Efficient Document Indexing Using Pivot Tree

Gaurav Singh[1] and Benjamin Piwowarski[2]

[1] UPMC, Paris,
gaurav.singh.15@ucl.ac.uk
[2] UPMC, Paris,
benjamin.piwowarski@lip6.fr

Abstract. We present a novel method for efficiently searching top-k neighbors for documents represented in high dimensional space of terms based on the cosine similarity. Mostly, documents are stored as bag-of-words tf-idf representation. One of the most used ways of computing similarity between a pair of documents is cosine similarity between the vector representations, but cosine similarity is not a metric distance measure as it doesn't follow triangle inequality, therefore most metric searching methods can not be applied directly. We propose an efficient method for indexing documents using a pivot tree that leads to efficient retrieval. We also study the relation between precision and efficiency for the proposed method and compare it with a state of the art in the area of document searching based on inner product.

1 Introduction and Related Work

There are two main areas of research in information retrieval 1.) Search in metric spaces 2.) Search in non-metric spaces. A metric space basically refers to a similarity measure which follows all the metric properties like reflexivity, symmetry, non-negativity and triangle-inequality. All other properties can be achieved by trivial transformations, but triangle inequality is considered most important out of all others, since it is difficult to achieve it using trivial transformations and it can be effectively used in pruning elements. On the other hand, a non-metric space basically refers to a similarity measure that doesn't follow triangle inequality. In such spaces, metric access methods using triangle inequality can not be applied directly. The retrieval approaches in non-metric spaces can be broadly categorized as *embedding* and *classification*.

A metric space is described by a similarity measure that follows the properties of reflexivity, nonnegativity, symmetry and most importantly triangle inequality. A number of methods have been developed in the past to search metric spaces[4, 11, 14], most metric spaces can be search efficiently using the triangle inequality. [12, 8] discuss the use of range queries and kNN in metric space. A lot of research has been done in the field of information retrieval from nonmetric spaces, but a lot of similarity measures do not follow triangle inequality. In such non metric spaces, there are broadly two approaches followed, embedding and classification

Embedding basically refers to the conversion of nonmetric space into metric spaces. There are certain embedding methods which perform exact conversion[5]

to metric space whereas others do approximate conversion to metric space[2]. A number of approximate embedding methods have been developed such as [2] that converts data objects into vector space. They introduce a query sensitive distance function together with the embedding method, in order to give different importance to different embedding dimensions for each query object. TriGen is another method developed to convert nonmetric spaces into metric spaces by using metric preserving and similarity invariant modifiers. But the author himself acknowledges, not all nonmetric measures are suitable to be converted by these methods. In the case of exact embedding methods, the most prominent is LCE[5], which tried to divide objects into groups and then adds a small local constant to all pairwise distances within a group to make them follow triangle inequality. This method although exact, becomes completely unscalable for large datasets, since it requires the computation of all possible triplets of objects within a cluster, which can be a huge computational cost. A number of other approximate embedding techniques like Fastmap[6], Metric Map[13], and Sparse Map[7] exist, but the only exact method with no false dismissals are LCE and CSE[10]

[3] presented a non-metric clustering method based on distances to the so-called fiduciary templates (some selected random objects from the set). The distances to these fiduciary templates form a vector, which is used to decide in which cluster a new object belongs. [1] proposed a k-median clustering algorithm for nonmetric functions (specifically, the Kullback-Leibler divergence) that computes $a(1+\epsilon)-$ approximation of the k-median problem.

Recently [9] published maximum inner product based appraoch for querying documents. Their approach is based on creating tigher bounds as the query object traverses down the tree because of reduced number of documents at each new level and therefore a reduced radius. In the proposed method, we project query object on a set of orthogonal pivots as we descend down the pivot tree. We use the previous pivots to construct an orthogonal pivot to all other pivots in the descend path of the query object. We avoid any euclidean addition/subtraction operations that are expensive in high dimensional spaces. The proposed method is based on maximimizing the projection for group of documents on a set of orthogonal projectors.

2 Proposed Method

We observe experimentally that the following relation holds for any given query $q \in \mathbb{R}^v$, where v is the vocabulary size, projector $S \in \mathbb{R}^{v \times v}$, document $d \in \mathbb{R}^v$ and orthogonal projector $S^{\perp} \in \mathbb{R}^{v \times v}$

$$q^T d \leq \|Sq\|\|Sd\| + \|S^{\perp}q\|\|S^{\perp}d\| \tag{1}$$

$$\leq 1 + 2\|Sq\|\|Sd\| - \|Sq\| - \|Sd\| \tag{2}$$

We can bound the distance between a given document d and query q using the above inequality. We use the above inequality to bound $\|q^T d\|$ for all documents

contained in the subtree rooted at node N_p, we select a radom pivot p_{n+1} from all such documents.

