Template Independent Object extraction using MDL and MinHash Techniques

Mr. Balaji T K
Assistant Professor
S V P C E T., Puttur
JNTA university, Anantapur.

Prof. V Janardhan Babu
Hod of CSE,
S V P C E T., Puttur
JNTA university, Anantapur.

Mr. E. Siva Prasad
PG scholar
Quba Engineering College
JNTA university, Anantapur.

Abstract— We have great useful information on web, with the increased features of web 2.0 we can present the page with rich context. As we have several tools to develop web page without knowing programming skills and providing great design using predefined templates. To improve the performance of search engines, clustering, and classification of web documents we use template detection techniques. We cluster the web documents based on the similarity of underlying template structures in the documents so that the template for each cluster is extracted simultaneously. Templates similarity of tags we are finding out with intersection areas. Using MinHash find out the unique tags and gives optimal solution with minimized cost. In Future we are going to extending with sum clustering algorithms with SAX parser implementation. Compare to DOM parser, SAX parser are detect templates with less amount of time. It can show the performance of templates detection process with clustering process. SAX parser can provides the good partition technique compare to DOM parser representation performance and scalability with time consuming process.

Index Terms—, clustering, minimum description length principle, MinHash, Template extraction.

1 INTRODUCTION

WORLD Wide Web (WWW) is widely used to publish and access information on the Internet. In order to achieve high productivity of publishing, the webpage’s in many websites are automatically populated by using common templates with contents. For human beings, the templates provide readers easy access to the contents guided by consistent structures even though the templates are not explicitly announced. However, for machines, the unknown templates are considered harmful because they degrade the accuracy and performance due to the irrelevant terms in templates. Thus, template detection and extraction techniques have received a lot of attention recently to improve the performance of web applications, such as data integration, search engines, classification of web documents, and so on [3], [10], [22]. For example, bio gene data are published on the Internet by many organizations with different formats and scientists want to integrate these data into a unified database. For price comparison services, the price information is gathered from various Internet marketplaces. Good template extraction technologies can significantly improve the performance of these applications.

The problem of extracting a template from the web documents conforming to a common template has been studied in [3], [10], [22]. Due to the assumption of all documents being generated from a single common template, solutions for this problem are applicable only when all documents are guaranteed to conform to a common template.

To overcome the limitation of the techniques with the assumption that the web documents are from a single template, the problem of extracting the templates from a collection of heterogeneous web documents, which are generated from multiple templates, was also studied. In this problem, clustering of web documents such that the documents in the same group belong to the same template is required, and thus, the correctness of extracted templates depends on the quality of clustering.

Since an HTML document can be naturally represented with a Document Object Model (DOM) tree, web documents are considered as trees and many existing similarity measures for trees have been investigated for clustering [12], [22], [24]. However, clustering is very expensive with tree-related distance measures. For instance, tree-edit distance has at least O(n1n2) time complexity [12], [22], where n1 and n2 are the sizes of two DOM trees and the sizes of the trees are usually more than a thousand. Thus, clustering
on sampled web documents is used to practically handle a large number of web documents.

Reis et al. [12] presented a method in which a small number of sampled documents are clustered first, and then, the other documents are classified to the closest clusters. In both clustering and classifying, a restricted tree-edit distance is used to measure the similarity between documents.

However, it is not easy to select proper training data of small size, since we do not have any knowledge about given data in advance. Moreover, it is hard to decide how many clusters are to be generated from the web documents. In [12], they empirically suggested 80 percent similarity threshold within a cluster but it does not work for all the time. In [24], it is assumed that labelled training data are given for clustering. However, this assumption is not valid either in many cases. In this paper, in order to alleviate the limitations of the state-of-the-art technologies, we investigate the problem of detecting the templates from heterogeneous web documents and present novel algorithms called TEXT (automatic tEmplate eXTraction). We propose to represent a web document and a template as a set of paths in a DOM tree. As validated by the most popular XML query language XPATH [2], paths are sufficient to express tree structures and useful to be queried. By considering only paths, the overhead to measure the similarity between documents becomes small without significant loss of information. For example, let us consider simple HTML documents and paths in Fig. 2 and Table 1. We will formally define the paths later. Document d1 is represented as a set of paths p1; p2; p3; p4; p6g and the template of both d1 and d2 is another set of paths p1; p2; p3; p4.

