are assuming every item in the query set *and* the item to be scored, $\mathbf{x}$, all come i.i.d. from our model with unknown, but the same parameters, $\theta$. Thus, given these marginal likelihoods, we can interpret the score, equation 2.3, as the ratio of the probability that $\mathcal{D}_q$ and $\mathbf{x}$ belong to the *same* model with the same, though unknown, parameters $\theta$ (hypothesis $H_0$), to the probability that $\mathcal{D}_q$ and $\mathbf{x}$ belong to models with *different* parameters, $\theta$ and $\theta'$ (hypothesis $H_1$). This has a nice intuitive graphical representation:

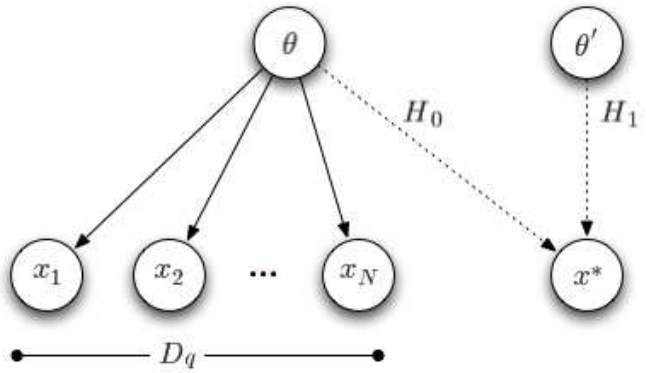


Figure 2.1: Graphical Model for score($\mathbf{x}$)

## 2.3 Bayesian Sets on Sparse Binary Data

Computing the integrals defined above for the score function is, in general, computationally expensive. However, if we assume the data collection we are searching is sparse binary, we can compute the exact score very efficiently. Assuming each item $\mathbf{x}_i \in \mathcal{D}_q$ is represented as a binary vector $\mathbf{x}_i = (x_{i1}, \ldots, x_{iJ})$ and $x_{ij} \in \{0, 1\}$, we define a model in which each element of $\mathbf{x}_i$ has an independent Bernoulli distribution:

$$p(\mathbf{x}_i|\theta) = \prod_{j=1}^{J} \theta_j^{x_{ij}} (1 - \theta_j)^{1-x_{ij}} \tag{2.7}$$

The conjugate prior for the parameters of a Bernoulli distribution is the Beta distribution:

$$p(\theta|\alpha,\beta) = \prod_{j=1}^{J} \frac{1}{Z(\alpha_j,\beta_j)}\ \theta_j^{\alpha_j-1}\,(1-\theta_j)^{\beta_j-1} \tag{2.8}$$

where $\alpha$ and $\beta$ are hyperparameters of the prior, and $Z(\cdot)$ is a normalization constant such that

$$\begin{aligned} Z(\alpha,\beta) &= \int_0^1 \theta^{\alpha-1}\,(1-\theta)^{\beta-1} d\theta \\ &= \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} \end{aligned}$$

where the Gamma function, $\Gamma(\cdot)$, is a generalization of the factorial function. The hyperparameters $\alpha$ and $\beta$ are set empirically from the data, $\alpha = \kappa\mathbf{m}$, $\beta = \kappa(\mathbf{1}-\mathbf{m})$, where $\mathbf{m}$ is the mean of $\mathbf{x}$ over all items in the dataset $\mathcal{D}$, and $\kappa$ is a scaling factor representing how many times we have observed the dataset before retrieval; a measure of confidence in the priors.

With our statistical model defined, we can now derive an efficient method of computing score($\mathbf{x}$). For a query $\mathcal{D}_q = \{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$, consisting of $N$ vectors, we can compute its prior as such:

$$\begin{aligned} p(\mathcal{D}_q|\alpha,\beta) &= \int \left[\prod_{i=1}^{N} p(\mathbf{x}_i|\theta)\right] p(\theta|\alpha,\beta)d\theta \\ &= \int \left[\prod_{i=1}^{N}\prod_{j=1}^{J} \theta_j^{x_{ij}}(1-\theta_j)^{1-x_{ij}}\right] \prod_{j=1}^{J} \frac{1}{Z(\alpha_j,\beta_j)}\ \theta_j^{\alpha_j-1}\,(1-\theta_j)^{\beta_j-1} d\theta \\ &= \prod_j \frac{\Gamma(\alpha_j\ +\ \beta_j)}{\Gamma(\alpha_j)\,\Gamma(\beta_j)}\ \frac{\Gamma(\tilde{\alpha}_j)\Gamma(\tilde{\beta}_j)}{\Gamma(\tilde{\alpha}_j+\tilde{\beta}_j)} \end{aligned}$$

where $\tilde{\alpha} = \alpha + \sum_{i=1}^{N} x_{ij}$ and $\tilde{\beta} = \beta + N - \sum_{i=1}^{N} x_{ij}$.

It is not difficult to compute the rest of the marginal likelihoods in the scoring criterion of equation 2.3, and the reader is advised to review Appendix A for such derivations. By expanding and combining the marginal likelihoods, our

scoring criterion becomes:

$$\begin{aligned}\text{score}(\mathbf{x}) &= \frac{p(\mathbf{x}, \mathcal{D}_q)}{p(\mathbf{x})p(\mathcal{D}_q)} \\ &= \prod_j \frac{\frac{\Gamma(\alpha_j+\beta_j+N)}{\Gamma(\alpha_j+\beta_j+N+1)} \frac{\Gamma(\tilde{\alpha}_j+x_{\cdot j})\Gamma(\tilde{\beta}_j+1-x_{\cdot j})}{\Gamma(\tilde{\alpha}_j)\Gamma(\tilde{\beta}_j)}}{\frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j+\beta_j+1)} \frac{\Gamma(\alpha_j+x_{\cdot j})\Gamma(\beta_j+1-x_{\cdot j})}{\Gamma(\alpha_j)\Gamma(\beta_j)}}\end{aligned} \tag{2.9}$$

We can simplify this expression by using the fact that $\Gamma(x) = (x-1)\,\Gamma(x-1)$ for $x > 1$, and further, for each $j$, we can consider the two cases $x_{\cdot j} = 0$ and $x_{\cdot j} = 1$ separately. If $x_{\cdot j} = 0$, then

$$\text{score}(\mathbf{x}^*) = \prod_j \left(\frac{\alpha_j + \beta_j}{\alpha_j + \beta_j + N}\right) \left(\frac{\tilde{\beta}_j}{\beta_j}\right)$$

and if $x_{\cdot j} = 1$, then

$$\text{score}(\mathbf{x}^*) = \prod_j \left(\frac{\alpha_j + \beta_j}{\alpha_j + \beta_j + N}\right) \left(\frac{\tilde{\alpha}_j}{\alpha_j}\right)$$

Putting these two cases together, we get

$$\text{score}(\mathbf{x}^*) = \prod_j \frac{\alpha_j + \beta_j}{\alpha_j + \beta_j + N} \left(\frac{\tilde{\alpha}_j}{\alpha_j}\right)^{x_{\cdot j}} \left(\frac{\tilde{\beta}_j}{\beta_j}\right)^{1-x_{\cdot j}}$$

and taking the log of this score we find it reduces to simply:

$$\log \text{score}(\mathbf{x}) = c + \sum_j q_j x_{\cdot j} \tag{2.10}$$

where

$$c = \sum_j \log(\alpha_j + \beta_j) - \log(\alpha_j + \beta_j + N) + \log \tilde{\beta}_j - \log \beta_j$$

and

$$q_j = \log \tilde{\alpha}_j - \log \alpha_j - \log \tilde{\beta}_j + \log \beta_j$$

This end result is *linear* in $\mathbf{x}$ and if we put the data collection $\mathcal{D}$ into a matrix $\mathbf{X}$ with $J$ columns, we can compute the vector $\mathbf{s}$ of log scores for all items using a single matrix-vector multiplication

$$\mathbf{s} = c + \mathbf{X}\mathbf{q} \tag{2.11}$$

which is very efficient to compute for sparse binary data.

## 2.4 Summary

A general summary of the Bayesian Sets algorithm is given in the following pseudocode:

**Algorithm 1** Bayesian Sets

**background:** a set of items $\mathcal{D}$, a probabilistic model $p(\mathbf{x}|\theta)$ where $\mathbf{x} \in D$, a prior on the model parameters $p(\theta)$
**input:** a query, $\mathcal{D}_q = \{\mathbf{x}_i\} \subset D$
**for all** $\mathbf{x} \in \mathcal{D}$ **do**
  compute $\text{score}(\mathbf{x}) = \dfrac{p(\mathbf{x}, \mathcal{D}_q)}{p(\mathbf{x})p(\mathcal{D}_q)}$
**end for**
**output:** sorted list of top scoring items in $D$

As previously noted in the Introduction, Bayesian Sets have been used in a variety of domains with good results. In the next chapter, we describe the application of Bayesian Sets for content-based image retrieval (CBIR) as it will be the basis for ideas used in the remainder of the thesis.

# Chapter 3

# Baseline Evaluation System

In this chapter we outline a Bayesian Sets system for use with content-based image retrieval which will become our baseline evaluation. We also describe the experimental setup for testing our algorithms, consisting of the dataset and evaluation measures we will use.

## 3.1 Bayesian Sets for CBIR

This thesis explores extensions of Bayesian Sets through the domain of content-based image retrieval (CBIR). To ease the process of development, we use a previously developed system for CBIR under the Bayesian Sets framework to create our gold standard for evaluation (Heller & Ghahramani, 2006). The data collection $\mathcal{D}$ that we will be using is the the Corel image database of 31,992 images (Corel, 2010). To create the query sets, a random selection of 10,000 images were set aside as labelled training data. Thus when a user performs search on a particular concept, the algorithm first returns all images from the training data with labels that correspond to the concept and use this as the query set, denoted as $\mathcal{D}_q = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. Then, using binary feature vectors which represent the images, the algorithm iterates through each of the 21,992 images in the test set, denoted $\mathbf{x}^*$, and computes the following score:

$$\text{score}(\mathbf{x}^*) = \frac{p(\mathbf{x}^*, \mathcal{D}_q)}{p(\mathbf{x}^*)p(\mathcal{D}_q)} \tag{3.1}$$

A general summary of the Bayesian Sets CBIR algorithm is given in the following pseudocode:

**Algorithm 2** CBIR in Bayesian Sets

**background:** a set of labelled images $\mathcal{D}_\ell$, a set of unlabelled images $\mathcal{D}_u$, a probabilistic model $p(\mathbf{x}|\theta)$ defined on binary feature vectors representing images, a prior on the model parameters $p(\theta)$

**preprocess:** compute texture and color features for each image, binarize feature vectors across images

**input:** a text query, $q$

find images corresponding to $q$, $\mathcal{D}_q = \{\mathbf{x}_i\} \subset \mathcal{D}_\ell$

**for all** $\mathbf{x}^* \in \mathcal{D}_u$ **do**

compute $\text{score}(\mathbf{x}^*) = \dfrac{p(\mathbf{x}^*, \mathcal{D}_q)}{p(\mathbf{x}^*)p(\mathcal{D}_q)}$

**end for**

**output:** sorted list of top scoring images in $\mathcal{D}_u$

Example images retrieved from this algorithm used on the Corel image database are shown in figures 3.1 and 3.2. These examples show how the algorithm does reasonably well, but can give a bias to certain features of a query set. For example, the results for the query: sunset, shown in figure 3.1, appear to have a focused region of very high contrast (the sun) on top of a dark horizontal region (the horizon). Similarly, the results for the query: desert, shown in figure 3.2, have very similar color histograms, with blue sky in the top half of the image and a tan region spanning the bottom half of the image. While this does seem indicative of the images relevant to the concept of desert, it is clear that there are instances of non-desert concepts that also have this distinctive color segmentation (ie: the mountain lion).

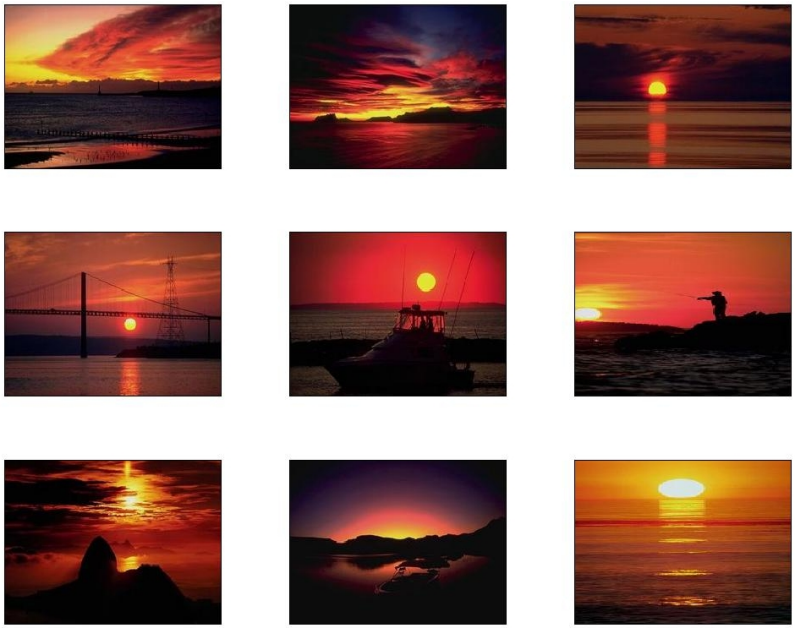

Figure 3.1: CBIR Results for Query: Sunset

Figure 3.2: CBIR Results for Query: Desert

## 3.2 Dataset

For the purpose of investigating and developing the algorithms presented in this thesis, we chose to focus on the task of content-based image retrieval (CBIR) over the 31,992 images in the Corel dataset (Corel, 2010). While we are aware of the criticisms of this particular dataset (Müller *et al.*, 2002), we will not be comparing our algorithms to any other system besides the baseline system presented in this chapter. This method of CBIR algorithm development has been cited as quite useful for testing out the efficacy of ideas as is the main aim of the thesis (Howarth, 2007).

### 3.2.1 Features

We represent images from the Corel dataset as 240-dimensional binary feature vectors. To create these vectors, each image is first processed into 48 Gabor texture features computed as outlined in (Howarth & Rüger, 2004), 27 Tamura texture features computed as outlined in (Tamura *et al.*, 1978), and 165 color histogram features computed as outlined in (Heesch *et al.*, 2003). After each image has been initially processed into a 240-dimensional feature vector, the entire dataset is then further processed to create a sparse binary matrix. To binarize the data in an informative way, the skewness of each feature is calculated across the dataset. For each image, if a specific feature is positively skewed above the 80th percentile, it is assigned the value '1', and if a specific feature is negatively skewed below the 20th percentile, it is assigned the value '1'. If these conditions are not met, the feature in question is assigned the value '0'. This binarization stage transforms the entire image dataset into a sparse binary matrix that represents the features which most distinguish each image from the rest of the dataset.

