Application of BIRCH to text clustering

© Ilya Karpov © Alexandr Goroslavskiy
Federal state unitary enterprise "Institute "KVANT", Moscow
karpovilia@gmail.com sashka_airok@mail.ru

Abstract

This work represents a clustering technique, based on the Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) algorithm and LSA-methods for clustering large, high dimensional datasets. We present a document model and a clustering tool for processing texts in Russian and English languages and compare our results with other clustering techniques. Experimental results for clustering the datasets of 10'000, 100'000 and 850'000 documents are provided.

1 Introduction

Many important tasks in IR involve clustering large datasets of unstructured text. Although there is a large set of efficient techniques for clustering of abstract data sets, few of them have been applied to clustering textual data. The task is specific due to the following reasons: (1) large number of data points, (2) large number of clusters and (3) high clustering feature dimensionality. The other problem is requirement in high performance and precise linguistic techniques to deal with a large set of documents. This work represents a clustering technique, based on the Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) algorithm and LSA-methods for clustering these large, high dimensional datasets in Russian and English languages.

Input documents are stored as a set of normalized terms vectors as described in Bag of words model. Initial dimensionality of nearly 300'000 terms is reduced by the TF-IDF dispersion threshold for each term of the document. Then, term vectors are analyzed with latent semantic analysis (LSA) as described in [2] and [3]. After that, clustering with BIRCH algorithm is performed

Proceedings of the 14th All-Russian Conference "Digital Libraries: Advanced Methods and Technologies, Digital Collections" — RCDL-2012, Pereslavl Zalesskii, Russia, October 15-18, 2012.

2 Related work

There has been proposed number of approaches for clustering large collections of arbitrary data. MST [7], DBSCAN [1], CLOPE [4] and BIRCH [8] are the most suitable techniques for text clustering according to the (1)-(3) criteria. All of them are suitable high feature dimensionality and have complexity $O(n \log n)$ for MST, DBSCAN and CLOPE and $O(n \log k)$ for BIRCH. Another method for clustering text datasets is called Canopies algorithm and described in [4]. The key idea of Canopies is to divide the data into overlapping subsets (canopies). Then clustering is performed by measuring exact distances only between points that occur in a common canopy. The algorithm requires $O(nk)$ distance comparisons per iteration of clustering, where n is the number of documents and k - number of canopies.

3 Preprocessing

3.1 Building vector space representation

We use the "bag-of-words" model, where a document is represented as an unordered collection of terms, disregarding grammar and even word order. In this work a term is a normal form of the word with its part-of-speech tag. $t =< normalized_word, POS >$; Note that in this case different part of speech are normalized as follows:

ADJECTIVE - subj case, singular num, masculine
NOUN - subjective case, singular num
VERB - infinitive

We checked two strategies to resolve word polysemy: The naive one was to add all possible terms to the term-vector. The second way was to use lexical compatibility and choose the most probable variant. Performance and quality will be discussed at the RESULTS section. After the normalization, a document in the collection is represented as a vector of term weights, counted according to the term frequency − inverse document frequency (TF/IDF) model. $D = (w_1, ..., w_n)$ The result of this stage is a term matrix containing word weights per document (rows represent unique words

and columns represent documents) as shown below:

$$\begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,n} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m,1} & w_{m,2} & \cdots & w_{m,n} \end{pmatrix}$$

where m is the number of documents and n is the number of unique terms.

3.2 Reducing the clustering space dimension

First step in dimension reduction is removing "noisy" parts of speech from the document model. It stands to reason that adjectives and verbs bring rather noise than useful information when they are disconnected from nouns, so we used only nouns in our experiments.

The next step is selecting the most informative terms in the model. There are several methods for choosing a threshold, based on the TF/IDF: M. Kiselev uses the sum of term weights in all documents [5], or the total frequency of the term [6]. In this work we use the dispersion of weight for each term in the collection as the threshold: $T = \max \sigma(w_i)$, where i is term number.

The last step is Latent semantic analysis (LSA) of the term matrix. LSA assumes that words that are close in meaning will occur close together in text and uses singular value decomposition (SVD) to reduce the number of terms while preserving the similarity structure among documents.

4 BIRCH algorithm

BIRCH algorithm is detaily specified in [8], we only describe the basic ideas and concepts here. The core concepts of BIRCH are Clustering feature, CF-tree and Cluster radius. A Clustering feature is a triple summarizing the information we maintain about the cluster:

Def. 1 Given N d-dimensional data points in a cluster $\{\overrightarrow{X_i}\}$ where $i = 1, 2, .., N$, the Clustering Feature (CF) vector of the cluster is defined as a triple: $CF = \left(N, \overrightarrow{LS}, SS\right)$, where N is the number of data points in the cluster, $\overrightarrow{LS}$ is the linear sum of the N data points, i.e. $\sum_{i=0}^{N} \overrightarrow{X_i}$, and SS is the square sum of the N data points, i.e. $\sum_{i=0}^{N} \overrightarrow{X_i^2}$

