# Torch: a tool for building topic hierarchies from growing text collections*

Ricardo M. Marcacini
Mathematical and Computer Sciences Institute - ICMC
University of São Paulo - USP
Sao Carlos, SP, Brazil
rmm@icmc.usp.br

Solange O. Rezende
Mathematical and Computer Sciences Institute - ICMC
University of São Paulo - USP
Sao Carlos, SP, Brazil
solange@icmc.usp.br

## ABSTRACT

Topic hierarchies are very useful for managing, searching and browsing large repositories of text documents. Several studies have investigated the use of hierarchical clustering methods to support the construction of topic hierarchies in an unsupervised way. In this paper we present the **Torch - Topic Hierarchies** software tool that implements the main features of the method IHTC - Incremental Hierarchical Term Cluster, aiming to build topic hierarchies from growing text collections. A simple experiment is described in order to illustrate the use of the Torch to support the construction of digital libraries.

## Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

## General Terms

Algorithms, Experimentation

## Keywords

Topic Hierarchies, Incremental Clustering, Digital Libraries

## 1. INTRODUCTION

The online platforms for publishing and storing digital content has contributed significantly to the growth of several text repositories. In this scenario, large numbers of new documents are generated everyday, resulting in the so-called growing text collections. Due to the need to extract useful knowledge from these repositories, methods for automatic and intelligent organization of text collections have received great attention in the literature [1]. The use of topic hierarchies is one of the most popular approaches for this organization because they allow users to explore the collection interactively through topics that indicate the contents of the available documents.

*This work was sponsored by FAPESP and CNPq.

Hierarchical clustering methods have been used to support the construction of topic hierarchies in a wider variety of applications such as digital libraries, web directories and document engineering [3, 2, 5]. The main motivation for the use of hierarchical clustering is the ability to organize the textual collection in different levels of granularity. Thus, the most generic knowledge is represented by clusters of higher hierarchy levels while their details, or more specific knowledge, by clusters of lower levels. This result is similar to the concept of topic hierarchies, facilitating the process of building these hierarchies. However, despite recent advances, there are still some challenges related to building topic hierarchies, as presented below.

• **Incremental clustering**: most algorithms require that all documents are available before starting the clustering process. Furthermore, when new documents are inserted, the cluster tree needs to be rebuilt and this is computationally expensive when dealing with growing text collections. Thus, it is necessary to update the clusters dynamically as new documents are inserted, ie, performing clustering in an incremental way.

• **Understandable cluster labels**: the interpretation of the results of clustering is a difficult task for users. Therefore, for the building topic hierarchies it is important that clusters have simple descriptions, ie, terms (words) that indicate the contents of the documents in the clusters.

• **Topics overlapping**: a particular feature of textual collections is that documents can belong to more than one topic. Thus, the topics overlapping is a desirable effect, since it allows to maintain the multi-topic property of the texts.

Since these challenges are requirements in many real scenarios, we proposed and evaluated in [4] a method called IHTC (Incremental Hierarchical Term Clustering). The IHTC method provides algorithms to incrementally perform preprocessing and hierarchical clustering of textual data, therefore allowing the construction of topic hierarchies with the requirements identified above.

In view of this, in this paper we present the **Torch - Topic Hierarchies** software tool that implements the main features of the method IHTC aiming to build topic hierarchies from growing text collections. The Torch tool helps users to organize and manage text collections and can be used in various applications.

The outline of this paper is as follows. In Section 2, we present the architecture and the main functionalities of the Torch. A simple experiment is described in Section 3 in order to illustrate the use of the tool to support the construction

of digital libraries. Finally, Section 4 summarizes the paper and outlines some directions for future work.

## 2. TORCH - TOPIC HIERARCHIES

The Torch tool (*TO*pic Hiera*RCH*ies) was developed to help users to "see hidden topics" in unstructured textual collections. The tool was implemented in the Java language and its architecture is organized into three main modules, as illustrated in Figure 1: (1) Text Preprocessing, (2) Hierarchical Term Cluster and (3) Discovering Topic Hierarchies.

An overview of the tool can be summarized as follows. First, a text data source is monitored to identify when new documents are added. Each new document is preprocessed (**Text Preprocessing module**) and the most significant terms are selected for the construction of a co-occurrence network. Then the cluster of terms (candidate topics) are identified in the **Hierarchical Term Clustering** module through a hierarchical clustering algorithm that is applied in the co-ocurrence network. Finally, the **Discovering Topic Hierarchies** module selects the best clusters of terms for the building of the final topic hierarchy.