For the purpose of this thesis, a pre-binarized dataset was supplied by the authors from (Heller & Ghahramani, 2006).

## 3.3 Evaluation Measures

We use the same fifty query sets as described in (Heller & Ghahramani, 2006) and evaluate our results on both the labels provided by Corel and through human judgements. As with the original work on CBIR in Bayesian Sets, we consider only the top nine ranked results for evaluation. The following sections explicitly outline the evaluation measures we will be using.

### 3.3.1 Precision at Position $n$

*Precision at n* is a commonly used measure to evaluate the relevance of the top $n$ items in a ranked result with respect to a given query. The computation is as follows:

$$P@n = \frac{\#\ of\ relevant\ items\ in\ top\ n\ results}{n} \tag{3.2}$$

Over a set of queries, the precision at $n$ values are averaged to obtain a mean P@$n$ score. Although this measure has widespread use, it is known to be unstable since the total number of relevant items for a given query has a strong influence on P@$n$ yet may vary greatly over a set of queries. We take this into consideration and only concern ourselves with P@9, giving a simple assessment of how many of the top nine images retrieved were relevant. Within the thesis we refer to this measure as *precision*.

### 3.3.2 Normalized Discount Cumulative Gain

*Normalized Discount Cumulative Gain* (NDCG) is another evaluation measure for ranked results and has been gaining increasing adoption by the TREC (Text REtrieval Conference) research community. It is designed under the assumptions that 1.) highly relevant items/documents/images are more valuable than marginally relevant ones, and 2.) the lower ranking position an item/document/image has, the less valuable it is for the user. Similar to P@$n$, the NDCG value at position $n$ of a ranked list is computed as:

$$NDCG(n) = Z_n \sum_{j=1}^{n} \frac{2^{relevant(j)} - 1}{log(1+j)} \tag{3.3}$$

where relevant($j$) is the relevance (0 or 1) of the $j^{th}$ image in the ranked list and $Z_n$ is a normalization constant chosen so that a perfect list (all top $n$ images are relevant) gets a NDCG score of 1. Though the primary desire of NDCG is that the relevant($j$) function can be non-binary, we prefer it over precision at $n$ since it factors in the ranked positions of relevant images. Thus we consider NDCG(1)-NDCG(9) in our evaluations.

### 3.3.3 Amazon Mechanical Turk

Amazon Mechanical Turk (MT) is an online marketplace which allows workers to perform simple judgement tasks for a minor fee through a web interface (Amazon, 2010). MT has had reported success for gathering large sets of relevance judgements in a cheap and quick manner. In particular, it has been cited as a method to develop gold-standard data for natural language processing (Snow *et al.*, 2008), measuring the semantic meaning in inferred topic models (Boyd-Graber *et al.*, 2009), and of most interest to us, as a method of annotating large image sets (Deng *et al.*, 2009; Sorokin & Forsyth, 2008).

To use Amazon MT, we displayed the top nine results for a query and asked the user to mark which images were relevant based on the following criterion:

- Does the concept "query" reasonably apply to some main visual aspect of this image?
- Would the concept "query" be a valuable member of a set of words describing the main visual elements, composition or style of this image?

An example task is presented in figure 3.3.

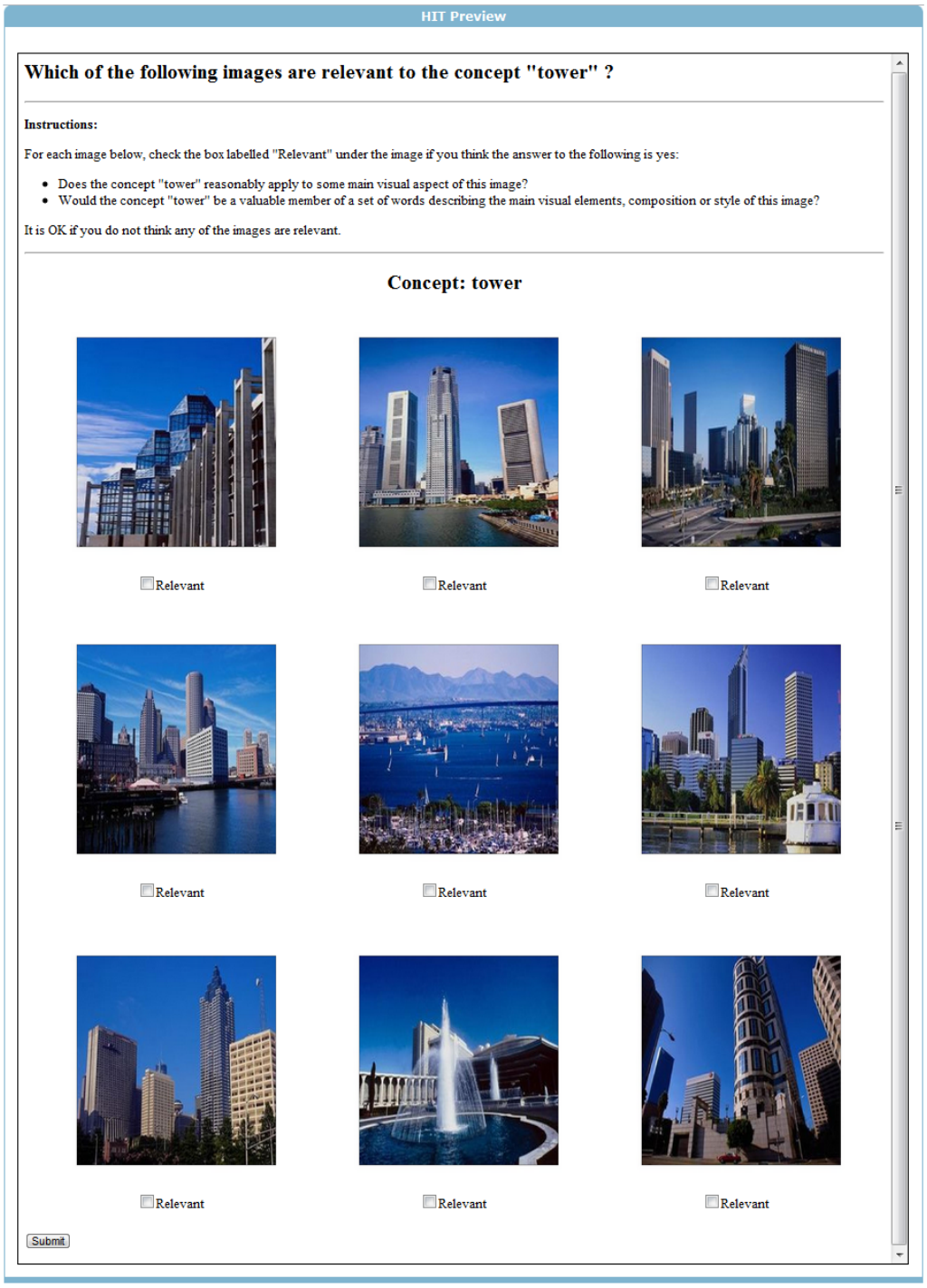


Figure 3.3: MTurk Example

## 3.4 Results

The Bayesian CBIR system from (Heller & Ghahramani, 2006) was tested over fifty queries on the Corel dataset to gather a baseline evaluation for the algorithms we will be developing in the forthcoming chapters. Table 3.1 lists the precision for each of the fifty queries as judged relevant by Amazon Mechanical Turk workers and the labels given from Corel. Table 3.2 lists the average precision over all fifty queries and the NDCG@$n$ values for both labelling systems.

| Query | MTurk | Labels | Query | MTurk | Labels |
|---|---|---|---|---|---|
| abstract | 7.6 | 3 | mountain | 5.5 | 0 |
| aerial | 4.3 | 1 | mountains | 6.7 | 6 |
| animal | 7.7 | 7 | penguins | 6.3 | 6 |
| ape | 3.5 | 4 | people | 5.7 | 6 |
| boat | 1.8 | 0 | person | 4.2 | 2 |
| building | 5.9 | 5 | pet | 3.6 | 3 |
| butterfly | 5.1 | 5 | reptile | 2.7 | 2 |
| castle | 4.9 | 3 | river | 4.4 | 1 |
| cavern | 4.3 | 4 | sea | 5.4 | 1 |
| cell | 5.6 | 6 | sign | 8.4 | 8 |
| church | 4 | 1 | snow | 5.7 | 5 |
| clouds | 5.3 | 4 | stairs | 3.6 | 4 |
| coast | 5.9 | 2 | sunset | 8.7 | 8 |
| desert | 5.5 | 3 | textures | 6.8 | 5 |
| door | 8.3 | 7 | tool | 4.1 | 3 |
| drawing | 4.3 | 4 | tower | 6.3 | 0 |
| eiffel | 5.5 | 5 | trees | 5.7 | 5 |
| fireworks | 8.3 | 9 | turtle | 2.8 | 1 |
| flower | 8.1 | 5 | urban | 5.8 | 0 |
| fractal | 5.7 | 0 | volcano | 2.3 | 2 |
| fruit | 5 | 4 | water | 7.7 | 8 |
| house | 5 | 2 | waterfall | 2.3 | 2 |
| kitchen | 6.1 | 5 | white | 7.2 | 6 |
| lights | 6.3 | 4 | woman | 4.1 | 4 |
| model | 4.9 | 5 | zebra | 2 | 2 |

Table 3.1: CBIR Baseline Precision for 50 queries

| Eval Set | Precision | N@1 | N@2 | N@3 | N@4 | N@5 | N@6 | N@7 | N@8 | N@9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Labels | 3.76 | 0.340 | 0.371 | 0.411 | 0.412 | 0.405 | 0.407 | 0.419 | 0.414 | 0.411 |
| Mturk | 5.34 | 0.596 | 0.586 | 0.612 | 0.618 | 0.610 | 0.601 | 0.603 | 0.601 | 0.597 |

Table 3.2: Baseline evaluation over 50 queries

## 3.5 Discussion

It is interesting to note the large difference in evaluation between the Corel labels and the responses from MT workers. Not only was there a difference in precision of more than 1.5 images (explaining why the NDCG@$n$ values for MT responses were consistently about .2 greater), but careful inspection of table 3.1 reveals numerous occasions where the MT workers judged some of the results as relevant yet none of the results had relevant Corel labels. A particular case of this is the query *tower*, where MT workers responded that roughly 6 of the 9 images were relevant, but none of these images was labelled as a tower in the Corel set. Figure 3.4 below displays the top 9 images that the Bayesian CBIR algorithm returned for query *tower*.

Figure 3.4: CBIR Results for Query: Tower

This dichotomy between evaluation sets brings into question the quality of the Corel labels. Since the training data used as query sets was created from these labels, this motivates us to investigate using Bayesian Sets to detect outliers in its own query sets.

# Chapter 4

# Novelty Detection

Novelty detection is the identification of unknown or noisy data, generally used to clean a corpus used for training learning algorithms (Hodge & Austin, 2004; Markou & Singh, 2003). Our motivation for developing a novelty detection algorithm inside Bayesian Sets stems from concerns over the quality of the Corel labels. In this chapter we develop a system within the Bayesian Set framework to rank items in labelled training data by how prototypical each element is of the given set and then use this ranking to find outliers. We test our system on sets of images from the Corel dataset and investigate the efficacy of incorporating novelty detection in the Bayesian CBIR system.

## 4.1 Novelty Detection in Bayesian Sets

Given a set of items with a particular label, we aim to find any outliers that do not represent the global distribution of features over the set. Our proposed implementation in Bayesian Sets is to simply iterate through each element of a particular set, remove it from the set, and compute a leave-one-out score based on the Bayesian scoring criterion outlined in section 2.2. This will result in a ranking of the items in the set by how 'prototypical' they are compared with the global properties of the set.

A general summary of this novelty detection algorithm under the Bayesian Set Framework is given in the following pseudocode:

**Algorithm 3** Novelty Detection in Bayesian Sets

**input:** a set of items, $\mathcal{D}_w$, for a particular label $w$
**for** each item $\mathbf{x}_i \in \mathcal{D}_w$ **do**
  let $\mathcal{D}_{wi} = \{\mathcal{D}_w \setminus \mathbf{x}_i\}$
  compute score$(\mathbf{x}_i, \mathcal{D}_{wi})$
**end for**
rank items in $\mathcal{D}_w$ by this score
**output:** sorted list of top scoring items in $\mathcal{D}_w$

## 4.2 Results

We ran a number of experiments with the Bayesian Novelty Detection algorithm on the Corel training data as are described in the following sections.

### 4.2.1 BND on the Corel Training Sets

We used our Bayesian Novelty Detection (BND) system on the training sets for our fifty queries and returned the top nine and bottom nine ranked images for evaluation. Since the training sets were created from the Corel labels, we performed evaluation solely with Amazon Mechanical Turk (MT).

Examples of the top nine and bottom nine images returned from our algorithm are given in figures 4.1 - 4.4, where the query labels are specified in the captions.

