Document Classification with Hierarchically Structured Dictionaries

Remya R.K. Menon and P. Aswathi

Abstract Classification, clustering of documents, detecting novel documents, detecting emerging topics etc in a fast and efficient way, is of high relevance these days with the volume of online generated documents increasing rapidly. Experiments have resulted in innovative algorithms, methods and frameworks to address these problems. One such method is Dictionary Learning. We introduce a new 2-level hierarchical dictionary structure for classification such that the dictionary at the higher level is utilized to classify the K classes of documents. The results show around an 85% recall during the classification phase. This model can be extended to distributed environment where the higher level dictionary should be maintained at the master node and the lower level ones should be kept at worker nodes.

1 Introduction

Applications like Novel document detection and clustering, Topic modeling, Spam filtering etc are still some challenging areas due to the ever increasing volume of on-line documents. The commercial value of classifying text documents have lead to many machine learning techniques in the area. The performance of such techniques can be improved by exploring domain-specific term features. Classification models have to be generated out of a set of labeled data and frequencies of class relevant terms of each document. Here, a basic necessity is to represent these extracted class specific information efficiently such that it can be used to express or approximate any input data belonging to its corresponding class at a later stage.

In areas like signal processing, image processing etc, any data can be represented or approximated in terms of a *dictionary* (a pre-determined set of data points) which is learned from the training data itself [1]. The same concept has been adopted for

R.R.K. Menon(✉) · P. Aswathi
Amrita Vishwa Vidyapeetham, Amritapuri Campus, Kollam 690525, Kerala, India
e-mail: ramya@am.amrita.edu, aswathipv45@gmail.com

S. Berretti et al. (eds.), *Intelligent Systems Technologies and Applications*,
Advances in Intelligent Systems and Computing 385,
DOI: 10.1007/978-3-319-23258-4_34

text documents representation by some works [2][3][6]. The dictionary learned out of the data is then used for classification and novelty detection [2]. Given a set of data $Y = \{y_i\}_{i=1,..m}$ in R^n where n is the number of relevant terms and m is number of documents in the dataset, the goal of dictionary learning is to find a dictionary $D \in R^{n \times k}$ such that each data point y in the set can be represented as a sparse linear combination of its columns. This can be represented as

$$Y \approx DX \quad satisfying \quad \|Y - DX\|_F \le \varepsilon \;\; \& \;\; \|x\|_0 \le \tau \ , \tag{1}$$

where $x \in R^k$ contains the ordered coefficient values and $X \in R^{k \times m}$ is the sparse representation matrix.

Any dictionary learning technique basically deals with two problems - Sparse coding and Dictionary composition. Sparse coding is the technique with which sparse representational values x for y with the atoms of D is obtained. The atoms to form the linear combination and its coefficient values are to be chosen well to obtain the optimal minimal representation[1][9]. There are many optimization algorithms to calculate x like Pursuit algorithms. The simplest one among them are Matching Pursuit algorithms that follows a greedy approach where the dictionary atoms are chosen sequentially and its commonly used variations are *Basis Pursuit* and *Orthogonal Matching Pursuit*(OMP). OMP solves x by using l_0 norm, as expressed below.

$$\min_x \|x\|_0 \quad subjectto \quad \|Y - DX\|_F \le \varepsilon \tag{2}$$

Given X after the sparse coding phase, with retrieval error $\|Y - DX\|_F$, the aim of the update step is to update the dictionary D or to find a better dictionary D that can accommodate the error. There are many techniques for this like the generalized k-means, MOD (Method of Optimal Directions), Maximum A-Posteriori Probability approach, K-SVD etc. All these methods prove to learn dictionaries well but varies in their convergence, efficiency and implementation effort. In K-SVD algorithm [9, 10, 11] this is done by finding SVD of the error matrix and the matrices D and X are updated simultaneously using the factors U and V.

Once a dictionary is generated, a major question to be considered is whether a new set of data of the same class can be represented with D. The given dataset may contain documents of different classes or labels. When there are K distinct classes then K dictionaries are to be learned [2]. The dependency and relationship among the classes should be considered while learning these dictionaries such that performance is better during classification and/or clustering.

