Information Fusion For Entity Matching in Unstructured Data

Omar Ali and Nello Cristianini

Intelligent Systems Laboratory, Bristol University
Merchant Venturers Building, Woodland Road, Bristol, BS8 1UB, United Kingdom
{Omar.Ali,Nello.Cristianini}@bristol.ac.uk
http://patterns.enm.bris.ac.uk

Abstract. Every day the global media system produces an abundance of news stories, all containing many references to people. An important task is to automatically generate reliable lists of people by analysing news content. We describe a system that leverages large amounts of data for this purpose. Lack of structure in this data gives rise to a large number of ways to refer to any particular person. Entity matching attempts to connect references that refer to the same person, usually employing some measure of similarity between references. We use information from multiple sources in order to produce a set of similarity measures with differing strengths and weaknesses. We show how their combination can improve precision without decreasing recall.

Key words: Information fusion, entity matching, information extraction, social network, data cleaning, web mining

1 Introduction

The global media system produces a vast amount of news articles which discuss named entities, such as people, places and organisations. Extracting these references to named entities can enable us to study interactions between them and the news stories that they appear in.

In this paper we describe a complete system for the extraction and matching of named entities from unstructured news data, where large numbers and multiple sources of information are used to improve the precision of the extracted output. Our aim is to investigate how the fusion of information can improve the performance of the system.

To this end, the system has been collecting on-line news articles from multiple sources over a period of nearly two years. These come from RSS feeds, which are visited many times per day. Articles are passed through a named entity extraction tool, Gate [1], which provides us with references to named entities in each article. These references do not have one-to-one correspondences with named entities as there are many ways to refer to the same entity.

Currently the system contains 9.3 million English articles, which contain references to 3.8 million people. Amongst these references are many sources of noise, such as rare misspellings of names and errors in the entity extraction process.

Any study based on the people that appear in the media will be affected by the problem of multiple references, so these must first be resolved. Given the scale of the input this must be automated and we should be very careful not to merge similar-looking, yet different references, such as 'Tim Johnson' and 'Jim Johnson'.

We describe our approach to entity matching, fusing three sources of information to improve precision, without reducing recall and we test a method for the removal of ambiguous references based on their degree in a co-reference network.

2 Methods and Problem Formulation

In this section we formalise the problem of entity matching and discuss the similarity measures that will be used in our experiments.

Problem. We consider a set of references to named entities, $\{r_1, r_2, ...r_n\} \in R$. References may occur in many articles and pairs that appear in the same article are said to co-occur. The number of articles that a reference A appears in is its frequency, $f(A)$ and the number of times that any pair of references co-occur is their co-occurrence frequency, $f(A, B)$.

This definition allows us to construct a graph of cooccurrences, $G(V, E)$, which consists of references from set R as vertices and co-occurrences between references as edges. In this graph we wish to find pairs of references that represent the same person.

String Similarity. Most references to the same person share something in common. These range from multiple words, for example 'Hillary Clinton' and 'Hillary Rodham Clinton', to small differences in spelling like 'Hilary' or 'Hillary'. We use three measures of string similarity from the SimMetrics library [13]: **Jaro-Winkler**, **Q-Gram** and **Levenshtein**.

Titles, such as 'Mr', 'President', or 'Governor' should not be matched, therefore we maintain a list of titles and remove them prior to comparison. This leads us to rate 'President Barack Obama' and 'Barack Obama' very highly, whilst avoiding matches such as 'President Obama' and 'President Bush'.

Note that string similarity alone cannot resolve cases where similar names belong to different entities, as is the case for 'George W. Bush' and 'George H. W. Bush'. In order to resolve this ambiguity we require additional information.

Neighbourhood Similarity. Matching references are likely to be either directly linked, or to share a very similar set of neighbours. For this reason we employ measures of neighbourhood similarity in order to locate potential matches.

Finding references with similar neighbours could require us to compare all pairs, which is infeasible on large inputs. Fortunately we do not need to measure similarity between references that are more than two steps away from each other in the input graph, as we prove that their similarity must be zero.