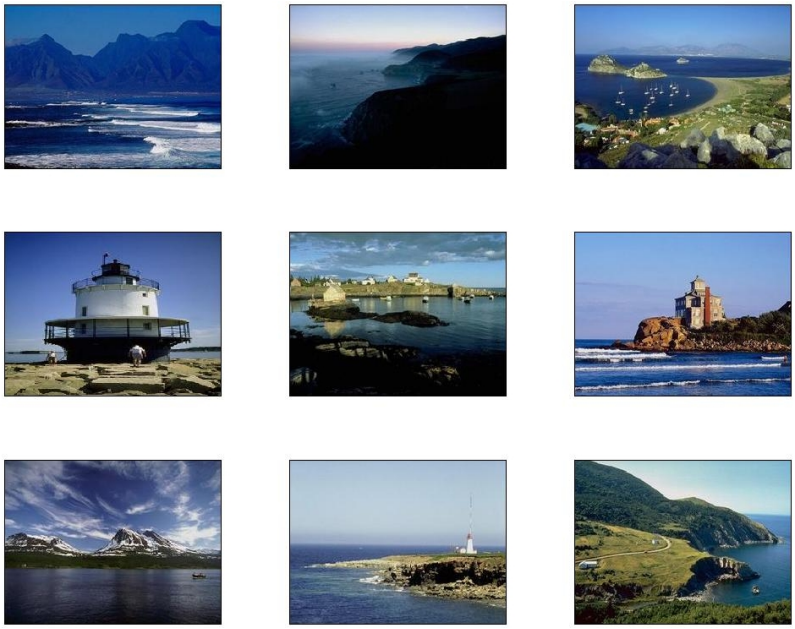

Figure 4.1: Top 9 images from Coast training set

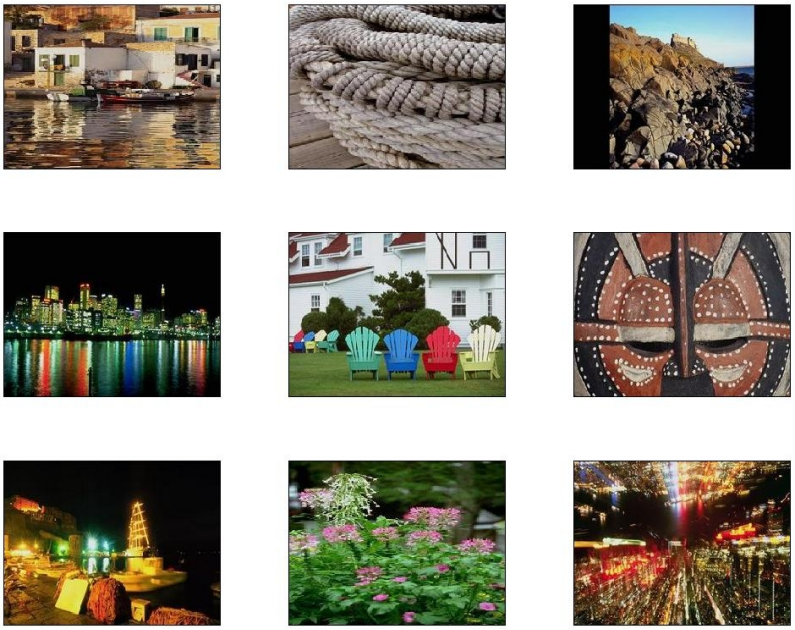

Figure 4.2: Bottom 9 images from Coast training set

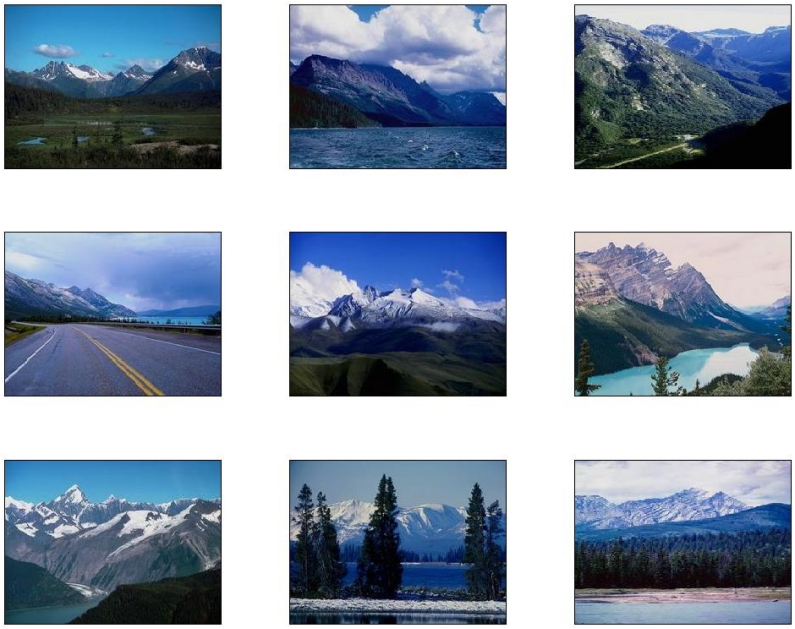

Figure 4.3: Top 9 images from Mountains training set

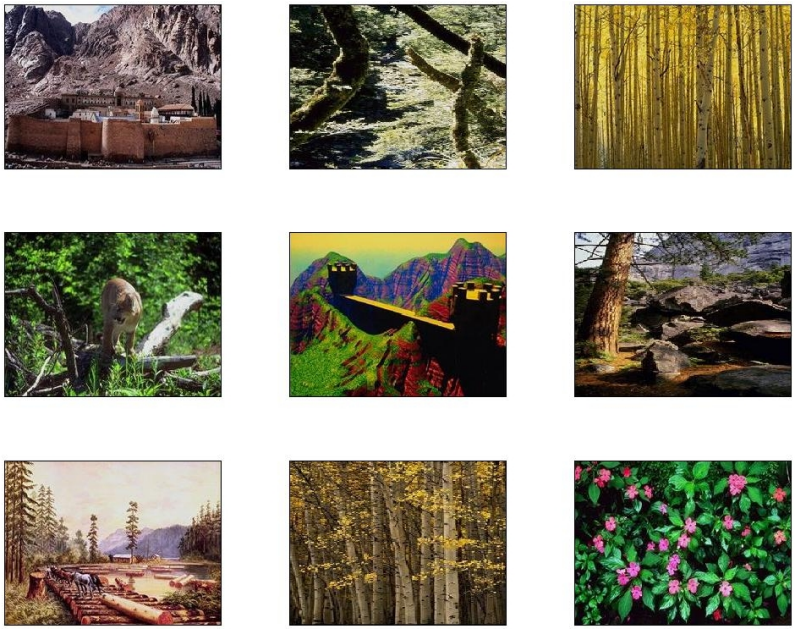

Figure 4.4: Bottom 9 images from Mountains training set

The results of this experiment are listed in table 4.1, which displays the average precision, as judged over the ten MT workers, for the top nine and bottom nine images of each query.

| Query | Top 9 | Bottom 9 | Query | Top 9 | Bottom 9 |
|---|---|---|---|---|---|
| abstract | 6.6 | 4.9 | mountain | 7.6 | 4.1 |
| aerial | 7.4 | 6.1 | mountains | 8.4 | 3.5 |
| animal | 7.5 | 8.1 | penguins | 8.4 | 8.4 |
| ape | 4.4 | 6.7 | people | 6.2 | 7.2 |
| boat | 4.8 | 6.1 | person | 8.9 | 7.1 |
| building | 6.8 | 6.5 | pet | 8.5 | 8.0 |
| butterfly | 8.5 | 7.8 | reptile | 7.8 | 8.0 |
| castle | 5.8 | 6.0 | river | 7.5 | 4.7 |
| cavern | 7.6 | 6.8 | sea | 7.9 | 5.3 |
| cell | 7.3 | 6.6 | sign | 8.6 | 7.6 |
| church | 8.7 | 4.6 | snow | 7.7 | 5.7 |
| clouds | 6.7 | 3.4 | stairs | 7.9 | 6.3 |
| coast | 8.2 | 3.6 | sunset | 8.9 | 6.3 |
| desert | 8.0 | 4.2 | textures | 8.3 | 5.2 |
| door | 8.4 | 7.5 | tool | 8.2 | 6.6 |
| drawing | 8.3 | 7.1 | tower | 7.6 | 4.4 |
| eiffel | 7.7 | 7.2 | trees | 7.0 | 5.8 |
| fireworks | 9.0 | 7.2 | turtle | 8.4 | 9.0 |
| flower | 8.5 | 4.2 | urban | 6.4 | 6.8 |
| fractal | 7.6 | 7.1 | volcano | 8.1 | 4.3 |
| fruit | 8.5 | 5.8 | water | 7.8 | 2.6 |
| house | 6.3 | 4.5 | waterfall | 8.4 | 8.2 |
| kitchen | 8.2 | 7.5 | white | 5.9 | 3.1 |
| lights | 8.6 | 4.8 | woman | 8.9 | 7.0 |
| model | 8.3 | 7.6 | zebra | 9.0 | 6.4 |
| Average Precision | | Top 9: 7.72 | | Bottom 9: 6.07 | |

Table 4.1: Average Precision of top 9 and bottom 9 sets over 50 queries

Below, in figure 4.5, we represent entries in table 4.1 as (x,y) pairs with the precision of the bottom nine images on the x-axis, and the precision of the top nine images on the y-axis. We show a plot of $y = x$ to illustrate that the images in the top nine set were deemed more relevant than those in the bottom nine.

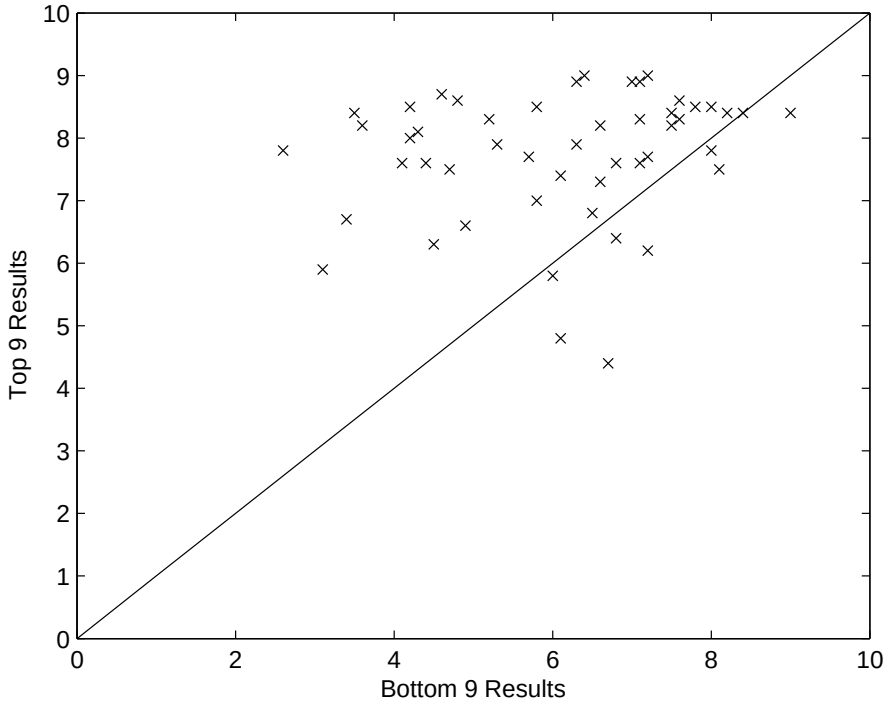


Figure 4.5: Top and Bottom Results Plot

We then added outlier images into the training sets for particular categories and re-ranked the training sets. For example, figure 4.6 displays the bottom nine images ranked by BND on the training set for the label *fireworks*, while figure 4.7 displays the bottom nine images ranked by BND on the *fireworks* training set with two outlier images introduced into the set: an image of two colorful doors, and a desert scene with a cactus.

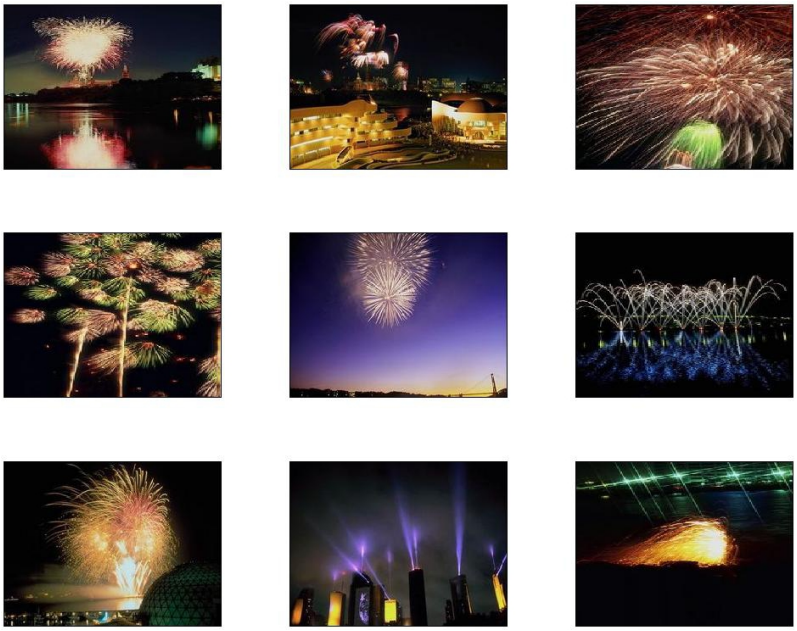

Figure 4.6: Bottom 9 images from Fireworks training set

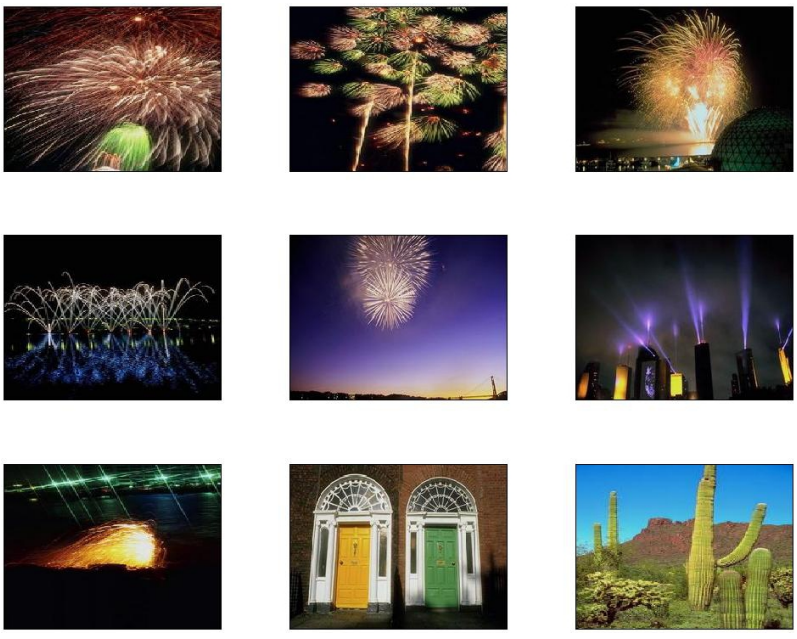

Figure 4.7: Bottom 9 images from Fireworks training set with outliers

### 4.2.2 BND within CBIR

We then re-ran our CBIR system over the fifty queries, but with our BND system first ranking the training set of a given query and returning varying percentages of the total set to act as the CBIR query set, $D_q$. We examined the efficacy of using the top 90%, 80%, 70%, 60%, and 50% of the BND ranked training sets as the query set for a particular query. We evaluated the results of this modified CBIR system on both the Corel labels and MT. The results of these experiments are given in tables 4.2 and 4.3 as NDCG@n values and average number of images out of the nine images returned by the CBIR algorithm for the respective evaluation set. We limited evaluation with MT to the top 90%, 80%, and 70% cases for time and cost efficiency.

| Eval Set | Precision | N@1 | N@2 | N@3 | N@4 | N@5 | N@6 | N@7 | N@8 | N@9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Baseline (100%) | 3.76 | 0.340 | 0.371 | 0.411 | 0.412 | 0.405 | 0.407 | 0.419 | 0.414 | 0.411 |
| Top 90% | 3.74 | 0.400 | 0.400 | 0.414 | 0.415 | 0.416 | 0.420 | 0.430 | 0.421 | 0.415 |
| Top 80% | 3.86 | 0.380 | 0.388 | 0.405 | 0.431 | 0.422 | 0.417 | 0.415 | 0.419 | 0.422 |
| Top 70% | 3.76 | 0.380 | 0.388 | 0.433 | 0.441 | 0.441 | 0.426 | 0.423 | 0.418 | 0.418 |
| Top 60% | 3.54 | 0.360 | 0.368 | 0.394 | 0.405 | 0.412 | 0.405 | 0.397 | 0.397 | 0.393 |
| Top 50% | 3.48 | 0.340 | 0.348 | 0.374 | 0.389 | 0.390 | 0.393 | 0.383 | 0.390 | 0.383 |

Table 4.2: Evaluation over 50 queries on Labels

| Eval Set | Precision | N@1 | N@2 | N@3 | N@4 | N@5 | N@6 | N@7 | N@8 | N@9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Baseline (100%) | 5.338 | 0.596 | 0.586 | 0.612 | 0.618 | 0.610 | 0.601 | 0.603 | 0.601 | 0.597 |
| Top 90% | 4.904 | 0.598 | 0.582 | 0.582 | 0.572 | 0.565 | 0.557 | 0.565 | 0.559 | 0.555 |
| Top 80% | 5.088 | 0.610 | 0.571 | 0.584 | 0.597 | 0.590 | 0.580 | 0.580 | 0.575 | 0.573 |
| Top 70% | 5.286 | 0.614 | 0.617 | 0.633 | 0.633 | 0.622 | 0.612 | 0.611 | 0.603 | 0.599 |

Table 4.3: Evaluation over 50 queries on MTurk

## 4.3 Discussion

Our novelty detection algorithm works rather well for detecting outliers in the training sets. However, when integrating the ranked training sets in the CBIR system, using a simple percentage threshold over all queries regardless of the number of elements in the training set does not seem beneficial. With such a simple heuristic, we may be removing breadth in the diversity of concepts that is actually desired. In future work we will examine methods to find label-specific thresholds based off the scores for each label. For example, the following figures are the scores from the BND algorithm for the Coast and Mountains training sets previously shown in figures 4.1 - 4.4.