Our contribution includes hierarchically structured dictionaries and a method to classify a document into any of the K classes, based on this structure. The hierarchy consists of 2-levels. The lower level consist of K *sub-dictionaries* corresponding to K classes and the higher level has a single dictionary referred to as *parent-dictionary*(refer Fig. 1 with value of K as 3).

The structure is generated in a bottom-up fashion and classification is done with the parent dictionary. Parent dictionary is constructed out of all the sub-dictionaries,

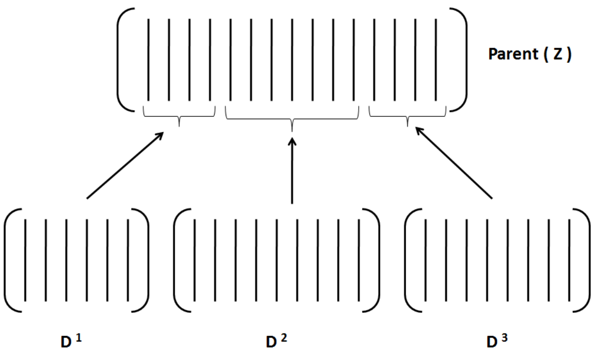

Fig. 1 2-level Hierarchical Dictionary Structure

using Principal Component Analysis(PCA), and hence contains a set of representative atoms from each of the classes. Experiments show that the parent dictionary alone is enough for classification among K classes. We perform our experiments on two different datasets those when evaluated gives around 85% recall and precision.

2 Related Work

Dictionary learning is an area which has scope for exploration in the domain of text mining. Few research has been done with regard to the usage of learning dictionaries from the vast amount of text piling up in the web/database. Sparse representation of data through sparse coding is a byproduct of this approach.

Dictionary learning has been utilized for novel document detection both for static data [2] as well as streaming data [3] using a distributed environment. It has also been used for clustering [1] but by the signal processing community. In their work, a dictionary is constructed for each cluster using which the signals are best reconstructed using the sparse code. Data points belonging to the same class can belong to different subspaces based on the generated sparse code. A dictionary incoherence term is added to the general condition of reconstructed error to make the dictionaries to represent a specific class while at the same time differentiate it from other classes. Sharing of atoms between dictionaries of two distinct classes can also occur [5]. Emerging topic detection [2] proposes an Alternative directions method (ADM). ADM is applied by representing the l_1-norm reconstruction error in the augmented Lagrangian form and then approximating it. ADM has been applied for the detection stage as well as the learning stage.

Online l_1-dictionary learning [4] uses the l_1-norm for the reconstruction error based on a variation of the ADM for novel document detection. The paper proposes to obtain bounds on the regret for updating dictionary. Experiments have been conducted on the Twitter data.Fast on-line $l1$-dictionary learning [7] compares 3 algorithms for on-line dictionary learning - dual averaging scheme (D_A), projected gradient scheme

(P_G) and online ADM. The paper claims that D_A has better predictive power were as the other two algorithms score in terms of running time and parameter stability. Kasiviswanathan et.al. [3] extends the novel document detection to a distributed environment by choosing the SPMD (Simple Program Multiple Data) model which uses the MPI concept for message passing. Here also, dictionary learning is done using the ADM approach.

Utilization of dictionary learning for classification especially in the document/text domain is yet to be well explored. Unlike signal and image domains, application of dictionary learning to the text domain will have a varied set of problems like maintainability of document structure.

