Organizing Hidden-Web Databases by Clustering Visible Web Documents

Luciano Barbosa
University of Utah
lbarbosa@cs.utah.edu

Juliana Freire
University of Utah
juliana@cs.utah.edu

Altigran Silva
Universidade Federal do Amazonas
alti@dcc.ufam.edu.br

Abstract

In this paper we address the problem of organizing hidden-Web databases. Given a heterogeneous set of Web forms that serve as entry points to hidden-Web databases, our goal is to cluster the forms according to the database domains to which they belong. We propose a new clustering approach that models Web forms as a set of hyperlinked objects and considers visible information in the form context—both within and in the neighborhood of forms—as the basis for similarity comparison. Since the clustering is performed over features that can be automatically extracted, the process is scalable. In addition, because it uses a rich set of metadata, our approach is able to handle a wide range of forms, including content-rich forms that contain multiple attributes, as well as simple keyword-based search interfaces. An experimental evaluation over real Web data shows that our strategy generates high-quality clusters—measured both in terms of entropy and F-measure. This indicates that our approach provides an effective and general solution to the problem of organizing hidden-Web databases.

1 Introduction

As the volume of information in the hidden Web grows, there is an increased need for techniques and tools that allow users and applications to uncover and leverage this information. Several applications attempt to make hidden-Web information more easily accessible, including metasearchers [13, 39], hidden-Web crawlers [2, 27], database directories [6, 11], and Web information integration systems [9, 18]. A key requirement for these applications is the ability to locate relevant hidden-Web sources. But doing so at a large scale is a challenging problem that has been largely overlooked in the literature.

The Web is estimated to contain millions of online databases [22]. Because the Web is so vast and dynamic—with new sources constantly being added and old sources removed and modified, a scalable solution for finding online databases must automatically discover the searchable forms that serve as entry points to those databases. Web crawlers have been proposed which efficiently locate online databases [3]. But even crawlers that focus the search on specific database domains retrieve an invariably diverse set of forms. These include non-searchable forms (e.g., forms for login, quote request, etc.) as well as searchable forms from multiple online-database domains.

As an important step towards uncovering hidden information, we consider the problem of online-database organization. Given a set of heterogeneous searchable forms, we aim to group together forms that correspond to *similar databases*, so that people and applications can more easily find the *right* databases and consequently, the hidden information they are seeking on the Web.

There are several challenges in organizing these forms. Notably, a scalable solution must be able to automatically parse, process and group form interfaces that are designed primarily for human consumption. In addition, because there is a very wide variation in the way Web-site designers model aspects of a given domain, it is not possible to assume certain standard form field names and structures [15]. Even in simple domains such as job search, the heterogeneity in forms is amazing. Consider, for example, the forms in Figure 1(a) and 1(b). Different terms are used to represent the same attributes: the first form uses "Job Category" and "State", whereas the second uses "Industry" and "Location" to represent the same concepts. Simple search interfaces often have a single attribute with generic labels such as "Search", and some have no labels. For example, the text field of the form in Figure 1(c) has no label—the string "Search Jobs" which appears above the text field, actually occurs outside the `FORM` tags. There are also forms that do not contain any parseable attribute names, GIF images are used instead.

The problem of organizing hidden-Web sources has received a lot of attention in the recent literature. Approaches have been proposed for both classifying [4, 10, 14, 21] and clustering hidden-Web sources [17]. These approaches can be broadly characterized as *pre-query* and *post-query* [28]. Post-query techniques issue probe queries and the retrieved results (i.e., the database contents) are used for classification (or clustering) purposes. Techniques such as [4, 14] are effective for simple, keyword-based interfaces, which

326

Figure 1. Forms in the Jobs domain. Different attribute names are used to represent the same concepts in (a) and (b). The form in (c) has no attribute labels—the string seen above the text field actually resides outside the FORM tags.

are easy to fill out automatically [2] and have little or no information pertinent to the underlying database (e.g., a form with a single attribute labeled "search"). These techniques, however, cannot be easily adapted to (structured) multi-attribute interfaces. To help automatically fill out multi-attribute forms, paradoxically, it is often necessary to first discover and organize forms in the domain, so that attribute correspondences can be found and possible values for attributes collected [16, 38].

Pre-query techniques, on the other hand, rely only on visible features of forms (see e.g., [17, 21]). Previous techniques based their classification and clustering decisions on a subset of the form content: attribute labels and available values. Thus, they are only suitable for forms whose contents are indicative of the database domain. They cannot handle, for example, simple keyword-based interfaces that are widely used to query online databases. In addition, the effectiveness of these techniques is highly dependent on the ability to extract descriptive labels for form attributes, a task that is hard to automate [17, 27]. Although the HTML standard provides a label attribute to associate descriptive information with individual form elements, it is not widely used. Since the nesting relationship between forms and labels in the HTML markup is not fixed, approaches to label extraction often use heuristics (e.g., based on the layout of the page) to *guess* the appropriate label for a given form attribute [27]. Finally, as discussed above, forms may not have descriptive labels. This makes techniques based solely on attribute names brittle.

Contributions and Outline. In this paper we describe *Context-Aware Form Clustering* (*CAFC*), a new framework for clustering online databases. Since our goal is to build a scalable solution for this problem, our approach attempts to cluster these databases based on visible features that can

be automatically and reliably extracted from the context of searchable Web forms that serve as entry points to these databases[1]. Instead of attempting to extract labels for individual form attributes, we look at a broader set of metadata associated with the database: the text (the bag of words) in the form and in the page where the form is located; and the hyperlink structure around the form pages. Not only can these features be automatically extracted, but they also provide a good indication of the domain of the hidden database. However, these features also contain information that is not directly related to the database schema and contents, and thus, constructing accurate clusters becomes more challenging due to the presence of noise in the data. To build an effective clustering solution, it is critical that the relevant pieces of the database context be identified and used as the basis for similarity computation.

