The PARIS Algorithm for Determining Latent Topics

Conference Paper · December 2010

DOI: 10.1109/ICDMW.2010.187 · Source: DBLP

5 authors, including:

Ira Cohen
HP Inc.

Ron Banner
HP Inc.

2010 IEEE International Conference on Data Mining Workshops

The PARIS Algorithm for Determining Latent Topics

Michal Aharon, Ira Cohen, Arik Itskovitch, Inbal Marhaim and Ron Banner
Hewlett-Packard Israel Labs
Email: michal.aharon / ira.cohen @hp.com

***Abstract*—We introduce a new method for discovering latent topics in sets of objects, such as documents. Our method, which we call PARIS (for Principal Atoms Recognition In Sets), aims to detect principal sets of elements, representing latent topics in the data, that tend to appear frequently together. These latent topics, which we refer to as 'atoms', are used as the basis for clustering, classification, collaborative filtering, and more. We develop a target function which balances compression and low error of representation, and the algorithm which minimizes the function. Optimization of the target function enables an automatic discovery of the number of atoms, representing the dimensionality of the data, and the atoms themselves, all in a single iterative procedure. We demonstrate PARIS's ability to discover latent topics, even when those are arranged hierarchically, on synthetic, documents and movie ranking data, showing improved performance compared to existing algorithms, such as LDA, on text analysis and collaborative filtering tasks.**

***Keywords*-machine learning, automatic discovery, keyword extraction, document clustering, set representation, topic extraction, latent concepts**

I. Introduction

In many problems we face the need to characterize objects according to the elements that make them up. Examples can be found in various fields, e.g., document characterization aims to describe documents in a corpus according to the topics they discuss, represented by their words. In a different scenario, software event stream analysis aims at discovering sequences of events that describe different states in a system running complex applications.

In this work we introduce a new approach for solving a characterization problem in sets. We identify a dictionary of *atoms* in the input data sets, where an *atom* is a principal set of elements that is common in many of the input sets contained in a data, and therefore is potentially "meaningful" in some sense. The number of atoms, which can be interpreted as a measure of dimensionality of the data is identified automatically from the data. For this purpose, we introduce a new data generation model and derive a corresponding data representation and cost function. We then design a cost function that balances between data representation error and compression. We design the PARIS algorithm to minimize this cost function. PARIS automatically discovers the number of atoms for the given data, and computes the data representation using these atoms in a single iterative procedure.

Identifying principal atoms in a set of given objects may be useful for several purposes. It can be exploited for analysis, clustering, collaborative filtering, set characterization, compression and noise filtering in sets. We demonstrate PARIS in this paper on both synthetic data and for the tasks of text characterization, clustering and collaborative filtering. We compare our algorithm to Latent Dirichlet Allocation [5]. We demonstrate how it is able to recover latent topics accurately, even when there is an underlying hierarchical structure in the topics.

Existing approaches to solve similar problems have been previously introduced in the literature in various fields. In document characterization works such as Latent Dirichlet Allocation [5] and probabilistic latent semantic indexing [12] produce models that capture latent topics in documents using a corpus of training documents and different finite mixture models. [10] further expand the work to infinite mixture models. Hierarchical LDA [4] represents latent topics in an hierarchy, representing each document as the latent topics along a branch in the hierarchical topic tree. In all of the existing approaches, a compressed representation of the data is learned from data through probability distributions over words and topics.

A different problem, although with similar aspects, is found in the field of signal and image processing. The *Method of Optimal Directions* [9] and the K-SVD algorithm [2] solve the problem of over-complete dictionary extraction for sparse linear representation of real valued signals such as images. There, a dictionary that promises sparse linear representation of all data elements leads to various applications such as denoising [8], [15] and compression [6]. The algorithm we introduce here actually leverage ideas of sparse representation of signals to domain of set representation. Similar analogy was done for the purpose of characterization of streams of system event logs in [1]. In [1], a cost function which penalizes only for representation error was defined, thus requiring that the number of atoms be predefined, and leading to a substantially different algorithm compared to the one proposed in this work.

In fact, most of the existing approaches require the knowledge of the number of underlying latent topics [5], [12], [2], [9], [1]. This knowledge is unavailable and often unpredictable in most real data problems. In this work we suggest to handle this shortcoming automatically by enabling the algorithm to reveal the number of atoms requires for

978-0-7695-4257-7/10 $26.00 © 2010 IEEE
DOI 10.1109/ICDMW.2010.187

the data representation. Unlike the hierarchical LDA [4], PARIS maintains a "flat" representation of the atoms, thus allowing maintaining the freedom to assign multiple atoms (i.e., topics) to a set, and not limit it to a single branch in a hierarchy. In our experiments, we demonstrate that despite this degree of freedom, PARIS is able to identify latent topics from various levels in a topic hierarchy (if such an hierarchy exists), while LDA is incapable of distinguishing between the different levels in the hierarchy.

II. Problem Description

Our method optimizes a cost function, which is based on a data generation model assumption. We describe the data generation model and the cost function in this section.