3 Notations Used

Notation	Explanation
y	an input document as a vector
D	general representation for any dictionary
$D^c \in R^{n\times m}$	sub-dictionary corresponding to c^{th} class where n - number of terms m - number of examples in class c
d_i^c	i^{th} atom of c^{th} class sub-dictionary
$Z \in R^{n\times Q}$	Parent dictionary with n rows and Q atoms together from all classes
$Z^c \in R^{n\times q_c}$	atoms of parent that represents c^{th} class q_c is number of vectors/atoms taken from D^c
z_i^c	i^{th} atom of Z^c
T	coefficient matrix obtained after sparse coding with parent Z
T^c	part of T that contains values of c^{th} class
t_i^c	i^{th} coefficient value of T^c

4 Hierarchical Dictionary Structure

4.1 Structure of the Dictionaries

We introduce a new 2-level hierarchical structure of dictionaries which can be used later for classification purposes. The lower level consists of K *sub-dictionaries* and the higher level consist of a parent-dictionary. Given a set of labeled documents the system generates K *sub-dictionaries* for the K distinct classes in bottom up fashion. Applying PCA on each of these sub-dictionaries individually gives the most relevant vectors from each classes. These features or vectors from K classes are appended together to form a master/parent dictionary that suffices all the K classes.

The entire work is designed as 3 modules - Pre-processing data, Generating the dictionaries, Classification of documents.

4.2 Preprocessing

Given a set of documents each labeled with any of the K classes, this module processes them to generate K classes of input matrices containing numerical values. Relevant terms of each class is derived from its corresponding documents and their TF-IDF value is calculated to form the input Y^c matrix. Before the calculation, stemming is done along with removal of stop words and duplicate words to obtain more accurate values.

4.3 Procedure of Dictionary Learning

The mentioned dictionary structure is generated in 2 stages - learning K sub-dictionaries and learning the parent dictionary out of the K sub-dictionaries.

$$Y^c \approx D^c X^c \quad suchthat \;\; \|Y^c - D^c X^c\|_F \leq \varepsilon \tag{3}$$

As a new sub dictionary is generated, PCA is applied over it to generate the sub part Z^c of the parent dictionary Z.

$$Z = Z^1 \cup Z^2 \cup ... \cup Z^K \tag{4}$$

4.3.1 Learning a Sub-dictionary

An input matrix $Y \in R^{n \times m}$ consists of n terms and m number of sample labeled documents. This matrix is generated using the TF-IDF measure and contains data of a single class. As we have K classes we have K such input matrices. We have adopted the K-SVD algorithm along with considering the dependency among the sub-dictionaries. (3) is solved in this section by implementing algorithm 1.

The input matrix itself is initially considered as the dictionary D^c and is used for sparse coding in the first iteration. The K sub-dictionaries have same number of rows but varying numbers of columns. Once the sparse representation X^c is obtained, it is used to update the dictionary to accommodate the error term $\|Y^c - D^c X^c\|_F$ using SVD decomposition. Considering each atom individually, all documents that uses the atom is identified from X. Excluding the contribution of this atom in the representation, the error in retrieval is obtained on which SVD is applied. While U and V are the factors obtained during SVD, the first column of U is used to replace the selected atom of D^c and the column of V along with its corresponding singular value is used to update the corresponding row of X^c that uses the atom. This procedure is done repeatedly until convergence.

Before generating a parent dictionary from these sub-dictionaries, we have to ensure that the dictionary is as independent as possible from each other such that the classifier can distinguish the classes well. To obtain this, we add a incoherence inducing term that can be expressed as

Algorithm 1. generating a sub-dictionary

```
Input: $Y$, initial dictionary $D_0$, $\tau$ := number of sparse coefficients , $K$:= no of labels ,$\eta$ := weight value
Output: Sub-dictionaries $\{D^1, D^2, .., D^K\}$ and sparse matrices $\{X\}$
for $c = 1 : K$ do
    $D^c := D_0$
    for $q = 1 : h$ do
        $[i, x_i^c] := Argmin(x)\|y_i - D^c x_i^c\|_2^2$ Subject To $\|x^c\|_0 \leq \tau$
        if $c! = 1$ then
            for $p = 1 : c - 1$ do
                $coh := coh + getIncoherence(D^c, D^p)$
            end for
            $coh := coh * \eta$
            $D^c[n, m] := D^c[n, m] + coh \quad \forall n = 1, 2, .., Nandm = 1, 2, .., M$
        end if
        for $j = 1 : k$ do
            $(D^c)' = (D^c)^{(q-1)}$
            $I$ := { indices of the docs in Y uses atom $(D^c)'$}
            $E_I := Y - D_j^{c\,\prime} x_T^j$
            $[D^c, X^c] = \text{SVD}(E)$
        end for
    end for
end for
```

$$D_i = D_i + \eta \sum_{j=1}^{i-1} \|D_i^T D_j\|_2^2 \qquad \forall\, i = 1, 2, .., K \tag{5}$$

Each element in the dictionary is added with this value making the dictionary matrix incoherent from others[5].