Our goal is to manage an unknown number of templates and to improve the efficiency and scalability of template detection and extraction algorithms. To deal with the unknown number of templates and select good partitioning from all possible partitions of web documents, we employ Rissanen’s Minimum Description Length (MDL) principle in [20], [21]. Intuitively, each candidate partitioning (i.e., clustering) is ranked according to the number of bits required to describe a clustering model and the partitioning with the minimum number of bits is selected as the best one. In our problem, after clustering documents based on the MDL principle, the model of each cluster is the template itself of the web documents belonging to the cluster. Thus, we do not need additional template extraction process after clustering. In order to improve efficiency and scalability to handle a large number of web documents for clustering, we extend MinHash [7]. While the traditional MinHash is used to estimate the Jaccard coefficient between sets, we propose an extended MinHash to estimate our MDL cost measure with partial information of documents. Moreover, our proposed algorithms are fully automated and robust without requiring many parameters. Experimental results with real life data sets confirm the effectiveness of our algorithms.

In summary, our contributions are as follows:

- We apply the MDL principle to our problem to effectively manage an unknown number of clusters (i.e., an unknown number of templates).

- In our method, document clustering and template extraction are done together at once. The MDL cost is the number of bits required to describe data with a model and the model in our problem is the description of clusters represented by templates.

- Since a large number of web documents are massively crawled from the web, the scalability of template extraction algorithms is very important to be used practically. Thus, we extend MinHash technique to estimate the MDL cost quickly, so that a large number of documents can be processed. Experimental results with real life data sets up to 15 GB confirmed the effectiveness and scalability of our algorithms. Our solution is much faster than...
related work and shows significantly better accuracy.

2 RELATED WORKS

The template extraction problem can be categorized into broad areas. The first area is the site-level template detection where the template is decided based on several pages from the same site. Crescenzi et al. Our algorithms to be presented later represent web documents as a matrix and find clusters with the matrix. Biclustering or coclustering is another clustering technique to deal with a matrix [13], [17], [18]. Co-clustering algorithms find simultaneous clustering of the rows and columns of a matrix and require the numbers of clusters of columns and rows as input parameters. However, we cluster only documents not paths, and moreover, the numbers of clusters of columns and rows are unknown.

3. PRELIMINARIES

3.1 HTML Documents and Document Object Model

The DOM defines a standard for accessing documents, like HTML and XML [1]. The DOM presents an HTML document as a tree structure. The entire document is a document node, every HTML element is an element node, the texts in the HTML elements are text nodes, every HTML attribute is an attribute node, and comments are comment nodes. However, we do not distinguish the type of nodes, since, as defined in [3], any type of node can be a part of a template in our problem. For instance, the DOM tree of a simple HTML document d2 in Fig. 2b is given in Fig. 3. For a node in a DOM tree, we denote the path of the node by listing nodes from the root to the node in which we use “n” as a delimiter between nodes. For example, in the DOM tree of d2 in Fig. 3.

![Fig. 3. DOM Tree of d2 in Fig. 2.](image)

the path of a node “World” is “Document<html><body><h1>Liet
World.”