Definition 1. *A graph $G(V,E)$ is composed of a set of vertices (references), V and a set of edges (co-occurring references), E. Each edge connects exactly two vertices. The* ***distance****, $d(u,v)$ between vertices u and v is the length of the shortest path between them. The* ***neighbours*** *of a vertex, $N(v)$ are the set of all vertices that are* ***adjacent*** *to v. If u and v are adjacent, $d(u,v)=1$.*

A ***neighbourhood similarity measure****, $S_{nbr}(u,v)$ describes the amount of overlap of the neighbourhoods of u and v. In general, if $N(u)\cap N(v)=\emptyset$ then $S_{nbr}(u,v)=0$ and if $N(u)\cap N(v)=N(u)\cup N(v)\neq\emptyset$ then $S_{nbr}(u,v)=1$.*

Proposition 1. *Any pair of vertices that have some neighbours in common, $N(u)\cap N(v)\neq\emptyset$ must be within two steps of one another, i.e.:*
$S_{nbr}(u,v)>0, \forall\, u,v\in V,$ *s.t.* $d(u,v)\leq 2$

Proof. Assume we have a pair of vertices, (u,v), where $S_{nbr}(u,v)>0$ and $d(u,v)>2$. If $S_{nbr}(u,v)>0$, then $N(u)\cap N(v)\neq\emptyset$, $\exists\, w\in V$ s.t. $w\in N(u), w\in N(v)$. Therefore $d(w,u)=1, d(w,v)=1$ and $d(u,v)\leq d(w,u)+d(w,v)=2$, which contradicts our assumption. □

We use the **Jaccard** and **Adar** neighbourhood similarity measures [3], which are defined in (1) and (2). Adar is similar to Jaccard with the addition of a uniqueness score for each reference.

$$S_{jaccard}(r_i,r_j)=\frac{|N(r_i)\cap N(r_j)|}{|N(r_i)\cup N(r_j)|} \tag{1}$$

$$S_{adar}(r_i,r_j)=\frac{\sum_{c\in(N(r_i)\cap N(r_j))}u(c)}{\sum_{c\in(N(r_i)\cup N(r_j))}u(c)} \text{ where } u(c)=\frac{1}{log(|N(c)|)} \tag{2}$$

Co-reference Similarity. Named entities are extracted using a named entity recognition tool that is part of the Gate distribution [1]. Gate also finds co-references [2], which are references to the same entity that occur within the same article. This information is used as another clue in our entity matching process. We treat this as a binary measure as shown in (3).

$$S_{coref}(r_i,r_j)=\begin{cases}1, & \text{if } f(r_i,r_j)>0\\ 0, & \text{otherwise}\end{cases} \tag{3}$$

Note that since this can only link references that appear in the same article it will not resolve systematic spelling differences from different sources, such as ‘Gaddafi’ and ‘Kaddafi’. For these we can only rely on string similarity.

The degree a node in this network reveals information about its ambiguity and is used to filter out ambiguous references, as we show in the next section.

Combination of Measures. All measures are normalised to be in the range $[0,1]$ and we test weighted combinations of them as shown, for example, in (4).

$$S(r_1,r_2)=\alpha S_1(r_1,r_2)+\beta S_2(r_1,r_2)+\gamma S_3(r_1,r_2)$$
$$\text{where } \alpha,\beta,\gamma\in[0,1],\ \text{s.t. } \alpha+\beta+\gamma=1 \tag{4}$$

3 Experimental Set-Up

In this section we describe how our experiments were set up, then present and discuss our findings. We also discuss the difficulties of measuring performance in an unstructured setting. This is a setting where references are not retrieved from a database where a certain degree of standardisation has been enforced, but are extracted from free text using statistical methods.

3.1 Measuring Performance

Matching entities that are automatically extracted from unstructured data is difficult due to the variability of references and errors in extraction and spelling. Freely available data sets for this task are difficult to find.

In order to be able to measure performance on this type of data we produced our own data set, consisting of 1417 pairs of marked references. This breaks down into two sets containing 901 correct and 516 incorrect pairs[1].