A. Data Generation Model

We assume that each object in a data set, be it a document, event sequences, or other objects comprised of elements from a finite alphabet, has a model which generated it. Plainly, the model assumes that each object is a union of a portion of the elements of a small number of atoms from a larger dictionary of atoms. In addition, an object includes a potentially large number of elements that are not necessarily related to any atom.

Formally, let $\mathbf{A} = \{A_1, A_2, \ldots, A_K\}$[1] be a set of principal atoms (or atoms, in short). Each atom consists of a finite set of elements (e.g., words) from an alphabet T. Our data is a set of N sets $\mathbf{D} = \{D_1, D_2, \ldots, D_N\}$, where $N >> K$, such that each set was generated in the same way. For a set D_i, L_i atoms from $\mathbf{A}$ are chosen, where L_i is random, but much smaller than the size of D_i. From each of the L_i selected atoms, a ratio of r ($0 \leq r \leq 1$) or more of its elements is chosen and inserted into a union that constructs D_i. Therefore for each i, D_i is actually a union of L_i sets of elements, each constitutes a portion of r or more from a different atom in $\mathbf{A}$. In addition, D_i may include other elements, that do not belong to any of the L_i atoms, which are considered noise.

The ratio r is an attempt to apply this model to real applications, where we do not always expect all the elements in the atom to appear in a data sample. For example, when handling documents, we would expect the atoms to hold words on a certain concept, so that each document holds a portion (r) of the words of a few atoms, and noise (other words that are not necessarily in any of the atoms). This portion, in the case of documents, is relatively small, as not all the words that describe one concept should be included in a document discussing this concept. The smaller we would set the value of r, the larger should we expect the resulting atoms to be. In a different scenario, when analyzing logs of large systems, we might expect that when a problem occurs, almost all the event logs that relates to this problem should appear, which results in larger r value.

The problem addressed in this work is the extraction of the set of atoms $\mathbf{A}$ knowing only the set of data $\mathbf{D}$ and the ratio r.

For better defining our problem we first formalize this model by defining the concepts of representation, representation cost, and PARIS cost function.

B. Data Representation

A representation of a data sample D_i by a set of atoms $\mathbf{A}$ is a set R_i of atoms' indices. The represented set is therefore, $Rep(\mathbf{A}, R_i) = \bigcup_{j \in R_i} A_j$. The set of all representations is denoted by $\mathbf{R} = \{R_i | 1 \leq i \leq N\}$. The **quality** of a representation of a data sample is measured according to its similarity to this data sample. We define this distance function as the number of elements included in the data sample and not in the representation, plus the number of elements included in the representation and not in the data sample,

$$d(D_i, Rep(\mathbf{A}, R_i)) = |(D_i \otimes Rep(\mathbf{A}, R_i))|, \quad (1)$$

where $\otimes$ is the known 'xor' operation that outputs all elements that are included in only one of the two sets. Furthermore, we modify the above distance function in order to account for r in the data generation model and add normalization,

$$d_r(D_i, Rep(\mathbf{A}, R_i)) = \max_{\tilde{\mathbf{A}}} \frac{\left|D_i \otimes Rep\left(\tilde{\mathbf{A}}, R_i\right)\right|}{|D_i|}, \quad (2)$$

where $\tilde{\mathbf{A}} = \left\{\tilde{A}_1, \tilde{A}_2, \ldots, \tilde{A}_K\right\}$, and each $\tilde{A}_j$ holds: $\tilde{A}_j \subseteq A_j$ and $\left|\tilde{A}_j\right| \geq r \cdot |A_j|$. Therefore $\tilde{\mathbf{A}}$ is a new set of atoms derived from $\mathbf{A}$ according to the data sample D_i. In other words, we allow $(1-r)$ elements from each atom not to be included in the data sample represented by this atom without penalizing for it.

C. PARIS cost function

Given the above data generation model and definition of data representation error, we now turn to defining a cost function we would like to optimize for discovering the set of atoms of a given data set, without knowledge of either the number of atoms in the dictionary, nor the number of atoms representing each data sample. Two trivial solutions always exist. First, the sets of atoms equal the data set ($\mathbf{A} = \mathbf{D}$), making the representation sets very simple $R_i = \{i\}$. In this case, $K = N$, and for each i, $|R_i| = 1$. A second trivial case is its dual, $A_i = \{i\}$, and $R_i = D_i$. This results in $K = T$ and $|R_i| = |D_i|$. Each such trivial case results in a zero representation error but very large set of atoms or large representations. To avoid these uninformative trivial solutions, we would like our cost function to result in some sense of **compression**, while maintaining a low representation error.