3.2 Essential Paths and Templates

Given a web document collection \(D = \{d_1, d_2,...,d_n\}\), we define a path set \(P_0\) as the set of all paths in \(D\). Note that since the document node is a virtual node shared by every document, we do not consider the path of the document node in \(P_0\). The support of a path is defined as the number of documents in \(D\), which contain the path. For each document \(d_i\), we provide a minimum support threshold \(t_{a_i}\). Notice that the thresholds \(t_{a_i}\) and \(t_{a_j}\) of two distinct documents \(d_i\) and \(d_j\), respectively, may be different. If a path is contained by a document \(d_i\) and the support of the path is at least the given minimum support threshold \(t_{a_i}\), the path is called an essential path of \(d_i\). We denote the set of essential paths of an HTML document \(d_i\) by \(E(d_i)\). For a web document set \(D\) with its path set \(PD\), we use a \(PD_1*PD_2\) matrix \(M_E\) with \(0/1\) values to represent the documents with their essential paths. The value at a cell \((i; j)\) in the matrix \(M_E\) is 1 if a path \(p_i\) is an essential path of a document \(d_j\). Otherwise, it is 0.

Example 1. Consider the HTML documents \(D = \{d_1, d_2,...,d_n\}\) in Fig. 2. All the paths and their frequencies in \(D\) are shown in Table 1. Assume that the minimum support thresholds \(t_{d_1}, t_{d_2}, t_{d_3}, t_{d_4}\) are 3, 3, 3, and 4, respectively. The essential path sets are \(E(d_1) = \{p_1, p_2, p_3, p_4\}\), \(E(d_2) = \{p_1, p_2, p_3, p_4, p_5\}\), \(E(d_3) = \{p_1, p_2, p_3, p_4, p_5\}\) and \(E(d_4) = \{p_1, p_2\}\). We have the path set \(PD = p_1 [1 < i < 8]\) and the matrix \(M_E\) becomes as follows:

\[
M_E = \begin{bmatrix}
    d_1 & d_2 & d_3 & d_4 \\
    p_1 & 1 & 1 & 1 \\
    p_2 & 1 & 1 & 1 \\
    p_3 & 1 & 1 & 0 \\
    p_4 & 1 & 1 & 1 \\
    p_5 & 0 & 1 & 1 \\
    p_6 & 0 & 0 & 0 \\
    p_7 & 0 & 0 & 0 \\
    p_8 & 0 & 0 & 0 \\
\end{bmatrix}
\]

We next discuss how to determine the proper minimum support threshold of each document. The goal of introducing essential paths is to prune the paths away in advance which cannot be a part of any template. It is a kind of pre-processing to improve the correctness of clustering. If we use the same threshold for all pages, it is not reasonable because the number of documents generated by each template is not the same. Thus, we may need to use a different threshold for each page.

The template of a document cluster is a set of paths which commonly appear in the documents of the cluster. If a path is contained in most pages of the cluster, we can assume that the occurrence of the path is not probably by chance, and thus, the path should be considered as a part of the template. Contents are the paths which are not members of
the template. If a document is generated by a template, the document contains two types of paths: the paths belonging to the template and the paths belonging to the contents. To separate the paths in contents from the paths in the template, we assume that 1) the support of a path in a template is generally higher than that of a path in contents and 2) the number of the paths belonging to the template is generally greater than that of paths belonging to the contents. For the first assumption, the paths in a template are shared by the documents generated by the template but those in contents are usually unique in each document. Thus, the support of the former is higher than that of the latter. For the second assumption, the paths from the template are typically dominant in a document. For example, a snippet given in Fig. 4 consists of 42 nodes including attributes in the DOM tree. Among them, only nine nodes are contents and the others are from the template. Based on our assumption, we found empirically that the mode of support values (i.e., the most frequent support value) of paths in each document is very effective to make templates survive, while contents are eliminated. Therefore, in this paper, we use the mode of support values of paths in each document as the minimum support threshold for each document. If there are several modes of support values, we will take the smallest mode.

**Example 2.** In Fig. 2 and Table 1, the paths appearing at the document d2 are p1, p2, p3, p4, p5, and p7 whose supports are 4, 4, 3, 3, 3, and 1, respectively. Since 3 is the mode of them, we use 3 as the minimum support threshold value td2. Then, p1, p2, p3, p4, and p5 are essential paths of d2.