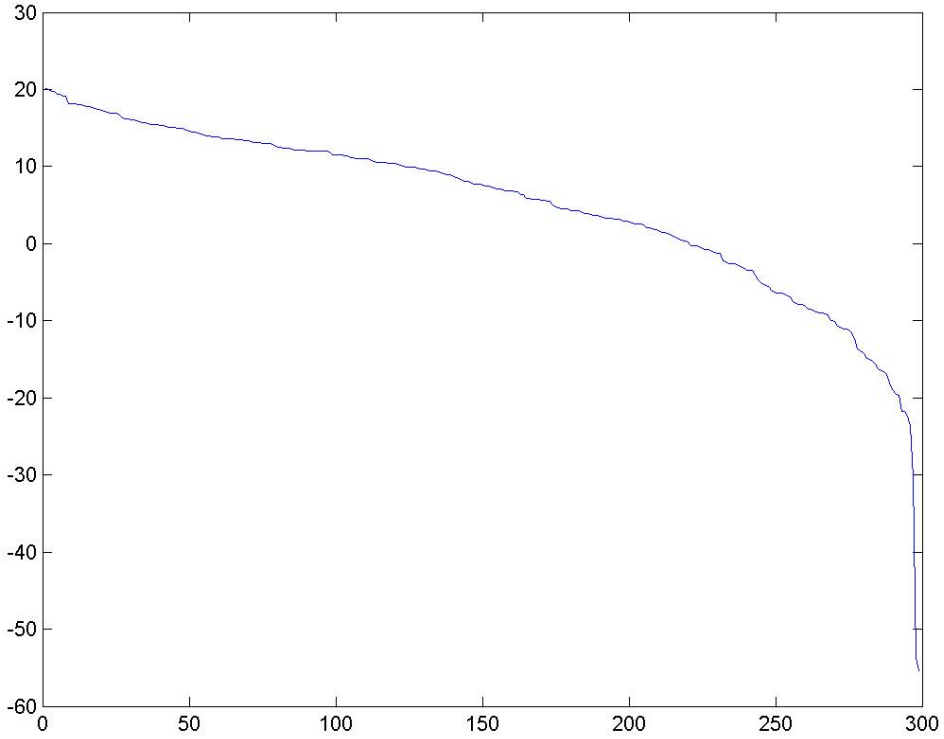


Figure 4.8: BND scores plotted against no. of training examples for Coast

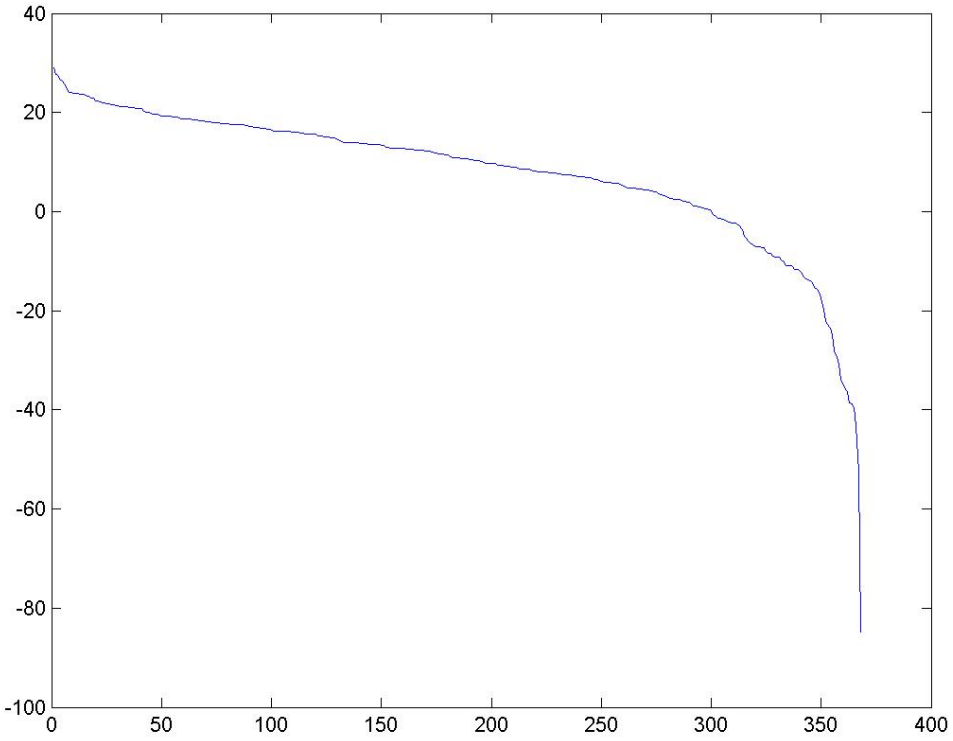


Figure 4.9: BND scores plotted against no. of training examples for Mountains

These graphs indicate that there are few examples within the training sets that are very atypical, as was displayed in the bottom 9 results of figures 4.1 - 4.4. This is a very nice result that should lead to better integration within the CBIR system.

# Chapter 5

# Relevance Feedback

A different approach to inferring a user's intended target in an information retrieval task is to actually involve the user in the retrieval process, known as *relevance feedback* (Manning *et al.*, 2008). The basic procedure is that the user makes an initial query and the retrieval system returns a set of results as usual, however, then the user interacts with this result set by marking which items he or she deems relevant or nonrelevant. The retrieval system then uses these relevance markings to get a better focus on the intended information need and returns a revised set of retrieval results. This feedback process can reiterate indefinitely as the user sees fit until his or her information need has been satisfied.

In this chapter we develop a system within the Bayesian Set framework that incorporates user-defined positive and negative examples to increase the performance of an information retrieval task. We test our relevance feedback algorithm by incorporating it within the Bayesian content-based image retrieval (CBIR) system presented in chapter 3.

## 5.1 Relevance Feedback in Bayesian Sets

To implement relevance feedback in the Bayesian Set framework, we must first assume the user has performed a query search on some collection of items $\mathcal{D}$ and retrieved a set of K ranked results using Bayesian Sets as described in algorithm 1. Then, through some interface, we assume the user has selected a

subset of M items that are relevant. We shall denote the set of relevant (positive) items as $\mathcal{D}_p = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\}$ and the K-M = N nonrelevant (negative) items as $\mathcal{D}_n = \{\mathbf{x}_1^*, \ldots, \mathbf{x}_N^*\}$. As before, we iterate through the elements in $\mathcal{D}$ we would like to rescore, with the current item to be scored denoted $\mathbf{x}^{**}$.

The scoring criterion must now consider multiple hypotheses: the hypothesis that $\mathbf{x}^{**}$ *and* $\mathcal{D}_p$ were generated by the same statistical model, the N hypotheses that $\mathbf{x}^{**}$ *and* a negative item $\mathbf{x}_i^* \in \mathcal{D}_n$ were generated by the same statistical model, and the hypothesis that $\mathbf{x}^{**}$ was generated independently from its own statistical model. Figure 5.1 depicts the graphical model of this setup.

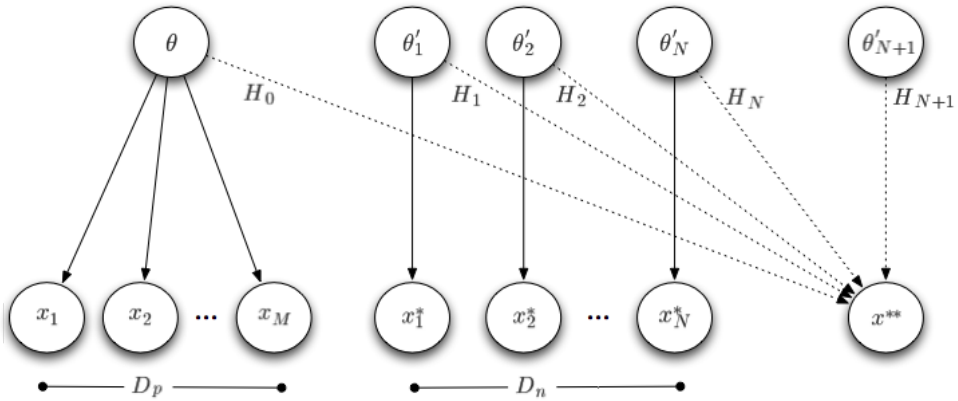


Figure 5.1: Graphical Model for rescore($\mathbf{x}^{**}$)

From this setup, our new scoring criterion becomes:

$$\text{rescore}(\mathbf{x}^{**}) = \frac{H_0}{\sum_{i=1}^{N+1} H_i} \tag{5.1}$$

where

$$
\begin{aligned}
H_0 &= p(\mathbf{x}^{**}, \mathcal{D}_p) \prod_{i=1}^{N} p(\mathbf{x}_i^*) \\
H_1 &= p(\mathcal{D}_p) p(\mathbf{x}^{**}, \mathbf{x}_1^*) p(\mathbf{x}_2^*) \dots p(\mathbf{x}_N^*) \\
H_2 &= p(\mathcal{D}_p) p(\mathbf{x}_1^*) p(\mathbf{x}^{**}, \mathbf{x}_2^*) \dots p(\mathbf{x}_N^*) \\
&\dots \\
H_N &= p(\mathcal{D}_p) p(\mathbf{x}_1^*) p(\mathbf{x}_2^*) \dots p(\mathbf{x}^{**}, \mathbf{x}_N^*) \\
H_{N+1} &= p(\mathcal{D}_p) \prod_{i=1}^{N} p(\mathbf{x}_i^*) p(\mathbf{x}^{**})
\end{aligned}
$$

If we include the $T$ items from the query set $\mathcal{D}_q$ of the initial search with our set of relevant items $\mathcal{D}_p$, then the preceeding and following derivations are similiar. One needs to simply set $M' = M + T$.

We can now rewrite the scoring criterion as follows:

$$
\text{rescore}(\mathbf{x}^{**}) = \frac{p(\mathbf{x}^{**}, \mathcal{D}_p) \prod_{i=1}^{N} p(\mathbf{x}_i^*)}{p(\mathcal{D}_p) \prod_{i=1}^{N} p(\mathbf{x}_i^*) \left[ \sum_{k=1}^{N} \frac{p(\mathbf{x}^{**}, \mathbf{x}_k^*)}{p(\mathbf{x}_k^*)} + p(\mathbf{x}^{**}) \right]} \tag{5.2}
$$

The marginal likelihoods in equation 5.2 can be written as integrals similar to those of equations 2.4, 2.5, and 2.6 in chapter 2. The prior of our current item can be expressed as:

$$
p(\mathbf{x}^{**}) = \int p(\mathbf{x}^{**}|\theta) p(\theta) d\theta \tag{5.3}
$$

where, as before, $\theta$ are the parameters of some distribution which has been chosen to model the item feature vectors, $p(\theta)$ is the prior over these parameters, and $p(\mathbf{x}^{**}|\theta)$ is the likelihood; the probability of observing $\mathbf{x}^{**}$ given that our model is parameterized by $\theta$. Similarly, the prior of a nonrelevant item is expressed:

$$
p(\mathbf{x}_i^*) = \int p(\mathbf{x}_i^*|\theta) p(\theta) d\theta \tag{5.4}
$$

For the prior of the relevant set, $\mathcal{D}_p$, we have:

$$
p(\mathcal{D}_p) = \int \left[ \prod_{i=1}^{M} p(\mathbf{x}_i|\theta) \right] p(\theta) d\theta \tag{5.5}
$$

where every item $\mathbf{x}_i$ in $\mathcal{D}_p$ comes from the same unknown parameters $\theta$. The joint probabilities we are concerned with are:

$$p(\mathbf{x}^{**}, \mathbf{x}_i^*) = \int p(\mathbf{x}_i^*|\theta)p(\mathbf{x}^{**}|\theta)p(\theta)d\theta \tag{5.6}$$

and

$$p(\mathbf{x}^{**}, \mathcal{D}_p) = \int \left[\prod_{i=1}^{M} p(\mathbf{x}_i|\theta)\right] p(\mathbf{x}^{**}|\theta)p(\theta)d\theta \tag{5.7}$$

Equation 5.6 assumes the item to be scored $\mathbf{x}^{**}$ *and* some nonrelevant item $\mathbf{x}_i^* \in \mathcal{D}_n$ are drawn i.i.d from the same unknown parameters $\theta$, and analogously, equation 5.7 assumes the item to be scored $\mathbf{x}^{**}$ *and* the relevant set, $\mathcal{D}_p$, are drawn i.i.d from the same unknown parameters $\theta$.