Def. 2 A CF tree is a height-balanced tree with two parameters: branching factor B and threshold T. Each nonleaf node contains at most B entries of the form $[CF_i, child_i]$, where $i = 1, 2, ..., B$, "$child_i$" is a pointer to its $i-th$ child node, and CF_i is the CF of the subcluster represented by this child. So a nonleaf node represents a cluster made up of all the subclusters represented by its entries. A leaf node contains at most L entries, each of the form $[CF_i, child_i]$, , where $i = 1, 2, ..., L$. In addition, each leaf node has two pointers, "$prev$" and "$next$" which are used to chain all leaf nodes together for efficient scans. A leaf node also represents a cluster made up of all the subclusters represented by its entries. But all entries in a leaf node must satisfy a *threshold requirement*, with respect to a threshold value T: *the diameter (or radius) has to be less than T*

The tree size is a function of T. The larger T is, the smaller the tree is. We require a node to fit in a page of size *P*. Once the dimension *d* of the data space is given, the sizes of leaf and nonleaf entries are known, then B and L are determined by P. So P can be varied for performance tuning.

Def. 3 A cluster radius R is an average Euclidean distance between the data points and the cluster centroid.

$$R = \left(\frac{\sum_{i=1}^{N} \left(\overrightarrow{x_i} - \overrightarrow{x_0}\right)^2}{N} \right)^{1/2}$$

where $\overrightarrow{x_i}$ - documents in cluster, $\overrightarrow{x_0}$ - center of the cluster (arithmetical mean of all documents in the clustering space), N - number of points in the cluster. BIRCH algorithm has three main stages:

1. Given the threshold T, the clustering space is divided into the *initial clusters* in such a way that cluster radius is smaller than R. BIRCH uses extremely high performance method based on *CF trees* to form the initial clusters. See [8] for details.

2. *Initial cluster* centroids are once again clustered with agglomerative clustering tool. New cluster centroids are determined after that.

3. New cluster centroids are used as seeds for clustering space redistribution. Each document is reassigned to the nearest seed.

5 Modifications

5.1 Defining the clustering threshold

The main advantage of BIRCH algorithm is its high performance: given a threshold T value, the cost of clustering will be only $O(2n)$, but choosing bad threshold may cause significant performance reduction.

BIRCH authors proposes the following method: Initially T is set to 0. If the number of leafs reaches the maximum amount, T is extended and CF tree is being rebuild. An assumption that the number of points N_i, that can be contained in CF tree at the i-th stage of the algorithm is in linear fashion with T_i^d where d is the clustering dimension. The next threshold T_{i+1} is determined by linear regression with regard to $N_{i+1} = \min(2N_i, N)$, where N is the total amount of points in the clustering space.

An implementation of this method shows that either the threshold grows too slowly that causes multiple tree reconstruction or the threshold grows too fast that causes the loss of accuracy. We provide a new method for determining the T value:

As in the previous solution, initially T is set to 0. The next threshold T_{i+1} is determined as a function of the cluster size for a random set of clusters: First, a cluster radius $R = \max(dist(X_0, Xi))$ is determined as the maximum distance between the point and the cluster centroid for a number Num_r of randomly selected clusters. Then, the threshold $T_{k,i+1} = Round(\alpha * R_k)$ is determined for each cluster. Resulting threshold T_{i+1} is determined as arithmetical mean of $T_{k,i+1}$.

5.2 Splitting tree nodes

The original BIRCH algorithm proposes splitting node into to new nodes when the limit of childs B is reached. Such a technique has disadvantages when inner childs are miscellaneous node need to be divided into three, four or more new node. We propose agglomerative clustering to split the node: the minimum number of nodes is set as 2, and the maximum as the *(B - parent node childs + 2)*.

6 Results

Method has been tested on the following collections: Lenta.ru news articles collection (approximately 100'000 text documents for 2007–2012) Russian volume of Wikipedia (850'000 articles at May 2012)

6.1 Performance results

Experimental clustering results for sets of various sizes from Wikipedia are shown in table 1.

Table 1: Clustering time for BIRCH* and MATLAB k-means algoritms

N docs	k-means (rand)	k-means (10%)	BIRCH*
1 000	3,2604	3,4008	1,2034
2 000	8,8453	9,5005	1,3329
5 000	30,6698	46,2855	2,1813
10 000	158,8558	170,1035	5,1964
20 000	396,1333	337,009	8,8000
100 000	—	—	19,66
500 000	—	—	52,89
850 000	—	—	104,59

Clustering has been performed with Intel Core i7 - 2600 CPU (3,4 GHz), 16 Gb DDR 3 RAM. The first and the second column shows time in seconds

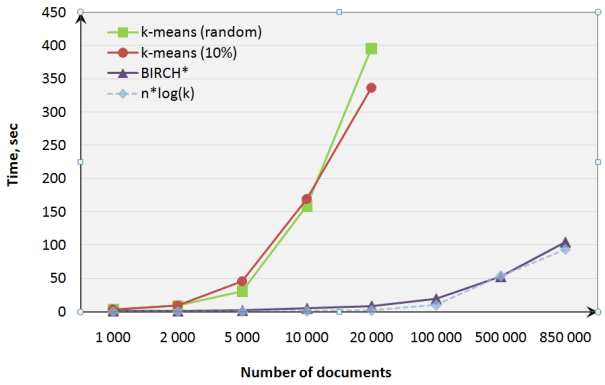

Figure 1: Clustering time vs Number of documents dependency

for **one** iteration of the default MATLAB k-means algorithm with different methods for centroids calculating. The first one selects centers randomly and the second one uses a random 10% subsample of the entire training set. The rest of the parameters are as follows: number of clusters - 100, number of iterations - 1, number of dimensions - 200. The third column is based on the modified BIRCH clustering algoritm with the same parameters.