[1] Throughout this work, we symbolize sets of elements by capital letters, and sets of sets (of elements) by **bold** capital letters

We denote the cost function of PARIS as PCF (for PARIS cost function), and define it as follows,

$$PCF_r(\mathbf{D},\mathbf{A},\mathbf{R}) = \underbrace{\sum_{i=1}^{N}\left(d_r\left(D_i, Rep\left(\mathbf{A}, R_i\right)\right)\right)}_{PCF_A} + \underbrace{\sum_{i=1}^{N}\mu_i\left|R_i\right|}_{PCF_B} + \underbrace{\tau\left|\mathbf{A}\right|}_{PCF_C}, \quad (3)$$

where $\mu_i = \frac{1}{|D_i|}$. The principles for designing this cost function are explained next.

- PCF_A - Minimization of the representation error is required. Otherwise there might not be any connection between the dictionary, representations and the data.
- PCF_B - Minimizing the size of all representations is important to prevent the algorithm from converging to the second trivial solution, in which $R_i = D_i$. This part also adds a sense of compression to the resulted representation, as it justifies adding another atom to the representation only if this addition reduces the representation error in **more** than one element.
- PCF_C - Minimizing the number of atoms in the dictionary is important to prevent the algorithm from converging to the first trivial solution, in which $A_i = D_i$. The value we set for τ will eventually control how **frequent** we would want some set repetition in the data, in order for it to become an atom in the dictionary. With the above settings of d_r (Eq. 2) and μ_i, setting $\tau = 1$ means that additional atom will be added only if it succeeds to eliminate the representation error of **more** than one data sample (the representation error of one complete data sample is $\frac{|D_i|}{|D_i|} = 1$ and the representation cost for using one atom is $\frac{1}{|D_i|}$) or alternatively, to eliminate part of the representation error of **a few** data samples.
  If we refer to r as part of the underlying model, τ is the only free parameter in the cost function, and its value reflects the 'resolution' of atoms we would like to reveal in our solution. As a rule of thumb, if the average size of the data samples is n, the expected atoms size is m, assuming a value r in the model , and we expect each atom to represent at least s data samples, the value of τ should be set to $\tau = \frac{s\cdot(r\cdot m-1)}{n}$.

D. Simple case illustration

In order to illustrate the effect of our cost function on the overall solution we would like to simulate it on an example. We choose to concentrate on a very simple case, in which the solution for $\mathbf{A}$ for each given number of atoms (K) is relatively simple, although we know that in a more general case such a problem is very complicated.

We assume the number of data samples is $N = 1000$. Each data sample is a union of exactly 5 atoms from a dictionary of size $K = 50$, in which each atom consists of exactly 5 different elements from an alphabet of size $T = 250$, and $r = 1$. Therefore, each data sample includes exactly 25 elements. Moreover, we assume a uniform distribution for using all atoms (each set of 5 atoms is equally probable). We demonstrate the behavior of the three parts in PCF as a function of the number of atoms in the left side of Figure 1, showing the values of the different components in PCF as a function of the number of atoms in the dictionary (K). For example, when $K = 0$ the representation cost, as also the atoms' cost are zero, but the representation error is very large. For each data sample 25 elements are not represented. Normalizing by the size of the data samples we get a total error of 1000 for all sets. When K increases, we must set the content of the atoms in order to calculate the overall cost. Reasonable assumptions force us to set the atoms as the original atoms' content (as all other settings will result in a larger representation error). For example, for $K = 1$, one original atom will decrease the representation error of ≈ 100 data samples in 5, and lead to an overall representation error of $\frac{900\cdot 25+100\cdot 20}{25} = 980$. In parallel, the representation cost is now $\sum_{i=1}^{N}\mu_i\left|R_i\right| = 100/25 = 4$, and the atoms' cost is $\tau\cdot|\mathbf{A}| = \tau\cdot 1$. Due to the simplifying assumption we took, one could see that the representation error keeps decreasing in the same ratio as K increases ($PCF_A = 1000 - 20K$), until $K = 50$. This is because each new atom reduces the error of approximately 100 data samples in exactly $5/25$ (while any other content of atoms leads to a greater representation error). As for the representation cost - for each new atom when $K \leq 50$ the representation increases in $100/25$ for all data samples (as 100 data samples are represented by this atom). When $K > 50$ we set the content of each new atom as the union of two original atoms, so that it decreases the representation cost for data samples that hold both atoms. We don't prove here that this is the best possible content. Doing so, the representation error remains zero, and the representation cost reduces slightly. Taking simple probability assumptions, the number of pairs that use two specific atoms is approximately 8, therefore, each such atom saves a total of 8 coefficients, which translates to a saving of $8/25$ in the total representation cost (PCF_B).

