AUTOMATIC TAXONOMY GENERATION FOR SPEECH ARCHIVES

Lee-Feng CHIEN, Chien-Chung HUANG, Jei-Wen TENG, Shui-Lung CHUANG

Institute of Information Science,
Academia Sinica, Taipei
{lfchien, villars, jackteng, slchuang} @iis.sinica.edu.tw

ABSTRACT
To facilitate browsing of speech archives, we will investigate a new research problem called taxonomy generation for speech archives in this paper. Speech archives are considered difficult to be browsed and navigated. Although the whole transcription of a spoken document might not be well recognized in a normal case, some key terms still can be recognized. In this study we propose an approach to grouping similar key terms extracted from the transcription of a speech archive into clusters and similar clusters into super clusters to form a subject taxonomy for the archive. We will report the potential merits and challenges of the proposed approach.

1. INTRODUCTION

Recently, with the rapid growth of spoken document resources, speech technology has been incorporated into information retrieval systems and has improved human computer interaction in accessing spoken information. However, most of the works in this area were focused on indexing and searching speech archives [6]. In this paper, we would like to investigate a new research problem called taxonomy generation for speech archives. In general, taxonomy-generation tools create a Yahoo!-like directory structure for navigating content in a portal or digital archive [7]. Taxonomy-generation tools complement search engines and provide high-quality searches. The demand of a well-organized taxonomy is not only in managing a text archive but also a speech archive. Especially, speech archives are considered more difficult to be browsed and navigated.

Conventional approaches for automatic taxonomy generation are document-based, i.e. the taxonomies are constructed from the documents that are used/indexed. These approaches rely on term extraction and document clustering techniques for analyzing the content similarity of the indexing documents. Similar documents can be grouped together and form a category. The subject terms of the category are extracted from that represent the key concepts of the documents. Due to some speech recognition errors, these approaches suffer from difficulties in dealing with spoken documents. Document clustering varies in accuracy is one of the major difficulties. Scalability is another concern. Whether the generated categories are comprehensive and adaptable with users’ preferences and usages is the most important problem to be dealt with.

In application to organizing speech archives, our work carefully considers the above difficulties and is exploring a new approach for taxonomy generation. We take the key terms extracted from the contents of the documents as the subject terms and develop a set of automatic clustering techniques to exploit Web resources to organize the subject terms into classes of different subject domains. The required term extraction and document clustering processes are performed under the transcribed data strings.

Although the whole transcription of a spoken document might not be well recognized in a normal case, some key terms still can be recognized. To facilitate browsing of speech archives, in our study we try to group similar key terms into clusters and similar clusters into super clusters to form a subject taxonomy for a speech archive. Since a term is short in length and simple in structure, its corresponding subject category (or categories) is difficult to judge. To cluster key terms with similar subject domains, the approach exploits the highly ranked speech documents retrieved by a key term, and extract other co-occurring terms as its feature terms and their frequency as feature values from these documents. For each key term to be clustered, it is, therefore, assigned a term vector. The term vector of a key term is constituted by the feature terms and their frequency values in the retrieved documents. A hierarchical clustering algorithm is developed to cluster the key term set extracted from a given speech archive into hierarchical clusters and creating a taxonomy for them. In fact, the above approach has been successfully applied to creating taxonomy for text archives [1, 2]. This paper is an extension of the approach for spoken document retrieval application. There might face with some problems not occurring in dealing with text archive. In this initial study, we will report the potential merits and challenges of the proposed approach in more details.

2. THE PROPOSED APPROACH

To automatically generate a taxonomy for speech archives, a set of key terms representing the spoken documents are pursued to be hierarchically clustered in our study. The proposed hierarchical term clustering approach is mainly composed of three processes: feature source collection, feature extraction, and term clustering. First, the feature source collection process retrieves the most relevant documents for the candidate key terms from real-world search engines. Next, the feature extraction process gathers a set of feature terms from the retrieved documents to characterize the feature space of the key terms. Finally, the term clustering process clusters similar key terms and generates appropriate cluster hierarchies. In the following subsections, we will describe the details of these processes.
2.1 Feature Source Collection and Feature Extraction

Since the key terms are short in length and simple in structure, we take their highly-ranked documents retrieved from on-line search engines as their feature source to judge the relevance of the key terms. To perform this process, we adopt Google as the back-end search engine. Each key term is used as a query to search and up to 100 result entries are collected. The title and description of each entry are extracted as the representation of the corresponding document, i.e. as the feature source for the key terms of concern.