As with the original Bayesian Sets algorithm, each item $\mathbf{x}_i \in \mathcal{D}_p$ is represented as a binary vector $\mathbf{x}_i = (x_{i1}, \ldots, x_{iJ})$ where $x_{ij} \in \{0, 1\}$. We define a model in which each element of $\mathbf{x}_i$ has an independent Bernoulli distribution:

$$p(\mathbf{x}_i|\theta) = \prod_{j=1}^{J} \theta_j^{x_{ij}}(1-\theta_j)^{1-x_{ij}}$$

The conjugate prior for the parameters of a Bernoulli distribution is the Beta distribution:

$$p(\theta|\alpha, \beta) = \prod_{j=1}^{J} \frac{1}{Z(\alpha_j, \beta_j)}\ \theta_j^{\alpha_j - 1}\,(1-\theta_j)^{\beta_j - 1}$$

where $\alpha$ and $\beta$ are hyperparameters of the prior, and $Z(\cdot)$ is a normalization constant such that:

$$\begin{aligned} Z(\alpha, \beta) &= \int_0^1 \theta^{\alpha-1}\,(1-\theta)^{\beta-1}d\theta \\ &= \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} \end{aligned}$$

where the Gamma function, $\Gamma(\cdot)$, is a generalization of the factorial function. The hyperparameters $\alpha$ and $\beta$ are set empirically from the data, $\alpha = \kappa\mathbf{m}$, $\beta = \kappa(\mathbf{1} - \mathbf{m})$, where $\mathbf{m}$ is the mean of $\mathbf{x}$ over all items, and $\kappa$ is a scaling

factor. We note in passing that $\kappa$ can be set differently for the positive and negative query sets depending on the particular application.

With our statistical model defined, we can now derive an efficient method of computing rescore($\mathbf{x}^{**}$). For a relevant set of items $\mathcal{D}_p = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\}$, consisting of M vectors, we can compute its prior as so:

$$\begin{aligned} p(\mathcal{D}_p|\alpha,\beta) &= \int \left[\prod_{i=1}^{M} p(\mathbf{x}_i|\theta)\right] p(\theta|\alpha,\beta)d\theta \\ &= \int \left[\prod_{i=1}^{M}\prod_{j=1}^{J} \theta_j^{x_{ij}}(1-\theta_j)^{1-x_{ij}}\right] \prod_{j=1}^{J} \frac{1}{Z(\alpha_j,\beta_j)}\ \theta_j^{\alpha_j-1}\,(1-\theta_j)^{\beta_j-1} d\theta \\ &= \prod_j \frac{\Gamma(\alpha_j\ +\ \beta_j)}{\Gamma(\alpha_j)\,\Gamma(\beta_j)}\ \frac{\Gamma(\tilde{\alpha}_j)\Gamma(\tilde{\beta}_j)}{\Gamma(\tilde{\alpha}_j+\tilde{\beta}_j)} \end{aligned} \tag{5.8}$$

where $\tilde{\alpha} = \alpha + \sum_{i=1}^{M} x_{ij}$ and $\tilde{\beta} = \beta + M - \sum_{i=1}^{M} x_{ij}$.

It is not difficult to compute the rest of the marginal likelihoods in equation 5.2, and the reader is advised to review Appendix B for such derivations. By expanding and combining the marginal likelihoods, our criterion for rescoring becomes:

$$\text{rescore}(\mathbf{x}^{**}) = \frac{p(\mathbf{x}^{**},\mathcal{D}_p)\prod_{i=1}^{N} p(\mathbf{x}_i^*)}{p(\mathcal{D}_p)\prod_{i=1}^{N} p(\mathbf{x}_i^*)\left[\sum_{k=1}^{N}\frac{p(\mathbf{x}^{**},\mathbf{x}_k^*)}{p(\mathbf{x}_k^*)} + p(\mathbf{x}^{**})\right]}$$

$$= \frac{\frac{\Gamma(\tilde{\alpha}_j+x_{\cdot j})\Gamma(\tilde{\beta}_j+1-x_{\cdot j})}{\Gamma(\alpha_j+\beta_j+M+1)}\frac{\Gamma(\alpha_j+\beta_j+M)}{\Gamma(\tilde{\alpha}_j)\Gamma(\tilde{\beta}_j)}}{\left(\sum_{k=1}^{N}\frac{\frac{\Gamma(\alpha_j+x_{kj}+x_{\cdot j})\Gamma(\beta_j+1-x_{kj}+1-x_{\cdot j})}{\Gamma(\alpha_j+\beta_j+1+1)}}{\frac{\Gamma(\alpha_j+x_{kj})\Gamma(\beta_j+1-x_{kj})}{\Gamma(\alpha_j+\beta_j+1)}}\right) + \frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)}\frac{\Gamma(\alpha_j+x_{\cdot j})\Gamma(\beta_j+1-x_{\cdot j})}{\Gamma(\alpha_j)\Gamma(\beta_j)}}$$

Through a process of simplification via the property $\Gamma(x) = (x-1)\Gamma(x-1)$ and explicitly computing the possible values of $x_{\cdot j}$, we find the log of this score is *linear* in $\mathbf{x}$:

$$\log \text{rescore}(\mathbf{x}^{**}) = c + \sum_j q_j x_{\cdot j} \tag{5.9}$$

where

$$c = \sum_j \log(\tilde{\beta}_j) - \log(\alpha_j + \beta_j + M) - \log\left[\left(\frac{\sum_{k=1}^N \hat{\beta}_{kj}}{\alpha_j + \beta_j + 1}\right) + \left(\frac{\beta_j}{\alpha_j + \beta_j}\right)\right]$$

and

$$\begin{aligned} q_j = &\log(\tilde{\alpha}_j) - \log(\tilde{\beta}_j) \\ &- \log\left[\left(\frac{\sum_{k=1}^N \hat{\alpha}_{kj}}{\alpha_j + \beta_j + 1}\right) + \left(\frac{\alpha_j}{\alpha_j + \beta_j}\right)\right] \\ &+ \log\left[\left(\frac{\sum_{k=1}^N \hat{\beta}_{kj}}{\alpha_j + \beta_j + 1}\right) + \left(\frac{\beta_j}{\alpha_j + \beta_j}\right)\right] \end{aligned}$$

If the the dataset $\mathcal{D}$ is sparse and expressed as a single matrix $\mathbf{X}$, with $J$ columns, the log score for all items can be computed efficiently as a single matrix-vector multiplication. As before, we simply sort these scores and return the highest ranking results to the user.

A general summary of our relevance feedback algorithm under the Bayesian Set framework is given in the following pseudocode:

**Algorithm 4** Relevance Feedback in Bayesian Sets

**background:** a set of items $\mathcal{D}$, a probabilistic model $p(\mathbf{x}|\theta)$ where $\mathbf{x} \in D$, a prior on the model parameters $p(\theta)$, a set of K results for a query $\mathcal{D}_q = \{\mathbf{x}_i\} \subset D$

**input:** a set of relevant items $\mathcal{D}_p$ from the K results for query $\mathcal{D}_q$

**for all** $\mathbf{x} \in \mathcal{D}$ **do**

compute $\quad \text{rescore}(\mathbf{x}^{**}) = \dfrac{p(\mathbf{x}^{**}, \mathcal{D}_p) \prod_{i=1}^N p(\mathbf{x}_i^*)}{p(\mathcal{D}_p) \prod_{i=1}^N p(\mathbf{x}_i^*) \left[\sum_{k=1}^N \frac{p(\mathbf{x}^{**}, \mathbf{x}_k^*)}{p(\mathbf{x}_k^*)} + p(\mathbf{x}^{**})\right]}$

**end for**

**output:** sorted list of top scoring items in $D$

## 5.2 Results

We tested our relevance feedback (RF) algorithm by incorporating it within the Bayesian content-based image retrieval (CBIR) system presented in chapter 3. We performed an offline experiment where the Amazon Mechanical Turk (MT) evaluations from the baseline system were used as input to the RF algorithm and the resulting output was evaluated by another set of MT workers. Though this does not test if the RF results were more relevant for a particular user that judged the original results, it allows us to gather a large set of judgements quickly and we can then simply compare which set of results had better evaluations for a given query.

Examples for two queries that demonstrate the effects of relevance feedback follow in figures 5.2 - 5.5.

Figure 5.2: Baseline Results for Query: Aerial

Figure 5.3: RF Results for Query: Aerial

Figure 5.4: Baseline Results for Query: Penguins

Figure 5.5: RF Results for Query: Penguins

The results of this experiment are listed in tables 5.1 and 5.2. Table 5.1 lists the precision of the baseline and RF results for each of the fifty queries. Table 5.2 lists the average precision over all fifty queries and the NDCG@$n$ values for both systems.

| Query | Baseline | After RF | Query | Baseline | After RF |
|---|---|---|---|---|---|
| abstract | 7.6 | 6.4 | mountain | 5.5 | 6.2 |
| aerial | 4.3 | 5.4 | mountains | 6.7 | 6.7 |
| animal | 7.7 | 7.3 | penguins | 6.3 | 7.9 |
| ape | 3.5 | 4.6 | people | 5.7 | 7.6 |
| boat | 1.8 | 2.8 | person | 4.2 | 6.9 |
| building | 5.9 | 7.5 | pet | 3.6 | 4.3 |
| butterfly | 5.1 | 6.9 | reptile | 2.7 | 4.9 |
| castle | 4.9 | 4.2 | river | 4.4 | 5.7 |
| cavern | 4.3 | 6.3 | sea | 5.4 | 7.0 |
| cell | 5.6 | 5.2 | sign | 8.4 | 8.4 |
| church | 4.0 | 3.8 | snow | 5.7 | 5.3 |
| clouds | 5.3 | 5.7 | stairs | 3.6 | 3.9 |
| coast | 5.9 | 5.3 | sunset | 8.7 | 8.4 |
| desert | 5.5 | 4.0 | textures | 6.8 | 6.6 |
| door | 8.3 | 6.7 | tool | 4.1 | 4.4 |
| drawing | 4.3 | 6.2 | tower | 6.3 | 6.9 |
| eiffel | 5.5 | 4.8 | trees | 5.7 | 7.5 |
| fireworks | 8.3 | 8.5 | turtle | 2.8 | 3.0 |
| flower | 8.1 | 7.2 | urban | 5.8 | 5.4 |
| fractal | 5.7 | 6.3 | volcano | 2.3 | 4.8 |
| fruit | 5.0 | 5.6 | water | 7.7 | 7.6 |
| house | 5.0 | 5.3 | waterfall | 2.3 | 4.0 |
| kitchen | 6.1 | 7.0 | white | 7.2 | 5.5 |
| lights | 6.3 | 6.1 | woman | 4.1 | 4.0 |
| model | 4.9 | 3.8 | zebra | 2.0 | 5.0 |

Table 5.1: Baseline and RF results over 50 queries

| Eval Set | Precision | N@1 | N@2 | N@3 | N@4 | N@5 | N@6 | N@7 | N@8 | N@9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Baseline | 5.338 | 0.596 | 0.586 | 0.612 | 0.618 | 0.610 | 0.601 | 0.603 | 0.601 | 0.597 |
| RF | 5.816 | 0.840 | 0.814 | 0.782 | 0.754 | 0.736 | 0.721 | 0.703 | 0.694 | 0.686 |

Table 5.2: Evaluation of RF over 50 queries on MTurk

## 5.3 Discussion

The results from our experiment reveal the proposed algorithm works quite well. The average precision increased over our fifty queries and the mean NDCG@n values for RF display a monotonically decreasing trend that begins much higher than Baseline and ends almost .1 greater as well. This means there were more images relevant in the top nine results after RF and a majority of the relevant images were in the highest ranked positions.

However, we feel these particular results are actually a conservative lower bound and underestimate how well the algorithm performed. Our experiment did not properly represent a relevance feedback task because the MT workers that evaluated the RF results were not the same workers that evaluated the original CBIR results. Thus there were instances where images marked relevant in the intial CBIR result and ranked in the top nine results from RF were not marked relevant in the subsequent round of MT evaluation. For example, figure 5.6 is one such case. The original CBIR MT precision for the image on the left was 5.6, whereas the MT precision of the RF results on the right was 5.2. In future work, we aim to control our test environment with relevance feedback in a more restricted approach where the same user marking the CBIR results will mark the RF results.

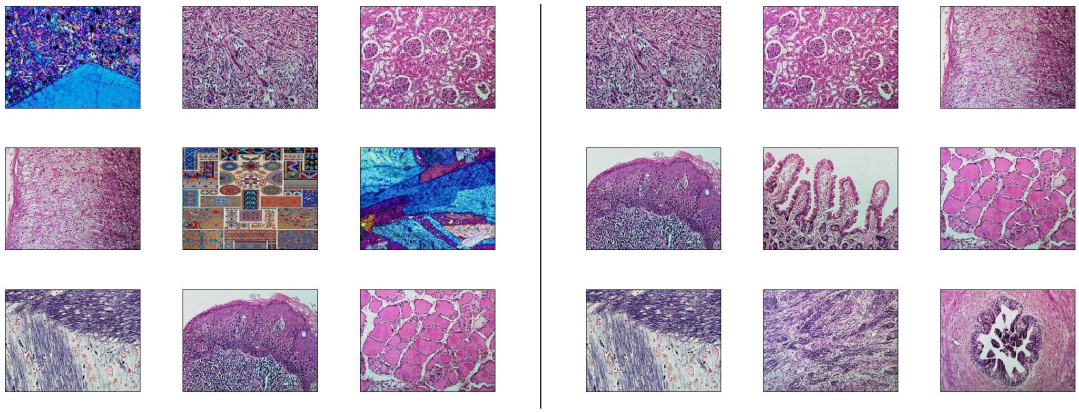

Figure 5.6: Top 9 Results for 'cell' in Baseline and RF

# Chapter 6

# Summary and Future Work

In this thesis, we have developed algorithms for novelty detection and relevance feedback under the Bayesian Set framework. Using content-based image retrieval as the particular task for evaluation, we have found our algorithms perform quite well. In Chapter 4, we presented our novelty detection algorithm that successfully found outliers in the given training sets. We found that incorporating this algorithm in the existing CBIR system by using a simple percentage threshold over all queries did not aid the task of image retrievel. For future work, we will investigate better methods of incorporating our ranked training sets into the Bayesian CBIR system. In Chapter 5, we presented our relevance feedback algorithm that successfully retrieved more relevant images than the baseline system and with a majority of these relevant images found at higher ranks. In the future, we would like to use these algorithms on other datasets to see how well they generalise across sparse-binary data.

We are currently exploring algorithms under Bayesian Sets to perform automated dataset annotation and developing clustering methods on both the query sets and results sets to improve the CBIR system.