We now continue with the former simple simulation, and add noise. Again, for simplicity reasons, we assume that each data sample is now 'infected' by 25 different elements that do not belong to the former dictionary (T is increased). We also assume that the 'noisy' entries are different in each data sample, and therefore there cannot be any atom that represents more than one set of noisy entries in one data sample ($|T| = 250 + N\cdot 25 = 25250$. In this case, when $K = 0$ the representation error of all samples is $N\cdot 50/50 = 1000$. Again, one can see that the best settings

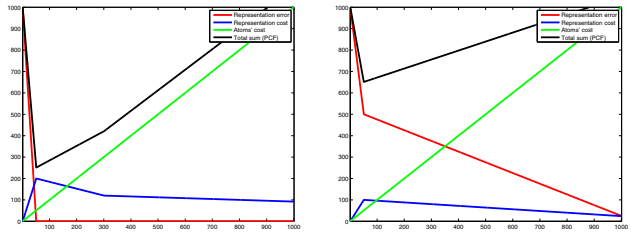


Figure 1. Each figure represents the behavior of the three components of the PCF for a simple illustrative case, with and without the presence of noise.

for the first 50 atoms is choosing the content of the original atoms. However, now the representation error decreases more slowly, and does not reach zero when $K = 50$. When $K > 50$, the most reasonable setting for each new atom would be to assign it as one of the data samples. Such a choice will decrease the representation error of this data sample to zero, and also decrease the representation cost for this data sample to $1/50$. The relevant graphs are presented on the right of Figure 1

Taking the sum of the three functions above we say the following,

- when the representation error decreases sharper than the increase of both the representation cost and the atoms' cost, larger K is preferred.
- when the representation error stops decreasing, or starts to decrease more slowly, the value of τ mainly controls whether K should stop increasing or not. In the above two simple scenarios, any value of τ between 0.6 to 8 results in the extraction of the original set of atoms.

III. The PARIS Algorithm

As the cost function in Equation 3 cannot be solved analytically, we designed an iterative procedure to solve it.Each iteration is composed of four steps:

1. Representation: Fix the atoms $\mathbf{A}$ and compute $\{R_i | 1 \leq i \leq N\}$.
2. Update: Update each atom A_i separately, assuming all other atoms are fixed.
3. Reduction: Reduce the size of the dictionary by omitting unused atoms, joining atoms that are used in common, and joining similar atoms.
4. Expansion: Expand the dictionary if regularities in the representation error sets are detected.

We next discuss each stage in details.

A. Data representation stage

In the representation stage we assume a fixed set of atoms $\mathbf{A}$, and compute the representation of each data sample based on these atoms. Formally, when fixing $\mathbf{A}$, we are left with the following problem,

$$\mathbf{R} = \underset{\{R_i | 1 \leq i \leq N\}}{\operatorname{argmin}} \sum_{i=1}^{N} \left(d_r \left(D_i, Rep \left(\mathbf{A}, R_i \right) \right) + \mu_i \left| R_i \right| \right). \quad (4)$$

Equation 4 can be minimized by minimizing for each i separately, therefore solving N times,

$$R_i = \underset{\widetilde{R_i}}{\operatorname{argmin}} \left(d_r \left(D_i, Rep \left(\mathbf{A}, \widetilde{R_i} \right) \right) + \mu_i \left| \widetilde{R_i} \right| \right). \quad (5)$$

For each i, we should find a set of atoms indices, whose union, according to the definition of r is closest to D_i. This is a combinatorial problem that we proved to be a NP-hard problem, using a reduction from the set-cover problem. Consequently, we adopt a greedy iterative approximation algorithm that remines the *orthogonal Matching Pursuit* (OMP) algorithm for sparse signal representation [14]. In each iteration we choose the atom that, after considering r or more of its elements, best reduces the expression in 5. This procedure is described in a pseudo-code in Figure 2.

Task: Find a representation for the set D_i
Input: A set of atoms $\mathbf{A}$, a parameter r.

Set $E_1 = D_i$.
For $l = 1, 2, \ldots$, do:

- denote,

$$RC(j_l) = d_r(E_l, Rep(A, \{j_l\})) + \mu_i \cdot l, \quad (6)$$

and find

$$j_l = \underset{\widetilde{j_l}}{\operatorname{argmin}}\ RC(\widetilde{j_l}) \quad (7)$$

- If $RC(j_l) > RC(j_{l-1})$, then the additional atom increases the representation error. We exclude it ($l = l - 1$) and finish (break the for loop).
- update $E_{l+1} = D_i \setminus \bigcup_{\{j_i | i \leq l\}} A_{j_i}$, where $X \setminus Y$ is the set subtraction operation that returns the set of all elements in X that are not elements of Y.

Set $R_i = \{j_n | n \leq l\}$.

Figure 2. The Greedy Data Representation algorithm

B. Update Stage

In this stage we update each atom in $\mathbf{A}$, assuming all other atoms are fixed. When updating A_j, we solve the following problem,

$$A_j = \underset{A_j}{\operatorname{argmin}} \sum_{i=1}^{N} \left(d_r \left(D_i, Rep \left(\mathbf{A}, R_i \right) \right) + \mu_i \left| R_i \right| \right). \quad (8)$$

Notice here, just like in Equation 4, the atoms' cost was omitted, as the total number of atoms is not changed.