After collecting the feature source, we then extract a set of feature terms from it. We adopt the n-gram method and choose bi- and tri-grams extracted from the feature source to represent the feature space of the key terms.

2.2 Term Clustering Algorithm

Clustering can be defined as a process of grouping objects into clusters whose members are closely associated in some way. There exist many different clustering algorithms. Two major styles are flat and hierarchical clustering. K-means and hierarchical agglomerative clustering (HAC) are well-known representatives respectively [3]. To generate a Yahoo!-like taxonomy to facilitate browsing of speech archives, we adopt the HAC approach to first produce a binary tree hierarchy and then hierarchically partition the binary tree into a multi-way tree. Briefly speaking, the proposed term clustering approach has two components: HAC algorithm and hierarchical cluster partitioning method. We will further describe the two components in the following subsections.

2.2.1 HAC Algorithm

An HAC algorithm groups a set of objects by using their inter-object distance matrix and constructs a binary tree from leaves to root [4]. Each non-leaf node is a cluster merged from two clusters, i.e. each cluster is a set of two clusters. First, each object forms a singleton cluster, and all the singleton clusters \( C_1, C_2, \ldots, C_n \) become the leaves of the binary tree. Second, the closest pair of clusters \( C_i \) and \( C_j \) are chosen to merge and form a new cluster \( C_{i,j} = \{ C_i, C_j \} \). The process of the two steps is iterated on the remaining unmerged clusters (including the new formed cluster) until only one unmerged cluster is left.

To execute the HAC algorithm, we first define the inter-object distance. We adopt vector-space model to represent each key term as a vector of weights of feature terms, therefore the dimension of the term vector is the number of feature terms. Given \( n \) key terms, we have \( n \) vectors \( v_1, v_2, \ldots, v_n \). Let \( T \) be the set of feature terms, \( t_j \) be the \( j \)-th feature term in \( T \), and \( v_{i,j} \) be the \( j \)-th element of the term vector \( v_i \). By adopting the conventional \( tf-idf \) term weighting scheme [5], \( v_{i,j} \), the weight of a feature term, is defined as:

\[
v_{i,j} = (0.5 + 0.5 \frac{tf_{i,j}}{\max_{i \in T} tf_{i,k}}) \log \frac{n}{n_j}
\]

where \( tf_{i,j} \) the term frequency, is the number occurrences of feature term \( t_j \) in the \( v_i \)'s corresponding feature source, and \( n_j \) is the number of key terms whose feature source contain feature term \( t_j \). The similarity of a pair of key terms is then defined as the cosine measure of the two vectors of key terms:

\[
sim(v_a, v_b) = \frac{\sum_{i \in T} v_{a,i} v_{b,i}}{\sqrt{\sum_{i \in T} v_{a,i}^2} \sqrt{\sum_{i \in T} v_{b,i}^2}}
\]

The distance between a pair of key terms is then defined as one minus the similarity of the pair of key terms:

\[
dist(v_a, v_b) = 1 - \sim(v_a, v_b)
\]

The core of an HAC algorithm is to choose a specific inter-cluster distance function. Table 1 lists three well-known inter-cluster distance functions.

<table>
<thead>
<tr>
<th>Method</th>
<th>Distance function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average-Linkage (AL)</td>
<td>( \frac{1}{</td>
</tr>
<tr>
<td>Complete-Linkage (CL)</td>
<td>( \max_{v_a \in C, v_b \in C} dist(v_a, v_b) )</td>
</tr>
<tr>
<td>Single-Linkage (SL)</td>
<td>( \min_{v_a \in C, v_b \in C} dist(v_a, v_b) )</td>
</tr>
</tbody>
</table>

In the HAC algorithm, the two children of each non-leaf node in the generated binary tree are the two merged closest clusters. Each cluster can be left or right. Since each cluster except the singleton one is a set of two clusters, we can represent the whole binary tree as a set of clusters, i.e. all nodes, composing the tree. Figure 1 shows the HAC algorithm.

2.2.2 Hierarchical Cluster Partitioning

The HAC algorithm produces a binary tree of clusters. However, we are more interested in an approach to producing a natural and comprehensive hierarchical structure such as Yahoo!. This multi-way tree representation, instead of the binary tree one, is easier and more suitable for human to browse, interpret, and do some deeper analysis.