2.1 Updating the Projector

We construct basis B_n for the subspace spanned by the vectors (pivots) $p_1, ..., p_n$ in the descend path to the node N_p from the root of the tree.

$$B_n = P_n A_n \text{ with } P_n = \begin{pmatrix} p_1 \dots p_n \end{pmatrix}$$

Let p_{n+1} be the new vector to be added to the subspace then, we have the new basis B_{n+1}:

$$B_{n+1} = \begin{pmatrix} B_n \ x \end{pmatrix}$$

such that:

$$x = \frac{y}{\|y\|} \text{ with } y = \left(\mathrm{Id} - B_n B_n^\dagger\right) p_{n+1}$$

We can get a projection vector(y) orthogonal to B_n using the relation:

$$\|y\|^2 = \|p_{n+1}\|^2 - \left\|B_n B_n^\dagger p_{n+1}\right\|^2 = \|p_{n+1}\|^2 - \left\|B_n^\dagger p_{n+1}\right\|^2$$

Then, denoting $\alpha = \|y\|^{-1}$,

$$B_{n+1} = \begin{pmatrix} P_n A_n \ \alpha \left(\mathrm{Id} - B_n B_n^\dagger\right) p_{n+1} \end{pmatrix} \tag{3}$$

$$= \begin{pmatrix} P_n \ p_{n+1} \end{pmatrix} \begin{pmatrix} A_n & -\alpha A_n A_n^\dagger P_n^\dagger p_{n+1} \\ 0 & \alpha \end{pmatrix} \tag{4}$$

2.2 Updating the Similarity

We compute the value of $\left\|B_{n+1}^\dagger D\right\|$ from $\left\|B_n^\dagger D\right\|$ for all the documents(D) contained in the subtree rooted at node N_p. Each node of pivot tree contains $max(\left\|B_{n+1}^\dagger D\right\|^2)$ and $min(\left\|B_{n+1}^\dagger D\right\|^2)$ $\forall D \in D_p$ where D_p is the set of all documents contained in the subtree rooted at node N_p.

$$\left\|D^\dagger B_{n+1}\right\|^2 = \left\| \begin{pmatrix} D^\dagger P_n \ D^\dagger p_{n+1} \end{pmatrix} \begin{pmatrix} A_n & -\alpha A_n A_n^\dagger P_n^\dagger p_{n+1} \\ 0 & \alpha \end{pmatrix} \right\|^2 \tag{5}$$

$$= \left\| \begin{pmatrix} D^\dagger P_n A_n \ \alpha D^\dagger p_{n+1} - \alpha D^\dagger P_n A_n A_n^\dagger P_n^\dagger p_{n+1} \end{pmatrix} \right\|^2 \tag{6}$$

$$= \left\|D^\dagger B_n\right\|^2 + \left\|\alpha D^\dagger p_{n+1} - \alpha D^\dagger P_n A_n A_n^\dagger P_n^\dagger p_{n+1}\right\| \tag{7}$$

2.3 Algorithm

In this section we describe the algorithm we use to construct the pivot tree. We then describe an algorithm to search the pivot tree using a given query.

1. Algorithm SelectPivot(Data S)

```
SelectPivot(Data S)
    Pick some random pivots P ∈ S
    Choose a random pivot p ∈ P s.t. argmax_p(∑ ||p^T p S_i||^2)   ∀S_i ∈ S
    return (p)
```

2. Algorithm MakeSplit(Data S, Pivot p)

```
MakeSplit(Data S, Pivot p)
    A ← {s ∈ S: ||D^T p_{n+1}||^2 > c}
    B ← S/A
    return (A,B)
```

3. Algorithm UpdateProjections(Data D_l, Pivot p, A)

```
UpdateProjections((Data D, Pivot p, A)
    D_i.Projections ← update(D_i,p,A) ∀D_i ∈ D; # Using eqn. 5
```

4. Algorithm BuildTree(Data S)

```
BuildTree(Data S)
   Input ← S
   Output ← Tree T
   T.S ← S
   T.min ← min(S.Projections)
   T.max ← max(S.Projections)
   if (|S| ≤ N_o)
        return T
   T.p ← SelectPivot(Data S)
   D_l,D_r ← MakeSplit(T.S, T.p)
   # Using eqn. 4
   T.A ← UpdateA(pivot p)
   # Using eqn. 4
   T.P ← UpdateP(pivot p)
   # Using eqn 7.
   D_l.Projections ← UpdateProjections(Data D_l, Pivot p, T.A)
   T.left ← BuildTree(Data D_l)
   T.right ← BuildTree(Data D_r)
   return T
```