In web extraction applications it is more important to guarantee the quality of extracted information than it is to extract it all. As there is plenty of data available, we focus on high precision rather than high recall, in a similar vein to [15]. In addition we focus on relative changes in performance resulting from information fusion, rather than on absolute performance.

Precision and Recall. The input is a set of entity references, R from our entity database. All pairs of references could potentially be linked, $P = R \times R$.

Our algorithms produce a set of linked pairs, $P_L \subseteq P$. The ground-truth data consists of two sets of pairs, correct and incorrect, or $P_T \subseteq P$ and $P_F \subseteq P$, respectively. Precision and recall are calculated according to (5).

$$\text{Precision } = \frac{|P_T \cap P_L|}{|P_T \cap P_L| + |P_F \cap P_L|}, \text{ Recall } = \frac{|P_T \cap P_L|}{|P_T - P_L|} \tag{5}$$

3.2 Basic Set Up

For experimental purposes we focus on one week of references from the 20th to the 27th December 2009. During this time we observed 114,388 references and 2,008,225 co-occurring pairs, in N = 96,880 articles. The size of this graph is reduced by discarding edges that co-occur less than 5 times. This reduces the size of the vertex and edge sets to, $|V| = 8,264$ and $|E| = 57,361$.

χ-Square Test of Independence. To ensure that the remaining edges are statistically significant we use Pearson's χ-square test of independence [14]. In this setting we examine the contents of an article and make a pair of observations on the presence of references A and B. Our null hypothesis is that references A and B are statistically independent so their expected frequency of co-occurrence should equal $\frac{f(A)f(B)}{N}$, where N is the number of articles and $f(A)$ is the number of articles reference A is seen in.

[1] Data available at: `http://patterns.enm.bris.ac.uk/entity-matching-data`.

Our test makes multiple comparisons between reference pairs and the test statistic is calculated for all pairs of references which co-occur. In reality we implicitly compare $g = \frac{n(n+1)}{2} - n$ pairs, for n total references. Therefore we apply the Bonferroni correction and adjust the significance level to $1/g$.

For this network we have $g \approx 3.42 \times 10^7$ and an adjusted significance of 2.93×10^{-10}. Applying the χ-square test of independence at this significance level leaves us with $|V| = 8,230$ and $|E| = 53,462$.

3.3 Results

We now present results for string, neighbourhood and co-reference similarities, along with weighted combinations of the three.

Note that all measures are implicitly aided by neighbourhood information, as comparisons are made between references that are up to two steps apart in the co-occurrence network. This step ensures that we do not calculate similarities for all pairs of references. When we discuss neighbourhood similarity we are referring to *further* specific measures on the immediate neighbours of each reference.

We quote values of precision and recall which demonstrate the peak precision of each method. This excludes any results where precision becomes unstable when the number of output pairs becomes too small.

String Similarity. Table 1 shows results for string similarity measures, along with weighted combinations of them. All measures exhibit reasonably high precision but have subtly different types of reference-pairs that they can and cannot match. JaroWinkler is biased toward matching the beginning of a string, which leads to incorrect matches such as, 'Robert Casey' and 'Robert Costa'.

Q-gram is able to handle more variation in the input as it operates on sets of 3-grams, so order and positioning of characters is less important. This works in the case of 'Thomas James Leggs Jr.' and 'Thomas J. Leggs Jr.', but fails on pairs such as 'James Anderson' and 'Pamela Anderson'.

Levenshtein is a lot more selective, as each insertion, modification or deletion incurs a cost. Very similar strings like 'Tim Johnson' and 'Jim Johnson' are still a problem, however this is true for all of the measures we have described.

Neighbourhood Similarity. The addition of neighbourhood to string similarity did not lead to any improvement. We ran a number of weighted combinations of these measures and in all cases precision fell to around 50%.

This loss of precision was due to the inclusion of many false positives, which were typically strongly-connected pairs such as 'Warren Taylor' and his ex-wife 'Karen Taylor'. Neighbourhood measures have no way of discriminating references that are similar due to being related in some way from those which are the same. Similar to findings in [3], references which are similar but not matching are scored highly and incorrectly matched.

