Mining Entity Attribute Synonyms via Compact Clustering

Yanen Li[1], Bo-June (Paul) Hsu[2], ChengXiang Zhai[1], Kuansan Wang[2]

[1]Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
{yanenli2, czhai}@illinois.edu
[2]Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA
{paulhsu, kuansan.wang}@microsoft.com

ABSTRACT

Entity attribute values, such as "lord of the rings" for movie.title or "infant" for shoe.gender, are atomic components of entity expressions. Discovering alternative surface forms of attribute values is important for improving entity recognition and retrieval. In this work, we propose a novel compact clustering framework to jointly identify synonyms for a set of attribute values. The framework can integrate signals from multiple information sources into a similarity function between attribute values. And the weights of these signals are optimized in an unsupervised manner. Extensive experiments across multiple domains demonstrate the effectiveness of our clustering framework for mining entity attribute synonyms.

1. INTRODUCTION

The Web contains a wealth of structured data, such as various entity databases, web tables, etc. There is a growing trend in commercial search engines to match unstructured user queries to these structured data sources. However, user expressions of such entities often do not match the canonical specifications from the data providers. For example, in the *movie* domain, the full title "the lord of the rings: the return of the king" can be specified by users as "lotr 3", "lotr: return of the king", or "the return of the king". For shoes, people may describe the standard gender value "infant" as "baby" or "toddler". Thus, entity synonym identification, the discovery of alternative ways people describe entities, has become a critical problem to bridge the above mentioned gap between data providers and consumers.

Traditionally, entity synonym research has focused on finding synonyms of named entities, where the entity itself is completely specified by the referent string. Here we are interested in finding synonyms of entity attribute values (also referred to as entity attribute synonyms throughout this paper). While the attribute values can be entity mentions, they can also be arbitrary strings (adjectives, verbs, etc). In fact, our problem definition is a generalization of finding named entity synonyms, because the named entity expression is often just an attribute of the entity. Fig. 1 illustrates an example of such general cases collected from a product title and two user issued queries. Here "canon" is a named entity, but it also matches attribute *digital-camera.brand*. And "12.1 mega pixel" is an attribute value; but it cannot be interpreted as a stand-alone entity. As seen in Fig. 1, there are a lot of variations in describing the same attribute values. Successful identification of their surface forms will enable better query intent understanding and better normalization of products from different providers, *etc*. In the case

Color Codes	brand	product-line	model	resolution	color
Provider	canon	power-shot	sd1300-is	12.1 mega pixels	black
User Query 1	canon	powershot	sd1300is	12.1 mp	black
User Query 2	cannon	ps	sd1300is	12 mp	blk

Figure 1: Entity Attribute Value Variations

the attribute value itself is an entity mention, our problem setup is the same as traditional entity synonym finding. Previous research has addressed the synonym identification problem from multiple perspectives. For example, [10, 4] tried to reconcile different references to the same database record. Other works identified alternative forms of a query for web search by measuring query similarity [6, 12], query-document click-graphs [1] and query-entity click-graphs [9]. For non-entity attribute values (arbitrary strings), there are also research efforts from the Natural Language Processing community on finding semantic synonyms based on distributional similarity [14, 15], syntactic patterns [11, 5] *et. al*.

However, two major challenges remain. First, finding synonyms without context can not handle semantic ambiguity. There are recent research attempts to identify synonyms with additional context, such as paragraph context in [22]. But for structured database, such information is not always available. Second, previous approaches usually focus on utilizing a single signal, such as distributional similarity [14, 15], syntactic patterns [11, 5], or query-entity clicks [9]. Some recent works explored more information sources [17, 7]. However, the weights for combining these information sources are usually manually tuned largely based on experience.

In this work we focus on finding synonyms for a set of entity attribute values simultaneously. Our problem setup is a generalization of the entity synonym identification problem, in which the input can be an entity mention or an arbitrary string. To address the deficiencies of existing approaches discussed above, we propose a compact clustering model that enables the integration of multiple heterogeneous information sources. Our main contributions are summarized as follows:

- ***Joint synonym mining from a set of attribute values***. Most previous synonym identification methods search for synonyms one entity at a time. However, processing a set of entity attribute values simultaneously has several advantages. First, as the values are from the same attribute, they exhibit distinctive contextual pat-

CIKM'13, Oct. 27-Nov. 1, 2013, San Francisco, USA

terns. Mining such patterns allows us to define a novel **categorical pattern similarity** function to tackle the ambiguity problem. Second, joint modeling of multiple attribute values also provides prior knowledge about the relationship among candidates. For example, for entity mention "lord of the rings 2", "lord of the rings 3" could be identified as synonym mistakenly. But if we learn synonyms from those two values jointly, this error could be corrected easily because of the awareness of "lord of the rings 3".

