Binarized Mode Seeking for Scalable Visual Pattern Discovery

Wei Zhang[1], Xiaochun Cao[1,2*], Rui Wang[1], Yuanfang Guo[1], Zhineng Chen[3]
1: SKLOIS, Institute of Information Engineering, Chinese Academy of Sciences
2: School of Cyber Security, University of Chinese Academy of Sciences
3: Institute of Automation, Chinese Academy of Sciences
{wzhang, caoxiaochun, wangrui, guoyuanfang}@iie.ac.cn, zhineng.chen@ia.ac.cn

Abstract

This paper studies visual pattern discovery in large-scale image collections via binarized mode seeking, where images can only be represented as binary codes for efficient storage and computation. We address this problem from the perspective of binary space mode seeking. First, a binary mean shift (bMS) is proposed to discover frequent patterns via mode seeking directly in binary space. The binomial-based kernel and binary constraint are introduced for binarized analysis. Second, we further extend bMS to a more general form, namely contrastive binary mean shift (cbMS), which maximizes the contrastive density in binary space, for finding informative patterns that are both frequent and discriminative for the dataset. With the binarized algorithm and optimization, our methods demonstrate significant computation ($50\times$) and storage ($32\times$) improvement compared to standard techniques operating in Euclidean space, while the performance does not largely degenerate. Furthermore, cbMS discovers more informative patterns by suppressing low discriminative modes. We evaluate our methods on both annotated ILSVRC *(1M images) and unannotated blind* Flickr *(10M images) datasets with million scale images, which demonstrates both the scalability and effectiveness of our algorithms for discovering frequent and informative patterns in large scale collection.*

1. Introduction

Pattern discovery is one of the most fundamental problems in computer vision and pattern analysis. Given a large-scale un-ordered image collection (e.g., images crawled from an anonymous website), the first question is to ask *"What kind of images are in the dataset? What's the difference with other 'common' datasets?"* Visual pattern discovery aims to automatically find dominant items in an unsupervised setting. A lot of researchers have studied this problem due to its wide applications in various vision tasks such as classification [7], retrieval [27], summarization [32]. In the context of big data, visual pattern discovery is becoming even more important, as it provides an efficient way characterizing a large image collection. Particularly, this problem is even important as the data explodes in photo sharing websites, such as Instagram, Flickr, Imgur.

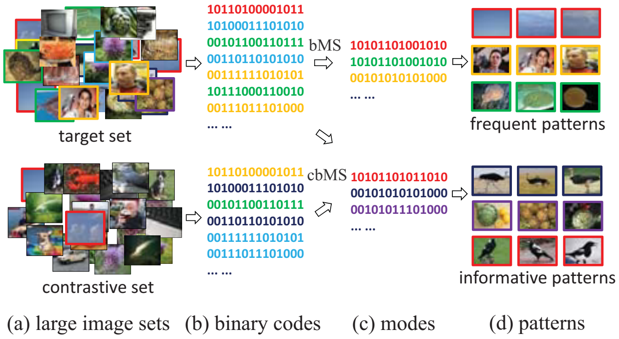

Figure 1. Given a large collection of images (a), our goal is to discover patterns (d) by seeking modes (c) in the binary space (b). Our *bMS* seeks frequent patterns from the target set, while *cbMS* discovers more informative patterns by referencing an additional contrastive set.

Among the techniques for pattern discovery, cluster analysis serves as the core part. By projecting data into the feature space, patterns are closely related to the high density regions. Previous techniques are mostly based on clustering directly in Euclidean space. However, there are still two fundamental problems seldomly addressed.

One problem is the scalability issue. It is quite inefficient to cluster large datasets in Euclidean space. First, large amount of real-value arithmetic operations is involved during distance evaluation, which makes it impossible for analyzing large dataset. Second, it is also difficult to keep all the data in memory for large datasets. For example, keeping one million real value vectors, say $R^{4,096}$ - FC7 activations of AlexNet[11], takes 30GB memory, not to mention the

[*] Corresponding author.

additional memory required for the mining algorithm. This is quite unaffordable for most computers. Therefore as data grows, it becomes infeasible for current approaches to analyze the data in terms of computational and storage costs.

The other problem lies in the informativeness of the discovered patterns. By $informative$, we indicate that the patterns need to be both frequent and discriminative. In other words, a pattern is considered informative, only if it is frequent in the target dataset, and rare in other contrastive datasets. In a large scale dataset, directly mining frequent items only gives patterns appearing everywhere (e.g., trees and skies), which are not always informative. Note that most images in the dataset are actually not that interesting, either because they are too rare to characterize the dataset, or too common in almost every dataset. It is worth noting that our informativeness is an analog to the notion of TF-IDF[1] used in information retrieval, where a word is weighted by both the frequency and discrimination.

This paper proposes two versions of binarized mode seeking algorithms for scalable visual pattern discovery (Figure 1). On one hand, the computation and storage bottlenecks are jointly addressed by binarized data representation as well as our binarized algorithm. On the other hand, we suppress less informative patterns via contrastive binary mode seeking. The modes of a probability distribution are the values where the probability density function (PDF) takes its (local) maximum values.