The main contribution of this paper is a scalable solution to the problem of clustering online databases. We address two key issues: how to identify relevant information among the broad set of metadata associated with a database; and how to combine this information to organize online databases based on their domains. Section 2 details how we model the features related to the contents of a form page, i.e., the page in which the form is embedded. We define the form-page model, which partitions the textual contents of form pages into two feature spaces: the contents of the form and the contents of the page. We also describe *CAFC-C*, a clustering strategy based on k-means [32], which uses the form-page model and obtains clusters that are more homogeneous than if either form or page contents are used in isolation. In Section 3, we discuss how similarity induced by the presence of hubs that point to the same set of form pages can lead to substantial improvement in the homogeneity of resulting clusters. We describe *CAFC-CH*, an algorithm that combines hub information and textual contents in a novel way. In contrast to previous works on Web document clustering [20, 34], which use a similarity measure that combines these features, in *CAFC-CH*, similarity induced by hubs is reinforced by the content similarity. In Section 4, we present the results of experiments we have carried out with a set of heterogeneous form pages that includes both single- and multi-attribute forms. The results show that our strategy generates high-quality clusters—measured both in terms of entropy and F-measure. This indicates our approach is effective for clustering online databases: high-quality clusters are obtained through a completely automated process that does not require complex label extraction or manual pre-processing. Related work is discussed in Section 5 and we conclude in Section 6, where we outline directions for future work.

---

[1]We assume that the input to our clustering algorithm consists of only searchable forms. Non-searchable forms can be filtered out using techniques such as the generic form classifier proposed in [3].

2 Clustering Hidden-Web Sources using Form and Page Contents

In this section, we describe an approach for clustering online databases based on the textual contents of forms and of the pages in which forms are located. In contrast to the categorical clustering strategy proposed by He et al. [17], whose effectiveness depends on the accurate extraction of attribute labels, we cast the problem of online database categorization as document clustering. This makes our approach robust, scalable and general: the clustering process is fully automatic, requiring no manual pre-processing; and it is able to handle a wide range of forms, including single-attribute forms and forms with little or no descriptive textual attributes. However, because we use a broader and larger set of metadata, identifying the relevant portions of this metadata which are good discriminators for different databases becomes more challenging.

In what follows, we define the form-page model which serves as the basis for our clustering strategies. We discuss how it represents the textual information associated with online databases and in particular, how it models the importance of individual terms. After defining how the similarity between form pages is computed, we describe *CAFC-C*, an algorithm which applies k-means to partition the form objects using the form-page similarity measure. We conclude the section with a discussion about the pros and cons of the form-page model for clustering online databases.

2.1 The Form-Page Model

In our framework, a Web form is associated with a form page *FP*. A *FP* consists of a tuple $FP(PC, FC)$, where PC and FC correspond to two distinct feature spaces: PC represents the page contents and FC represents the form contents. Since we view both PC and FC as text, we use the vector-space model [29] as the underlying model for these feature spaces—each feature space has an associated vector, which consists of a set of (distinct) terms and their weights.

Constructing Form-Page Vectors. To construct the FC feature space, the HTML page is parsed and the contents in between the FORM tags are extracted. After the HTML markup is removed from the form contents, the terms are obtained by stemming all the distinct words. For PC, a similar process is used, except that the contents of the page are considered (i.e., all words within the HTML tags).

To generate homogeneous clusters, it is important to identify terms that are good discriminators for the database domain. Ideally, one would like to assign higher weights to *anchors*, terms that are unique to a given domain [17], and lower weights to generic terms that have high frequency in multiple domains. The TF-IDF (term frequency/inverse document frequency) measure, which is widely used in information retrieval [1], provides a natural way to model the

importance of terms as well as to eliminate noise from the context vectors. To illustrate this point, we randomly selected 30 form pages from each of the following domains: Music, Movie and Book. For each form page, we extracted and stemmed all the words within the HTML tags. Generic terms such as *privaci*, *shop*, *copyright*, *help*, have high frequency in form pages of all three domains. Clearly, these terms are not good discriminators for any domain. This is captured by the TF-IDF measure—generic terms tend to have a very low IDF value. In contrast, descriptive terms for a domain are likely to have higher IDF. For example, terms such as *flight*, *return* and *travel* have high frequency within the Airfare domain, but they have low overall frequency in the whole collection.

In the form-page model, we have adapted the traditional TF-IDF measure to also take into consideration the location of the term. Not all terms in a Web page are equally important. For example, for ranking purposes, search engines often give higher weights to terms that appear in certain places in the document, such as the document title and the anchor text in links [7]. We use the same idea here to prioritize terms that are more likely to be good domain discriminators. Term weights for both the FC and PC feature spaces are computed as follows:

$$w_i = LOC_i * TF_i * log(\frac{N}{n_i}) \qquad (1)$$

LOC_i is a small integer whose value depends on the location of the term i in the form (FC). As usual in IDF estimation, N is the total number of documents and n_i is the document frequency—the number of documents in the collection where term i appears. TF_i is the frequency of term i in the document, as it is used in vector-space model.

In the case of Web forms, by assigning a lower LOC_i value to content inside option tags, more importance is given to terms that are more likely related to the schema of the database as opposed to the database contents, which can vary widely across sites.