To generate a multi-way tree from a binary tree, a top-down approach is to decompose the binary tree into several major sub-hierarchies first, and then, recursively apply the same procedure to each sub-hierarchy. To create a particular major sub-hierarchy, a suitable level of the binary tree hierarchy is chosen to be cut, where each level of the binary tree generated by HAC corresponds to each iteration of the HAC algorithm and a new cluster is also generated. Therefore, the set of clusters of a cut level \( l \) is defined as the set of remaining unmerged clusters after \( l-1 \) iterations of the HAC algorithm. The core of hierarchical cluster partitioning algorithm is thus the determination of the quality of a set of clusters of a cut level.

Let \( QC(C) \) be a function to measure the quality of a set of clusters \( C \), and \( cutClusters(l) \) be a set of clusters of a cut level \( l \),
then the hierarchical cluster partitioning problem is to find a suitable cut level \( l \) where \( QC(C_{cutClusters(l)}) \) is maximized. The generally accepted requirement of "natural" clusters is that they should be cohesive and isolated from other clusters [4]. The criterion to determine the appropriate cut level of a binary tree cluster hierarchy is then to realize this intuition. Thus, \( QC(C) \) can be defined as a product of three components: (1) \( F(C) \): a function to measure the cohesion of the clusters; (2) \( S(C) \): a function to measure the isolation of the clusters; and (3) \( M(C) \): a function to measure whether the number of clusters are proper.

\[
QC(C) = F(C)S(C)M(C)
\]

After defining the quality function of a set of clusters, we formulate the three components.

\[
F(C) = \frac{1}{N} \sum_{C \subseteq C} n_f(C)
\]

\[
f(C) = \begin{cases} 
\frac{2}{n_f(n_f-1)} \sum_{i \neq j} \text{sim}(v_i, v_j), & \text{if } n_f > 1; \\
\frac{en}{\max_{v \in v_i} \text{sim}(v, v_j)}, & \text{otherwise}.
\end{cases}
\]

Given two clusters \( C_i \) and \( C_j \), we define the isolation between them as the minimum inter-object distance between the two clusters. Note that this definition is equivalent to their single-linkage distance. Let the single-linkage distance of \( C_i \) and \( C_j \) be \( SLdist(C_i, C_j) \), the isolation of a set of clusters \( C = \{C_1, C_2, \ldots, C_k\} \) is defined as the average of the single-linkage inter-cluster distances of all member clusters in \( C \):

\[
S(C) = \frac{2}{k(k-1)} \sum_{i \neq j} \sum_{i \neq j} SLdist(C_i, C_j)
\]

Usually, a partition with neither too few nor too many clusters are preferred. Given total \( n \) initial input objects of HAC, there are at least one cluster and at most \( n \) clusters in a cut level. In a hierarchical cluster partitioning approach, we expect that the number of top-level clusters should be small, but a proper number is really hard to anticipate automatically because we have no idea of how many meaningful groups exist among the objects. However, after we set our expected number of clusters \( en \), we can define the function to measure the appropriateness of number of clusters of a cut level as follows:

\[
M(C) = 1 - \frac{nc - en}{n}
\]

where \( nc \) is the number of clusters in \( C \). This is a simple and intuitive measure of the appropriateness of number of clusters of a cut level. The measure has the maximum value at our expected number of clusters \( en \), and the larger the difference between \( nc \) (the number of clusters in \( C \)) and \( en \), the smaller the value of the measure. Now, all three components of the quality measure are defined. To choose a suitable cut level, we just need to compute \( QC \) value for each cut level and then select the one with maximum \( QC \) value. Let \( CH(C) \) be the set of clusters rooted at \( C_n \), the hierarchical cluster partitioning algorithm and the term clustering algorithm can be illustrated as in Figure 2.

![Figure 1: The hierarchical agglomerative clustering algorithm.](image)

The cohesion measure of a set of clusters can be defined as the weighted average cohesion measure of its member clusters. The cohesion measure of each member cluster \( C_i \) with \( n_i \) objects can be defined as the average similarity of all its object pairs. For a singleton cluster, the cohesion measure is positive related to the expected number of clusters \( en \) and can be assumed to be at most the maximum similarity of all inter-object similarities of all \( n \) objects (initial input objects of HAC). Thus, let \( N \) be the number of objects in a set of clusters \( C \), the formal definition can be stated as follows:

\[
F(C) = \frac{1}{N} \sum_{C \subseteq C} n_f(C)
\]

\[
f(C) = \begin{cases} 
\frac{2}{n_f(n_f-1)} \sum_{i \neq j} \text{sim}(v_i, v_j), & \text{if } n_f > 1; \\
\frac{en}{\max_{v \in v_i} \text{sim}(v, v_j)}, & \text{otherwise}.
\end{cases}
\]

2.2.3 Cluster Naming

It is not easy to automatically generate an appropriate name for a cluster. However, for browsing purpose, we choose top 5 significant feature terms that minimize the inter-object distances of the cluster to be the cluster name. Through the concepts expressed by these feature terms, users may have more clues to browse the automatically generated taxonomy.