- ***Integrating multiple information sources.*** Synonym values generally exhibit similarities in more than one aspect. Some synonym values only differ in a few characters, due to spelling errors or morphological differences. Also, queries that differ only in synonym values tend to have clicks on similar sets of documents. In addition, synonym values generally have similar surrounding contexts within queries and documents. Among these signals, some are more important than others in determining synonym relations. Furthermore, the relative importance of these signals also depends on the domain: a feature that is crucial in the movie domain might be only marginal in the camera domain. Therefore automatic determination of the weights of different information sources is critical. In this work we propose to automatically learn these weights via **compact clustering** – a novel clustering procedure that maximizes the similarity of points within a cluster.

2. RELATED WORK

There is a rich body of work on the general topic of automatic synonym discovery. This topic can be divided into sub-areas, including finding word synonyms, entity/attribute synonyms, and related query identification. Identifying word level synonyms from text is a traditional topic in the NLP community. Such synonyms can be discovered using simple dictionary based methods such as WordNet; distributional similarity based methods [14, 15]; and approximate string matching approaches [18]. In this work we focus on finding entity attribute synonyms, which usually have more domain context than plain words.

Researchers have employed several similarity metrics to find synonyms from web data. Such similarities include distributional similarity [14, 15], coclick similarity [9, 7], pointwise mutual information [21], and co-occurrence statistics [3]. Unlike these works, our work introduces a novel similarity metric called categorical pattern similarity for jointly finding synonyms from a set of attributes.

Although several similarity metrics have been introduced to find synonyms, most previous approaches use only a single metric. [7] tries to combine multiple metrics, however they manually choose a set of thresholds for individual metrics, leading to a precision oriented approach. Instead, our approach combines the metrics with weights, and learns these weights automatically in an unsupervised manner. In this sense our work is also related to the previous works on semi-supervised metric learning [23, 20, 2]. We differ from these works in that our metric learning approach is embedded in the compact clustering framework.

3. PROBLEM DEFINITION

In our problem setup, the input consists of (1) a set of canonical entity attribute values from a domain; (2) candidate synonyms of the canonical attribute values. And the output is the true synonyms of this set of attribute values. As there are multiple interpretations of "alternative expressions", we focus on synonyms that convey the equivalent meaning of the canonical value in the (implied) domain, including semantic alterations, abbreviations, acronyms, permutations, spelling errors, *etc*. For example, for the input "IBM", the synonyms include "International Business Machines", "Big blue", "IBM corporation" *etc*.

Formally, given a set of K semantically distinct values $\mathbb{V} = \{v_1, v_2, ..., v_K\}$ from an unspecified entity attribute, where each value $v \in \mathbb{V}$ is represented by a canonical string expression, such as "5d mark iii" for *camera.model*. From a set of N candidate synonym values $\mathbb{X} = \{x_1, \ldots, x_N\}$, we can define an oracle mapping $\mathcal{F} : \mathbb{X} \to \mathbb{V} \cup \{v_0\}$, which assigns each candidate value x to its unique canonical synonym value v, or if x is not a synonym of any value in $\mathbb{V}$, to the special background value v_0. Note that we assume each candidate synonym expression maps to at most one canonical value. Now, we can define the synonym identification problem as follows:

Definition 1: For each canonical attribute value $v \in \mathbb{V}$, find the subset $\mathbb{X}_v = \{x \in \mathbb{X} | \mathcal{F}(x) = v\}$, representing the set of synonym expressions for value v.

Note that we assume values in $\mathbb{V}$ are semantically distinct and homogeneous. This assumption is reasonable in several application scenarios. For instance, for product providers such as eBay and Amazon, a set of distinct and homogeneous canonical attribute values can be easily obtained from product catalog. The homogeneous assumption implies that the inputs are from the same domain, which can be leveraged for mining their synonyms collectively.

4. COMPACT CLUSTERING

As mentioned in the introduction, most previous synonym identification methods search for synonyms one input at a time. However, such strategy have two major drawbacks. First, without modeling the attribute values jointly, it's very difficult to tackle the ambiguity problem since the category context implied by a set of attribute values is lost. Second, this strategy doesn't leverage the prior knowledge multiple attribute values bring to the candidates. In order to take advantage of a set of canonical attribute values, we propose to identify synonyms of attribute values by a clustering model with multiple similarity kernels called ***compact clustering***. In this model, attribute values $\mathbb{X} = \{x_1, x_2, ..., x_N\}$ are modeled as data points. And points are connected with others with similarity function f. Data points form clusters such that points in the same cluster are considered synonyms. In this section we first define the similarity kernel functions. Then we introduce a basic model by motivating the concept of cluster compactness. By addressing the limitations of this model, we propose several extensions that lead to the standard compact clustering model.