Computing Form-Page Similarity. The similarity between different form pages is determined by computing the distance between the corresponding vectors in each feature space. In our algorithm, we use the cosine similarity measure [1]:

$$cos(\vec{d_1}, \vec{d_2}) = \frac{\vec{d_1} \bullet \vec{d_2}}{\|\vec{d_1}\| \times \|\vec{d_2}\|} \qquad (2)$$

The cosine distance between vectors $\vec{d_1}$ and $\vec{d_2}$ is computed by dividing the dot product of the vectors by the product of their lengths. To combine the similarities of the two feature spaces, we take the weighted average of the similarity in each space:

$$sim(FP_1, FP_2) = \frac{C_1 * cos(\vec{PC_1}, \vec{PC_2}) + C_2 * cos(\vec{FC_1}, \vec{FC_2})}{C_1 + C_2} \qquad (3)$$

Algorithm 1 *CAFC-C*

1: **Input:** *formPages*, *k*
2: *centroids* = *selectSeeds*(*formPages*, *k*)
  {Randomly select seeds}
3: **repeat**
4:   *clusters* = *assignPoints*(*formPages*, *centroids*)
    {Assign form page to the closest centroid}
5:   *centroids* = *recomputeCentroids*(*clusters*)
    {Recomputing centroids}
6: **until** stop criterion is reached
7: return *clusters*

2.2 The *CAFC-C* Algorithm

To cluster similar form pages, *CAFC-C* uses k-means as the basic clustering strategy (see Algorithm 1). K-means is a partition centroid-based clustering algorithm. It is widely used in document clustering because of its effectiveness and simplicity [31].

CAFC-C takes as input the number *k* of desired clusters and a set of form pages. Initially, *k* clustering seeds are randomly selected and their centroids computed (line 2). Each cluster has an associated centroid vector (in *PC* and *FC* feature spaces) that is used to represent the cluster. The centroid vector of a cluster *C* is computed by taking the average of the weights of the terms in the different form pages in *C*:

$$\vec{c} = (\frac{\sum_{\vec{PC} \in C} \vec{PC}}{|C|}, \frac{\sum_{\vec{FC} \in C} \vec{FC}}{|C|}) \qquad (4)$$

The distance between clusters, as well as between a form page and a cluster, can then be computed using Equation 3. For simplicity, in our implementation, we assign the same weights to both form and page contents, $C_1 = C_2 = 1$.

The algorithm then iterates over the remaining form pages, assigning each form page to the cluster whose centroid is most similar to it (line 4). Then, the cluster centroids are recomputed (line 5). This process is repeated until the clusters become stable—in our implementation, until fewer than 10% of the form pages move across clusters.

2.3 The Pros and Cons of *CAFC-C*

As we discuss in detail in Section 4, *CAFC-C* leads to homogeneous clusters that have high F-measure and relatively low entropy. This indicates that textual content in and around forms, if properly used, is effective for discriminating different online database domains. By taking into account both the content of forms and of the pages in which the forms are located, *CAFC-C* is able to accurately cluster content-rich (multi-attribute) as well as simple (single-attribute) forms. In addition, the clusters obtained by combining the two feature spaces are of higher quality than clusters derived using either feature set in isolation.

CAFC-C, however, has two important limitations. Since it applies the k-means strategy, the quality of the resulting clusters is highly dependent on the initial seeds. In addition, since it considers only the textual contents associated with forms, it is prone to make mistakes for domains with highly heterogeneous vocabularies and when there are large vocabulary overlaps across different domains. Vocabulary heterogeneity in a domain leads to difficulties in creating good clusters since individual form pages in the same domain can be very distant from each other; and when there are large vocabulary overlaps, clusters tend to include similar forms from different domains. This motivated us to explore the use of hyperlink structure around form pages as a contributing factor to their similarity.

3 Using Hubs to Improve Clusters

Strategies for clustering Web documents have been proposed which, in addition to textual contents, use hyperlink structure to improve clustering effectiveness. Existing works, however, have focused either on clustering Web search results [20, 34] or on clustering document collections held by search engines [35]. For both tasks, it is assumed that detailed information is available about the hyperlink structure (e.g., the adjacency matrix for the documents) and similarity functions are proposed that capture some notions derived by hyperlink structure that may imply semantic relations. For example, HyPursuit [35] defines a measure that captures the existence of a path between two documents, the number of common ancestors, and the number of common descendants.

Although we also aim to cluster Web documents, we are dealing with a special kind of document—form pages which are very sparsely distributed over the Web [3]. And, in contrast to previous approaches to Web document, we do not assume that detailed information about the Web graph is available. Below, we explore a particular similarity notion that can be easily obtained from the hyperlink structure around form pages: the existence of common ancestors. We show that this notion gives a good indication of two forms belonging to the same database domain, and propose a new algorithm which uses this information in a pre-clustering step, which deterministically derives seed clusters for the k-means-based clustering strategy described in Section 2.

3.1 Hub-Induced Similarity

Intuitively, if a set of pages share a common backlink, they are likely to be related—the existence of a hub *H* that points to a set of pages $P_1, P_2, \ldots, P_n$ serves as an indication that these pages P_i may be related. Backlink information can be retrieved through the `link:` API provided by search engines such as AltaVista, Google and Yahoo! [5]. If it is possible to identify *good* hubs which point to form pages in the same database domain, this information can potentially be used to reinforce content-based similarity.

1-4244-0803-2/07/$20.00 ©2007 IEEE.   329

To verify the effectiveness of hub-induced similarity, we performed the following experiment. For each page in a collection of 454 form pages[2], we extracted 100 backlinks from AltaVista. Although backlink information is readily available, it is very incomplete. AltaVista returned no backlinks for over 15% of forms in our collection. To deal with this problem, we also retrieved backlinks to the root page of the site where the form is located. Using this information, we identified 3,450 distinct sets of pages that are co-cited by a hub, the *hub clusters*. Among these clusters, 69% were homogeneous, i.e., they contained form pages which belong to a single domain. In addition, there were representative homogeneous hub clusters in all domains. The high-precision of the hub-induced clusters and their high coverage of the domains indicate that this information can contribute to the derivation of homogeneous clusters. The question that remains is how to combine the content-based similarity with the hub-cluster induced similarity.