# References


Amazon (2010). Amazon mechanical turk. `http://www.mturk.com`. 17

Boyd-Graber, J., Chang, J., Gerrish, S., Wang, C. & Blei, D. (2009). Reading tea leaves: How humans interpret topic models. *Advances in Neural Information Processing Systems*. 17

Corel (2010). Corel photo cds image database. `http://www.corel.com`. 12, 15

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K. & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. *IEEE Conference on Computer Vision and Pattern Recognition*. 17

Ghahramani, Z. & Heller, K.A. (2005). Bayesian sets. *Advances in Neural Information Processing Systems*. 1

Heesch, D., Pickering, M., Rüger, S. & Yavclinsky, A. (2003). Video retrieval with a browsing framework using key frames. *Proceedings of TRECVID*. 15

Heller, K.A. (2008). *Efficient Bayesian Methods for Clustering*. Ph.D. thesis, University College London, Gatsby Computational Neuroscience Unit. 3

Heller, K.A. & Ghahramani, Z. (2006). A simple bayesian framework for content-based image retrieval. *IEEE Conference on Computer Vision and Pattern Recognition*. 3, 12, 15, 16, 19

Hodge, V. & Austin, J. (2004). A survey of outlier detection methodologies. *Artificial Intelligence Review*, **22**, 85–126. 22

Howarth, P. & Rüger, S. (2004). Evaluation of texture features for content-based image retrieval. *International Conference on Image and Video Retrieval*. 15

Howarth, P.D. (2007). *Discovering images: features, similarties and subspaces*. Ph.D. thesis, University of London, Department of Computing. 1, 15

Jones, K.S. (1993). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, **28**, 11–21. 6

Ksikes, A. (2010). Cloud mining. `http://imdb.cloudmining.net/`. 3

Manning, C.D., Raghavan, P. & Shütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press, Cambridge. 1, 32

Markou, M. & Singh, S. (2003). Novelty detection: a review–part 1: statistical approaches. *Signal Processing*, **83**, 2481 – 2497. 22

Müller, H., Marchand-Maillet, S. & Pun, T. (2002). The truth about corel - evaluation in image retrieval. *International Conference on Image and Video Retrieval*. 15

Rocchio, J. (1971). Relevance feedback in information retrieval in the smart retrieval system - experiments in automatic document processing. *Salton ed*, 313–323.

Rui, Y., Huang, T.S., Ortega, M. & Mehrotra, S. (1998). Relevance feedback: A power tool for interactive content-based image retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, **8**.

Shepard, R.N. (1987). Toward a universal law of generalization for psychological science. *Science*, **237**, 1317–1323. 5

Silva, R., Heller, K.A. & Ghahramani, Z. (2007). Analogical reasoning with relational bayesian sets. *International Conference on AI and Statistics*. 3

Snow, R., O'Connor, B., Jurafsky, D. & Ng, A.Y. (2008). Cheap and fast-but is it good? evaluating non-expert annotations for natural language tasks. *Conference on Empirical Methods in Natural Language Processing*. 17

Sorokin, A. & Forsyth, D. (2008). Utility data annotation with amazon mechanical turk. *IEEE Conference on Computer Vision and Pattern Recognition*. 17

Su, Z., Zhang, H., Li, S. & Ma, S. (2003). Relevance feedback in content-based image retrieval: Bayesian framework, feature subspaces, and progressive learning. *IEEE Transactions on Image Processing*, **12**.

Tamura, H., Mori, S. & Yamawaki, T. (1978). Textual features corresponding to visual perception. *IEEE Transactions on Systems, Man and Cybernetics*, **8**, 460–472. 15

Tenenbaum, J.B. & Griffiths, T.L. (2001). Generalization, similarity, and bayesian inference. *Behavioral and Brain Sciences*, **24**, 629–641. 6

Vasconcelos, N. (2007). From pixels to semantic spaces: advances in content-based image retrieval. *Computer*, **40**, 20–26. 1

Zhou, X.S. & Huang, T.S. (2003). Relevance feedback in image retrieval: A comprehensive review. *Multimedia Systems*, **8**.

# Appendix A

# Bayesian Sets Derivations

Here we derive the scoring function used for Bayesian Sets. For sake of clarity, the content-based image retrieval (CBIR) system used in the main text of this dissertation is assumed to be the retrieval problem under investigation. However, the following derivations apply to any sparse binary data.

## A.1 Bayesian Sets for CBIR

We assume the user has performed a query and the Bayesian Sets algorithm has located the collection of training images corresponding to this query, denoted as $\mathcal{D}_q = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$. The algorithm then considers each unlabelled image, $\mathbf{x}^*$, and computes the following score:

$$\text{score}(\mathbf{x}^*) = \frac{p(\mathbf{x}^*, \mathcal{D}_q)}{p(\mathbf{x}^*)p(\mathcal{D}_q)} \tag{A.1}$$

Each of the three terms in equation A.1 above are marginal likelihoods and from the discussion in section 3.1, can be expressed as the following integrals:

$$p(\mathbf{x}^*) = \int p(\mathbf{x}^*|\theta)p(\theta)d\theta \tag{A.2}$$

$$p(\mathcal{D}_q) = \int \left[\prod_{i=1}^{N} p(\mathbf{x}_i|\theta)\right] p(\theta)d\theta \tag{A.3}$$

$$p(\mathbf{x}^*, \mathcal{D}_q) = \int \left[ \prod_{i=1}^{N} p(\mathbf{x}_i|\theta) \right] p(\mathbf{x}^*|\theta) p(\theta) d\theta \tag{A.4}$$

Assuming each image $\mathbf{x}_i \in \mathcal{D}_q$ is represented as a binary vector $\mathbf{x}_i = (x_{i1}, \ldots, x_{iJ})$ with $x_{ij} \in \{0, 1\}$, we define a model in which each element of $\mathbf{x}_i$ has an independent Bernoulli distribution:

$$p(\mathbf{x}_i|\theta) = \prod_{j=1}^{J} \theta_j^{x_{ij}} (1 - \theta_j)^{1 - x_{ij}} \tag{A.5}$$

The conjugate prior for the parameters of a Bernoulli distribution is the Beta distribution:

$$p(\theta|\alpha, \beta) = \prod_{j=1}^{J} \frac{1}{Z(\alpha_j, \beta_j)} \, \theta_j^{\alpha_j - 1} \, (1 - \theta_j)^{\beta_j - 1} \tag{A.6}$$

where $\alpha$ and $\beta$ are hyperparameters of the prior, and $Z(\cdot)$ is a normalization constant such that

$$\begin{aligned} Z(\alpha, \beta) &= \int_0^1 \theta^{\alpha - 1} \, (1 - \theta)^{\beta - 1} d\theta \\ &= \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha + \beta)} \end{aligned}$$

where the Gamma function, $\Gamma(\cdot)$, is a generalization of the factorial function. The hyperparameters $\alpha$ and $\beta$ are set empirically from the data, $\alpha = \kappa \mathbf{m}$, $\beta = \kappa(\mathbf{1} - \mathbf{m})$, where $\mathbf{m}$ is the mean of $\mathbf{x}$ over all images, and $\kappa$ is a scaling factor. The following section provides the derived efficient computational forms of the marginal likelihoods in equations A.2, A.3, and A.4.

## A.2 Derivations of Marginal Likelihoods

### A.2.1 Computing the marginal likelihood $p(\mathbf{x}^*)$

Assuming the current image we are observing, $\mathbf{x}^*$, is represented as a binary vector $\mathbf{x}^* = (x_{\cdot 1}, \ldots, x_{\cdot J})$ where $x_{\cdot j} \in \{0, 1\}$, and defined under a model in which each element of $\mathbf{x}^*$ has an independent Bernoulli distribution and conjugate Beta

prior analogous to equations A.5 and A.6, the marginal likelihood $p(\mathbf{x}^*)$ from equation A.2 can be expressed as:

$$\begin{aligned}
p(\mathbf{x}^*|\alpha,\beta) &= \int p(\mathbf{x}^*|\theta)p(\theta|\alpha,\beta)d\theta \\
&= \int \prod_{j=1}^{J} \theta_j^{x_{\cdot j}}(1-\theta_j)^{1-x_{\cdot j}} \prod_{j=1}^{J} \frac{1}{Z(\alpha_j,\beta_j)}\ \theta_j^{\alpha_j-1}\,(1-\theta_j)^{\beta_j-1}d\theta \\
&= \prod_{j=1}^{J} \frac{1}{Z(\alpha_j,\beta_j)}\ \int \theta_j^{\alpha_j+x_{\cdot j}-1}(1-\theta_j)^{\beta_j-x_{\cdot j}}d\theta \\
&= \prod_{j=1}^{J} \frac{1}{Z(\alpha_j,\beta_j)}\ Z(\alpha_j + x_{\cdot j}\,,\ \beta_j + 1 - x_{\cdot j}) \\
&= \prod_{j} \frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)} \frac{\Gamma(\alpha_j+x_{\cdot j})\Gamma(\beta_j+1-x_{\cdot j})}{\Gamma(\alpha_j+\beta_j+1)}
\end{aligned} \tag{A.7}$$

### A.2.2 Computing the marginal likelihood $p(\mathcal{D}_q)$

For a query $\mathcal{D}_q = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ consisting of $N$ vectors, we show that:

$$p(\mathcal{D}_q|\alpha,\beta) = \int \left[\prod_{i=1}^{N} p(\mathbf{x}_i|\theta)\right] p(\theta|\alpha,\beta)d\theta$$

Expanding the bracketed term based off our assumed Bernoulli model from equation A.5 and expanding $p(\theta|\alpha,\beta)$, the given Beta conjugate prior in equation A.6, we get:

$$p(\mathcal{D}_q|\alpha,\beta) = \int \left[\prod_{i=1}^{N}\prod_{j=1}^{J} \theta_j^{x_{ij}}(1-\theta_j)^{1-x_{ij}}\right] \prod_{j=1}^{J} \frac{1}{Z(\alpha_j,\beta_j)}\ \theta_j^{\alpha_j-1}\,(1-\theta_j)^{\beta_j-1}d\theta$$

Through simplification we derive the following:

$$\begin{aligned}
p(\mathcal{D}_q|\alpha,\beta) &= \prod_{j=1}^{J} \frac{1}{Z(\alpha_j,\beta_j)} \int \left[\theta_j^{\sum_{i=1}^{N} x_{ij}} (1-\theta_j)^{N-\sum_{i=1}^{N} x_{ij}}\right] \theta_j^{\alpha_j-1} (1-\theta_j)^{\beta_j-1} d\theta \\
&= \prod_{j=1}^{J} \frac{1}{Z(\alpha_j,\beta_j)} \int \theta_j^{\alpha_j+\sum_{i=1}^{N} x_{ij}-1} (1-\theta_j)^{\beta_j+N-\sum_{i=1}^{N} x_{ij}-1} d\theta \\
&= \prod_{j=1}^{J} \frac{1}{Z(\alpha_j,\beta_j)} \; Z(\alpha_j + \sum_{i=1}^{N} x_{ij} \,,\, \beta_j + N - \sum_{i=1}^{N} x_{ij}) \\
&= \prod_{j=1}^{J} \frac{\Gamma(\alpha_j \,+\, \beta_j)}{\Gamma(\alpha_j)\,\Gamma(\beta_j)} \; \frac{\Gamma(\alpha_j + \sum_{i=1}^{N} x_{ij})\,\Gamma(\beta_j + N - \sum_{i=1}^{N} x_{ij})}{\Gamma(\alpha_j + \sum_{i=1}^{N} x_{ij} \,+\, \beta_j + N - \sum_{i=1}^{N} x_{ij})} \\
&= \prod_{j} \frac{\Gamma(\alpha_j \,+\, \beta_j)}{\Gamma(\alpha_j)\,\Gamma(\beta_j)} \; \frac{\Gamma(\tilde{\alpha}_j)\Gamma(\tilde{\beta}_j)}{\Gamma(\tilde{\alpha}_j + \tilde{\beta}_j)}
\end{aligned} \tag{A.8}$$

where $\tilde{\alpha} = \alpha + \sum_{i=1}^{N} x_{ij}$ and $\tilde{\beta} = \beta + N - \sum_{i=1}^{N} x_{ij}$.

### A.2.3 Computing the marginal likelihood $p(\mathbf{x}^*, \mathcal{D}_q)$

From equation A.4 above, the joint probability that a given unlabelled image $\mathbf{x}^*$ and the query set $\mathcal{D}_q$ were produced by the same parameters $\theta$ can be expressed as:

$$p(\mathbf{x}^*, \mathcal{D}_q|\alpha,\beta) = \int \left[\prod_{i=1}^{N} p(\mathbf{x}_i|\theta)\right] p(\mathbf{x}^*|\theta) p(\theta|\alpha,\beta) d\theta$$

As derived in the previous subsections, the three individual expressions above expand as such:

$$\begin{aligned}
\prod_{i=1}^{N} p(\mathbf{x}_i|\theta) &= \prod_{j} \theta_j^{\sum_{i=1}^{N} x_{ij}} (1-\theta_j)^{N-\sum_{i=1}^{N} x_{ij}} \\
p(\mathbf{x}^*|\theta) &= \prod_{j} \theta_j^{x_{\cdot j}} (1-\theta_j)^{1-x_{\cdot j}} \\
p(\theta|\alpha,\beta) &= \prod_{j} \frac{1}{Z(\alpha_j,\beta_j)} \; \theta_j^{\alpha_j-1} (1-\theta_j)^{\beta_j-1}
\end{aligned}$$

Multiplying through and simplifying, we get:

$$\begin{aligned} p(\mathbf{x}^*, \mathcal{D}_q|\alpha, \beta) &= \prod_{j=1}^{J} \frac{1}{Z(\alpha_j, \beta_j)} \int \theta_j^{\alpha_j + \sum_{i=1}^{N} x_{ij} + x_{\cdot j} - 1} (1-\theta_j)^{\beta_j + N - \sum_{i=1}^{N} x_{ij} - x_{\cdot j}} \, d\theta \\ &= \prod_{j=1}^{J} \frac{1}{Z(\alpha_j, \beta_j)} \; Z(\alpha_j + \sum_{i=1}^{N} x_{ij} + x_{\cdot j} \, , \, \beta_j + N - \sum_{i=1}^{N} x_{ij} + 1 - x_{\cdot j}) \\ &= \prod_{j} \frac{\Gamma(\alpha_j + \beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)} \frac{\Gamma(\tilde{\alpha}_j + x_{\cdot j})\Gamma(\tilde{\beta}_j + 1 - x_{\cdot j})}{\Gamma(\alpha_j + \beta_j + N + 1)} \end{aligned} \tag{A.9}$$

where $\tilde{\alpha} = \alpha + \sum_{i=1}^{N} x_{ij}$ and $\tilde{\beta} = \beta + N - \sum_{i=1}^{N} x_{ij}$.

## A.3 Efficient Computation of score($\mathbf{x}^*$)

Combining the three marginal likelihoods from equation A.1, as computed in the previous section, we can derive an efficient computation of the score:

$$\begin{aligned} \text{score}(\mathbf{x}^*) &= \frac{p(\mathbf{x}^*, \mathcal{D}_q)}{p(\mathbf{x}^*)p(\mathcal{D}_q)} \\ &= \prod_{j} \frac{\frac{\Gamma(\alpha_j+\beta_j+N)}{\Gamma(\alpha_j+\beta_j+N+1)} \frac{\Gamma(\tilde{\alpha}_j+x_{\cdot j})\Gamma(\tilde{\beta}_j+1-x_{\cdot j})}{\Gamma(\tilde{\alpha}_j)\Gamma(\tilde{\beta}_j)}}{\frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j+\beta_j+1)} \frac{\Gamma(\alpha_j+x_{\cdot j})\Gamma(\beta_j+1-x_{\cdot j})}{\Gamma(\alpha_j)\Gamma(\beta_j)}} \end{aligned} \tag{A.10}$$

We can simplify this expression by using the fact that $\Gamma(x) = (x-1)\,\Gamma(x-1)$ for $x > 1$. Also, for each $j$ we can consider the two cases $x_{\cdot j} = 0$ and $x_{\cdot j} = 1$ separately.