For solving Eq. 8 we try to design a new atom that will best reduce the representation error. We start by considering the set of data samples that used the inspected atom in their representation $S_j = \{i | j \in R_i\}$. We define the set of representation errors $\mathbf{E}^j$ by the differences between

the original sets $\{D_i | i \in S_j\}$ and their representations, excluding the atom A_j,

$$E^j{}_i = \{ \ D_i \setminus Rep(\mathbf{A}, R_i \setminus \{j\}) | i \in S_j \ \} \tag{9}$$

A new atom A_j is then designed iteratively in order to best represent the set $\mathbf{E}^j$,

$$A_j = \underset{\widetilde{A_j}, R_i^j}{\operatorname{argmin}} \sum_{i=1}^{|S_j|} \left(d_r \left(\mathbf{E}_i^j, Rep \left(\{\widetilde{A_j}\}, R_i^j \right) \right) + \mu_i \left| R_i^j \right| \right), \tag{10}$$

where R_i^j is either $\{1\}$ or $\{\}$, as we do not force all error sets to be represented by the new designed atom. We proved that a simplified version of this problem is NP-hard by showing a reduction from the Balanced Complete Bipartite Subgraph (BCBS) problem.

We approximate the solution by designing a greedy method which we call the *Atom Design Procedure*. First, we select the most frequent pair of elements in the sets $\mathbf{E}^j$ to construct A_j. Then, in each iteration we compute the samples in $\mathbf{E}^j$ that benefit from using A_j in their representation (and set $R_i^j = \{1\}$ for each such sample) and add another element to A_j as the next frequent element in those samples $\left\{ i | R_i^j = \{1\} \right\}$. The algorithm is described in Figure 3

Task: Design a new atom A_j
Input: $\mathbf{E}$ - a set of samples we would like to best represent by the new atom, the parameter r.

Find a pair of elements $\{e_i, e_j\} \in T$ that appears most frequently together in the samples $\mathbf{E}$. Set: $A_j^2 = \{e_i, e_j\}$
For $l = 2, 3, \ldots$

- Calculate the samples' representation $\mathbf{R} = \{R_l^j\}$ by selecting the samples that benefit from using A_j^l in their representation,

$$R_l^j = \{1\} \tag{11}$$
$$\Updownarrow$$
$$PCF_r(\{E_l\}, \{A_j^l\}, \{1\}) < PCF_r(\{E_l\}, \{A_j^l\}, \{\}).$$

- Calculate $e_l = PCF_r(\mathbf{E}, \{A_j\}, \mathbf{R})$
- If $e_l > e_{l-1}$, break.
- Set $A_j^{l+1} = A_j^l \cup t$, where t is the most frequent element in the samples that use A_j^l, $\{i | R_i^j = \{1\}\}$.

Output: A_j^{l-1}, $\mathbf{R}^j$.

Figure 3. Atom Design Procedure - Designing a new Atom

C. Reduction stage

Reducing the number of atoms is important for minimization of Equation 3, as long as it does not increase the representation error expression (PCA_A) and the representation cost (PCA_B). There are three cases where an atom reduction is performed:

1) Reduce atoms that are not being used for representing any (or almost any) of the data objects. These atoms are easily identified by observing the representation sets.
2) Join sets of atoms that are used to represent together many of the objects (a merge of such a pair might decrease the representation cost and the atoms cost). We identify such potential sets by inspecting the representations R^i, and consider merging any sets of atoms that appear together more than twice it would have been in a uniform distribution.
3) Join pairs of atoms that share many of their elements, and are mostly used separately. These pairs can be joined without significantly increasing the representation error or cost. We identify them by inspecting the intersections between atoms: $\{\{n, m\} | \, |A_n \cap A_m| > \rho \cdot r \cdot \max(|A_n|, |A_m|)\}$, where ρ is some factor close to one.

For each atom or set of atoms that holds the above conditions do:

- let $C_{start} = PCF$ of the current settings.
- Simulate reduction of the atoms by omitting the atom in case (1), or designing a single atom instead the set of atoms in cases (2) and (3) by applying the *Atom Design Procedure* in Figure 3.
- check $C_{end} = PCF$ of the new setting.
- if $C_{end} < C_{start}$ - perform the reduction. Otherwise, reconstruct the former settings and continue with the next suspicious atom / set of atoms.

D. Expansion stage

In each iteration we consider the possibility of adding atoms to represent existing regularities in the data. Such an act will increase the atoms' cost (PCF_C), and might even increase the representation cost (PCF_B). Therefore, it is done as long as the decrease in the representation error (PCF_A) justifies it. Each new atom is designed using the *Atom Design Procedure*, which is given as input the set of representation errors, $\mathbf{E} = \{ \ D_i \setminus Rep(\mathbf{A}, R_i) | 1 \leq i \leq N \ \}$.

E. PARIS execution summary

The algorithm starts with an empty set of atoms, and therefore the first three stages of the first iteration are skipped. In cases where some knowledge of the underlying principle atoms exists, it can be exploited by setting them as the initial set of atoms. The algorithm runs for a fixed number of iterations or until convergence is reached, and the resulted dictionary is taken from the iteration with the minimal cost value.