An effective solution for combining the similarity measures must take into account the fact that although hubs provide a good indication of similarity among form pages, they are not perfect. Some hub clusters are too small and do not provide enough evidence for the similarity of the forms they contain. As we discuss in Section 4, we address this problem by elimating small clusters. There are also clusters which are heterogeneous and point to form pages in multiple domains, e.g., online directories. In addition, the effectiveness of a particular similarity measure varies across form pages: whereas hub-induced similarity may be effective for some pages, content may be better for others. For example, two forms that have similar content may not share any hub. Finally, the hyperlink structure we obtain through backcrawling is incomplete, as a result, some form pages are not represented in any cluster. Thus, if a term is added to represent hub-induced similarity in Equation 3, it can be hard to determine appropriate weights for each measure.

Below, we describe a new algorithm, which instead of using a similarity measure that combines textual and page linkage information, composes these distinct feature spaces in two clustering steps. In the first step, clusters are derived based on hub-induced similarity. In the second step, these clusters are refined and expanded based on their content similarity. In essence, the content of the form pages is used to *reinforce* or *negate* the hub-induced similarity.

3.2 The *CAFC-CH* Algorithm

The *CAFC-CH* algorithm uses an extension of the form-page model that also includes backlink information. Each form page FP is now represented by a triple $FP(Backlink, PC, FC)$, where $Backlink$ consists of a list of URLs that point to FP; PC and FC represent the page and form contents, respectively. As shown in Algorithm 2,

---

[2]Details about these forms are given in Section 4.

Algorithm 2 *CAFC-CH*

1: **Input:** $formPages$, k
   {$formPages$: set of form pages and k: number of clusters required}
2: $hubClusters$ = SelectHubClusters($formPages, k$)
3: $clusters$ = CAFC-C ($formPages, k, hubClusters$)
   {Compute k-means using $hubClusters$ instead of random seeds}
4: return $clusters$

Algorithm 3 SelectHubClusters

1: **Input:** $formPages$, k {$formPages$: set of form pages and k: number of clusters required}
2: $hubs$ = generateHubs($formPages$)
3: $distanceMatrix$ = createDistanceMatrix($hubs$)
   {Compute distance between hubs}
4: $finalSeeds$ = twoMostDistant($distanceMatrix$)
   {Select two hubs that are most far apart}
5: **while** $finalSeeds.length < k$ **do**
6:    $finalSeeds$ = addDistantPoint($finalSeeds, distanceMatrix$)
7: **end while**
8: return $finalSeeds$

CAFC-CH uses the *Backlink* set to construct a set of *hub clusters* (details are given in Algorithm 3). The hub clusters are used as seeds for the content-based clustering process. The intuition behind the effectiveness two-phase approach is that the use of hub-induced similarity allows the creation of clusters that have large and accurate centroid vectors which are better representatives for the various domains than randomly selected seeds whose centroids are constructed from a single form page (see Algorithm 1).

3.3 Selecting Hub Clusters

As discussed above, hubs extracted from the set of backlinks lead to the creation of a large number of hub-clusters. A key issue that must be addressed is how to select among these the *k-best* hub clusters. Ideally, one would like to select clusters that represent all different domains, and that have representative centroids (i.e., which cover a significant subset of the vocabulary in the corresponding domain).

Algorithm 3 describes our approach to selecting the k best hub clusters. First, backlinks in the set of $formPages$ are used to generate hub clusters. We then eliminate *useless* clusters. For some form pages, all backlinks belong to the same site as the page they point to. Because these intra-site hubs do not add much information about the topic of the form page, they are eliminated from the hub collection.

After the hub clusters are created (line 2), our goal is to identify the k most representative clusters. Intuitively the k most distant clusters are more likely to represent distinct domains than clusters that are closer together. We use a greedy approach to select these.

Initially, the distance between all hub cluster pairs is computed (line 3) as described in Equation 3 and the two most distant clusters, c_1 and c_2, are selected (line 4). A cluster c is then added to the seed set (line 6) if the sum of the distances between c and the clusters c_i in the seed set is maximal, i.e., $max(\sum_{i=1}^{n} dist(c, c_i))$ where n is the size of the current seed set. The algorithm continues this process until k clusters are selected.

A potential problem with this approach for cluster selection is the presence of outliers which can cause *bad* clusters to be selected. Note however, that we are not dealing with individual documents—instead the selection process is performed over clusters, which correspond to multiple documents and which have relatively large centroids. The experimental results described Section 4 confirm that, by selecting clusters with representative centroids (in other words, by avoiding clusters that are too small), *CAFC-CH* obtains substantial improvements in cluster homogeneity.

4 Experimental Evaluation

In the experimental evaluation described below, our goals are: to verify whether visible information that can be automatically extracted from a form context provides enough features that can discriminate different database domains; and to assess whether our approach to identifying and combining these features is effective. To better understand how the individual factors contribute to the overall effectiveness of our approach, we tested different configurations, varying the algorithms used (*CAFC-C* and *CAFC-CH*); and the content considered (FC–only the form contents, PC–only the page contents, and FC+PC–the two combined). We also examined the effect of alternative clustering strategies and of differentiated weight assignment to terms.

4.1 Experimental Setup

Data Set. We tested the algorithms described in Sections 2 and Sections 3 over a set of 454 form pages. We gathered these pages from two sources: roughly half of the forms were retrieved from the UIUC repository [33]—we gathered all the forms in the repository whose pages still exist on the Web; the other half was automatically retrieved by a Web crawler [3]. The collection contains both single- and multi-attribute forms: 56 have a single attribute, and 398 have more than one attribute.[3] Forms in the collection were manually classified, and the resulting groups serve as the gold standard to evaluate our techniques. They belong to eight distinct domains: Airfare search; search for new and used automobiles; books for sale; hotel availability; job search; movie titles and DVDs; music titles and CDs; rental car availability.