4.1 Similarity Kernels

In our clustering framework, data points (attribute values) are related to each other in different aspects. For example, in the domain of *movie.title*, two titles are similar if people click on the same set of documents after querying for these titles. Two titles also are similar if they follow similar lexical distribution. In fact, there are heterogeneous types of information that can be leveraged to infer the synonym relationship. Suppose from information type t, the similarity of points x_i and x_j is defined as a similarity kernel $f_t(x_i, x_j) \in [0, 1]$, and each type of similarity kernel is associated with a weight w_t reflecting its relative importance, then the overall distance between x_i and x_j can be defined by these similarity kernels. Here we define the distance between x_i and x_j as the combination of the similarity kernels with weights:

$$d(x_i, x_j) = \sum_{t=1}^{T} w_t^{\alpha} \cdot d_t(x_i, x_j) \quad (1)$$
$$= \sum_{t=1}^{T} w_t^{\alpha} \cdot (1 - f_t(x_i, x_j))$$

where $f_t(x_i, x_j) \in [0, 1]$ is the similarity kernel of x_i and x_j cal-

culated based on evidence from information source $t \in \{1, ..., T\}$. Likewise $d_t(x_i, x_j) = 1 - f_t(x_i, x_j) \in [0, 1]$ is the distance between x_i and x_j. α is a constant whose value is set to 2 in this work. And $w_t \geq 0$ are the weights needed to be learned, following constraint $\sum_{t=1}^{T} w_t = 1$.

The special choice of α is to make the optimal w_t easier to solve under the above constraint, as introduced in previous work [8]. In the following, we specifically define four similarity kernels according to four types of information. Note that our framework is not restricted to these kernels. In fact our model can support arbitrary number of similarities from different information sources.

1. **Categorical pattern similarity**. This is a novel similarity kernel which leverages a set of attribute values simultaneously. A key insight is that canonical values in the same category should share common lexical or semantic patterns. Table. 1 illustrates the pattern distribution over 50 attribute values from *shoe.brand*. These patterns are found by extracting the left and right lexical context from a set of search queries. It clearly shows that the brand names are much more likely to appear at the beginning of a query (#EMPTY# pattern on the left); and the word "shoes" is frequently following the brand name. By mining the this context, we are able to discover categorical patterns, which would otherwise be impossible had we looked for synonyms one attribute value at a time due to data sparseness. Specifically, given data points x_i, x_j and the left and right categorical pattern distributions $\bar{\Omega}_l, \bar{\Omega}_r$ derived from the canonical attribute values, we define the categorical pattern similarity between x_i and x_j as:

$$f_1(x_i, x_j) = 1 - |Jaccard(\Omega_i, \bar{\Omega}) - Jaccard(\Omega_j, \bar{\Omega})| \quad (2)$$

where $Jaccard(\Omega_i, \bar{\Omega})$ is the average $Jaccard$ similarity of the left context and right context between x_i ($\Omega_{i,l}, \Omega_{i,r}$) and the category ($\bar{\Omega}_l, \bar{\Omega}_r$):

$$Jaccard(\Omega_i, \bar{\Omega}) = \frac{1}{2} \cdot \left(\frac{||\Omega_{i,l} \cap \bar{\Omega}_l||}{||\Omega_{i,l} \cup \bar{\Omega}_l||} + \frac{||\Omega_{i,r} \cap \bar{\Omega}_r||}{||\Omega_{i,r} \cup \bar{\Omega}_r||} \right) \quad (3)$$

Note that the categorical pattern similarity kernel is large only if both x_i and x_j share similar context distributions with the categorical patterns, which is especially effective for excluding the ambiguous candidate strings. For example, for the canonical value "Apple" in the domain of IT companies (implied by inputs "Apple", "IBM", *etc.*), a candidate "Apple fruit" will have very low categorical pattern similarity because this candidate has very different query context.

Table 1: Categorical Patterns in Shoe.brand

Left Patterns	Count	Right Patterns	Count
1. #EMPTY#	3823	1. #EMPTY#	333
2. www	55	2. shoes	109
3. cheap	38	3. com	67
4. discount	35	4. boots	60
5. women	30	5. sandals	42
...		...	