First, we propose a binarized mean shift algorithm addressing the scalability challenge by operating directly in binary space. Frequent patterns can be efficiently discovered as modes in binary space, where distances can be efficiently evaluated with "XOR" and "PopCnt" operations. Although hashing technique has been extensively studied [29, 28, 22] for generating binary codes, studies on binarized data analysis are still quite limited. In this paper, we propose to analyze the data in binary space. Concretely, a binary mean shift is developed by enforcing the binary constraint and exploring the suitable kernels. Second, we propose to suppress less informative patterns by mining on the contrastive density. For a natural image dataset, the dominating modes usually correspond to very common items (e.g., face, sky, tree) that are not that informative. We further extend the binary mean shift to operate on the contrastive density, i.e., the density ratio between the target dataset (as positive or foreground distribution) and another contrastive dataset (as negative or background distribution). By iteratively maximizing the density ratio, patterns that best discriminate the target set from contrastive set can be discovered.

For evaluation, we test our methods on the annotated ILSVRC dataset [21], including one million images, as well as another unlabeled Flickr dataset, including over 10 million images crawled from Flickr website. Compared to existing methods, our methods find frequent and informative patterns with much faster speed and much smaller memory cost, by operating in the binary space and contrastive density. To the best of our knowledge, this is the pilot work studying the binarized pattern mining technique. Our main contributions are summarized as follows:

- We propose a binarized mode seeking algorithm (*bMS*) operating in binary space, where binomial-based kernel and binary constraint are explored for binary space.

- We further extend to contrastive version (*cbMS*) for discovering more informative patterns, and show that *bMS* can be generalized to contrastive density by optimizing with a non-uniform background density.

- Experiments show that our algorithms run much faster (50×) with less memory cost (32×), and does not degenerate too much in performance.

2. Related Work

Our work is closely related to both visual pattern discovery and binary space data analysis. In this section, we review relevant literatures based on this categorization.

Visual pattern discovery. Visual pattern discovery has been studied by many researchers in multimedia and computer vision communities.

Early studies on Common Pattern Discovery [13, 25, 30] model this as an optimization problem. These methods are computationally expensive, thus can only operate on small scale dataset (up to few hundreds images). Frequent Itemset Mining [31, 18] mostly work on small scale dataset (∼10k images), despite a few works [14, 20] optimize the scalability for large dataset. These methods are effective in pattern discovery, but slow in the support-counting step.

Methods for large scale visual pattern discovery can be roughly grouped into two streams. The first stream is based on clustering. The most popular technique, k-means, has two inherent limitations: the clusters are constrained to be spherically symmetric and their number has to be known a priori [8]. Sivic [24] directly clusters small patches extracted around local features, such that frequent patterns can be grouped together. Philbin [17] proposes to discover patterns via Spectral Clustering [15] on a matching graph, which is constructed by searching every image in turn against the whole dataset. As a result, dense sub-graphs are considered as patterns. As for the speed, it takes around two hours for clustering a 37k dataset. Chum [3] mines visually similar photos by clustering images with Min-Hash algorithm, where hash-key collisions are extracted as patterns. It requires large number of hash tables for a decent collision probability.

[1] TF-IDF: Term Frequency-Inverted Document Frequency

The second stream is based on mode seeking, which does not require prior knowledge of the number of clusters, or constrain the shape of clusters. One typical method is by [7], which mines mid-level visual elements by discriminative mode seeking in Euclidean space.

Binarized data analysis. Although there have been many hashing techniques for data binarization, only a few works directly analyze data in binary space to fully explore the efficient nature of binary representation.

Fast exact nearest neighbor search in binary space [16] improves retrieval efficiency by building multiple hash tables on binary code substrings. Recent work on BKmeans [10] performs fast clustering by adapting K-Means into binary space. Though much effective, still it needs to process all the data before generating any patterns. It is worth noting that clustering-based method needs to process all the data before generating any patterns, while mode seeking has the advantage of partially generating patterns in limited time budget.

Deep learning based techniques have been showing promising results on various tasks. Usually it requires considerable computation resources for good performance. Interestingly, recent advancements in binarized deep architecture [12, 6, 5] leverage the binary operations and connections in deep neural networks, which demonstrate the efficiency and effectiveness of data binarization in visual recognition task. A more recent work [19] exploring the "XNOR" operation has shown similar results on ImageNet classification task compared to the full-precision AlexNet, but with 58× speedup and 32× memory savings. These promising results have demonstrated great potential of binarized analysis in multimedia content analysis.

As a comparison, our method efficiently discovers patterns via mode-seeking in the binary space, i.e., binarized mode seeking technique that targets for large dataset.

3. Binarized Visual Pattern Discovery

3.1. Data Binarization

Theoretically, any hashing techniques can be adopted in our algorithm. Our aim is to best approximate the data with binary codes, such that the following process could benefit from the binarization with minimum precision loss. Thus we adopt Iterative Quantization (ITQ) [9], since it minimizes loss-of-precision while quantizing real-value vectors V to binary codes B by rotating the data with R:

$$Q(B,R) = \|B - VR\|_F^2, \tag{1}$$

which can be optimized by alternatively updating B and R. Details on this binarization technique can be found in [9].

Performing mean shift on real-value vectors is trivial. However, in practice, we may not be able to keep large number of real-valued vectors in memory. The only affordable representation is the binary hash keys. To perform mean shift, one can construct a hash-table from compact hash keys to real-value vectors, as done in [10]. During optimization, the real-value vectors can be efficiently retrieved in $O(1)$ for numeric calculation. In such case, this method shares the same result and similar running speed as standard mean shift.