4.3.2 Dictionary Learning - Learning the Parent Dictionary

Given the sub dictionaries, we generate the Parent Dictionary there by solving (4). By applying PCA on a sub-dictionary D^c we obtain a set of eigenvalues and their corresponding eigenvectors, that constitute the basis of the dictionary(refer algorithm 2). PCA projects the data to a lower dimensional representation with least error while preserving largest variances. As all the eigenvalues are not necessary for an approximation, we choose those eigenvalues greater than δ. The corresponding eigenvectors of the chosen eigenvalues forms the set $Z^c \in R^{n \times q_c}$. q_c indicates the number of eigenvectors included to the parent. The final parent dictionary $Z \in R^{n \times Q}$ is obtained by vertically concatenating $Z^1, Z^2, .., Z^K$ where Q = $q_1 + q_2 + .. + q_k$ as given in algorithm 2.

Algorithm 2. Generating Parent Dictionary

```
Input: $D$ - Set of all k dictionaries, $\delta$ - threshhold for eigen value
Output: $DParent$ - Parent dictionary
for $a = 1 : k$ do
    $[P, Ev] := PCA(D_a)$
    for $\forall e \in Ev \geq \delta$ do
        $DParent_a := DParent_a \cup (P_e)$
    end for
end for
```

4.4 Classification

4.4.1 Parent-dictionary Based Classification

The parent dictionary Z contains set of vectors from all the sub-dictionaries and it represents all the K classes together. Parent based classification algorithm(refer algorithm 3) finds the sparse coding T of the new test document with parent Z. Based on the atoms being used and its coefficient values the algorithm identifies the class. Z^c indicates the part of the parent that is derived from class c^{th} dictionary. T^c indicates the coefficient values corresponding to Z^c. Thus $\sum_{i=1}^{q_c} z_i^c t_i^c$ calculates the linear combination of all atoms of c^{th} class and then its frobenious norm is taken. The document belongs to class c for which the representation error is minimum (6).

$$\arg\min_c \| y - \sum_{i=1}^{q_c} z_i^c t_i^c \|_F^2 \tag{6}$$

By bounding this error with a threshold, this can be used for classification of a document into multiple labels.

5 Experimentation and Analysis

5.1 Datasets and Preprocessing

Experiments are conducted using two datasets dataset I and dataset II. The training samples of dataset I contains 200 documents belonging to a ‘Question Bank’ on the topics of computer science, collected from our academic collection. It consists of 4 labels ($K = 4$) - ‘Computer Organization And Architecture’, ‘Computer Networks’, ‘Programming’, and ‘Database Management System’. The training samples of dataset II includes 75 documents belonging to two entirely distinct labels ($K = 2$)- ‘Indian Climate’ and ‘Dictionary Learning’.

Algorithm 3. Classification with parent-dictionary

```
Input: = parent dictionary, testY, no_sparse_coef
Output: list of novel documents , labels for rest of documents
           errors = error from closer dictionaries
Init: $N$ =No.of documents in testY, $r = ()$, $doclabels = ()$, $noveldocs = ()$, $errors = ()$
for $i = 1 : N$ do
    $y = testY_i$
    $T := sparseCoding(Z, y, no\_sparse\_coef)$
    for $c = 1 : K$ do
        $res := \| \sum_{i=1}^{q_c} Z_i^c \, T_i^c \|_F^2$
        if $res <= \alpha$ then
            $r := r \cup c$
        end if
    end for
    if $r$ is empty then
        alert("Document to not belong to any class !")
    end if
end for
```

Each document in the set of training documents taken as input belongs to any one of the classes in its domain. These labeled documents becomes the input to the preprocessing module were they are processed to generate an output file corresponding to each class/label. These files contains the TF-IDF values, to form the input Y^c matrix corresponding to each classes. Each i^{th} column of the matrix represents i^{th} training document in the class. Removal of duplicate words and stopwords followed by stemming ensures that only relevant terms are considered in this calculation. The relevant terms identified for each class in the dataset and their corresponding TF-IDF weight is written into a file.