2. **Coclick similarity.** Two attribute values are similar if users click on similar documents when they issue queries containing the two attribute values (proxy queries). Let the set of proxy queries of x_i be $Q_i = \{q_1^i, q_2^i, ..., q_{n_i}^i\}$. For each query q_l^i, the users have clicks on a set of documents, which is denoted as $\Phi^l = \{\phi_1^l, \phi_2^l, ..., \phi_M^l\}$, where M is the total number of documents. And let the accumulation of these clicks be:

$$\Phi = \sum_l \phi^l = \left\{ \sum_l \phi_1^l, \sum_l \phi_2^l, ..., \sum_l \phi_M^l \right\} \quad (4)$$

Then for points x_i and x_j, we define their coclick similarity as the cosine similarity of Φ_i and Φ_j:

$$f_2(x_i, x_j) = \frac{\Phi_i \cdot \Phi_j}{||\Phi_i|| \cdot ||\Phi_j||} \quad (5)$$

3. **Lexical context similarity.** Under the distributional similarity assumption [16], two strings will carry similar meaning if they share similar context. We observe that for true synonyms, the two attribute values will share common left and right context in web search queries. However this similarity is different from the categorical pattern similarity in that the lexical context similarity is more specific to a particular attribute value while the categorical pattern similarity is related to the patterns of a set of values. We define the lexical context similarity of x_i and x_j as the Jaccard similarity of their left and right context:

$$f_3(x_i, x_j) = \frac{1}{2} \cdot \left(\frac{||\Omega_{i,l} \cap \Omega_{j,l}||}{||\Omega_{i,l} \cup \Omega_{j,l}||} + \frac{||\Omega_{i,r} \cap \Omega_{i,r}||}{||\Omega_{i,r} \cup \Omega_{j,r}||} \right) \quad (6)$$

4. **Pseudo document similarity.** This similarity kernel has been successfully applied to finding entity synonyms [7]. It essentially measures the similarity between two attribute values based on the number of co-occurrences in the query-clicked pseudo document pairs. Please refer to [7] for more detail.

4.2 Basic Model

After defining the overall distance function and similarity kernels, we now describe the formulation of the clustering model. As for a clustering model, we must specify the cluster centers. For the attribute synonym finding problem it's natural to nominate the canonical attribute values as the cluster centers since they should be close to their synonyms. Moreover, synonymous attribute values should be close with each other in a cluster and far away from other clusters, which motivates our **compact clustering** model. Formally, in the basic model we aim at minimizing the following objective function:

$$\begin{aligned} & g_0(R, Z, W) \quad (7) \\ &= \sum_{k=1}^{K} \sum_{i=1}^{N} r_{i,k} \cdot d(x_i, z_k) + \sum_{i=1}^{N} r_{i,0} \cdot d(x_i, z_0) \\ &= \sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{t=1}^{T} r_{i,k} \cdot w_t^2 \cdot d_t(x_i, z_k) + \sum_{i=1}^{N} r_{i,0} \cdot \gamma \end{aligned}$$

subject to:

$$\begin{cases} \sum_{k=0}^{K} r_{i,k} = 1, 1 \leq i \leq N \\ r_{i,k} \in \{0, 1\}, 1 \leq i \leq N, 0 \leq k \leq K \\ w_t \geq 0, \sum_{t=1}^{T} w_t = 1, 1 \leq t \leq T \end{cases} \quad (8)$$

The above objective function is the sum of within-cluster dispersions. In *Eq.* (7), the first term is the overall within-cluster distances of the normal clusters, and the second term is the within-cluster distances in the background cluster. Such formulation is to make the resulting clusters more compact. Note that in our model there is no need to represent data points with explicit feature vectors, instead, we only require that $d(x_i, x_j) \geq 0$. The notations of variables in the formula are listed below:

- $d(x_i, z_k)$ is the overall distance function between x_i and z_k, as defined in *Eq.* (1);
- R is an $N \times (K+1)$ partition matrix, where N is the total number of points and $K+1$ is the number of clusters; $r_{i,k} \in \{0,1\}$ indicates whether object x_i is in k^{th} cluster;
- $Z = \{z_0, z_1, ..., z_K\}$ are the medoids of the clusters. In the basic model, the first K medoids are fixed to the target attribute values $\{v_1, v_2, ..., v_K\}$ for which we look for synonyms;
- $W = \{w_1, w_2, ..., w_T\}$ are the weights of distance kernels;
- γ is a constant measuring the distance of $x \in \mathbb{X}$ to the background cluster.

Rationale of the objective function: The above objective function is similar to K-medoids[13]. The advantage of this framework compared to K-means is that the distance function between data points can be defined in arbitrary form. However, there are important differences between our basic model and the K-medoids model: firstly, the first K medoids in our model are fixed to the canonical attribute values, assuming they are best representatives of these clusters. Secondly, in our model the distance between points is a weighted distance function, which is very different from the standard K-medoids model. Thirdly, in our model we add a background cluster in order to attract the random points.