3. THE EXPERIMENT

3.1 Experimental Environment

There were totally 76 spoken documents used in our experiment, which were four hours recordings of a TV-news channel. These recordings were transcribed both by human indexers and by machine. It is easy to imagine that in the machine-transcribed documents there were many meaningful words missing or
distorted. The obtained average accuracy are of character recognition is 0.43. As this paper aims at constructing a taxonomy tree of key terms, our first task is to extract key terms from both of the human-transcribed and machine-transcribed documents. To avoid the increasing of complexity by using an automatic term extraction method, the term extraction process was basically preformed manually. Its procedure was run as follows. First, list all character N-grams. Second, choose the terms whose document frequencies are above a certain threshold. Finally, human indexers inspect the remaining terms. All terms judged by the indexers as meaningless are discarded. Table I lists the number of terms that were identified as the key terms in the experiment. We then used the approach described in Section 2 to conduct the experiment. We employed three distance functions to construct hierarchical binary trees: complete-linkage, average-linkage, and single-linkage. Finally, the F-measure of any class is the maximum value it attains at any node in the tree, and an overall F-measure is computed by taking the weighted average of all values for the F-measure with respect to class i is defined as

$$F_{i,j} = \frac{2 R_{i,j} P_{i,j}}{R_{i,j} + P_{i,j}}$$

where $R$ and $P$ are recall and precision which are defined as $n_{ij}/n_i$ and $n_{ij}/n_j$ respectively, in which $n_{ij}$ is the number of members of class i in cluster j, $n_i$ is the number of members in cluster j, and $n_j$ is the number of members of class i. For an entire hierarchical clustering, the F-measure of any class is the maximum value it attains at any node in the tree, and an overall F-measure is computed by taking the weighted average of all values for the F-measure as given by the following:

$$F = \frac{1}{n} \sum_{i} \frac{n_i}{n} \text{max} \{F_{i,j}\}$$

3.2 The Experimental Result

Table II lists the F-measure values obtained in our experiment. The F-measure values for the machine-transcribed documents are not high enough as that for human-transcribed documents. This shows the performance of the algorithm is still affected by the not well recognition accuracy. However, the obtained clustering accuracy is still high in terms of precision rate of term clustering. To show this point, we list all the precision and recall rates for some classes in machine-transcribed documents in Table III.

Table I. The number of terms identified as the key terms.

<table>
<thead>
<tr>
<th>Term Length</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>#n-grams</td>
<td>2,385</td>
<td>445</td>
<td>397</td>
<td>116</td>
<td>40</td>
<td>17</td>
</tr>
<tr>
<td>#key terms</td>
<td>75</td>
<td>29</td>
<td>26</td>
<td>7</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>141</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table II. The obtained F-measure.

<table>
<thead>
<tr>
<th></th>
<th>Complete-Linkage</th>
<th>Single-Linkage</th>
<th>Average-Linkage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human-Transcribed</td>
<td>0.55</td>
<td>0.52</td>
<td>0.60</td>
</tr>
<tr>
<td>Machine-Transcribed</td>
<td>0.16</td>
<td>0.17</td>
<td>0.15</td>
</tr>
</tbody>
</table>

Table III. The precision and recall rate for some classes obtained from the machine-transcribed documents.

<table>
<thead>
<tr>
<th>Term</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Place</td>
<td>0.8</td>
<td>0.17</td>
</tr>
<tr>
<td>Politic</td>
<td>0.17</td>
<td>0.56</td>
</tr>
<tr>
<td>Education</td>
<td>0.5</td>
<td>0.18</td>
</tr>
<tr>
<td>Culture</td>
<td>1</td>
<td>0.07</td>
</tr>
<tr>
<td>Weather</td>
<td>0.5</td>
<td>0.15</td>
</tr>
<tr>
<td>Society</td>
<td>1</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Except Politic class and Place class, one can observe from Table III that the precision of all classes are very high, even for the machine-transcribed documents. We may draw from this a primitive conclusion: if the size of a class is appropriate, even for the machine-transcribed documents, the proposed algorithm can obtain rather good performance in the precision rate of term clustering. However, our research on the considered problem is still in the initial stage. More in-depth investigations are really necessary.

4. REFERENCES