### 3.3 Matrix Representation of Clustering

We next illustrate the representation of a clustering of web documents. Let us assume that we have m clusters such as \( C = \{c_1, c_2, \ldots, c_m\} \) for a web document set \( D \). A cluster \( c_i \) is denoted by a pair \((T_i, D_i)\), where \( T_i \) is a set of paths representing the template of \( c_i \) and \( D_i \) is a set of documents belonging to \( c_i \). In our clustering model, we allow a document to be included in a single cluster only. That is, we have \( D_i \cap D_j = \emptyset \); for all distinct clusters \( c_i, c_j, \) and \( U1 \leq s_i < mD_i = D \). In addition, we define \( E_i \) for a cluster \( c_i \) as \( \square D_k \in D_i E (d_k) \). To represent a clustering information \( C = \{c_1, c_2, \ldots, c_m\} \) for \( D \), we use a pair of matrices \( MT \) and \( MD \), where \( MT \) represents the information of each cluster with its template paths and \( MD \) denotes the information of each cluster with its member documents. If the value at a cell \((i; j)\) in \( MT \) is 1, it means that a path \( p_i \) is a template path of a cluster \( c_j \) (i.e., \( p_i \in T_j \)). Otherwise, \( p_i \) does not belong to the template paths of \( c_j \) (i.e., \( p_i \notin T_j \)). Similarly, the value at a cell \((i; j)\) in \( MD \) is 1 if a document \( d_j \) belongs to a cluster \( c_i \) (i.e., \( d_j \in D_i \)). Regardless of the number of clusters, we fix the dimension of \( MT \) as \( |P| \times |D| \) and that of \( MD \) as \( |D| + |D| \). Columns and rows in \( MT \) and \( MD \) exceeding the number of clusters are filled with zeros. In other words, for a clustering with \( C = \{c_1; c_2; \ldots, c_m\} \), all values from \((m + 1)\)th to \(|D|\)th columns in \( MT \) are zeros, and all values from \((m + 1)\)th to \(|D|\)th rows in \( MD \) are zeros. We will represent \( M_e \) by the product of \( MT \) and \( MD \). However, the product of \( MT \) and \( MD \) does not always become \( M_e \). Thus, we reconstruct \( M_e \) by adding a difference matrix \( M_s \) with \( 0/1\)-1 values to \( M_e \) in \( C \), i.e., \( M_e = M_t \cdot M_d + M_s \).

### 3.4 Minimum Description Length Principle

In order to manage the unknown number of clusters and to select good partitioning from all possible partitions of HTML documents, we employ Rissanen’s MDL principle [20], [21]. The MDL principle states that the best model inferred from a given set of data is the one which minimizes the sum of 1) the length of the model, in bits, and 2) the length of encoding of the data, in bits, when described with the help of the model. We refer to the above sum for a model as the MDL cost of the model. In our setting, the model is a clustering \( C \), which is described by partitions of documents with their template paths (i.e., the matrices \( MT \) and \( MD \)), and the encoding of data is the matrix \( M_A \). The MDL costs of a clustering model \( C \) and a matrix \( M \) are denoted as \( L(C) \) and \( L(M) \), respectively.
Considering the values in a matrix as a random variable $X$, $Pr(1)$ and $Pr(-1)$ are the probabilities of 1s and -1s in the matrix and $Pr(0)$ is that of zeros. Then, the entropy $H(X)$ of the random variable $X$ [9], [19] is as follows:

$$H(X) = \sum_{x \in \{1, 0, -1\}} -Pr(x) \log_2 Pr(x)$$

The MDL costs of $MT$ and $M\Delta$ (i.e., $L(MT)$ and $L(M\Delta)$) are calculated by the above formula. For $MD$, we use another method to calculate its MDL cost. The reason is that the random variable $X$ in MD is not mutually independent, since we allow a document to be included in a single cluster only (i.e., each column has only a single value of 1). Thus, we encode $MD$ by $|D|$ number of cluster IDs. Since the number of bits to represent a cluster ID is $\log_2 |D|$, the total number of bits to encode $MD$ (i.e., $L(MD)$) becomes $|D| \cdot \log_2 |D|$. Then, the MDL cost of a clustering model $C$ is defined as the sum of those of three matrices (i.e., $L(C) = L(MT)$ ($L(MD)$), $L(M\Delta)$). According to the MDL principle, for two clustering models $C = (MT; MD)$ and $C' = (M'T; M'D)$, we say that $C$ is a better clustering than $C'$ if $L(C)$ is less than $L(C')$.

Then, the entropy $H(X)$ of the random variable $X$ [9], [19] is as follows:

$$H(X) = \sum_{x \in \{1, 0, -1\}} -Pr(x) \log_2 Pr(x)$$

The MDL cost of $MT$ and $M\Delta$ (i.e., $L(MT)$ and $L(M\Delta)$) is calculated by the above formula. For $MD$, we use another method to calculate its MDL cost. The reason is that the random variable $X$ in MD is not mutually independent, since we allow a document to be included in a single cluster only (i.e., each column has only a single value of 1). Thus, we encode $MD$ by $|D|$ number of cluster IDs. Since the number of bits to represent a cluster ID is $\log_2 |D|$, the total number of bits to encode $MD$ (i.e., $L(MD)$) becomes $|D| \cdot \log_2 |D|$. Then, the MDL cost of a clustering model $C$ is defined as the sum of those of three matrices (i.e., $L(C) = L(MT)$ ($L(MD)$), $L(M\Delta)$).

$$L(M) = |M| \cdot H(X).$$

### 3.5 Problem Formulation

The formal problem formulation is as follows:

**Problem 1.** Given a web document set $D$ and its essential path matrix $ME$, find the best clustering model $C$ with $MT$ and $MD$ to minimize the MDL cost $L(C)$. In the remainder of this paper, we shall investigate how to cluster a number of web documents to minimize the MDL cost.

**4 CLUSTERING WITH MDL COST**

### 4.1 Agglomerative Clustering Algorithm

Our clustering algorithm TEXT-MDL is presented in Fig. 5. The input parameter is a set of documents $D = \{d_1; \ldots; d_n\}$ where $di$ is the ith document. The output result is a set of clusters $C = \{\{c_1; \ldots; c_m\}\}$, where $ci$ is a cluster represented by the template paths $Ti$ and the member documents $Di$ (i.e., $ci = (Ti; Di)$). A clustering model $C$ is denoted by two matrices $MT$ and $MD$ and the goodness measure of the clustering $C$ is the MDL cost $L(C)$, which is the sum of $L(MT)$, $L(MD)$, and $L(M\Delta)$.

**Algorithm:**

```plaintext
algorithm TEXT-MDL(D) /* a set of document */
begin
1. $C := \{c_1, c_2, \ldots, c_m\}$ with $c_i = (E(d_i), \{d_i\})$;
2. $(c_i, c_j, c_k) := \text{GetBestPair}(C)$;
3. /* Let $c_i$ and $c_j$ be the best pair for merging */
4. /* Let $c_k$ be a new cluster made by merging $c_i$ and $c_j$ */
5. while $(c_i, c_j, c_k)$ is not empty do
6. $C := C - \{c_i, c_j\} \cup \{c_k\}$;
7. $(c_i, c_j, c_k) := \text{GetBestPair}(C)$;
8. end
9. return $C$
end
procedure GetBestPair($C$ /* a clustering model */) begin
1. $\text{MDLCost}_{\text{min}} := \infty$;
2. for each pair $(c_i, c_j)$ of clusters in $C$ do
3. $(\text{MDLCost}, c_k) := \text{GetMDLOpt}(c_i, c_j, C)$;
4. /* GetMDLOpt returns the optimal MDL cost */
5. when $c_k$ is made by merging $c_i$ and $c_j$ */
6. if $\text{MDLCost} < \text{MDLCost}_{\text{min}}$ then
7. $\text{MDLCost}_{\text{min}} := \text{MDLCost}$;
8. $(c_i^p, c_j^p, c_k^p) := (c_i, c_j, c_k)$;
9. end
10. end
11. return $(c_i^p, c_j^p, c_k^p)$;
end
```