Although the basic compact clustering model can partition the data points into synonym clusters, it suffers from the following limitations: (1) Using a single fixed representative for a cluster may be problematic. First, the canonical value is not always the most popular or most representative. It may have idiosyncrasies that are not shared by other members of the cluster. Second, because the similarity features are noisy, if we only compare a candidate against the canonical value, a noisy feature may bias it towards an incorrect cluster. (2) Manually setting the constant γ is very difficult. (3) No measurement of uncertainties of a point belonging to the background.

4.3 Standard Model

Generally, Limitation 1 can be addressed by employing a flexible representative or a small set of representatives for each cluster. However it's not desirable to have flexible medoids since in our problem setup the canonical values are good representatives and it is more robust to include them into the medoids. Therefore we propose to use a small subset of points, including the canonical value, to form a new pseudo-medoid. The subset is viewed as a committee that determines which other points belong to the cluster. A similar idea of clustering with committees of points has been successfully applied to the document clustering problem [19]. Specifically, in our new proposal, we form the new pseudo-medoid by including the $L-1$ most similar values to the canonical value as well as the canonical value itself.

To address Limitation 2, we propose to randomly select μ proportion of points from the background cluster, and estimate γ by taking the average of the distance from x to this random subset. Results show that the final synonyms are stable with respect to different setting of μ.

We address Limitation 3 by introducing a prior probability p that a given point x belongs to the background cluster. If we further assume x follows a uniform prior distribution for normal clusters, then the prior probability of x belonging to a normal cluster is $\frac{1-p}{K}$.

Based on these new proposals, we present the **standard compact clustering** model by minimizing the updated objective function:

$$
\begin{aligned}
& g_1(R, Z', W) \qquad (9)\\
=& \frac{1-p}{K} \sum_{k=1}^{K} \sum_{i=1}^{N} r_{i,k} \cdot d(x_i, z'_k) + p \sum_{i=1}^{N} r_{i,0} \cdot d(x_i, z_0) \\
=& \frac{1-p}{K} \sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{x_j \in z'_k} \sum_{t=1}^{T} \frac{1}{|z'_k|} \cdot r_{i,k} \cdot w_t^2 \cdot d_t(x_i, x_j) \\
& + p \sum_{i=1}^{N} \sum_{x_j \in A} \sum_{t=1}^{T} \frac{1}{|A|} r_{i,0} \cdot w_t^2 \cdot d_t(x_i, x_j)
\end{aligned}
$$

subject to *Eq.* (8). Where z'_k is the pseudo-medoid, A is the subset of random points in the background cluster, whose size is controlled by the parameter μ. And the prior probability p is a tunable parameter. The standard compact clustering model aims at inducing more compact clusters.

4.4 Solving the Standard Model

In the standard model there are three sets of unknown variables: R, Z' and W, which are dependent on each other. There is no exact solution to solve all of them at the same time. Instead we solve this optimization problem by iteratively solving the following minimization problems:

1. Fix $Z' = \hat{Z}'$ and $W = \hat{W}$; find the best R that minimizes $g_1(R, \hat{Z}', \hat{W})$
2. Fix $W = \hat{W}$ and $R = \hat{R}$; find the best medoids Z' that minimizes $g_1(\hat{R}, Z', \hat{W})$
3. Fix $Z' = \hat{Z}'$ and $R = \hat{R}$; solve the best parameters W that minimizes $g_1(\hat{R}, \hat{Z}', W)$

Sub-problem 1 (cluster assignment) can be solved by:

$$
\begin{cases} r_{i,k} = 1 & \text{if } \ d'(x_i, z'_k) \le d'(x_i, z'_l), 0 \le k, l \le K \\ r_{i,k} = 0 & \text{otherwise} \end{cases} \qquad (10)
$$

where

$$
\begin{cases} d'(x_i, z'_k) = \dfrac{1-p}{K} \cdot d(x_i, z'_k) & \text{if } \ k > 0 \\ d'(x_i, z'_k) = p \cdot d(x_i, z'_0) & \text{if } \ k = 0 \end{cases}
$$

For **sub-problem 2**, we update the pseudo-medoids of first K clusters by including up to the top $L-1$ most similar values to the canonical value as well as the canonical value itself:

$$
z'_k \leftarrow v_k \cup \{L-1 \text{ nearest neighbors of } v_k \text{ in cluster } k\} \qquad (11)
$$

For the background cluster, there is no need to calculate the updated medoid. We follow the basic ideas from weighted K-means [8] to solve **sub-problem 3**. Because after fixing R and Z, *Eq.* (9) is a convex quadratic function, we apply the Lagrange Multiplier method and obtain a closed form solution to W (not shown due to the page limitation). Intuitively, a larger weight is assigned to a feature function which makes the clusters more compact.