Co-Reference Similarity. Table 1 also shows precision and recall for co-reference similarity, followed by the addition of string and then neighbourhood similarity measures. Co-reference similarity alone shows strong performance, however we can improve this by adding in string measures.

Table 1. Precision and recall for measures, plus weighted combinations of them. Combining string measures seems to improve precision slightly, although Levenshtein performs best alone. Neighbourhood measures reduce performance (we show Levenshtein combined with both Jaccard and Adar — all other combinations perform comparably). Co-reference shows high precision and the addition of string similarity improves this, without a reduction in recall. The final line shows the effect of removing high-degree (ambiguous) references.

Method	Precision	Recall
Levenshtein	86.36	2.11
Q-Gram	80.77	2.33
JaroWinkler	79.17	2.11
$\frac{1}{4}$JaroWinkler + $\frac{3}{4}$Levenshtein	83.33	2.22
$\frac{1}{2}$JaroWinkler + $\frac{1}{2}$Levenshtein	84.21	1.78
$\frac{3}{4}$JaroWinkler + $\frac{1}{4}$Levenshtein	82.61	2.11
$\frac{1}{4}$JaroWinkler + $\frac{3}{4}$Q-Gram	71.43	2.77
$\frac{1}{2}$JaroWinkler + $\frac{1}{2}$Q-Gram	84.21	1.78
$\frac{3}{4}$JaroWinkler + $\frac{1}{4}$Q-Gram	82.35	1.55
$\frac{1}{4}$Q-Gram + $\frac{3}{4}$Levenshtein	83.33	1.66
$\frac{1}{2}$Q-Gram + $\frac{1}{2}$Levenshtein	86.36	2.11
$\frac{3}{4}$Q-Gram + $\frac{1}{4}$Levenshtein	85.00	1.89
$\frac{1}{2}$Levenshtein + $\frac{1}{2}$Jaccard	53.19	2.77
$\frac{1}{2}$Levenshtein + $\frac{1}{2}$Adar	48.78	2.22
CoRef	91.76	8.66
$\frac{1}{2}$Levenshtein + $\frac{1}{2}$CoRef	95.35	9.10
$\frac{1}{2}$Q-Gram + $\frac{1}{2}$CoRef	95.92	5.22
$\frac{1}{2}$JaroWinkler + $\frac{1}{2}$CoRef	94.87	8.21
$\frac{1}{3}$Levenshtein + $\frac{1}{3}$Jaccard + $\frac{1}{3}$CoRef	91.67	1.22
CoRef (ambiguous nodes removed)	91.49	19.09

Combining co-reference and string similarity gives some increase in performance. A co-reference error, which links ‘Vicki Kennedy’ to ‘Ted Kennedy’ is scored highly in terms of co-reference but relatively poorly by the Levenshtein measure, resulting in its removal. There is also the addition of pairs that cannot be found by co-reference alone, such as ‘Mahmud Abbas’ and ‘Mahmoud Abbas’.

This combination results in increased precision without decreasing recall over the baseline string and co-reference measures. The co-reference network map also allow us to identify and remove references that are ambiguous (high-degree nodes).

4 Related Work

Existing methods [6, 7] use on-line data, such as Wikipedia, to place references to entities into some context. An entity can then be disambiguated according to words or other entities that it appears alongside. Similarly, Bekkerman and McCallum [5] obtain context from a list of contacts in a user’s mailbox. They suggest that web pages related to a person will exist in a community of web pages related to that person's contacts.

Most methods [12] operate on structured data so do not have to handle noise associated with automatically extracting named entities from unstructured text.

Whilst state-of-the-art entity matching tools produce high accuracies, they do not scale well to large data sets. Rastogi [10] addresses this by defining a general framework for large scale entity matching. Dalvi [11] also discusses the more general problem of connecting related concepts.

Our system processes a high volume of unstructured data from the global media system, automatically extracting and storing named entities on a large scale. This introduces a lot of noise into our entity matching task and must be resolved with high precision in order that we may launch further studies with high confidence in our data.