IV. PARIS Results

We start by testing PARIS on a synthetic data that was generated according to the data generation model (in II-A), and examine a few of its properties. We continue with

another synthetic experiment based on a bit different data generation model, that explores the ability of PARIS to reveal the hierarchical structure of the latent topics. Then, we demonstrate the results with two real-data problems; text classification and collaborative filtering.

A. Synthetic data tests

We designed a dictionary of K atoms, each was generated by uniformly choosing a number of elements from the alphabet T, where the number of elements is randomly chosen from a range rA. Then, N data samples were generated. Each datum was generated by choosing a random number (from a range rL) of atoms from the dictionary. From each chosen atom, r of its elements were chosen, to form the datum as a union of all chosen elements from the atoms, together with additional x percent of noisy entries (randomly chosen over T), where the percentage is taken from the size of the clean datum. PARIS is executed on each set of data samples, given only the value of r. We also tested LDA over the data, providing LDA with the true dictionary size K.

We measure the similarity of the estimated dictionary to the true dictionary using a measure of similarity between sets of sets, *Total Atom Similarity Measure* (or TASM). The TASM is defined as follows: Each atom, both in the resulted dictionary (A_i) and in the original dictionary(oA_i), represented as a sparse vector with '1' for each of its elements, is normalized by dividing it in its L_2 norm $A_i/\left\|A_i\right\|_2$ or $oA_i/\left\|oA_i\right\|_2$. Then, each normalized atom in the original dictionary is multiplied by all the normalized resulted atoms. The atom that results in the highest product is assigned to that original atom, and the similarity between them is defined as the above inner product (only two identical atoms will result in an inner product of 1). The TASM is the average similarity between all atoms in the original dictionary:

$$TASM = \frac{\sum_{j=1}^{K} \max_{A_i} \frac{oA_j^T \cdot A_i}{\left\|oA_j\right\|_2 \cdot \left\|A_i\right\|_2}}{K}. \tag{12}$$

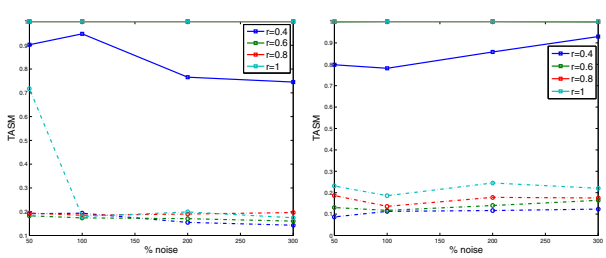
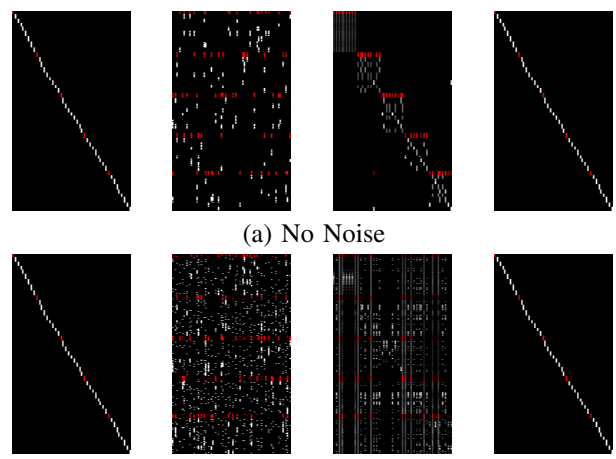


Figure 4. TASM for synthetic tests on PARIS (solid line) and LDA (dashed line) with uniform noise (left figure) and with uniform noise plus consisted noise (right figure). The plots for $r = 0.8, 1$ for PARIS are hidden under the plot for $r = 0.6$ (achieving 100% reconstruction)

In the first experiments we examined different values of r $(0.4, 0.6, 0.8, 1)$ and different noise levels (between 50% and 300%), with uniform distribution. We used $K = 40$, $|T| = 2000$, $rA = [5, 10]$, $rL = [3, 6]$ to generate the data, and set $\tau = 2$ in all tests. The results are shown as the solid lines in Figure 4; we see that PARIS is able to uncover the hidden principal atoms almost perfectly. The results slightly deteriorate when r decreases. Furthermore, the algorithm is robust to noise and the effect of the added noise can be observed only when r is small. Running LDA on these test cases, even after providing it with the true number of atoms, achieved much lower reconstruction scores.In another experiment, in addition to the above random noise, a "consistent noise" was added. That is, we picked 4 elements, each of which appears in different 80% of the data samples. This simulates consistent elements that often appears in real data, such as stop words in documents or items most users prefer when examining users' preferences. The results over the different r values are presented in 4 (both for PARIS and for LDA).



(a) No Noise

(b) 100% noise

Figure 5. Revealing the hierarchical data structure. In all figures the x-axis is for the different sets, the y-axes is for the different elements in the alphabet. Left most figure presents the original atoms (each atom appears in a different column), where colored dots mean the element exists in the atom and red columns are for the super-atoms. second from left shows 50 data samples (each sample in a different column), third from left shows the LDA latent topics and first from right shows the PARIS atoms (each topic or atom appears in a different column). The two figures are for the noiseless (top) and noisy (bottom) scenarios.