3.2. Binary Mean Shift: *bMS*

Given a set of real-value vectors X $= \{x_1, x_2, ..., x_n\}$, manifesting an underlying probability density function $p(x)$ in $\mathbb{R}^d$, the density can be estimated by:

$$p(x) \propto \sum_{i=1}^{n} K(\|\frac{x - x_i}{h_i}\|^2), \tag{2}$$

where h_i is the adaptive bandwidth associated with x_i's nearest neighbours [8], and $K(x)$ is a kernel function (e.g., Gaussian). Mean shift locates the modes by maximizing $p(x)$. However due to time efficiency, standard mean shift is not applicable for large datasets by operating in Euclidean space. We propose a much efficient binary mean shift algorithm which directly locates modes in the binary space.

Let B $= \{b_1, b_2, ..., b_n\}$ be the corresponding binary code of X embedded in the k-dimensional binary space $\{-1,1\}^k$, where b_i is generated using ITQ.

In binary space, the commonly adopted continuous kernels (e.g., Gaussian kernel) is not applicable. First, kernel designed for continuous space becomes less suitable for discrete binary space. Second, distance between binary codes has a clear interpretation, i.e., number of inconsistent bits, which also requires a kernel that fits this interpretation.

Without any prior knowledge, the binary codes are assumed to be uniformly distributed in $\{-1,1\}^k$. With this assumption, the Hamming distance between two random hash-codes follows the Binomial distribution $Bin(k, 1/2)$. We propose a kernel K_b to weight binary codes with different Hamming distances:

$$K_b(d) = -\log_2\left(\frac{1}{2^k}\sum_{i=0}^{d}\binom{k}{i}\right)/z, \tag{3}$$

where z is the normalization factor to ensure K_b a valid kernel, and the part in parentheses amounts to the cumulative distribution function CDF(d) of $Bin(k, 1/2)$. In this way, the kernel $K_b(d)$ has a clear interpretation: the probability of two random binary codes with their Hamming distance less than or equal to d. Since there are only $k+1$ finite values for K_b, it is implemented as a look-up table in our case. Moreover, our method has a clear advantage in running speed compared to standard mean shift.

Considering this binomial-based kernel as well as the bi-

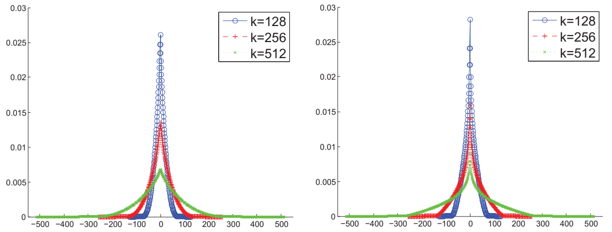

Figure 2. The binomial based kernel K_b (left) and the corresponding kernel G_b (right). Please note that all kernels are discrete and we line up the discrete points only for better visualization. This figure is best viewed in color.

Algorithm 1 - bMS: binary mean shift

```
Input: $B$: binary codes of the dataset; $b$: initial location;
    $h$: bandwidth; $r$: radius search threshold; $T$: maximum
    iterations;
Output: mode: $b^*$
Build a multi-index hash table $M$ on $B$;
for $iter$ = 1 ... $T$ do
    $\mathcal{N}(b)$ = $M$.r-search($b$, $r$);
    Update $\hat{b}$ using Eq. (6);
    if not converge then
        $b \leftarrow \hat{b}$;
    else
        break;
    end if
end for
$b^*$ = $b$;
```

nary constraint, we formulate the problem as:

$$b^* = \arg\max_b \sum_{i=1}^{n} K_b(\|\frac{b-b_i}{2h_i}\|^2) \\ \text{s.t. } b, b_i \in \{-1,+1\}^k, \tag{4}$$

where $\|\cdot\|$ denotes the $l2$-norm. With the binary constraints on b, we only shift the estimation among vertices of the Hamming hypercube.

For optimization, we iteratively update the estimation with mean shift. As pointed out in [2], mean shift on kernel $G_b = -K_b'$ is equivalent to gradient ascent on the density estimated with its shadow kernel K_b, where K_b' denotes the derivative of K_b. Figure 2 shows several example kernel profiles for K_b (left) and G_b (right) with different number of bits k. Please note that the kernel derived for binary space is quite different from the widely adopted Gaussian kernel for Euclidean space. The shape for G_b is binomial-like. Most energy is located in the area with small Hamming distances, and the peak is more sharper than Gaussian function. In our case, G_b is also implemented as a look-up table to facilitate fast computation. Although the differentiation is analytically complicated, the numeric computation is quite straightforward as long as we got K_b computed. As a result, performing mean shift with the binary constraint is to update the estimation $\hat{b}$ via:

$$\hat{b} = sgn\left(\frac{\sum_{i=1}^{n}\frac{b_i}{2h_i^2}G_b(\|\frac{b-b_i}{2h_i}\|^2)}{\sum_{i=1}^{n}\frac{1}{2h_i^2}G_b(\|\frac{b-b_i}{2h_i}\|^2)}\right), \tag{5}$$

where $sgn(\cdot)$ is the signum function for binarization.