5. Algorithm SearchTree(Query S, Tree T)

```
SearchTree(Query S, Tree T)
   Input ← Query S, Tree T
   Output ← Document Set D
   B_l ← ComputeBound(Tree T.left, Query q) # using eqn 2
   B_r ← ComputeBound(Tree T.right, Query q) #using eqn 2

   #getLast: Returns the element with least similarity with query
   if (B_l ≥ getLast(queue))
        searchL=True
   if (B_r ≥ getLast(queue))
        searchR=True

   if(searchL and searchR)
        if (B_l > B_r)
            queue ← SearchTree(Query S, Tree T.left)
        else
           queue ← SearchTree(Query S, Tree T.right)
    else if (searchL and !searchR)
        queue ← SearchTree(Query S, Tree T.left)
    else if (!searchL and searchR)
        queue ← SearchTree(Query S, Tree T.right)
    else
        return (queue)
```

3 Experimentation and Results

We present in this section experimental results for the proposed method based on the **MTA** (Maximized Trace Approach) against state of the art method **MIP**(Maximum Inner Product) appraoch. The precision versus prunes is drawn for both appraoches by reducing the bound artificially, reduction in bound leads to more prunes, but reduced precision. We can see in Figure 1 that MTA outperforms MIP[9] in terms of both ranking (as measured by spearman distance) and precision for different values of prunes.

References

1. Marcel R Ackermann, Johannes Blömer, and Christian Sohler. Clustering for metric and nonmetric distance measures. *ACM Transactions on Algorithms (TALG)*, 6(4):59, 2010.
2. Vassilis Athitsos, Marios Hadjieleftheriou, George Kollios, and Stan Sclaroff. Query-sensitive embeddings. *ACM Trans. Database Syst.*, 32(2), June 2007.
3. Glenn Becker and Mark Potts. Non-metric biometric clustering. In *Biometrics Symposium, 2007*, pages 1–6. IEEE, 2007.
4. Edgar Chávez, Gonzalo Navarro, Ricardo Baeza-Yates, and José Luis Marroquín. Searching in metric spaces. *ACM Comput. Surv.*, 33(3):273–321, September 2001.

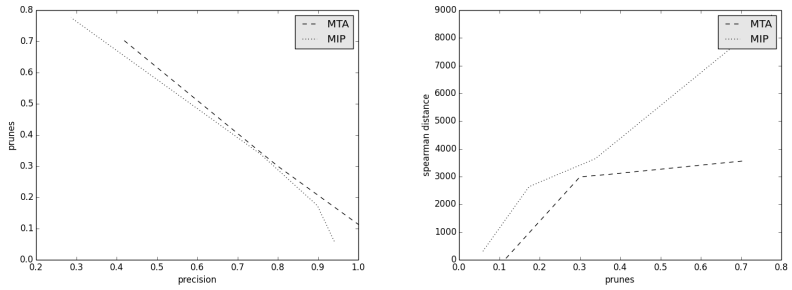

Fig. 1: The figure in the left presents the prunes against precision. The figure on the right presents ranking performance of the two methods for different number of prunes.

5. Lei Chen and Xiang Lian. Efficient similarity search in nonmetric spaces with local constant embedding. *IEEE Trans. on Knowl. and Data Eng.*, 20(3):321–336, March 2008.
6. Christos Faloutsos and King-Ip Lin. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. *SIGMOD Rec.*, 24(2):163–174, May 1995.
7. Hristescu Gabriela and Farach Martin. Cluster-preserving embedding of proteins. Technical report, 1999.
8. G.R. Hjaltason and H. Samet. Properties of embedding methods for similarity searching in metric spaces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):530–549, May 2003.
9. Parikshit Ram and Alexander G. Gray. Maximum inner-product search using cone trees. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 931–939, New York, NY, USA, 2012. ACM.
10. Volker Roth, Julian Laub, Joachim M Buhmann, and Klaus-Robert Müller. Going metric: Denoising pairwise data. In *NIPS*, pages 817–824, 2002.
11. Hanan Samet. *Foundations of Multidimensional and Metric Data Structures (The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.
12. Thomas Seidl and Hans-Peter Kriegel. Optimal multi-step k-nearest neighbor search. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, SIGMOD '98, pages 154–165, New York, NY, USA, 1998. ACM.
13. Xiong Wang, Jason T. L. Wang, King ip Lin, Dennis Shasha, Bruce A. Shapiro, and Kaizhong Zhang. An index structure for data mining and clustering. *Knowledge and Information Systems*, 2:161–184, 2000.
14. Pavel Zezula, Giuseppe Amato, Vlastislav Dohnal, and Michal Batko. *Similarity Search: The Metric Space Approach.* Springer Publishing Company, Incorporated, 1st edition, 2010.