**Case 1:**

Let $x_{\cdot j} = 1$. Then score($\mathbf{x}^*$) reduces to:

$$\text{score}(\mathbf{x}^*) = \prod_{j} \frac{\frac{\Gamma(\alpha_j+\beta_j+N)}{\Gamma(\alpha_j+\beta_j+N+1)} \frac{\Gamma(\tilde{\alpha}_j+1)}{\Gamma(\tilde{\alpha}_j)}}{\frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j+\beta_j+1)} \frac{\Gamma(\alpha_j+1)}{\Gamma(\alpha_j)}}$$

Using $\Gamma(x) = (x-1)\,\Gamma(x-1)$ for $x > 1$, we see:

$$\begin{aligned} \Gamma(\alpha_j + \beta_j + N + 1) &= (\alpha_j + \beta_j + N)\,\Gamma(\alpha_j + \beta_j + N) \\ \Gamma(\alpha_j + \beta_j + 1) &= (\alpha_j + \beta_j)\,\Gamma(\alpha_j + \beta_j) \\ \Gamma(\tilde{\alpha}_j + 1) &= (\tilde{\alpha}_j)\,\Gamma(\tilde{\alpha}_j) \\ \Gamma(\alpha_j + 1) &= (\alpha_j)\,\Gamma(\alpha_j) \end{aligned}$$

Thus, we derive the following for $x_{\cdot j} = 1$:

$$\begin{aligned}
\text{score}(\mathbf{x}^*) &= \prod_j \frac{\frac{\Gamma(\alpha_j+\beta_j+N)}{\Gamma(\alpha_j+\beta_j+N+1)}\frac{\Gamma(\tilde{\alpha}_j+1)}{\Gamma(\tilde{\alpha}_j)}}{\frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j+\beta_j+1)}\frac{\Gamma(\alpha_j+1)}{\Gamma(\alpha_j)}} \\
&= \prod_j \frac{\frac{\Gamma(\alpha_j+\beta_j+N)}{(\alpha_j+\beta_j+N)\,\Gamma(\alpha_j+\beta_j+N)}\frac{(\tilde{\alpha}_j)\,\Gamma(\tilde{\alpha}_j)}{\Gamma(\tilde{\alpha}_j)}}{\frac{\Gamma(\alpha_j+\beta_j)}{(\alpha_j+\beta_j)\,\Gamma(\alpha_j+\beta_j)}\frac{(\alpha_j)\,\Gamma(\alpha_j)}{\Gamma(\alpha_j)}} \\
&= \prod_j \left(\frac{\alpha_j+\beta_j}{\alpha_j+\beta_j+N}\right)\left(\frac{\tilde{\alpha}_j}{\alpha_j}\right)
\end{aligned}$$

**Case 2:**

Let $x_{\cdot j} = 0$. Then score($\mathbf{x}^*$) reduces to:

$$\text{score}(\mathbf{x}^*) = \prod_j \frac{\frac{\Gamma(\alpha_j+\beta_j+N)}{\Gamma(\alpha_j+\beta_j+N+1)}\frac{\Gamma(\tilde{\beta}_j+1)}{\Gamma(\tilde{\beta}_j)}}{\frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j+\beta_j+1)}\frac{\Gamma(\beta_j+1)}{\Gamma(\beta_j)}}$$

As above, using $\Gamma(x) = (x-1)\,\Gamma(x-1)$ for $x > 1$, we derive:

$$\begin{aligned}
\text{score}(\mathbf{x}^*) &= \prod_j \frac{\frac{\Gamma(\alpha_j+\beta_j+N)}{\Gamma(\alpha_j+\beta_j+N+1)}\frac{\Gamma(\tilde{\beta}_j+1)}{\Gamma(\tilde{\beta}_j)}}{\frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j+\beta_j+1)}\frac{\Gamma(\beta_j+1)}{\Gamma(\beta_j)}} \\
&= \prod_j \frac{\frac{\Gamma(\alpha_j+\beta_j+N)}{(\alpha_j+\beta_j+N)\,\Gamma(\alpha_j+\beta_j+N)}\frac{(\tilde{\beta}_j)\,\Gamma(\tilde{\beta}_j)}{\Gamma(\tilde{\beta}_j)}}{\frac{\Gamma(\alpha_j+\beta_j)}{(\alpha_j+\beta_j)\,\Gamma(\alpha_j+\beta_j)}\frac{(\beta_j)\,\Gamma(\beta_j)}{\Gamma(\beta_j)}} \\
&= \prod_j \left(\frac{\alpha_j+\beta_j}{\alpha_j+\beta_j+N}\right)\left(\frac{\tilde{\beta}_j}{\beta_j}\right)
\end{aligned}$$

Putting these two cases together, we can see that:

$$\text{score}(\mathbf{x}^*) = \prod_j \frac{\alpha_j+\beta_j}{\alpha_j+\beta_j+N}\left(\frac{\tilde{\alpha}_j}{\alpha_j}\right)^{x_{\cdot j}}\left(\frac{\tilde{\beta}_j}{\beta_j}\right)^{1-x_{\cdot j}} \tag{A.11}$$

To complete this section on efficiently computing score($\mathbf{x}^*$), we can now take the log of this score and note it is *linear* in $\mathbf{x}$:

$$\begin{aligned}
\log \text{score}(\mathbf{x}^*) &= \log \left( \prod_j \frac{\alpha_j + \beta_j}{\alpha_j + \beta_j + N} \left( \frac{\tilde{\alpha}_j}{\alpha_j} \right)^{x_{\cdot j}} \left( \frac{\tilde{\beta}_j}{\beta_j} \right)^{1 - x_{\cdot j}} \right) \\
&= \sum_j \log \left( \frac{\alpha_j + \beta_j}{\alpha_j + \beta_j + N} \right) + x_{\cdot j} \log \left( \frac{\tilde{\alpha}_j}{\alpha_j} \right) + (1 - x_{\cdot j}) \log \left( \frac{\tilde{\beta}_j}{\beta_j} \right) \\
&= \sum_j \log (\alpha_j + \beta_j) - \log (\alpha_j + \beta_j + N) + x_{\cdot j} \log \tilde{\alpha}_j - x_{\cdot j} \log \alpha_j \\
&\quad + \log \tilde{\beta}_j - x_{\cdot j} \log \tilde{\beta}_j - \log \beta_j + x_{\cdot j} \log \beta_j \\
&= c + \sum_j q_j x_{\cdot j} \qquad \text{(A.12)}
\end{aligned}$$

where

$$c = \sum_j \log(\alpha_j + \beta_j) - \log(\alpha_j + \beta_j + N) + \log \tilde{\beta}_j - \log \beta_j$$

and

$$q_j = \log \tilde{\alpha}_j - \log \alpha_j - \log \tilde{\beta}_j + \log \beta_j$$

Thus if our entire image data is in one large sparse binary matrix $\mathbf{X}$, we can compute the log scores for all images very efficiently using a single matrix vector multiplication:

$$\mathbf{s} = c + \mathbf{X}\mathbf{q}$$

# Appendix B

# Relevance Feedback Derivations

Here we derive the scoring function used for our Bayesian Sets model of Relevance Feedback. For sake of clarity, the content-based image retrieval system (CBIR) used in the main text of this dissertation is assumed to be the retrieval problem under investigation. However, the following derivations apply to any sparse binary data.

## B.1 Relevance Feedback in Bayesian Sets

We assume the user has performed a query search with Bayesian Sets, has been shown a set of K results, and through some interface has selected a set of M images that are relevant. We denote the set of relevant images as $\mathcal{D}_p = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\}$ and the K-M = N nonrelevant images as $\{\mathbf{x}_1^*, \ldots, \mathbf{x}_N^*\}$. We can then iterate through each unlabelled image, denoted $\mathbf{x}^{**}$, and compute the following rescoring criterion:

$$\text{rescore}(\mathbf{x}^{**}) = \frac{p(\mathbf{x}^{**}, \mathcal{D}_p) \prod_{i=1}^{N} p(\mathbf{x}_i^*)}{p(\mathcal{D}_p) \prod_{i=1}^{N} p(\mathbf{x}_i^*) \left[ \sum_{k=1}^{N} \frac{p(\mathbf{x}^{**}, \mathbf{x}_k^*)}{p(\mathbf{x}_k^*)} + p(\mathbf{x}^{**}) \right]} \tag{B.1}$$

Based off the discussion in section 5.1, the marginal likelihoods in equation B.1 can be expressed as the following integrals:

$$p(\mathbf{x}^{**}) = \int p(\mathbf{x}^{**}|\theta)p(\theta)d\theta \tag{B.2}$$

$$p(\mathbf{x}_i^{*}) = \int p(\mathbf{x}_i^{*}|\theta)p(\theta)d\theta \tag{B.3}$$

$$p(\mathcal{D}_p) = \int \left[\prod_{i=1}^{M} p(\mathbf{x}_i|\theta)\right] p(\theta)d\theta \tag{B.4}$$

$$p(\mathbf{x}^{**}, \mathbf{x}_i^{*}) = \int p(\mathbf{x}_i^{*}|\theta)p(\mathbf{x}^{**}|\theta)p(\theta)d\theta \tag{B.5}$$

$$p(\mathbf{x}^{**}, \mathcal{D}_p) = \int \left[\prod_{i=1}^{M} p(\mathbf{x}_i|\theta)\right] p(\mathbf{x}^{**}|\theta)p(\theta)d\theta \tag{B.6}$$

As with the original Bayesian Sets algorithm, each image $\mathbf{x}_i \in \mathcal{D}_p$ is represented as a binary vector $\mathbf{x}_i = (x_{i1}, \ldots, x_{iJ})$ where $x_{ij} \in \{0, 1\}$. We define a model in which each element of $\mathbf{x}_i$ has an independent Bernoulli distribution:

$$p(\mathbf{x}_i|\theta) = \prod_{j=1}^{J} \theta_j^{x_{ij}}(1-\theta_j)^{1-x_{ij}}$$

The conjugate prior for the parameters of a Bernoulli distribution is the Beta distribution:

$$p(\theta|\alpha, \beta) = \prod_{j=1}^{J} \frac{1}{Z(\alpha_j, \beta_j)}\ \theta_j^{\alpha_j - 1}\,(1-\theta_j)^{\beta_j - 1}$$

where $\alpha$ and $\beta$ are hyperparameters of the prior, and $Z(\cdot)$ is a normalization constant such that

$$\begin{aligned} Z(\alpha, \beta) &= \int_0^1 \theta^{\alpha-1}\,(1-\theta)^{\beta-1}d\theta \\ &= \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} \end{aligned}$$

and where the Gamma function, $\Gamma(\cdot)$, is a generalization of the factorial function. The hyperparameters $\alpha$ and $\beta$ are set empirically from the data, $\alpha = \kappa\mathbf{m}$, $\beta = \kappa(\mathbf{1}-\mathbf{m})$, where $\mathbf{m}$ is the mean of $\mathbf{x}$ over all images, and $\kappa$ is a scaling factor. In our model for relevance feedback, we distinguish $\kappa$ for the positive and negative query sets. The following section provides the derivations for the marginal likelihoods above.

## B.2 Derivations of Marginal Likelihoods

### B.2.1 Computing the marginal likelihood $p(\mathbf{x}^{**})$

Assuming the current image we are observing, $\mathbf{x}^{**}$, is represented as a binary vector $\mathbf{x}^{**} = (x_{\cdot 1}, \ldots, x_{\cdot J})$ where $x_{\cdot j} \in \{0, 1\}$, and defined under the Bernoulli and Beta models given above, we can compute the marginal likelihood $p(\mathbf{x}^{**})$ as such:

$$\begin{aligned} p(\mathbf{x}^{**}|\alpha, \beta) &= \int p(\mathbf{x}^{**}|\theta)p(\theta|\alpha, \beta)d\theta \\ &= \int \prod_{j=1}^{J} \theta_j^{x_{\cdot j}}(1-\theta_j)^{1-x_{\cdot j}} \prod_{j=1}^{J} \frac{1}{Z(\alpha_j, \beta_j)}\ \theta_j^{\alpha_j - 1}\,(1-\theta_j)^{\beta_j - 1} d\theta \\ &= \prod_j \frac{\Gamma(\alpha_j + \beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)} \frac{\Gamma(\alpha_j + x_{\cdot j})\Gamma(\beta_j + 1 - x_{\cdot j})}{\Gamma(\alpha_j + \beta_j + 1)} \end{aligned} \tag{B.7}$$

### B.2.2 Computing the marginal likelihood $p(\mathbf{x}_i^*)$

Assuming nonrelevant images are defined with the models given above, the marginal likelihood of a particular non-relevant image $\mathbf{x}_i^*$, can be expressed as so:

$$\begin{aligned} p(\mathbf{x}_i^*|\alpha, \beta) &= \int p(\mathbf{x}_i^*|\theta)p(\theta|\alpha, \beta)d\theta \\ &= \int \prod_{j=1}^{J} \theta_j^{x_{ij}}(1-\theta_j)^{1-x_{ij}} \prod_{j=1}^{J} \frac{1}{Z(\alpha_j, \beta_j)}\ \theta_j^{\alpha_j - 1}\,(1-\theta_j)^{\beta_j - 1} d\theta \\ &= \prod_j \frac{\Gamma(\alpha_j + \beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)} \frac{\Gamma(\alpha_j + x_{ij})\Gamma(\beta_j + 1 - x_{ij})}{\Gamma(\alpha_j + \beta_j + 1)} \end{aligned} \tag{B.8}$$

### B.2.3 Computing the marginal likelihood $p(\mathcal{D}_p)$

For a relevant set of images $\mathcal{D}_p = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\}$, consisting of M vectors, we derive:

$$\begin{aligned} p(\mathcal{D}_p|\alpha,\beta) &= \int \left[\prod_{i=1}^{M} p(\mathbf{x}_i|\theta)\right] p(\theta|\alpha,\beta)d\theta \\ &= \int \left[\prod_{i=1}^{M}\prod_{j=1}^{J} \theta_j^{x_{ij}}(1-\theta_j)^{1-x_{ij}}\right] \prod_{j=1}^{J} \frac{1}{Z(\alpha_j,\beta_j)}\ \theta_j^{\alpha_j-1}\,(1-\theta_j)^{\beta_j-1} d\theta \\ &= \prod_j \frac{\Gamma(\alpha_j\ +\ \beta_j)}{\Gamma(\alpha_j)\,\Gamma(\beta_j)}\ \frac{\Gamma(\tilde{\alpha}_j)\Gamma(\tilde{\beta}_j)}{\Gamma(\tilde{\alpha}_j+\tilde{\beta}_j)} \end{aligned} \tag{B.9}$$

where $\tilde{\alpha} = \alpha + \sum_{i=1}^{M} x_{ij}$ and $\tilde{\beta} = \beta + M - \sum_{i=1}^{M} x_{ij}$.