5.2 Dictionary Learning

The obtained tf-idf value-based vectors are given into the sub-dictionary learning module. Initially the input matrix itself is taken as the dictionary. Then the algorithm identifies initial coefficient values(X) by using OMP, implemented with cholesky decomposition. Once X is obtained, it is used to update the dictionary atoms along with its coefficient values using SVD factorization over error matrix. The dictionary is updated iteratively until the error criterion is met. Frobenius norm is used here to calculate the error $\|Y - DX\|_F \leq \varepsilon$. The error obtained lies between the interval 1.5 and 3.5.

With dataset II first dictionary comes with a dimension of 10×35 (10 terms and 20 documents). Similarly the dimension of second dictionary is 10×40. The parent dictionary is created from the sub dictionaries using PCA. It identifies the independent basis vectors as eigenvectors and in this experiment we consider those with their corresponding eigenvalues that are greater than δ. For dataset II, $\delta = 0.005$.

Table 1 Sample Result with Dataset I (9 out of 22 test documents)

Actual class	Documents	COA (class 1)	Networks (class 2)	Programming (class 3)	DBMS (class 4)	classified to
1	Doc 1	0.08284	0.15171	0.14925	0.20732	1
1	Doc 2	0.04193	0.07972	0.11042	0.08160	1
1	Doc 3	0.04598	0.08545	0.09380	0.14708	1
2	Doc 4	0.07379	0.04642	0.04796	0.09865	2
2	Doc 5	0.11858	0.04978	0.12538	0.06616	2
2	Doc 6	0.09460	0.04962	0.08535	0.10455	2
3	Doc 7	0.44099	0.26587	0.16008	0.32346	3
4	Doc 8	0.11906	0.09827	0.15469	0.21753	2
4	Doc 9	0.14588	0.11382	0.10693	0.08081	4

Table 2 Confusion matrix for Dataset I

Cases	Predicted Negative	Predicted Positive
Negative cases	53	3
Positive Cases	3	19

Table 3 Confusion matrix for Dataset II

Cases	Predicted Negative	Predicted Positive
Negative cases	8	1
Positive Cases	1	8

From 1st and 2nd sub-dictionaries, we get 10 and 9 eigenvectors respectively. i.e., $q_1 = 10, q_2 = 9$. On concatenating them vertically we obtain total $Q = 19$ vectors in parent. Number of rows in parent remains as n.

Similarly with dataset I, the sub-dictionary dimensions are such that $Y^1 \in R^{20\times35}$, $Y^2 \in R^{20\times23}$, $Y^3 \in R^{20\times26}$ and $Y^4 \in R^{20\times24}$. After applying the algorithm 2 over them, we obtain $Z \in R^{20\times26}$.

5.3 Classification

Generally during classification, each document is to be matched with all the sub-dictionaries to identify the better one. In this work, we use the parent dictionary alone for classification. To find the resultant class, equation (7) is used and results are given in Table 1 and Table 4. With dataset I, the representation error $\| \sum_{i=1}^{q_c} z_i^c t_i^c \|_F^2$ of some of the test documents, with respect to 4 classes are given in Table 1. Similarly with dataset II, the error of the test documents, with respect to 2 classes are given in Table 4.