---

[3]Note that we do not consider hidden attributes, i.e., those corresponding to fields with type="hidden", which are invisible to users.

The procedure used for obtaining the hub clusters for the experiments is outlined in Section 3.3. To construct the Web graph in the vicinity of these forms, we use the link: facility available in some search engines [5] and crawl backward one step from each form page in the collection. We also retrieve the root page in the site where the form is located. For each form page, we extracted a maximum of 100 backlinks. From these, we generated 3,450 hub clusters.

Evaluation Metrics for Cluster Quality. To measure the quality of the clusters derived by our algorithms, we use two standard measures: entropy and the F-measure. For each cluster c_j, we compute the probability p_{ij} that a member of cluster j belongs to class i. Using this class distribution, the entropy of each cluster is calculated using the standard formula:

$$Entropy_j = -\sum_i p_{ij} log(p_{ij}) \qquad (5)$$

The total entropy for the set of all clusters is the sum of the entropies of each cluster, weighted by the size of each cluster. Intuitively, the better the clustering solution, the more homogeneous are the clusters, and consequently, the lower is the entropy.

The F-measure provides a combined measure of precision and recall [25]. For cluster j and class i, we define

$$Recall(i, j) = \frac{n_{ij}}{n_i} \qquad Precision(i, j) = \frac{n_{ij}}{n_j}$$

where n_{ij} is the number of members of class i in cluster j, n_j is the number of members in cluster j, and n_i is the number of members of class i. The F-measure is then computed by the following formula:

$$F(i, j) = \frac{2 \times Recall(i, j) \times Precision(i, j)}{Recall(i, j) + Precision(i, j)} \qquad (6)$$

The overall F-measure for a set of clusters is computed by the weighted average of the values for the F-measure of individual clusters. A perfect clustering solution will result in an F-score of one, and in general, the higher the F-measure value, the better the clustering solution is.

4.2 Effectiveness of *CAFC*

Combining Form and Page Contents. Figure 2 shows the entropy and F-measure values obtained by *CAFC-CH* and *CAFC-C*. For the latter, the values represent the average over 20 runs of the algorithm; and for the former, the minimum cardinality of the hub clusters was set to 8 (see discussion below in "Sensitivity to Hub-Cluster Cardinality"). Note that for both algorithms, combining the contents of forms and pages leads to F-measure values that are higher and entropy values that are lower than when either page or form contents are used in isolation.

The entropy and F-measure values (0.56 and 0.74, respectively), obtained by the FC+PC configuration for *CAFC-C*, indicate that the *textual contents associated with a form are good discriminators for the database domain to*
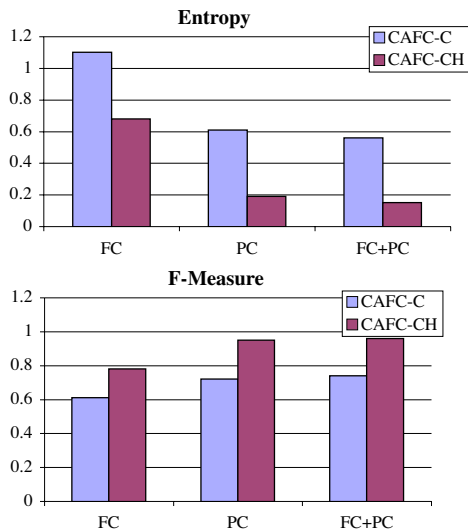
Figure 2. Entropy and F-measure values obtained by *CAFC-C* and *CAFC-CH* using only the form content (FC), only the page content (PC), and combining the page and form content (FC+PC).

Figure 3. Entropy values obtained by *CAFC-CH* varying the minimum cardinality of hub clusters. The entropy obtained by *CAFC-C* is also shown for comparison purpose.

Form size	Page terms - Form Terms
<10	162
[10,50)	131
[50, 100)	76
[100,200)	83
>=200	44

Table 1. Relationship between form and page sizes. Table shows the average number of terms in the page which are located outside the form for different form-size intervals.

which the form belongs. On the other hand, the high entropy (1.1) and low F-measure (0.61) for the FC configuration show that the form content alone does not provide sufficient information for discriminating different database domains. These results support our decision to combine the FC and PC feature spaces to compute the similarity of form pages. The intuition behind the effectiveness of this strategy is demonstrated in Table 1. The table shows, for different intervals of form sizes, the average number of terms in the form page that are located outside the form. Note that pages which contain small forms are often content-rich. In contrast, large forms are often located in pages that have little content in addition to the form. This shows that when FC is not sufficient to determine the similarity between form pages, PC has more information that may compensate, and vice-versa.

Benefits from Using Hubs. As Figure 2 shows, the use of hub-induced similarity in *CAFC-CH* leads to improvements in both entropy and F-measure values for all configurations (FC, PC and FC+PC). This indicates that, when combined with the textual content, hub clusters are very useful in determining the similarity among online databases. The benefits can be substantial. For the FC+PC configuration, the hub clusters leads to an increase of 29.7% in the F-measure, while the entropy drops to 0.15—almost one quarter of the value of the FC+PC configuration under *CAFC-C*.