5. EXPERIMENTS AND RESULTS

To test the effectiveness of our proposed compact clustering model, we first make direct comparison of our model to the baselines on the traditional setting that the attribute values are also entities mentions. We then conduct another set of experiments on the setting that the attribute values are arbitrary strings. After that, we will show results in cases where the attribute values have ambiguous senses. Furthermore, we investigate the relative importance of similarity kernels.

5.1 Datasets and Evaluation Metrics

In order to evaluate the proposed models, we have collected several attribute synonym datasets from multiple categories (see Table 2). Specifically, 3 datasets are constructed to test the traditional entity synonym finding. 3 other sets are selected to test the synonym identification where the attribute values don't look like entity mentions. Furthermore, we have collected a set of ambiguous attribute values to discuss the challenging issue of ambiguity. Because obtaining a set of ambiguous values from a single category is hard, we get the results from 5 datasets, select 18 such ambiguous values and then label them. In terms of evaluation metrics, We evaluate our system based on the standard expected precision, expected recall and expected F1 measure.

Table 2: Test Datasets

Type	Dataset	#Values	#Labels	%Positive
entity mentions	movie.title	50	3272	15.9
	shoe.brand	50	3370	17.2
	doctor.specialty	50	2105	12.5
arbitrary strings	shoe.gender	5	96	19.8
	babyclothing.age	15	129	21.0
	movie.genre	21	340	15.4
ambiguous values	shoe.brand	6		
	movie.title	3		
	movie.genre	3		
	itcompany.name	3	410	16.8
	insurance.provider	3		

5.2 Baselines

1. ***Individual features***. Individual features are included as baselines so as to reveal their strength and weakness on identifying entity attribute synonyms both in the form of entity mentions as well as arbitrary strings. Synonyms are identified by single attribute value at a time. We try several settings and manually choose the best thresholds for these feature functions.

2. ***Chakrabarti-2012***. We also include a strong baseline proposed by Chakrabarti *et. al* [7], which identifies entity synonyms by combining multiple similarity scores with manually tuned thresholds. We consider it a state-of-the-art multi-feature, single value at a time approach. For a fair comparison, this system works on the same set of query log and clickthroughs as our approach for calculating similarities. We collect final outputs in the form of an unordered list of synonyms for each input attribute value via the system interface provided by the authors of [7].

3. ***Clustering with Fixed Weights***. In order to reveal the effectiveness of the kernel weights learning, we add a baseline that uses the same clustering model, yet with fixed (equal) kernel weights.

5.3 Entity Mentions

We first evaluate the performance of the compact clustering model on attribute values that are also entity mentions. Among the three test datasets, *movie.title* and *shoe.brand* are from popular domains while *doctor.specialty* is from tail domain. Table 3 shows the expected precision, recall, and F1 scores. Firstly, the results show consistently across three datasets that using single feature doesn't achieve competitive results. Specifically, *categorical pattern* has somewhat good precision but suffers from very low recall. *pseudo document* similarity is a relatively robust method achieving balanced precision and recall. However it fails to get competitive performance compared to methods combining multiple features such as *Chakrabarti-2012* and our model. Secondly, the *Chakrabarti-2012* approach achieves relatively high on precision but low on recall, confirming its precision orientated nature. Thirdly, learning synonyms jointly in our clustering framework clearly demonstrates advantages: it achieves better F1 scores than *Chakrabarti-2012* across three datasets by simply fixing the weights to be all equal. Finally, our proposed *compact clustering* model is consistently obtaining balanced precision and recall, resulted in best F1 scores. It clearly outperforms the baseline of clustering with fixed weights, showing the benefit of automatic weight learning. Moreover, its F1 score also consistently outperforms *Chakrabarti-2012*. In fact, in two of the three datasets, the statistical T-Test indicates that there is statistically significant difference between our model and *Chakrabarti-2012* at confidence level $p = 0.01$. This reveals the effectiveness of our proposed model that identifies synonyms jointly with kernel weights automatically tuned.