Fig. 5. The TEXT-MDL algorithm.

**TEXT-MDL** is an agglomerative hierarchical clustering algorithm which starts with each input document as an individual cluster (in line 1). When a pair of clusters is merged, the MDL cost of the clustering model can be reduced or increased. The procedure GetBestPair finds a pair of clusters together and isolating $d_4$ is better than the other clustering $C$ grouping $d_1$, $d_2$, $d_3$, and $d_4$ altogether. Thus, we can see that it is intuitively reasonable to prefer $C$ to $C'$.
whose reduction of the MDL cost is maximal in each step of merging and the pair is repeatedly merged until any reduction is not possible. In order to calculate the MDL cost when each possible pair of clusters is merged, the procedure GetMDLCost(\(c_i, c_j, C\)), where \(c_i\) and \(c_j\) are a pair to be merged and \(C\) is the current clustering, is called in GetBestPair and \(C\) is updated by merging the best pair of clusters. As we will discuss later in detail, because the scale of the MDL cost reduction by merging a pair of clusters is affected by all the other clusters, GetBestPair should recalculate the MDL cost reduction of every pair at each iteration of while loop in line 7. Furthermore, the complexity of GetMDLCost is exponential on the size of the template of a cluster. Since it is not practical to use TEXT-MDL with a number of web documents, we will introduce an approximate MDL cost model and use MinHash to significantly reduce the time complexity.

4.2 Computation of Optimal MDL Cost

4.2.1 Optimal Template Paths of Clusters

As mentioned previously, we represent a clustering model \(C\) by two matrices \(MT\) and \(MD\) and the MDL cost \(L(C)\) is the sum of \(L(MT)\), \(L(MD)\), and \(L(M∆)\). Let us start by discussing the independence of \(L(MD)\) from \(L(C)\) and the sparsity of \(ME\). Independence of \(L(MD)\) from \(L(C)\). Because \(L(MD)\) is constant for every MD as \(|D| \approx \log2|D|\), the value of \(L(C)\) is not affected by that of \(L(MD)\). Thus, minimizing \(L(C)\) is the same as minimizing the sum of \(L(MT)\) and \(L(M∆)\) only.

Sparsity of \(ME\). Since web documents are made by different templates of various sites, the web documents seldom have common paths. In shallow depths, some paths can commonly occur in heterogeneous documents since the kinds of tags, which can be placed at the first or second depth, are limited. However, as the depth is deepened, the possibility that a path appears commonly in heterogeneous documents is decreased exponentially. Thus, in the rest of the paper, we assume that the templates, such as \(ME\), are sparse (i.e., zero is more frequent than other values in a matrix). If this assumption does not hold in an extreme case, we can add empty documents as many as the number of documents in \(D\). Then, the empty documents are represented by only zeros in \(ME\) and zeros in \(ME\) become more than a half of \(ME\). Candidacy of template paths. For a cluster \(c_i = (T_i, D_i)\) of a clustering model \(C\), only the essential paths of documents in \(D_i\) can be included in the optimal template paths \(T_i\) to minimize the MDL cost of \(C\), as shown in the following lemma:

**Lemma 1.** For a cluster \(c_i = (T_i, D_i)\) of a clustering model \(C\), if \(T_i\) is the optimal template of \(c_i\), then, to minimize \(L(C)\), \(T_i\) must be a subset of \(E_i = \{sdke, D_i, Ed(\text{dk})\}\).