Sensitivity to Hub-Cluster Cardinality. As discussed in Section 3.3, a potential problem with the hub-selection algorithm is the presence of outliers. However, this problem can be mitigated by eliminating small hub clusters. Hub clusters with too few form pages not only contain small (non-representative) centroids, but they also fail to provide sufficient evidence of similarity. To analyze the sensitiv-
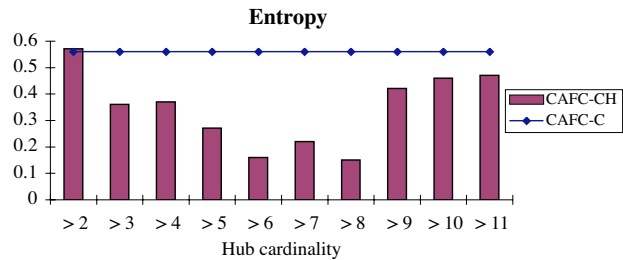
ity of our approach to different hub-cluster sizes, we ran *CAFC-CH* over hub clusters by varying the minimum cardinality allowed. The results shown in Figure 3 confirm our intuition: the best entropy values are obtained when small hub-clusters (with cardinalities less than 7) are eliminated before the greedy selection. Note that although there is a variation in the entropy for different cardinalities, *CAFC-CH* always leads to improvements over *CAFC-C*. A similar trend is observed in the F-measure—higher values are obtained when small clusters are not considered.

When small clusters are included, SelectHubClusters (Algorithm 3) selects clusters that have few elements. Although these clusters are mostly homogeneous, since they do not have enough information (their vectors are small), they lead to little improvement over *CAFC-C*. On the other hand, when only very large clusters are included (cardinality greater than 9) there are two problems: SelectHubClusters may select clusters that are too heterogeneous (e.g., directories that point to databases in many different domains); and the clusters may not represent all the domains. For instance, in our data set, hub clusters with 14 or more form pages only contain forms from Air and Hotel.

Bounding the hub cardinality is also useful for pruning the search space for SelectHubClusters. In our experiments, by eliminating small clusters, the total number of hub clusters is reduced from 3,450 to 164. This leads to a substantial reduction in the running time of the greedy selection.

Another point worthy of note is the interplay between the different components of the form-page model. In particular, the contribution of FC-induced similarity is greater when hub clusters are of low quality. When the selected hub clusters are of high-quality (e.g., when the minimum cardinality is set to 7), the decrease in entropy achieved by combining
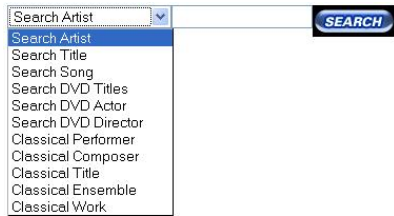
Figure 4. Example of an ambiguous form that belongs to two domains: Music and Movie domains.

FC and PC (compared to PC only) is 21% (see Figure 2). In contrast, if small clusters (with cardinality greater than 3) are selected, the decrease in entropy from PC to FC+PC is much higher—40%.

Derived Online-Database Clusters. Examination of the resulting clusters (which we do not show due to space limitations) has uncovered an interesting issue. Most of the incorrectly clustered form pages belong to the Music and Movie domains. Although, we expected this outcome, since we already knew that there was a large vocabulary overlap between the two domains, what we found is that there are forms which actually search databases that have information from both domains. An excerpt of such a form is shown in Figure 4. Another interesting observation is that, among the 17 form pages that were incorrectly clustered, only one is a single-attribute form, indicating that our approach is indeed effective for clustering single-attribute forms. Note that this particular form page is special, in the sense that it diverges from the trend shown in Table 1: there are few terms in the form as well as in the page.

4.3 Alternative Clustering Algorithms

Besides k-means, another technique that is widely employed to cluster documents is Hierarchical Agglomerative Clustering (HAC) [32]. HAC starts with the individual documents as initial clusters and, at each step, combines the closest pair of clusters. We ran variations of *CAFC-C* and *CAFC-CH* that use HAC instead of k-means as the basic clustering strategy. The results are shown in Table 2. The use of hubs leads to improvements in the homogeneity of the clusters regardless of the underlying clustering strategy: for k-means the entropy is reduced from 0.56 to 0.15 whereas for HAC it goes from 0.52 to 0.40. Note that the entropy obtained by k-means is less than half that of the HAC configuration. The smaller improvement observed for HAC is due to the fact that, unlike k-means, HAC makes local decisions. Even when HAC starts with high-quality hub-clusters, it can still make mistakes that are propagated during the agglomerative process over the remaining elements.

One widely-used technique to derive seeds for k-means is to take a sample of points and use HAC to cluster them [32]. To verify the effectiveness of this approach, we ran HAC with the best configuration (FC+PC) over

Measure/Technique	*CAFC-C* (k-means)	*CAFC-C* (HAC)
Entropy	0.56	0.52
F-measure	0.74	0.84
Measure/Technique	*CAFC-CH* (k-means)	*CAFC-CH* (HAC)
Entropy	0.15	0.4
F-measure	0.96	0.88

Table 2. HAC versus k-means.

the entire dataset and used the resulting clusters as seeds for *CAFC-C*. Although there is little difference in the F-measure values (0.93 versus 0.96), the entropy is 60% higher than the one obtained by *CAFC-CH*.

4.4 Differentiated Weight Assignment

As described in Section 2.1, we use differentiated weights for terms depending on their location on the page and in a form (the LOC_i factor in Equation 1). For the results described above, we used a simple strategy to assign weights to terms. For form contents, lower weights are given to terms inside `option` tags; and for page contents, weights given to terms inside the `title` tag are higher than for terms in the body. To verify the impact of differentiated weight assignment in the quality of the resulting clusters, we executed our best configuration (*CAFC-CH* over FC+PC) using uniform weights. Although there is little change in the F-measure value (0.96 to 0.91), there is an increase in entropy from 0.15 to 0.31. This shows that the use differentiated weights is indeed beneficial. Note, however, that the clusters derived by *CAFC-CH* with uniform weights are more homogeneous than the clusters derived by *CAFC-C* using differentiated weights.