There are two observations to further speed up the computation. First, although Eq. (5) takes the summation over the whole set of points, it is quite inefficient and unnecessary, since most of the points makes no contribution to the summation. In our implementation, we only consider the points in the neighborhood of b, i.e., points with their Hamming distances less than or equal to a threshold r. For efficient r-radius search in the Hamming ball, we adopt the Multi-Index Hashing [16] to index and search in binary space. From Figure 2, $k/4 \sim k/3$ is a reasonable range for r. This strategy significantly improves the running speed. Second, noticing that the kernel K_b as well as G_b are strictly positive, the denominator can be ignored since it is strictly positive. Therefore, the estimation can be further simplified as:

$$\hat{b} = sgn\left(\sum_{b_i \in \mathcal{N}(b)} \frac{b_i}{2h_i^2} G_b(\|\frac{b-b_i}{2h_i}\|^2)\right), \tag{6}$$

where $\mathcal{N}(b)$ denotes the neighborhood of b. As a result, our method is highly efficient in terms of computation. This algorithm (*bMS*) is summarized in Algorithm 1.

3.3. Contrastive Binary Mean Shift: *cbMS*

As data grows, modes in the feature space often correspond to common patterns (e.g., sky, people) appearing everywhere. For a dataset with a large number of categories that may overlap with each other, taking the modes directly as the patterns is not that informative, e.g., the sky and face patterns in Figure 1 (top-right). We further extend binary mean shift to find discriminative patterns, by contrasting against the background distribution defined by another set.

We take the target dataset as the positive set, and introduce another background dataset as the negative set for contrast. The density ratio between positive and negative sets makes more sense, since it suppresses the modes that also exist in the background sets. Formally, in binary space, some points are taken as the foreground (positive set P), and others are regarded as the background (negative set N). Inspired by [7], we can formulate the problem on contrastive density ratio $\hat{p}(b) = \frac{p_+(b)}{p_-(b)}$:

$$b_c^* = \arg\max_b \hat{p}(b) = \frac{\sum_{b_i \in P} K_b(\|\frac{b-b_i}{2h_i}\|^2)}{\sum_{b_j \in N} K_b(\|\frac{b-b_j}{2h_i}\|^2) + \lambda}, \\ \text{s.t. } b, b_i, b_j \in \{-1,+1\}^k \tag{7}$$

where the modes are estimated with the contrastive density between p_+ and p_-, and λ is an offset to avoid division-by-zero. Therefore, a pattern has to be frequent in the positive set, and rare in the negative set at the same time. Similarly, by maximizing $\hat{p}(b)$, the new mode can be estimated via:

$$\hat{b_c} = sgn\left(\frac{f_{\mathcal{N}_b,G_b} \cdot l_{\mathcal{P}_b,K_b} - f_{\mathcal{P}_b,G_b} \cdot l_{\mathcal{N}_b,K_b} - \lambda f_{\mathcal{P}_b,G_b}}{l_{\mathcal{P}_b,K_b} \cdot l_{\mathcal{N}_b,G_b} - l_{\mathcal{N}_b,K_b} \cdot l_{\mathcal{P}_b,G_b} - \lambda l_{\mathcal{P}_b,G_b}}\right), \tag{8}$$

where

$$\begin{aligned} f_{\mathcal{S},H} &= \sum_{b_i \in \mathcal{S}} \frac{b_i}{2h_i^2} H(\|\frac{b-b_i}{2h_i}\|^2) \\ l_{\mathcal{S},H} &= \sum_{b_i \in \mathcal{S}} H(\|\frac{b-b_i}{2h_i}\|^2), \end{aligned} \tag{9}$$

and $\mathcal{S} \in \{\mathcal{P}_b, \mathcal{N}_b\}$, $H \in \{K_b, G_b\}$, where $\mathcal{P}_b$, $\mathcal{N}_b$ denote the neighborhood of b in set P, N respectively.

Discussion. It is worth noting that Eq. (8) can be regarded as the generalized version of Eq. (5), and bMS is actually a special case of $cbMS$. Assuming $p_-(x)$ a uniform distribution, it is easy to show that Eq. (8) degenerates to (5) with the following equations hold:

$$\begin{aligned} &\sum_{b_i \in N} K_b(b_i) = 1/|N|, \\ &\sum_{b_i \in N} b_i G_b(b_i) = \sum_{b_i \in N} G_b(b_i) = 0, \end{aligned} \tag{10}$$

where $|N|$ is the cardinality of the set N, and $b_i \in N$ is uniformly distributed.

It is worth noting that a fundamental difference between *cbMS* and [7] is that our method is *binarized analysis* designed for large dataset, and the optimization is quite different. [7] operates on continuous space and targets for relative small data.

Optimization detail. In practice, optimizing Eq. (7) with (8) tends to be unstable when $p_-(b)$ shrinks to zero. Gradients are peaky around such locations and the optimization of Eq. (7) becomes difficult and unstable. Please note that the offset parameter λ also plays an important role in smoothing the estimation by adding a constant offset on the background density, such that the binarization via $sgn(\cdot)$ will not drastically drift $\hat{b_c}$ too far away. In extreme case where λ goes to very large number, the problem of *cbMS* (Eq. (7)) degenerates to *bMS* (Eq. (4)).

Though the computation in Eq. (8) involves both the kernel and its shadow, still it can be efficiently evaluated with two separate loop-up tables for K_b and G_b. Note that the binary codes for positive and negative sets are jointly indexed in one Multi-index Hash Table, but different neighborhoods in Eq. 8 are separately evaluated. This algorithm (*cbMS*) is summarized in Algorithm 2.