In a different synthetic experiment we tested a different data generation model than the original assumption of PARIS, which fits a hierarchical data structure. Here, we designed 5 'super atoms', each followed by another 8 'child atoms' associated with the corresponding 'super atom'. Each data sample was generated by first choosing randomly $1-2$ super atoms, and then accordingly, selecting $2-3$ child atoms of the chosen super atoms, resulting in $3-8$ atoms that construct each data set. We set $r = 0.8$ and noise of 0% and 100% of the original datum size. In this experiment we chose the original atoms specifically for presentation

purposes, without any intersection, in order to illustrate and emphasize the difference between PARIS and LDA, as seen in the left of each Figure in 5, where each atom is displayed in a different column, colored dots means the element exists in this atom. the red atoms are the super-atoms. Additionally, we tested LDA on the same data set, providing it with the exact number of atoms - 45. In both these experiments the TASM resulted by PARIS was 1 (exact recovery), while LDA results were 0.49 and 0.29 in the noiseless and noisy cases respectively. We present all results in Figure 5(a-b). The second image in each figure shows the first 50 data samples (out of 3000) that were given as input (again, each input set is presented in a different column, where the elements it consists of are colored). The two right most images, showing the LDA latent topics (β vectors), and the PARIS's atoms, where an atom or β_i vector are placed in the original atom they matched most to. What can be seen is that PARIS recovered exactly all the atoms, despite the fact that super-atoms always appear together with a subset of their child atoms, and despite the existing noise. LDA without any noise is able to recover the combination of super-atoms with their children, but not separate them; however, with the introduction of noise, even this structure does not appear.

B. Real data - text analysis

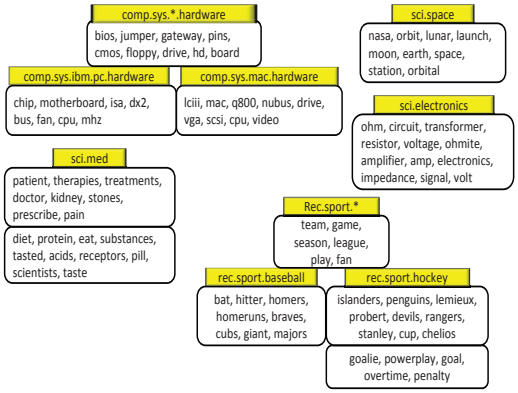


Figure 6. Sample atoms and related categories for the Newsgroup data

We evaluate the performance of PARIS by performing experiments over several real text collections: oh5, oh10 and oh15 datasets from OHSUMED (see [11] for a full description) and mini 20-Newsgroups dataset [3]. oh5, oh10 and oh15 datasets contain about 1000 documents each from 10 different categories. The 20-Newsgroups dataset contains 2000 documents from 20 categories.

Figure 6 shows a few examples of atoms, with the categories which they strongly represented in the documents of the 20-Newsgroup corpus. Some atoms represent well documents from multiple categories (see the general sports and computer hardware atoms), while others were specific to their category (e.g., the strong hockey and baseball atoms). These examples give credence to the ability of PARIS to uncover topic hierarchies, when those exist. Other examples show how atoms in a particular category (e.g., the sci.med ones), uncover multiple topics covered under the general topic (nutrition and medical treatments in medicine).

We further tested PARIS and LDA on each of the datasets for their ability to cluster the data. We use the F-measure as a goodness measure of topic recovery, based on the known topics provided to each document (one per document). We ran LDA with 15 different values of K between 20 and 400, and selected the best F-measure result from all runs. For each atom or latent topic, we computed the topic it most represents, by counting the number of documents that each atom (or latent topic) represents in the data, where, for simplicity, each document was represented by the most likely atom or latent topic. This provides a topic assignment rule for each document in the corpus (we also tested probabilistic assignments of documents to atoms/latent topics, and it provided worse F-measure results for both PARIS and LDA). The average results are shown in Table I. We see that in all these tests, PARIS produced better clustering results. In the case of the Newsgroup data set, we also computed the ability of both methods to predict the six super-categories in the data (science, politics, sports, computers, religion, misc), with the results showing that PARIS still outperforms LDA.

Dataset	*LDA F-measure*	*PARIS F-measure*
oh5	0.53 (K=20)	0.77 (r=0.3)
oh10	0.41 (K=30)	0.70 (r=0.6)
oh15	0.51 (K=30)	0.72 (r=0.6)
tr23	0.55 (K=350)	0.65 (r=0.5)
20-Newsgroups	0.43 (K=30)	0.70 (r=0.3)
20-Newsgroups (super categories)	0.59 (K=40)	0.81 (same dictionary as above)

Table I
F-MEASURE COMPARISONS OF PARIS AND LDA ON DIFFERENT DATASETS.