5 Related Work

Several works have addressed different issues related to the retrieval and integration of hidden-Web data. In particular, the problem of matching and merging Web query interfaces has received substantial attention (see e.g., [16, 19, 36, 37, 38]). Some of these techniques aim to find attribute-level matchings among form interfaces, whereas others focus on merging similar forms into a unified query interface. In both cases, they require as inputs groups of similar forms such as the ones derived by our approach.

As discussed in Section 1, approaches have been proposed for both classifying [4, 10, 14, 21] and clustering hidden-Web sources [17]. Among these, the most closely related to our work is the form-clustering strategy proposed by He et al. [17]. Although we share the same goal, there are key differences between our approaches. Based on the observation that form schemas are discriminative representatives of sources, they attack the problem of source organization by translating it into the problem of clustering schemas. Although this may simplify the clustering task, it makes the effectiveness of their approach dependent on the ability to extract of descriptive attribute labels, a task that is hard to automate [17, 27]. In addition, the use of attribute

labels makes this approach unsuitable for single-attribute forms which are commonplace on the Web. The form-page model used as the basis for our clustering strategies is more complex and contains a much larger set of features. But all these features can be automatically (and reliably) extracted. Having this more comprehensive set of features also enables *CAFC* to uniformly handle both single- and multi-attribute forms. Our experimental evaluation shows that the form-page model captures discriminating features of different database domains and leads to a meaningful organization of hidden-Web sources.

CAFC was inspired by approaches to Web document clustering [34, 20, 40, 23] and borrows some of the ideas successfully applied in this area, such as the use of TF-IDF and hyperlink structure. But there are important differences. The focus of these works has been on clustering Web search results and document collections held by search engines. We are dealing with a special kind of a Web document—Web forms, and we exploit characteristics of forms to derive high-quality clusters. For example, we split the textual contents into two feature spaces—the page and form contents, and as the results of our experimental evaluation show, combining these two feature spaces leads to better clusters than using either in isolation (see Figure 2). In addition, distinctly from what happens when clustering Web pages from search engine collections, form pages are sparsely distributed over the Web and we do not have access to detailed information about the hyperlink graph of the collection. This prevents the use of existing techniques that incorporate linking information for clustering Web documents (e.g., the direct path between two pages [35]). Instead, we use backlink information provided by search engines to identify hubs.

Another key difference between our work and existing Web document clustering techniques is the way we combine the different features. Although previous works have combined textual content and hyperlink structure to improve cluster quality, they have done so by combining the different feature spaces using a unique similarity measure. In contrast, our approach uses the similarity induced by different feature sets to reinforce each other. By doing so, *CAFC-CH* avoids the need to specify the contribution of the text and structure through a weighted formula. This is similar to the idea of mutual reinforcement proposed by Huang et al. [23], where intermediate clusterings in one feature space are used to provide additional information to enhance clustering in other spaces. However, they focus on the problem of feature selection for clustering using textual contents only. An interesting direction of future work is to investigate the benefits of using their techniques for the content-based clustering component of our approach (*CAFC-C*).

Our work is also related to approaches that aim to identify Web communities [12, 24]. Based on the assumption that pages in a Web community form a dense subgraph, these techniques use information derived by the hyperlink topology (e.g., hubs and authorities) to identify these dense subgraphs. Unlike these works, we do not assume the availability of detailed information about the hyperlink topology. Instead, we use a very coarse approximation given by backlinks. However, because we also take the pages' contents into account, this coarse approximation is sufficient to bootstrap the clustering process. Nonetheless, an interesting question we would like to investigate in future work is if and how the more sophisticated link analysys techniques used to find Web communities can improve the quality of our clusters.

Recently, a number of new directories have appeared that are specialized on hidden-Web sources, e.g., [6, 26, 30]. Hidden-Web directories organize pointers to online databases in a searchable topic hierarchy. Chang et al. [8] note that these directories cover a small percentage of the hidden-Web databases; and they suggest this low coverage is due to their "apparent manual classification". *CAFC* has the potential to help automate the process of classifying the hidden-Web sources. Once the clusters are built and properly labeled with the domain name, they can be used as the basis to automatically classify new sources.

6 Conclusion and Future Work

We have presented *CAFC*, a new approach to organizing online databases. *CAFC* clusters these databases by using a rich subset of the information in the context of forms that serve as their entry points. Our experimental results indicate that our approach is effective, and that by extracting and properly combining discriminating features in the form context, it is possible to automatically construct highly-homogeneous clusters.

Because it relies only on information that can be automatically and reliably extracted from the context of forms, *CAFC* is scalable—it requires no manual pre-processing of the forms. Another key feature of our approach is the way it combines similarity information induced by the different components of the form context. Since no single component is uniformly better at discriminating groups of forms as belonging to a domain, it is hard to determine appropriate weights that reflect their importance in a combined similarity measure. *CAFC* addresses this problem combining the associated feature spaces in a way that one complements and reinforces the other. Finally, the use of a broad set of metadata in the form context, instead of just the form contents, allows *CAFC* to uniformly handle hidden-Web databases accessible through both single- and multi-attribute forms.

Although *CAFC* has achieved remarkable results, as we are dealing with real, noisy Web data, it is unlikely that per-

fect clusters can be derived in general. Thus, it is important to provide means for applications and users to explore the resulting clusters. We are currently investigating visual and query-based interfaces for this purpose.

To further improve the quality of the resulting clusters, we plan to exploit a richer set of features provided by: the hyperlink structure, e.g., anchor text and the quality of hub pages; and form contents, e.g., structural information and automatically extracted labels. Another promising avenue we intend to pursue in future work is to explore the effectiveness of our approach for Web objects other than forms.