Algorithm 2 - cbMS: contrastive binary mean shift

Input: P: binary codes for the target dataset; N: binary codes for the contrastive dataset; b: initialization; h: bandwidth; T: maximum iterations; λ: smoothing term;

Output: mode: b_c^*

```
Build a multi-index hash table $M$ for $\{P \cup N\}$;
for $iter$ = 1 ... $T$ do
    [$\mathcal{P}_b$, $\mathcal{N}_b$] = $M$.r-search($b$, $r$);
    Update $\hat{b_c}$ using Eq. (8);
    if not converge then
        $b \leftarrow \hat{b_c}$;
    else
        break;
    end if
end for
$b_c^*$ = $b$;
```

4. Experiments

Benchmarking mode seeking algorithms is quite difficult. Due to the lack of large scale annotated dataset, quantitatively evaluating our algorithms becomes quite difficult. In this section, we first evaluate bMS and $cbMS$ in terms of purity-coverage. Then we analyze the results on a large-scale real-world Flickr dataset.

4.1. Datasets

We adopt ILSVRC because it is the largest *fully-annotated dataset* publicly available, such that quantitative evaluation can be conducted. On the other hand, Flickr10M is adopted to test our algorithm on real-world large-scale un-ordered dataset.

ILSVRC [21]. A large number of images crawled from the Internet are manually organized with the WordNet hierarchy. This dataset contains 1,000 object categories and 1.28 million images in total. For detailed evaluation, we sample three subsets by randomly picking 200, 500 and 1,000 object categories from the 1,000 categories, namely ILSVRC-200, ILSVRC-500, and ILSVRC-1000, respectively. Note that the ILSVRC-1000 corresponds to the full set of ILSVRC.

Flickr10M. This dataset contains 10 million images crawled from Flickr. Though randomly constructed, Flickr10M still contains lots of visual patterns due to the effect of big data, e.g., popular objects/landmarks shared by different users, consistent images uploaded in the same geo-location. Note that it is impossible for us to annotate this dataset, and this set serves as a "blind" dataset to test the applicability of our algorithms against real-world un-ordered Web data.

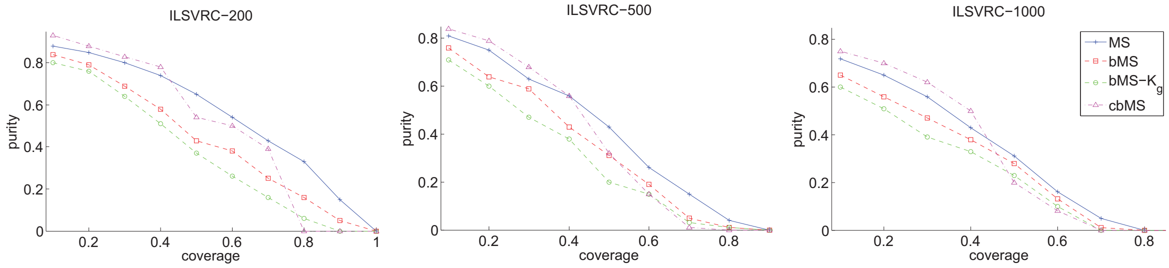

Figure 3. Comparison between different methods with the purity-coverage plot. Left: the subset with 200 categories. Middle: the subset with 500 categories. Right: the full set with 1,000 categories.

4.2. Experimental Setup

We use the deep features of VGGNet-19 [23] fc7 activations as the image features. Iterative Quantization [9] is applied to quantize features into binary codes, which are the inputs for all methods (except mean shift operating in Euclidean space). We follow the evaluation protocol by [7], Purity-Coverage plot. High quality patterns correspond to dense regions in feature space, where the supporting images locating in its neighborhood share the same category label. For a discovered pattern (or equivalently a mode in the feature space), we pool different sets of supporting images by varying the (Hamming) distances to the mode. Purity is defined as the percentage of true images in the set, and coverage is the fraction of true images included in the set. As an analog to the Precision-Recall curve, there is a also trade-off between purity and coverage.

To the best of our knowledge, this paper is the pioneer work on mode seeking / pattern discovery with binarized analysis, where the data is difficult to fit into the memory. Thus most methods operating in Euclidean space are not directly comparable. We include the standard Mean Shift operating on full precision data for evaluating the precision loss in our binarized algorithms. In this paper, the following methods are included for comparison:

- Mean Shift (*MS*) is the standard method operated in Euclidean space [4].
- Binary Mean Shift (*bMS*) corresponds to the method (with binomial-based kernel K_b) in Algorithm 1.
- Binary Mean Shift with Gaussian kernel (*bMS-K_g*) corresponds to the method in Algorithm 1. Different from bMS, bMS-K_g adopts a Gaussian-based kernel.
- Contrastive Binary Mean Shift (*cbMS*) is the extension of *bMS* on contrastive density as described in Algorithm 2.

4.3. Performance Comparison

For fair comparison, we initialize all the methods with the same set of initial positions, which contains 200 / 500 / 1,000 random locations for ILSVRC-200 / 500 / 1000. Upon convergence, the modes are extracted as patterns and the supporting images can be drawn by retrieving the neighborhoods.