Table 3: Attribute Values as Entity Mentions

Dataset	Method	Precision	Recall	F1
movie.title	categorical pattern	0.463	0.202	0.2811
	coclick	0.381	0.405	0.393
	lexical context	0.398	0.372	0.385
	pseudo document	0.412	0.437	0.424
	Chakrabarti-2012	**0.706**	0.400	0.470
	clst. w. fixed weights	0.503	0.525	0.514
	compact clustering	0.541	**0.572**	**0.556***
shoe.brand	categorical pattern	0.455	0.258	0.329
	coclick	0.425	0.446	0.435
	lexical context	0.431	0.418	0.424
	pseudo document	0.454	0.477	0.465
	Chakrabarti-2012	0.762	0.470	0.545
	clst. w. fixed weights	0.713	0.510	0.595
	compact clustering	**0.768**	**0.560**	**0.647***
doctor.specialty	categorical pattern	0.398	0.19	0.257
	coclick	0.359	0.337	0.348
	lexical context	0.365	0.328	0.346
	pseudo document	0.380	0.359	0.369
	Chakrabarti-2012	**0.683**	0.520	0.590
	clst. w. fixed weights	0.665	0.543	0.598
	compact clustering	0.673	**0.558**	**0.610**

Note: The F1 score marked by * means it has statistically significant difference compared to *Chakrabarti-2012* at confidence level $p = 0.01$. The same as in the remaining tables.

5.4 Arbitrary Strings

We then compare the results on attribute values that don't look like entity mentions. Such values include interesting instances like infant, women in *shoe.gender*, thriller in *movie.genre*, 2 year in *babyclothing.age*. We summarize the results in Table 4. As expected, *categorical pattern*, *pseudo document* behave similarly as in the previous experiment, confirming using them individually is not effective in both forms of attribute values. Also, the clustering with fixed weights performs slightly better than *Chakrabarti-2012*. Further, the compact clustering model achieves significantly better results than *Chakrabarti-2012* across three datasets. We list some interesting cases that our proposed model identifies successfully but *Chakrabarti-2012* fails. For example, for thriller, *Chakrabarti-2012* finds "michael jackson thriller" as its synonym while compact cluster doesn't. In fact, "michael jackson thriller" is not the synonym of thriller in the particular domain of *movie.genre*. And our model identifies "scary" as its synonym, which is more appropriate. The superior performance of compact clustering might be due to two reasons: first is that we aggregate all referent strings of the attribute value as proxies, therefore resulting in more robust estimate of similarity measures. And second, the joint modeling of multiple attribute values from the same implied domain effectively handles the ambiguity problem, which we will further discuss below.

Table 4: Attribute Values as Arbitrary Strings

Dataset	Method	Precision	Recall	F1
shoe.gender	categorical pattern	0.371	0.179	0.242
	coclick	0.343	0.362	0.352
	lexical context	0.360	0.336	0.348
	pseudo document	0.370	0.400	0.384
	Chakrabarti-2012	0.489	0.372	0.423
	clst. w. fixed weights	0.485	0.502	0.493
	compact clustering	**0.500**	**0.550**	**0.524***
babyclothing.age	categorical pattern	0.382	0.190	0.254
	coclick	0.412	0.455	0.432
	lexical context	0.462	0.421	0.441
	pseudo document	0.401	0.544	0.462
	Chakrabarti-2012	0.592	0.380	0.463
	clst. w. fixed weights	0.562	0.466	0.510
	compact clustering	**0.669**	**0.543**	**0.599***
movie.genre	categorical pattern	0.355	0.140	0.201
	coclick	0.325	0.355	0.340
	lexical context	0.343	0.312	0.327
	pseudo document	0.331	0.362	0.346
	Chakrabarti-2012	0.591	0.482	0.531
	clst. w. fixed weights	0.580	0.554	0.567
	compact clustering	**0.594**	**0.588**	**0.591***

5.5 Ambiguous Attribute Values

Ambiguous synonyms handling is important for finding domain specific synonyms. Here we compare our model to *Chakrabarti-2012* on a set of attribute values that are ambiguous. They include {jordan, coach, lv} from *shoe.brand*, {app, sun, adobe} from *it-company.name*, {aarp, advantage, aim} from *insurance.provider*, {thriller} from *movie.genre*, {matrix} from *movie.title*. Results on Table clearly indicate that compact clustering is much more effective than *Chakrabarti-2012* on handling ambiguous attribute values. Interestingly, the baseline of clustering with fixed weights also significantly outperforms *Chakrabarti-2012* in this case, suggesting that joint modeling of multiple attribute values is particularly effective for ambiguous synonyms handling.