**Proof.** Assume that the optimal template \(T_i\) to minimize \(L(C)\) contains a path \(p_k\) which is not included in \(E_i\). Then, in the product of \(MT\) and \(MD\) (i.e., \(MT \_MD\)), the \(k\)th row of the columns corresponding to \(D_i\) is filled with 1s. However, since \(p_k\) does not appear in any document in \(D_i\), the \(k\)th row of the columns corresponding to \(D_i\) in \(M∆\) should be filled with 1s. Now let us exclude \(p_k\) from \(T_i\). Then, in \(MT\), the number of 1s decreases by one, and thus, \(Pr(1)\) decreases. Since we assume the sparsity of the matrices (i.e., \(Pr(1) < Pr(0)\)), \(H(X)\) on \(MT\) will decrease. Now, in \(MT \_MD\), the Kth row of the columns corresponding to \(D_i\) is changed as 0s. Then, the number of 1s in \(M∆\) will decrease by \(|D|\) and that of 0s will increase by \(|D|\). It results in a smaller value of \(Pr(1)\), a larger value of \(Pr(0)\), and the same value of \(Pr(1)\) in \(M∆\). Therefore, \(H(X)\) on \(M∆\) decreases as well. Since \(H(X)\) on both \(MT\) and \(M∆\) decrease, \(L(C)\) decreases as well. It is contradictory to the assumption that \(T_i\) containing \(p_k\) is the optimal template of \(c_i\) to minimize \(L(C)\). Thus, we can conclude that \(T_i\) contains only the essential paths of the documents in \(D_i\) (i.e., \(T_i - E_i\)). Tu Decision of optimal template paths. According to Lemma 1, the optimal template \(T_i\) is a subset of the essential path set \(E_i = \{sdke, D_i, Ed(\text{dk})\}\) where \(Ed(\text{dk})\) is the set of essential paths of a document \(dk\) in \(D_i\).

However, the number of subsets of \(E_i\) is exponential on the size of \(E_i\), and furthermore, the optimal template \(T_i\) of \(c_i\) depends on the other clusters. The following example shows that the optimal template \(T_i\) of \(c_i\) can be changed depending on the template of the other cluster:

![Fig. 6. Different \(\Delta f(X)\) is for the same \(\Delta P(X)\).](image)

Recall that \(E_i\) of \(c_1\) is \(\{p_1; p_2; p_3; p_4; p_5\}\). If we examine all subsets of \(E_i\), \(L(C)\) with \(MT\) is minimized when \(T_i\) = \(\{p_1; p_2; p_3; p_4\}\) but that with \(MT\) is minimized when \(T_i\) = \(\{p_1; p_2; p_3; p_4\}\). Thus, we can see that the optimal template of a cluster can be different depending on the other
clusters. Therefore, we should consider all combinations of the subsets of Ti in C to calculate the minimum L(C) when merging a pair of clusters and it is impractical.

Example 5. For M2 in Example 1, consider two different template matrices M2, M2', and the common M2 as follows: Let us calculate the optimal Ti of α1 in each template matrix:

\[
\begin{align*}
M_2 &= \begin{bmatrix}
1 & 0 & 0 \\
1 & 0 & 0 \\
0 & 1 & 1
\end{bmatrix}, & M_2' &= \begin{bmatrix}
1 & 0 & 0 \\
1 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\end{align*}
\]

5. CONCLUSION

We introduced a novel approach of the template detection from heterogeneous web documents. We employed the MDL principle to manage the unknown number of clusters and to select good partitioning from all possible partitions of documents, and then, introduced our extended MinHash technique to speed up the clustering process. Experimental results with real life data sets confirmed the effectiveness of our algorithms.

REFERENCES