Figure 3 compares different methods on each subset of ILSVRC, in terms of purity-coverage plot. As shown, it is slightly easier to find patterns in the small subset with only 200 categories. As more images are embedded in the feature space, modes could be easily perturbed by noise images from other categories.

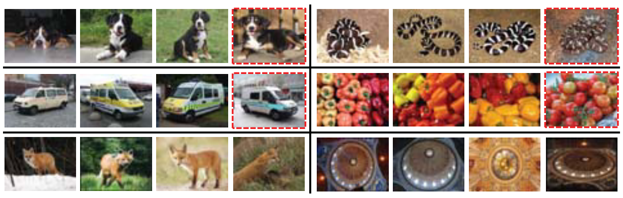

Figure 4. Example patterns discovered by bMS in ILSVRC. Images with red-dashed borders are false results from categories other than the dominating pattern.

As expected, MS works best by operating in the original Euclidean space, since other methods suffers from the loss-of-precision during data binarization as well as the binary constraint. In general, our method bMS works reasonably well, demonstrating slightly worse performance compared to MS. However, as we will show later in Section 4.5, bMS runs much faster than MS. Example patterns of bMS can be found in Figure 4. By operating in binary space, bMS does not show too much performance degeneration. First, data binarization via ITQ approximately preserves the density by minimizing the quantization error, although the neighborhood does not necessarily hold. Second, in mode seeking step, the binomial kernel K_b is effective to model the kernel for binary space, which can be confirmed by comparing bMS and bMS-K_g. The Gaussian kernel, however, does not fit well for binary codes. For $cbMS$, we take one category in turn as the positive set, and treat other categories as the negative set. Our $cbMS$ shows clear advantage in high purity areas (see top half of each figure). That is, $cbMS$ tends to locate the modes in the area with low negative density.

Informative Pattern Discovery Practically, both frequency and discriminability account for a "good" pattern for real users. We are more interested in the patterns telling things apart from each other.

Figure 5 shows example patterns discovered by bMS and $cbMS$ from the ILSVRC dataset. For bMS, we discovery patterns for each category. For $cbMS$, we take one category as the positive set and the rest as negative sets. Based on our observation, it is easier to discover patterns from categories with consistent visual appearances. In practice, some categories are with multiple patterns, and modes from different categories might overlap with each other. For example in the first row, the "space bar" has a pattern similar with a "typewriter keyboard", as well as another pattern with closed-up view space-bar. With patterns by *bMS*, it remains difficult to tell one category from another, since modes from "space bar" is quite similar with the mode from "typewriter keyboard". However, $cbMS$ addresses this problem by the contrastive density. As shown in the last column, modes discovered by $cbMS$ are more informative to characterize the category. Another typical example is as the last row. The pattern on the left is with the fish and a person inside. However, persons are rather "common" patterns throughout the whole dataset (middle). Therefore, the pattern by *bMS* on the left is suppressed and a more discriminative pattern (right) is discovered by *cbMS*.

4.4. Results on Flickr10M Dataset

We further conduct experiments on the un-annotated Flickr10M dataset to discover patterns in the "blind" setting, i.e., no prior knowledge on the dataset. Figure 6 shows several patterns discovered on this dataset by *bMS* (left) and *cbMS* (right). As shown on the left, most popular patterns in a real-world dataset are not that interesting, similar to the stop-words in information retrieval (e.g., is, are, of). Real-world data is mostly noisy and biased. As expected, our method *bMS* characterizes Flickr10M with such patterns, e.g., default place-holder image (1st row), skys (2nd row), uni-color photo (3rd row) and people (last rows). If we take a close look, most patterns exist due to certain reasons. For example, the place-holder pattern is due to dead-links.

We further take the top patterns from *bMS* as the negative set N, and run *cbMS* on the dataset again to obtain the top patterns as shown on the right. This time, the resulting patterns characterizes the Flickr10M dataset by showing more informative patterns, e.g., pets, planes, landmarks that are popularly shared among Flickr users.

4.5. Discussion

The experiments were conducted on a PC with 3.30GHz CPU and 8GB memory, except that the full precision mean shift (*MS*) is on a server machine with 64GB memory.

Table 1. Average running time (in seconds) for finding one mode in ILSVRC dataset (with one million images).

Methods	MS	bMS	bMS-K_g	$cbMS$
Time	9.78	0.17	0.17	0.21

Space complexity of our methods is $O(n)$, where n is the total number of images. In practice, keeping the same feature dimension, converting real-value data into binary codes gives 64× (double precision to bit) or 32× (single precision to bit) memory savings.

Time complexity of our algorithms is of $O(Tn^2)$, where T is the number of iterations. Table 1 summarizes the average running time for mining a pattern from one million images. Algorithms operating directly in binary space run significantly faster, i.e., 10 versus 0.2 seconds, benefiting from efficient "XOR" and "PopCnt" operations. In our experiment, bMS and bMS-K_g show similar running time, since both of them benefit from the discrete kernels implemented with look-up tables. On the contrary, MS needs to online evaluate the kernels in continuous space. Our $cbMS$ is slightly slower than bMS, by introducing extra computations in the optimization (Eq. (8)).

Theoretically, for modern CPU, one can perform 64 binary operations in a single CPU clock, and thus the speedup is around 64× faster [19]. Practically, we observe around 50× speedup in the experiment (Table 1).