### B.2.4 Computing the marginal likelihood $p(\mathbf{x}^{**}, \mathbf{x}_i^*)$

For a particular pairing of a nonrelevant image, $\mathbf{x}_i^*$, and the current image to be scored, $\mathbf{x}^{**}$, the probability that they come i.i.d. from a model with unknown, but *same* parameters $\theta$ is:

$$p(\mathbf{x}^{**}, \mathbf{x}_i^*|\alpha,\beta) = \int p(\mathbf{x}_i^*|\theta)p(\mathbf{x}^{**}|\theta)p(\theta|\alpha,\beta)d\theta$$

As derived in the previous subsections, the three individual expressions above expand as such:

$$\begin{aligned} p(\mathbf{x}_i^*|\theta) &= \prod_j \theta_j^{x_{ij}}(1-\theta_j)^{1-x_{ij}} \\ p(\mathbf{x}^{**}|\theta) &= \prod_j \theta_j^{x_{\cdot j}}(1-\theta_j)^{1-x_{\cdot j}} \\ p(\theta|\alpha,\beta) &= \prod_j \frac{1}{Z(\alpha_j,\beta_j)}\ \theta_j^{\alpha_j-1}\,(1-\theta_j)^{\beta_j-1} \end{aligned}$$

Multiplying through and simplifying, we get:

$$p(\mathbf{x}^{**}, \mathbf{x}_i^*|\alpha,\beta) = \prod_j \frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)} \frac{\Gamma(\hat{\alpha}_{ij}+x_{\cdot j})\Gamma(\hat{\beta}_{ij}+1-x_{\cdot j})}{\Gamma(\alpha_j+\beta_j+1+1)} \tag{B.10}$$

where $\hat{\alpha}_{ij} = \alpha_j + x_{ij}$ and $\hat{\beta}_{ij} = \beta_j + 1 - x_{ij}$.

### B.2.5 Computing the marginal likelihood $p(\mathbf{x}^{**}, \mathcal{D}_p)$

Here we derive the joint probability that a given unlabelled image $\mathbf{x}^{**}$ and the relevant query set $\mathcal{D}_p$ were produced by the same parameters $\theta$.

$$p(\mathbf{x}^{**}, \mathcal{D}_p|\alpha, \beta) = \int \left[\prod_{i=1}^{M} p(\mathbf{x}_i|\theta)\right] p(\mathbf{x}^{**}|\theta)p(\theta|\alpha, \beta)d\theta$$

As derived in the previous subsections, the three individual expressions above expand as such:

$$\begin{aligned}
\prod_{i=1}^{M} p(\mathbf{x}_i|\theta) &= \prod_j \theta_j^{\sum_{i=1}^{M} x_{ij}} (1-\theta_j)^{M-\sum_{i=1}^{M} x_{ij}} \\
p(\mathbf{x}^{**}|\theta) &= \prod_j \theta_j^{x_{\cdot j}} (1-\theta_j)^{1-x_{\cdot j}} \\
p(\theta|\alpha, \beta) &= \prod_j \frac{1}{Z(\alpha_j, \beta_j)} \, \theta_j^{\alpha_j - 1} \, (1-\theta_j)^{\beta_j - 1}
\end{aligned}$$

Multiplying through and simplifying, we get:

$$p(\mathbf{x}^{**}, \mathcal{D}_p|\alpha, \beta) = \prod_j \frac{\Gamma(\alpha_j + \beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)} \frac{\Gamma(\tilde{\alpha}_j + x_{\cdot j})\Gamma(\tilde{\beta}_j + 1 - x_{\cdot j})}{\Gamma(\alpha_j + \beta_j + M + 1)} \tag{B.11}$$

where $\tilde{\alpha}_j = \alpha_j + \sum_{i=1}^{M} x_{ij}$ and $\tilde{\beta}_j = \beta_j + M - \sum_{i=1}^{M} x_{ij}$.

## B.3 Efficient Computation of rescore($\mathbf{x}^{**}$)

As given above in equation B.1, our criterion for rescoring is:

$$\begin{aligned}
\text{rescore}(\mathbf{x}^{**}) &= \frac{p(\mathbf{x}^{**}, \mathcal{D}_p) \prod_{i=1}^{N} p(\mathbf{x}_i^*)}{p(\mathcal{D}_p) \prod_{i=1}^{N} p(\mathbf{x}_i^*) \left[\sum_{k=1}^{N} \frac{p(\mathbf{x}^{**}, \mathbf{x}_k^*)}{p(\mathbf{x}_k^*)} + p(\mathbf{x}^{**})\right]} \\
&= \frac{p(\mathbf{x}^{**}, \mathcal{D}_p) \frac{1}{p(\mathcal{D}_p)}}{\left[\sum_{k=1}^{N} \frac{p(\mathbf{x}^{**}, \mathbf{x}_k^*)}{p(\mathbf{x}_k^*)} + p(\mathbf{x}^{**})\right]}
\end{aligned}$$

Expanding these terms with the derived expressions from the previous subsections and removing like terms, we get:

$$\text{rescore}(\mathbf{x}^{**}) = \frac{\frac{\Gamma(\tilde{\alpha}_j+x_{\cdot j})\Gamma(\tilde{\beta}_j+1-x_{\cdot j})}{\Gamma(\alpha_j+\beta_j+M+1)}\frac{\Gamma(\alpha_j+\beta_j+M)}{\Gamma(\tilde{\alpha}_j)\Gamma(\tilde{\beta}_j)}}{\left(\sum_{k=1}^{N}\frac{\frac{\Gamma(\alpha_j+x_{kj}+x_{\cdot j})\Gamma(\beta_j+1-x_{kj}+1-x_{\cdot j})}{\Gamma(\alpha_j+\beta_j+1+1)}}{\frac{\Gamma(\alpha_j+x_{kj})\Gamma(\beta_j+1-x_{kj})}{\Gamma(\alpha_j+\beta_j+1)}}\right)+\frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)}\frac{\Gamma(\alpha_j+x_{\cdot j})\Gamma(\beta_j+1-x_{\cdot j})}{\Gamma(\alpha_j)\Gamma(\beta_j)}}$$

Now rearranging terms and simplifying via the identity $\Gamma(x) = (x-1)\Gamma(x-1)$, we reduce to:

$$\text{rescore}(\mathbf{x}^{**}) = \frac{\frac{\Gamma(\alpha_j+\beta_j+M)}{\Gamma(\alpha_j+\beta_j+M+1)}\frac{\Gamma(\tilde{\alpha}_j+x_{\cdot j})\Gamma(\tilde{\beta}_j+1-x_{\cdot j})}{\Gamma(\tilde{\alpha}_j)\Gamma(\tilde{\beta}_j)}}{\left(\sum_{k=1}^{N}\frac{\Gamma(\alpha_j+\beta_j+1)}{\Gamma(\alpha_j+\beta_j+1+1)}\frac{\Gamma(\hat{\alpha}_{kj}+x_{\cdot j})\Gamma(\hat{\beta}_{kj}+1-x_{\cdot j})}{\Gamma(\hat{\alpha}_{kj})\Gamma(\hat{\beta}_{kj})}\right)+\frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j+\beta_j+1)}\frac{\Gamma(\alpha_j+x_{\cdot j})\Gamma(\beta_j+1-x_{\cdot j})}{\Gamma(\alpha_j)\Gamma(\beta_j)}}$$

$$= \frac{\left(\frac{1}{\alpha_j+\beta_j+M}\right)\frac{\Gamma(\tilde{\alpha}_j+x_{\cdot j})\Gamma(\tilde{\beta}_j+1-x_{\cdot j})}{\Gamma(\tilde{\alpha}_j)\Gamma(\tilde{\beta}_j)}}{\left(\frac{1}{\alpha_j+\beta_j+1}\right)\left(\sum_{k=1}^{N}\frac{\Gamma(\hat{\alpha}_{kj}+x_{\cdot j})\Gamma(\hat{\beta}_{kj}+1-x_{\cdot j})}{\Gamma(\hat{\alpha}_{kj})\Gamma(\hat{\beta}_{kj})}\right)+\left(\frac{1}{\alpha_j+\beta_j}\right)\frac{\Gamma(\alpha_j+x_{\cdot j})\Gamma(\beta_j+1-x_{\cdot j})}{\Gamma(\alpha_j)\Gamma(\beta_j)}}$$

where $\hat{\alpha}_{ij} = \alpha_j + x_{ij}$ and $\hat{\beta}_{ij} = \beta_j + 1 - x_{ij}$.

Now we again have two choices for $x_{\cdot j}$. Letting $x_{\cdot j} = 1$, we see:

$$\text{rescore}(\mathbf{x}^{**}) = \frac{\left(\frac{1}{\alpha_j+\beta_j+M}\right)\left(\frac{\Gamma(\tilde{\alpha}_j+1)}{\Gamma(\tilde{\alpha}_j)}\right)}{\left(\frac{1}{\alpha_j+\beta_j+1}\right)\left(\sum_{k=1}^{N}\frac{\Gamma(\hat{\alpha}_{kj}+1)}{\Gamma(\hat{\alpha}_{kj})}\right)+\left(\frac{1}{\alpha_j+\beta_j}\right)\left(\frac{\Gamma(\alpha_j+1)}{\Gamma(\alpha_j)}\right)}$$

$$= \frac{\left(\frac{1}{\alpha_j+\beta_j+M}\right)(\tilde{\alpha}_j)}{\left(\frac{1}{\alpha_j+\beta_j+1}\right)\left(\sum_{k=1}^{N}\hat{\alpha}_{kj}\right)+\left(\frac{1}{\alpha_j+\beta_j}\right)(\alpha_j)} \tag{B.12}$$

Similarly, when $x_{\cdot j} = 0$, we derive:

$$\text{rescore}(\mathbf{x}^{**}) = \frac{\left(\frac{1}{\alpha_j+\beta_j+M}\right)\left(\tilde{\beta}_j\right)}{\left(\frac{1}{\alpha_j+\beta_j+1}\right)\left(\sum_{k=1}^{N}\hat{\beta}_{kj}\right)+\left(\frac{1}{\alpha_j+\beta_j}\right)(\beta_j)} \tag{B.13}$$

Thus, putting these two cases together, rescore($\mathbf{x}^{**}$) becomes:

$$\prod_j \left[ \frac{\left(\frac{1}{\alpha_j+\beta_j+M}\right)(\tilde{\alpha}_j)}{\left(\frac{1}{\alpha_j+\beta_j+1}\right)\left(\sum_{k=1}^N \hat{\alpha}_{kj}\right) + \left(\frac{1}{\alpha_j+\beta_j}\right)(\alpha_j)} \right]^{x_{\cdot j}} \left[ \frac{\left(\frac{1}{\alpha_j+\beta_j+M}\right)\left(\tilde{\beta}_j\right)}{\left(\frac{1}{\alpha_j+\beta_j+1}\right)\left(\sum_{k=1}^N \hat{\beta}_{kj}\right) + \left(\frac{1}{\alpha_j+\beta_j}\right)(\beta_j)} \right]^{1-x_{\cdot j}}$$

To complete this section on efficiently computing rescore($\mathbf{x}^{**}$), we see the log of this score is *linear* in $\mathbf{x}$:

$$\begin{aligned}
\log(\text{rescore}(\mathbf{x}^{**})) = \sum_j & x_{\cdot j} \log\left[\left(\frac{1}{\alpha_j+\beta_j+M}\right)(\tilde{\alpha}_j)\right] \\
& - x_{\cdot j} \log\left[\left(\frac{1}{\alpha_j+\beta_j+1}\right)\left(\sum_{k=1}^N \hat{\alpha}_{kj}\right) + \left(\frac{1}{\alpha_j+\beta_j}\right)(\alpha_j)\right] \\
& + (1-x_{\cdot j}) \log\left[\left(\frac{1}{\alpha_j+\beta_j+M}\right)\left(\tilde{\beta}_j\right)\right] \\
& - (1-x_{\cdot j}) \log\left[\left(\frac{1}{\alpha_j+\beta_j+1}\right)\left(\sum_{k=1}^N \hat{\beta}_{kj}\right) + \left(\frac{1}{\alpha_j+\beta_j}\right)(\beta_j)\right]
\end{aligned}$$

$$\begin{aligned}
= \sum_j & x_{\cdot j} \log(\tilde{\alpha}_j) - x_{\cdot j} \log(\alpha_j+\beta_j+M) \\
& - x_{\cdot j} \log\left[\left(\frac{\sum_{k=1}^N \hat{\alpha}_{kj}}{\alpha_j+\beta_j+1}\right) + \left(\frac{\alpha_j}{\alpha_j+\beta_j}\right)\right] \\
& + \log(\tilde{\beta}_j) - \log(\alpha_j+\beta_j+M) \\
& - x_{\cdot j} \log(\tilde{\beta}_j) + x_{\cdot j} \log(\alpha_j+\beta_j+M) \\
& - \log\left[\left(\frac{\sum_{k=1}^N \hat{\beta}_{kj}}{\alpha_j+\beta_j+1}\right) + \left(\frac{\beta_j}{\alpha_j+\beta_j}\right)\right] \\
& + x_{\cdot j} \log\left[\left(\frac{\sum_{k=1}^N \hat{\beta}_{kj}}{\alpha_j+\beta_j+1}\right) + \left(\frac{\beta_j}{\alpha_j+\beta_j}\right)\right]
\end{aligned}$$

Finally, we have derived our very simple dot product:

$$\log \text{rescore}(\mathbf{x}^{**}) = c + \sum_j q_j x_{\cdot j} \tag{B.14}$$

where

$$c = \sum_j \log(\tilde{\beta}_j) - \log(\alpha_j + \beta_j + M) - \log\left[\left(\frac{\sum_{k=1}^N \hat{\beta}_{kj}}{\alpha_j + \beta_j + 1}\right) + \left(\frac{\beta_j}{\alpha_j + \beta_j}\right)\right]$$

and

$$\begin{aligned} q_j = &\log(\tilde{\alpha}_j) - \log(\tilde{\beta}_j) \\ &- \log\left[\left(\frac{\sum_{k=1}^N \hat{\alpha}_{kj}}{\alpha_j + \beta_j + 1}\right) + \left(\frac{\alpha_j}{\alpha_j + \beta_j}\right)\right] \\ &+ \log\left[\left(\frac{\sum_{k=1}^N \hat{\beta}_{kj}}{\alpha_j + \beta_j + 1}\right) + \left(\frac{\beta_j}{\alpha_j + \beta_j}\right)\right] \end{aligned}$$

We note that when using relevance feedback and all returned images are marked relevant (M=K, N=0), then these equations all simplify back to the original Bayesian Sets equations.