Table 5: Ambiguous Attribute Values

Method	Precision	Recall	F1
Chakrabarti-2012	0.581	0.465	0.517
clst. w. fixed weights	0.613	0.574	0.593*
compact clustering	**0.677**	**0.582**	**0.626***

5.6 Contribution of Similarity Kernels

Our proposed model is able to learn the weights of similarity kernels. In this experiment we look into the learnt weights to see whether they reflect the relative importance of the similarity kernels. For this purpose, we have conducted the ablation test, in which we **remove** one similarity kernel at a time and run the model. We also report the weights learnt without removing any kernels. Results on three domains are shown in Table 6. These results indicate that pseudo document similarity seems to play relatively higher importance than other kernels. For example, both in *movie.title* and *doctor.specialty*, it carries the highest weights; and the F1 measures drop to the lowest when removing this kernel (the lowest F1 is marked in bold). Interestingly, the categorical pattern similarity plays an important role in *shoe.brand*. Note that in this domain there are more ambiguous inputs (6 values) than other domains, suggesting the importance of categorical pattern similarity for disambiguation.

6. CONCLUSIONS AND FUTURE WORKS

For the problem of finding entity attribute synonyms, we propose a **compact clustering** framework to simultaneously identify synonyms for a set of attribute values. In this framework, multiple sources of information are integrated into a kernel function and synonyms are learned via unsupervised clustering. We have also proposed a novel similarity kernel called Categorical Pattern Similarity, which has proven to be effective for improving the performance of the compact clustering model. Extensive experiments demonstrate the effectiveness of our clustering framework over previous approaches for identifying entity attribute synonyms, both in the cases where they are entity mentions or are arbitrary strings. We have also demonstrated the effectiveness of our model for ambiguity handling for identifying domain specific synonyms.

Further, besides attribute value synonym identification, our unsupervised framework of simultaneously modeling multiple inputs and integrating multiple kernels can be potentially applied to other applications, such as looking for related queries, product recommendation, question paraphrasing *et. al.*

Table 6: Relative Importance of Similarity Kernels

Dataset	W/F1	categorical pattern	coclick	lexical context	pseudo document
movie.title	W	0.20	0.20	0.27	0.33
	F1	0.505	0.510	0.489	**0.464**
shoe.brand	W	0.29	0.16	0.25	0.3
	F1	**0.569**	0.601	0.589	0.573
doctor.specialty	W	0.13	0.22	0.30	0.35
	F1	0.573	0.567	0.55	**0.538**

7. REFERENCES

[1] I. Antonellis *et. al.* Simrank++: query rewriting through link analysis of the click graph. *VLDB*, 2008.

[2] A. Bar-Hillel *et. al.* Learning a mahalanobis metric from equivalence constraints. *J. Mach. Learn. Res.*, 2005.

[3] M. Baroni *et. al.* Using cooccurrence statistics and the web to discover synonyms in technical language. In *LREC*, 2004.

[4] O. Benjelloun *et. al.* Swoosh: A generic approach to entity resolution. Technical Report, Stanford InfoLab, 2005.

[5] M. Berland *et. al.* Finding parts in very large corpora. In *ACL*, 1999.

[6] G. Cao *et. al.* Selecting query term alternations for web search by exploiting query contexts. In *ACL*, 2008.

[7] K. Chakrabarti *et. al.* A framework for robust discovery of entity synonyms. In *KDD*, 2012.

[8] E. Y. Chan *et. al.* An optimization algorithm for clustering using weighted dissimilarity measures. *Pattern Recognition*, 2004.

[9] T. Cheng *et. al.* Fuzzy matching of web queries to structured data. In *ICDE*, 2010.

[10] X. Dong *et. al.* Reference reconciliation in complex information spaces. In *SIGMOD*, 2005.

[11] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In COLING, 1992.

[12] R. Jones *et. al.* Generating query substitutions. In WWW, 2006.

[13] L. Kaufman *et. al. Clustering by Means of Medoids*. Reports. Delft University of Technology, 1987.

[14] D. Lin. Automatic retrieval and clustering of similar words. In *ACL*, 1998.

[15] D. Lin *et. al.* Identifying synonyms among distributionally similar words. In *IJCAI*, 2003.

[16] S. Mcdonald *et. al.* Testing the distributional hypothesis: The influence of context on judgements of semantic similarity. In *CogSci*, 2001.

[17] S. Mirkin *et. al.* Integrating pattern-based and distributional similarity methods for lexical entailment acquisition. In *COLING-ACL*, 2006.

[18] G. Navarro. A guided tour to approximate string matching. *ACM Comput. Surv.*, 2001.

[19] P. Pantel *et. al.* Document clustering with committees. In *SIGIR*, 2002.

[20] L. Si *et. al.* Collaborative image retrieval via regularized metric learning, 2006.

[21] P. D. Turney. Mining the web for synonyms: Pmi-ir versus lsa on toefl. In *ECML*, 2001.

[22] C. Wang, *et. al.* Targeted disambiguation of ad-hoc, homogeneous sets of named entities. In *WWW* 2012.

[23] E. P. Xing *et. al.* Distance metric learning, with application to clustering with side-information. In *NIPS*, 2002.