C. Collaborative filtering

We further tested PARIS for collaborative filtering. We used the MovieLens data set [7] which contained a million ratings for 3900 movies by 6040 users. Our input was $M \times N$ *interaction matrix* $A = (a_{ij})$, where M is the number of users and N is the number of movies. Each entry of this matrix can take the value of either 0 or 1. A value of 1 for a_{ij} represents the fact that user i has seen the movie j. The original dataset contains ratings (1-5), so the *interaction matrix* is created by collecting all the movies rated by users.

A comparison between seven different algorithms for collaborative filtering was described in [13]. The two algorithms performing best for the MovieLens data set were found to be *User-based* and *Item-based* algorithms. For this reason

we compared the PARIS results against results of these two algorithms and LDA.

We ran PARIS on the data, where the movies each user had seen represent the input set $\mathbf{D}$. The set of atoms representing each user, where each atom is a set of movies, expressed the relevancy of certain set of movies to that user. The representations were used to generate a user to user similarity matrix. After computing the user similarity matrix we created the ranked list of K movies as in the User-based algorithm. We applied the same procedure to LDA results for computing the movie ranking, where we computed each user's probabilistic representation by the latent topics using the variational inference method described in [5].

For evaluation of algorithms, the data set was divided to a training and testing set. The training set consisted of 80% of the movies rated by each user. The remaining 20% of the movies were used for testing. Only users who rated up to 100 movies were considered. For each user we generated a ranked list of K ($K = 10$) movies. For each user, the recommendation quality was measured on the basis of the number of *hits* (recommendations that matched the movies in the test set). We employed the same evaluation criteria from [13]: averaged *Precision*, *Recall* and *F-measure* over all users. For LDA we tested various values of number of latent topics and chose the value which produced the highest F-measure (30). For PARIS we set $\tau = 0.2$ and $r = 0.4$. The results are summarized in Table II. We can see that PARIS outperforms the other algorithms for all the three evaluation criteria.

	Precision	Recall	F-measure
User-based	0.1118	0.1232	0.1108
Item-based	0.1768	0.1957	0.1757
LDA	0.1469	0.1608	0.1452
PARIS	0.2021	0.225	0.2016

Table II
CF EXPERIMENT RESULTS

V. SUMMARY

We presented PARIS, an algorithm for sparse representation of sets using principal atoms recognized from data. One of the main advantages of the algorithm is that it identifies the number of atoms automatically, through the optimization of a cost function which balances compression and representation. We demonstrated the algorithm on both synthetic and real data, proving its advantage over LDA on both clustering and collaborative filtering tasks. A possible explanation to the success of the algorithm may stem from the data generation model it assumes - instead of the soft probability distributions over words and concepts in models such as LDA, it makes a hard decision on the assignment of a word to an atom, and an atom to a datum. These hard decisions lead to sparser representation of objects, and filter more of the noisy dimensions. In addition, we observed that these hard decisions enable the creation of atoms that can represent atoms in topic hierarchies, despite being a "flat" representation which assumes no hierarchy.

REFERENCES

[1] M. Aharon, G. Barash, I. Cohen, and E. Mordechai. One graph is worth a thousand logs: Uncovering hidden structures in massive system event logs. In *ECML/PKDD*, pages 227–243, 2009.

[2] M. Aharon, M. Elad, and A. Bruckstein. The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation. *IEEE Trans. On Signal Processing*, 54(11):4311–4322, 2006.

[3] C. L. Blake and C. J. Merz. UCI repository of machine learning databases, University of California, Irvine, Dept. of Information and Computer Sciences, 1998.

[4] D. Blei, T. Griffiths, and M. Jordan. The nested chinese restaurant process and bayesian inference of topic hierarchies. *Journal of the ACM*, 2010.

[5] D. Blei, A. Ng, M. Jordan, and J. Lafferty. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:2003, 2003.

[6] O. Bryt and M. Elad. Compression of facial images using the K-SVD algorithm. *Journal of Visual Communication and Image Representation*, 19(4):270–283, 2008.

[7] M. dataset. Grouplens research project at the university of minnesota. http://www.grouplens.org/taxonomy/term/14.

[8] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. on Image Processing*, 15(12):3736–3745, 2006.

[9] K. Engan, S. Aase, and J. Husoy. Method of optimal directions for frame design. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 5:2443–2446, 1999.

[10] T. Griffiths and Z. Ghahramani. Infinite latent feature models and the indian buffet process. In *NIPS*, 2006.

[11] E. Han and K. G. Centroid-based document classification: Analysis and experimental results. In *Principles Data Mining Knowledge Discovery*, 2000.

[12] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, 1999.

[13] Z. Huang, D. Zeng, and H. Chen. A comparison of collaborative-filtering recommendation algorithms for e-commerce. *IEEE Intelligent Systems*, 22(5):68–77, 2007.

[14] S. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Trans. Signal Processing*, 41(12):3397–3415, 1993.

[15] M. Protter and M. Elad. Image sequence denoising via sparse and redundant representations. *IEEE Trans. on Image Processing*, 18(1):27–36, 2009.