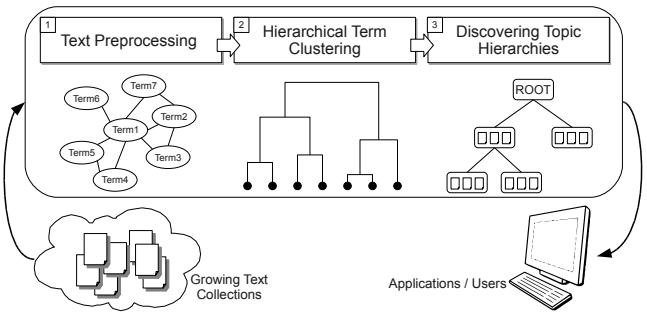


**Figure 1: Torch architecture**

The modules of the tool are implemented according to the method IHTC [4]. In the next sections we describe in more details the functionalities of each module and their relationships.

### 2.1 Text Preprocessing

Textual collections can easily contain thousands of terms, many of them redundant and unnecessary, that slow the process of building topic hierarchies and decreases the quality of results. Moreover, in this work we consider that the textual data is completely unstructured requiring specific tasks to obtain a structured representation of texts. Thus, the goals of the text preprocessing module is to maintain a subset of representative terms and organize them in a co-occurrence network. The co-occurrence network is a structured representation where terms are organized in a graph and two terms are connected if there is a significant co-occurrence value between them.

Each new document inserted to the text collection is preprocessed individually. To this end, the Text Preprocessing module has the following functionalities.

• **Stopwords removal:** non-significant words are removed from the text, such as articles, prepositions and conjunctions. The tool provides stopwords lists for English and Portuguese texts. However, the user can add new stopwords as needed.

• **Stemming:** performs the process of reducing words for their stems, ie, the portion of a word that is left after removing its prefixes and suffixes. The tool has options to stem Portuguese and English texts using the Porter algorithm[1].

• **Term Selection:** selects a small subset of terms for each document, providing a concise representation with little loss of information. This functionality is based on the Zipf's law to identify the most important terms of the texts.

• **Co-occurrence Network Construction:** builds a co-occurrence network from terms extracted from each document. The network is built incrementally using an algorithm that dynamically estimates the pairs of frequent terms [4]. Thus, if two terms occur with high frequency during the text preprocessing then a new edge is added to connect the pair of terms in the co-occurrence network.

Through the functionalities described above, the Text Preprocessing module builds a co-occurrence network incrementally as new documents are inserted. The co-occurrence network is used as input for the module Hierarchical Term Clustering, as described in the next section.

### 2.2 Hierarchical Term Clustering

The aim of this module is to organize the co-occurrence network into cluster of terms, where terms from the same cluster are similar compared with terms from different clusters. The Torch has two options to compute the similarity between terms.

• **Jaccard:** the similarity is computed as the ratio between the number of documents in which the two terms occur together (intersection) and the number of different documents (union).

• **SNN (Shared Nearest Neighbors):** uses only the topology of the co-ocurrence network and defines the similarity between a pair of terms in accordance of their shared nearest neighbors.

After choosing a similarity measure, a hierarchical clustering algorithm is executed to construct a dendrogram with cluster of terms. The dendrogram is a binary tree that organizes the terms into clusters and subclusters. This process is illustrated in Figure 2, where the co-ocurrence network (a) is processed resulting in the dendrogram (b).

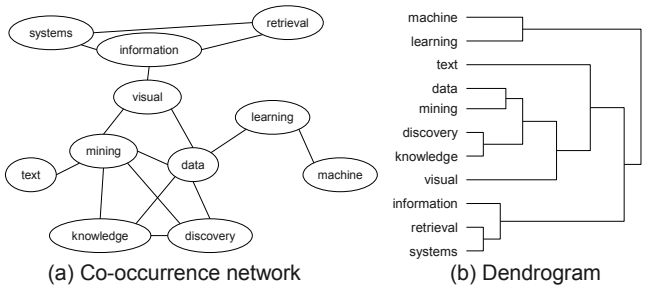


**Figure 2: Example of Hierarchical Term Clustering**

The hierarchical clustering algorithm available in the Torch tool is an algorithm for online clustering of terms [4]. This algorithm allows a fast reconstruction of the dendrogram as the co-occurrence network changes and, therefore, the update process can be performed in an incremental way.

[1] http://tartarus.org/~martin/PorterStemmer/

## 2.3 Discovering Topic Hierarchies

The dendrogram obtained from the previous module is used as the basis for building a topic hierarchy. This process involves the execution of two steps.

• **Documents mapping:** the clusters of terms of the dendrogram should be associated with sets of documents. This mapping is performed recovering all documents that have the terms of a given cluster. Thus, it is possible to obtain sets of documents with terms (labels) identifying the topics of textual collection.

• **Topics overlapping:** since a document may be associated with more than one cluster of terms during "Documents mapping", the multi-topic property is obtained naturally. However, some topics may be redundant because they can cover the same set of documents. Thus, the final topic hierarchy is constructed by combining the topics of the same level of the hierarchy according to the equation $\mid T_i \cap T_j \mid / \mid T_i \cup T_j \mid \leqslant \alpha$, where $\mid T_i \mid$ is the number of documents in the topic $T_i$ and $\alpha$ is a value that defines the level of overlap.