It is worth noting that, in terms of speed efficiency, mode seeking based methods are more flexible in running time. Even with limited time budget (one second for example), it could still return some interesting patterns from the large dataset, since it does not need to traverse all the data to find a single mode. With more time budget, it could gradually find more patterns as demanded. On the contrary, clustering based methods need to traverse all the data before generating any patterns, making it less suitable for real application.

Convergence analysis Although the convergence proof for *bMS* and *cbMS* is difficult[2], we provide some empirical analysis on the convergence property. Based on the experiments, our methods usually take 5∼15 iterations to converge. In particular, bMS always converges in 10 iterations, and cbMS generally takes more iterations in practice. The offset term (λ in Eq.7) helps a lot on the converge speed. Due to the discrete optimization (especially the sgn($\cdot$) operation), it is difficult to guarantee the convergence theoretically. In our case, each iteration through Eq. 6, 8 updates the estimation to a point with higher density.

[2] Note that a rigorous convergence proof for standard Mean-Shift in continuous space is still missing [1]

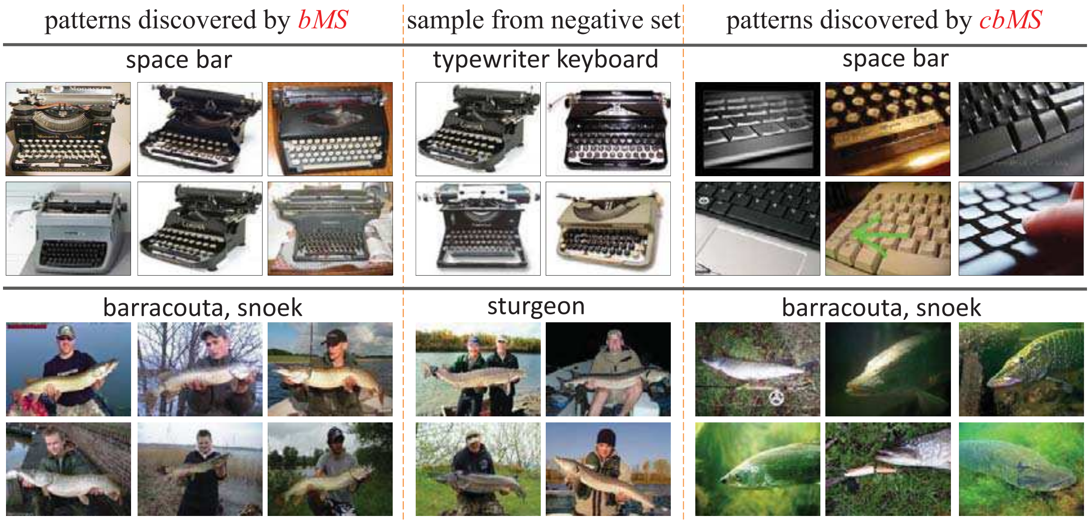

Figure 5. Comparison of patterns discovered by bMS (left) and $cbMS$ (right) in ILSVRC. Middle: sample images from other categories that are visually similar to the left column.

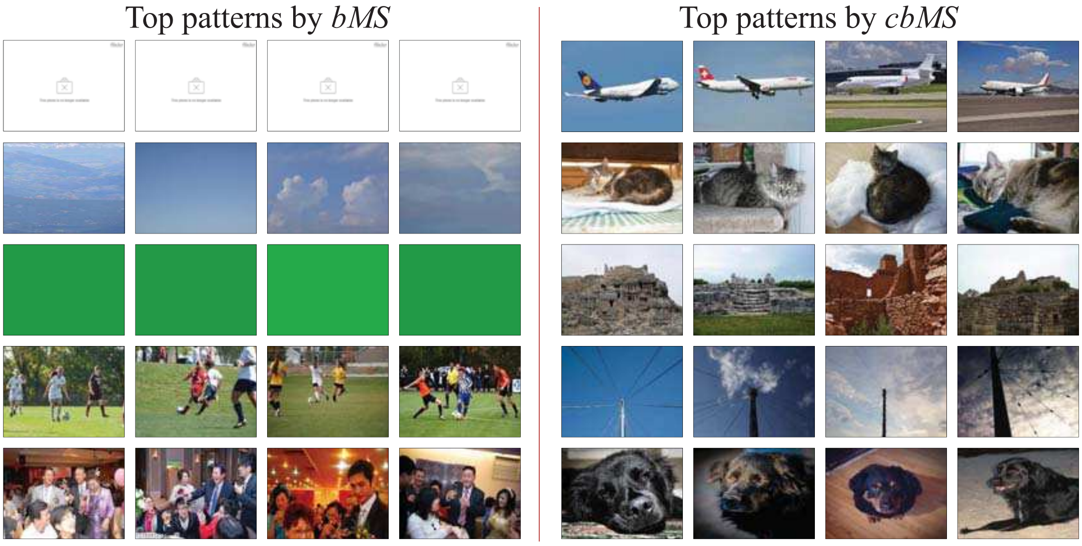

Figure 6. Top-5 patterns discovered by *bMS* (left) and *cbMS* (right) in the un-ordered Flickr10M dataset. Each row lists the supporting images of a discovered mode, ranked (top to bottom) based on the mode density. *bMS* tends to find common patterns shared by many images, while *cbMS* discovers more informative patterns characterizing the dataset.