The results are shown at figure 1. n*log(k) plot shows the estimating complexity of the BIRCH algorithm where n is the number of documents and k - the number of clusters. It is assumed that k = log(n).

6.2 Accuracy results

Accuracy results for a set of 600 articles from russian Wikipedia and 10000 news articles from Lenta.ru are provided in table 2.

Training sets were made as an intersection of three accessors hand-made clustering results. The results were measured with F-measuse as follows: Given i - some rubric from the document collection, j - some cluster, gained from clustering let D_{ri} be the documents of i rubric, and D_{ci} - documents of i cluster. *Precision* of j-th cluster about i-th rubric $P(i,j) = \frac{|D_{ri} \cap D_{cj}|}{|D_{cj}|}$. *Recall* of j-th cluster about i-th rubric $R(i,j) = \frac{|D_{ri} \cap D_{cj}|}{|D_{ri}|}$. *F-measure* of j-th cluster about i-th rubric is $F(i,j) = \frac{2*P(i,j)*R(i,j)}{P(i,j)+R(i,j)}$. *F-measure of the resulting clustering* is:

$$F = \sum_{i=1}^{M} \frac{|D_{ri}|}{N} max_j F(i,j),$$

where N – total amount of docs.

All algorithms have random factor, so we evaluated average F-measure for a set of 20 measurements. Thow BIRCH can predict the number of clusters, we fixed it to calculate the F-measure. It should be mentioned, that being used with one iteration,

k-means algorithm shows very poor quality results so we used 30 iterations for k-means with random centers and 3 iterations for k-means with 10% subsample. Total amount of iterations was found experimentaly as the function of the F-measure dispersion.

Table 2: F-measure results for BIRCH* MATLAB k-means algoritm

Measure	k-means (rand) 30 iterations	k-means (10%) 3 iterations	BIRCH*
$avg(F)$ $600docs$	$0,948$	$0,933$	$0,923$
$\sigma(F)$ $600docs$	$8,2*10^{-4}$	$1,2*10^{-4}$	$1,6*10^{-4}$
$avg(F)$ $10000docs$	$0,567$	$0,572$	$0,563$
$\sigma(F)$ $10000docs$	$5,4*10^{-4}$	$2,7*10^{-4}$	$1,7*10^{-4}$

Simularity of the results can be explained by simular methods of assigning points to the clusters at the last stage of all algorithms. Significant loss of quality for news collections can be caused by mixture of many topics in one document and different taxonomies of the human classification and statistical clustering. Better results can be achieved in the combination of clustering and topic-based classification.

6.3 Conclusions

This work has demonstrated the use of the modified BIRCH clustering method for clustering large datasets. In comparison with the naive clustering methods, such as k-means or EM-clustering computation time has been reduced by more than two orders with the same accuracy.

References

[1] Martin Ester, Hans-Peter Kriegel, Jorg Sander, Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In Evangelos Simoudis, Jiawei Han, Usama M. Fayyad. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96). AAAI Press. pp. 226−231.

[2] Thomas Hofman. Probabilistic Latent Semantic Analysis. EECS Department, Computer Science Division, University of California, Berkeley & International Computer Science Institute, 1999.

[3] Thomas Hofmann. Probabilistic Latent Semantic Indexing. Conference on Research and Development in Information Retrieval, 1999.

[4] Andrew McCallumzy, Kamal Nigamy. Lyle H. Ungar, Efficient Clustering of High-Dimensional Data Sets with Application to Reference Matching. //KDD '00 Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, 2000, pp. 169−178

[5] M. Kiselev, Text Clustering Procedure Based on Pair-wise Proximity of Key Terms and its Comparison with Metric Clustering Methods, "Internet-mathematics 2007", Moscow, Yandex publishing, 2007, pp. 74-83.

[6] M. Kiselev, M. Shmulevich, A. Ehrlich Automatic clustering method and its applications, Program products and systems No2, 2008, pp 47−50

[7] Zhou, Yan; Grygorash, Oleksandr; Hain, Thomas F. Clustering with Minimum Spanning Trees. International Journal on Artificial Intelligence Tools, Feb2011, Vol. 20 Issue 1, p139−177, 39p

[8] Tian Zhang, Raghu Ramakrishnan, Miron Livny BIRCH:an efficient data clustering method for very large databases. //Proceedings of the 1996 ACM SIGMOD international conference on Management of data,1996.