After performing these steps, a topic hierarchy is available for users and applications. The Torch uses a XML format as output of the results, since it is easy to understand and can be used in various applications.

It is important to note that the requirements discussed above have been met. Thus, the building of the topic hierarchies is performed incrementally and, moreover, documents can belong to more than one topic.

# 3. EXPERIMENT: BUILDING TOPIC HIERARCHIES FOR DIGITAL LIBRARIES

In order to illustrate the main functionalities of the tool Torch, we carried out a simple controlled experiment with the aim of supporting the construction of digital libraries. In our experiments we used eight different real text collection obtained from ACM's digital library. Each text collection contains about 500 articles about computer science and is organized into 40 predefined topics catalogued according to the ACM library. The aim of the experiment is to obtain a topic hierarchy from the tool Torch and compare it with the predefined topics. During this process, we inserted one article at a time to assess the incremental effect and to simulate a scenario involving growing text collections. The results were evaluated by the FScore index, which compares the topic hierarchy and predefined topics using precision and recall measures.

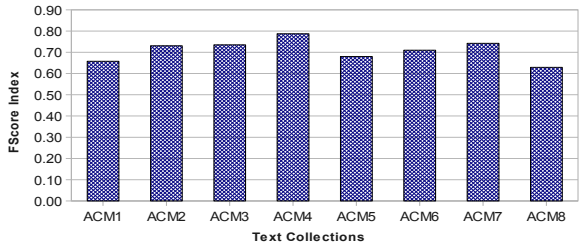


**Figure 3: FScore values obtained in the experiment**

Figure 3 shows the graph with the FScore values obtained from each text collection. The highest FScore value obtained is 0.79 while the lowest FScore is 0.63, indicating a good relationship between the topic hierarchy and predefined topics. Note that a perfect fit would have produced a FScore index of 1.0.

Some of the results are presented in Figure 4. In this figure, the left column is the predetermined topic according to ACM and the right column is the topic obtained using the Torch for the same set of documents.

| ACM Topic | Topics obtained using the Torch |
|---|---|
| VirtualReality | virtual, environ, realiti, augment |
| SoftwareReusability | softwar, reus, compon |
| MobileMultimedia | wireless, network, node |
| SoftwareEngine | test, case, studi, program, execut, suit |
| DistributedSimulation | simul, distribut, parallel |
| WebIntelligence | web, page, semant, ontolog, concept, relat, document |

**Figure 4: Examples of topics obtained using the Torch**

The results indicate that the Torch can effectively help users to build topic hierarchies from growing text collections. For example, the topic hierarchy obtained can be used as a first structure to be improved by the user or can also be used as metadata in information retrieval systems.

# 4. CONCLUDING REMARKS

In this paper we presented an overview of the Torch tool, developed with the goal of building topic hierarchies from growing text collections. The architecture of the tool and its main functionalities were described and also discussed how the tool meets the requirements: incremental clustering, understandable cluster labels, and topics overlapping.

A simple controlled experiment is carried out to illustrate the functionalities of the tool in the construction of digital libraries. In this experiment, we identified and evaluated the main topics of eight text collections.

The Torch tool and text collections are available online at: http://labic.icmc.usp.br/softwares-and-application-tools/.

In the future, we plan to improve the Torch tool by including new similarity measures between terms. Furthermore, we intend to investigate visualization techniques of topic hierarchies to facilitate analysis of results.

# 5. REFERENCES

[1] M. J. A. Berry and J. Kogan. *Text Mining: Applications and Theory*. Wiley, West Sussex, 2010.

[2] B. C. M. Fung, K. Wang, and M. Ester. *Encyclopedia of data warehousing and mining*, volume 1, chapter Hierarchical Document Clustering, pages 555–560. Information Science Reference - IGI Publishing, 2008.

[3] R. M. Marcacini, M. F. Moura, and S. O. Rezende. IFM's Digital Library: an application for organizing information using hierarchical clustering (In portuguese). In *III Workshop on Digital Libraries (WDL), XIII Webmedia*, pages 1–8, 2007.

[4] R. M. Marcacini and S. O. Rezende. Incremental construction of topic hierarchies using hierarchical term clustering. In *Proceedings of the 22nd International Conference on Software Engineering and Knowledge Engineering (SEKE)*, pages 553–558. KSI - Knowledge Systems Institute, 2010.

[5] M. F. Moura, R. M. Marcacini, B. M. Nogueira, M. S. Conrado, and S. O. Rezende. A proposal for building domain topic taxonomies. In *Proceedings of 1st International Workshop on Web and Text Intelligence (WTI), XIX SBIA*, pages 83–84, 2008.