5 Conclusions and Future Work

In this paper we have explored the use of information fusion for the task of information extraction from unstructured text. We have considered the problem of entity matching and have shown that combining multiple sources of information results in improved performance. Note that we have concentrated on relative changes in performance resulting from information fusion, rather than on absolute performance.

We looked at three similarity measures, all of which have different strengths and weaknesses. String similarity is a very effective method of comparison but is also very vulnerable to connecting similar-looking names. Co-reference information also gives high precision results and we have demonstrated that combining this with string similarity leads to increased precision, without reducing recall.

Neighbourhood similarity proved to be poor at this task as it assigns high scores to related references when we are looking for pairs which are the same. Whilst it does not prove useful as a similarity measure, the use of neighbourhood *information* is essential in order to reduce computation and produce a system which can scale.

In future work we will look to improve the recall of the system. The co-occurrence network is improved by considering greater volumes of news, for the simple reason that infrequent references can be disregarded, which greatly reduces noise in the input. We will also investigate further the performance improvements due to removing ambiguous (high-degree) nodes from the co-reference network. Examples include common forenames, the removal of which produces a much cleaner network. These noise-reduction steps will allow us to select lower thresholds to increase recall, without reducing precision.

On a more general level, our task is to find nodes in a network which are strongly related to one another. In this case the relation is one of identity, i.e., do these references refer to the same person? Other strong relations like family or work are regularly found and we wish to ignore these.

This also raises the issue of domain knowledge. One desirable property of an entity matching system is that it could be applied to other domains, as we also collect references to locations and organisations. It is very clear from our results that domain knowledge is very important and that a general system for matching entities of different types is a long way off. Increasing the amount of domain knowledge that we include should prove very effective at reducing the number of false positives that are produced.

Acknowledgements. We would like to give thanks to H. Cunningham, D. Maynard, K. Bontcheva and V. Tablan for creating and releasing Gate [1]. O. Ali is supported by a DTA grant, N. Cristianini is supported by a Royal Society Wolfson Merit Award and both are supported by the 'Pascal2' Network of Excellence.

References

1. Cunningham, H., Maynard, D., Bontcheva, K., Talban, V.: GATE: A framework and graphical development environment for robust NLP tools and applications. ACL (2002)
2. Dimitrov, M.: A light-weight approach to coreference resolution for named entities in text. MSc Thesis, University of Sofia (2002)
3. Bhattacharya, I., Getoor, L.: Collective entity resolution in relational data. TKDD (2007)
4. Newcombe, H., Kennedy, J., Axford, S., James, A.: Automatic linkage of vital records. Science (1959)
5. Bekkerman, R., McCallum, A.: Disambiguating web appearances of people in a social network. WWW (2005)
6. Han, X., Zhao, J.: Named entity disambiguation by leveraging Wikipedia semantic knowledge. CIKM (2009)
7. Cucerzan, S.: Large-scale named entity disambiguation based on Wikipedia data. EMNLP-CoNLL (2007)
8. Minkov, E., Cohen, W., Ng, A.: Contextual search and name disambiguation in email using graphs. SIGIR (2006)
9. Jijkoun, V., Khalid, M., Marx, M., De Rijke, M.: Named entity normalization in user generated content. AND (2008)
10. Rastogi, V., Dalvi, N., Garofalakis, M.: Large-scale collective entity matching. VLDB (2009)
11. Dalvi, N., Kumar, R., Pang, B., Ramakrishnan, R., Tomkins, A., Bohannon, P., Keerthi, S., Merugu, S.: A web of concepts. PODS (2009)
12. Köpcke, H., Rahm, E.: Frameworks for entity matching: A comparison. Data Knowledge Engineering (2009)
13. Chapman, S.: SimMetrics Library `http://www.dcs.shef.ac.uk/~sam/simmetrics.html` . NLP Group, University of Sheffield (2006)
14. Duda, R., Hart, P., Stork, D.: Pattern classification. Wiley Interscience (2nd Edition)
15. Talukdar, P., Brants, T., Liberman, M., Pereira, F.: A context pattern induction method for named entity extraction. CoNLL (2006)