References

[1] R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press/Addison-Wesley, 1999.

[2] L. Barbosa and J. Freire. Siphoning Hidden-Web Data through Keyword-Based Interfaces. In *SBBD*, pages 309–321, 2004.

[3] L. Barbosa and J. Freire. Searching for Hidden-Web Databases. In *WebDB*, pages 1–6, 2005.

[4] A. Bergholz and B. Chidlovskii. Crawling for Domain-Specific Hidden Web Resources. In *WISE*, pages 125–133, 2003.

[5] K. Bharat, A. Broder, M. Henzinger, P. Kumar, and S. Venkatasubramanian. The connectivity server: Fast access to linkage information on the Web. *Computer Networks*, 30(1-7):469–477, 1998.

[6] Brightplanet's searchable databases directory. http://www.completeplanet.com.

[7] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[8] K. C.-C. Chang, B. He, C. Li, M. Patel, and Z. Zhang. Structured Databases on the Web: Observations and Implications. *SIGMOD Record*, 33(3):61–70, 2004.

[9] K. C.-C. Chang, B. He, and Z. Zhang. Toward Large-Scale Integration: Building a MetaQuerier over Databases on the Web. In *CIDR*, pages 44–55, 2005.

[10] J. Cope, N. Craswell, and D. Hawking. Automated Discovery of Search Interfaces on the Web. In *ADC*, pages 181–189, 2003.

[11] M. Galperin. The molecular biology database collection: 2005 update. *Nucleic Acids Res*, 33, 2005.

[12] D. Gibson, J. M. Kleinberg, and P. Raghavan. Inferring web communities from link topology. In *UK Conference on Hypertext*, pages 225–234, 1998.

[13] L. Gravano, H. Garcia-Molina, and A. Tomasic. Gloss: Text-source discovery over the internet. *ACM TODS*, 24(2), 1999.

[14] L. Gravano, P. G. Ipeirotis, and M. Sahami. QProber: A system for automatic classification of hidden-Web databases. *ACM TOIS*, 21(1):1–41, 2003.

[15] A. Y. Halevy. Why your data don't mix. *ACM Queue*, 3(8), 2005.

[16] B. He and K. C.-C. Chang. Statistical Schema Matching across Web Query Interfaces. In *SIGMOD*, pages 217–228, 2003.

[17] B. He, T. Tao, and K. C.-C. Chang. Organizing structured web sources by query schemas: a clustering approach. In *CIKM*, pages 22–31, 2004.

[18] H. He, W. Meng, C. Yu, and Z. Wu. Wise-integrator: An automatic integrator of web search interfaces for e-commerce. In *VLDB*, pages 357–368, 2003.

[19] H. He, W. Meng, C. T. Yu, and Z. Wu. Automatic integration of Web search interfaces with WISE-Integrator. *VLDB Journal*, 13(3):256–273, 2004.

[20] X. He, H. Zha, C. H. Q. Ding, and H. D. Simon. Web document clustering using hyperlink structures. *Computational Statistics & Data Analysis*, 41(1):19–45, 2002.

[21] A. Hess and N. Kushmerick. Automatically attaching semantic metadata to web services. In *Proceedings of IIWeb*, pages 111–116, 2003.

[22] W. Hsieh, J. Madhavan, and R. Pike. Data management projects at Google. In *SIGMOD*, pages 725–726, 2006.

[23] S. Huang, G.-R. Xue, B. Zhang, Z. Chen, Y. Yu, and W.-Y. Ma. Multi-type features based web document clustering. In *WISE*, pages 253–265, 2004.

[24] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the Web for emerging cyber-communities. *Computer Networks*, 31(11-16):1481–1493, 1999.

[25] B. Larsen and C. Aone. Fast and effective text mining using linear-time document clustering. In *KDD*, pages 16–22, 1999.

[26] Profusion search engine directory. http://www.profusion.com/nav.

[27] S. Raghavan and H. Garcia-Molina. Crawling the Hidden Web. In *VLDB*, pages 129–138, 2001.

[28] Y. Ru and E. Horowitz. Indexing the invisible Web: a survey. *Online Information Review*, 29(3):249–265, 2005.

[29] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *CACM*, 18(11):613–620, 1975.

[30] Search engines directory. http://www.searchengineguide.com/searchengines.html.

[31] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 2000.

[32] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

[33] The UIUC Web integration repository. http://metaquerier.cs.uiuc.edu/repository.

[34] Y. Wang and M. Kitsuregawa. Evaluating contents-link coupled web page clustering for web search results. In *CIKM*, pages 499–506, 2002.

[35] R. Weiss, B. Vélez, and M. A. Sheldon. Hypursuit: a hierarchical network search engine that exploits content-link hypertext clustering. In *ACM Hypertext*, pages 180–193, 1996.

[36] P. Wu, J.-R. Wen, H. Liu, and W.-Y. Ma. Query selection techniques for efficient crawling of structured web sources. In *ICDE*, 2006.

[37] W. Wu, A. Doan, and C. Yu. Learning from the web to match query interfaces on the deep web. In *ICDE*, 2006.

[38] W. Wu, C. Yu, A. Doan, and W. Meng. An Interactive Clustering-based Approach to Integrating Source Query interfaces on the Deep Web. In *SIGMOD*, pages 95–106, 2004.

[39] C. Yu, K.-L. Liu, W. Meng, Z. Wu, and N. Rishe. A methodology to retrieve text documents from multiple databases. *IEEE TKDE*, 2002.

[40] O. Zamir and O. Etzioni. Web document clustering: a feasibility demonstration. In *SIGIR*, pages 46–54, 1998.