5. Conclusion

We have presented a technique addressing large scale visual pattern discovery via mode seeking in binary space, by exploring binary constraint and binomial kernel. Our method is much more scalable in terms of computation and storage than standard techniques. By further extending the algorithm on contrastive density, we are able to discover more informative patterns from the dataset. For now, our experiments were conducted only on full images for pattern discovery. A future direction is to extend for object level mining, i.e., finding representative parts of an object.

6. Acknowledgement

This work was supported by National Key Research and Development Plan (No.2016YFB0800603), National Natural Science Foundation of China (No. 61602463, No. 61422213, No. 61602344), Beijing Natural Science Foundation (4172068), Key Program of Chinese Academy of Sciences (No. QYZDB-SSW-JSC003).

References

[1] Y. Aliyari Ghassabeh. A sufficient condition for the convergence of the mean shift algorithm with Gaussian kernel. *Journal of Multivariate Analysis*, 135:1–10, 2015. 7

[2] Y. Cheng. Mean shift, mode seeking, and clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 790–799, 1995. 4

[3] O. Chum and J. Matas. Large-scale discovery of spatially related images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32:371–377, 2010. 2

[4] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(5):603–619, May 2002. 6

[5] M. Courbariaux and Y. Bengio. Binarynet: Training deep neural networks with weights and activations constrained to +1 or -1. *CoRR*, abs/1602.02830, 2016. 3

[6] M. Courbariaux, Y. Bengio, and J. David. Binaryconnect: Training deep neural networks with binary weights during propagations. *NIPS*, 2015. 3

[7] C. Doersch, A. Gupta, and A. A. Efros. Mid-level visual element discovery as discriminative mode seeking. In *Proc. NIPS*, pages 494–502, 2013. 1, 3, 4, 5, 6

[8] B. Georgescu, I. Shimshoni, and P. Meer. Mean shift based clustering in high dimensions: A texture classification example. In *ICCV*, pages 456–463. IEEE Computer Society, 2003. 2, 3

[9] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *Proc. CVPR*, pages 817–824, 2011. 3, 6

[10] Y. Gong, M. Pawlowski, F. Yang, L. Brandy, L. Bourdev, and R. Fergus. Web scale photo hash clustering on a single machine. In *Proc. CVPR*, 2015. 3

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. NIPS*, pages 1097–1105, 2012. 1

[12] Z. Lin, M. Courbariaux, R. Memisevic, and Y. Bengio. Neural networks with few multiplications. *ICLR*, 2016. 3

[13] H. Liu and S. Yan. Common visual pattern discovery via spatially coherent correspondences. In *Proc. CVPR*, pages 1609–1616, 2010. 2

[14] S. Moens, E. Aksehirli, and B. Goethals. Frequent itemset mining for big data. In *International Conference on Big Data*, pages 111–118, 2013. 2

[15] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Proc. NIPS*, pages 849–856. MIT Press, 2001. 2

[16] M. Norouzi, A. Punjani, and D. J. Fleet. Fast exact search in hamming space with multi-index hashing. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 1107–1119, 2014. 3, 4

[17] J. Philbin and A. Zisserman. Object mining using a matching graph on very large image collections. In *Proc. ICVGIP*, 2008. 2

[18] T. Quack, V. Ferrari, and L. V. Gool. Video mining with frequent itemset configurations. In *Proc. CIVR*, pages 360–369, 2006. 2

[19] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. Xnornet: Imagenet classification using binary convolutional neural networks. *CoRR*, abs/1603.05279v1, 2016. 3, 7

[20] Z. Rong and J. D. Knijf. Direct out-of-memory distributed parallel frequent pattern mining. In *International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications*, pages 55–62, 2013. 2

[21] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, pages 1–42, April 2015. 2, 5

[22] F. Shen, C. Shen, W. Liu, and H. Tao Shen. Supervised discrete hashing. In *CVPR*, June 2015. 2

[23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015. 6

[24] J. Sivic and A. Zisserman. Video data mining using configurations of viewpoint invariant regions. In *Proc. CVPR*, 2004. 2

[25] H.-K. Tan and C.-W. Ngo. Localized matching using earth mover's distance towards discovery of common patterns from small image samples. *Image Vision Computing*, 27(10), 2009. 2

[26] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L. Li. YFCC100M: the new data in multimedia research. *Commun. ACM*, 59(2):64–73, 2016.

[27] W. Voravuthikunchai, B. Crémilleux, and F. Jurie. Image reranking based on statistics of frequent patterns. In *International Conference on Multimedia Retrieval*, page 129, 2014. 1

[28] J. Wang, W. Liu, A. X. Sun, and Y.-G. Jiang. Learning hash codes with listwise supervision. In *Proc. ICCV*, 2013. 2

[29] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Advances in Neural Information Processing Systems*, pages 1753–1760, 2008. 2

[30] J. Yuan and Y. Wu. Spatial random partition for common visual pattern discovery. In *Proc. ICCV*, 2007. 2

[31] M. J. Zaki. Scalable algorithms for association mining. *IEEE Trans. on KDE*, pages 372–390, 2000. 2

[32] W. Zhang, H. Li, C.-W. Ngo, and S.-F. Chang. Scalable visual instance mining with threads of features. In *ACM Multimedia*, pages 297–306, 2014. 1