There are 22 documents in the test set for dataset I and 19 out of 22 are correctly classified. With 9 documents in the test set for dataset II, eight documents are classified correctly. Once *confusion matrix* is calculated (as given in Table 2 and Table 3), evaluation of classification results is performed with the measures of accuracy, precision and recall. Accuracy is the correctness of results, calculated as the total number

Table 4 Result with Dataset II (9 test documents)

Actual class	Documents	Dictionary-Learning (class 1)	Climate (class 2)	classified to
2	Document 1	0.03472702	0.01276724	2
2	Document 2	0.08271272	0.08000903	2
2	Document 3	0.45815999	0.21802396	2
2	Document 4	0.02490772	0.00474065	2
1	Document 5	0.05320239	0.43174093	1
2	Document 6	0.34723903	0.30103366	2
1	Document 7	0.05320239	0.23015204	1
1	Document 8	0.35465621	0.00542857	2
1	Document 9	0.05320239	0.41254825	1

Table 5 Evaluation with Datasets

Evaluation Criteria	Dataset I	Dataset II
Accuracy - % of correct prediction	92.3 %	88.89 %
Precision - % of positive predictions that are correct	86.36 %	88.89 %
Recall - % of correct prediction of positive cases	86.36 %	88.89 %

of correct classifications divided by total number of classifications. Precision is a measure calculated as number of true positives divided by sum of true positive and false negative. Precision is a measure calculated as number of true positives divided by sum of true positive and false positive. The evaluation results of experiments with both dataset is given in Table 5. Both experiments results in an effective classification with more than 85% of accuracy, precision and recall.

6 Conclusion

Classification is a well-saught after mechanism especially in this era when data gets piled up with no bounds and hinders the way for data analysis. So our aim in this paper is to harness the potential of dictionary learning for the process of classification. We have demonstrated that our proposed method that uses a hierarchical dictionary structure will pave the way for improving the classification process. This structure can be utilized for a distributed environment where the parent dictionary can act as the master node and the sub-dictionaries can be the slave nodes. So the parent dictionary can determine the corresponding child nodes to get the complete classification done.

Acknowledgments We express our fullest gratitude to the Department Chairman, Dr. M.R. Kaimal, who supported us throughout with his valuable comments. We thank our teachers and friends for giving us valuable information and our family for their moral support.

References

1. Bruckstein, A.M., Donoho, D.L., Elad, M.: From sparse solutions of systems of equations to sparse modeling of signals and images. SIAM Rev. **51**, 34–81 (2009)
2. Kasiviswanathan, S.P., Melville, P., Banerjee, A., Sindhwani, V.: Emerging topic detection using dictionary learning. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, CIKM 2011, pp. 745–754. ACM, New York (2011)
3. Kasiviswanathan, S.P., Cong, G., Melville, P., Lawrence, R.D.: Novel document detection for massive data streams using distributed dictionary learning. IBM Journal of Research and Development **57**(3/4), 9 (2013)
4. Kasiviswanathan, S.P., Wang, H., Banerjee, A., Melville, P.: Online l1-dictionary learning with application to novel document detection. In: Bartlett, P. L., Pereira, F.C.N., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) NIPS, pp. 2267–2275 (2012)
5. Ramrez, I., Sprechmann, P., Sapiro, G.: Classification and clustering via dictionary learning with structured incoherence and shared features. In: CVPR, pp. 3501–3508, IEEE (2010)
6. Menon, S.R., Nair, S.S.: Sparsity-based representation for categorical data. In: Recent Advances in Intelligent Computational Systems (RAICS). IEEE (2013)
7. Kasiviswanathan, S.P.: Fast online l 1-dictionary learning algorithms for novel document detection. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 8585–8589. IEEE (2013)
8. Berry, M.W., Drmac, Z., Jessup, E.R.: Matrices, vector spaces, and information retrieval. Society for Industrial and Applied Mathematics **41**, 335–362 (1999)
9. Aharon, M.: Overcomplete dictionaries for sparse representation of signals. PhD thesis, Technion-Israel Institute of Technology, Faculty of Computer Science (2006)
10. Aharon, M., Elad, M., Bruckstein, A.: Svdd: An algorithm for designing overcomplete dictionaries for sparse representation. Trans. Sig. Proc. **54**, 4311–4322 (2006)
11. Rubinstein, R., Zibulevsky, M., Elad, M.: Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit. CS Technion **40**(8), 1–15 (2008)
12. Jolliffe, I.: Principal component analysis. Wiley Online Library (2002)