Automatic Affiliation Extraction from Calls-For-Papers

Xinyu Li, Roya Rastan, John Shepherd, Hye-Young Paik
School of Comp Sci and Eng, UNSW, Sydney, Australia
{rrastan,jas,hpaik}@cse.unsw.edu.au

ABSTRACT

In this paper, we describe a system to collect information about academic affiliation (organisations where researchers work) from Calls-for-Papers for academic conferences. The system uses a range of heuristic approaches and open-source tools in order to extract and identify entities, and to incorporate the information into a pre-defined database schema. This forms part of a larger project to automatically populate and maintain a range of data related to academic research. The proposed system is currently being tested and some promising preliminary results are available.

1. INTRODUCTION

Research data management systems (RDMSs) store information related to academic research: publications, people, projects, events, etc. One useful kind of information in such systems is affiliation information for researchers. Maintaining affiliation information is currently a potentially time-consuming manual process, relying either on researchers themselves to maintain their own data or on field experts to keep track of researchers' changing affiliations.

One useful, easily-available and up-to-date source of data on affiliations is program committee (PC) lists in calls-for-papers (CFPs) for academic conferences. CFPs are published frequently, typically in a partially-structured natural language format. CFPs contain more than just program committee information (e.g. title, location, dates, website, topics of interest, etc.), but our primary concern in this work is extracting information about researcher affiliations from the program committee section.

The value of RDMSs depends critically on the volume and accuracy of the information they contain. Given the large volumes of data involved, such resources cannot feasibly be maintained entirely manually. They require systems that can collect and integrate information from a variety of (typically unstructured) online sources. Information extraction is a well-known approach for this task, but has been applied so far largely in the context of documents consisting of paragraphs of natural language text. CFP documents have some structure which can be exploited to more accurately discover and extract information. Part of a typical CFP email is shown in Figure 1.

```
Paper Submission Guidelines:
Submission instructions will be announced at ...
papers will be published by Springer. A selected ...

Important Dates:
  Paper Submission Deadline: July 16
  Notification of Acceptance: September 12
...
Program Committee Members:
  Andrea Arcuri, Simula Research Laboratory, Norway
  David Benavides, University of Seville, Spain
  Yu Cao, California State University, Fresno, USA
  Bogdan Filipic, Jozef Stefan Institute, Slovenia
  Du Zhang, California State Univ., Sacramento, USA
...
Track Co-Chairs
   Shih-Hsi Liu, California State Univ., Fresno, USA
```

Figure 1: An example of informal structure in CFPs

This paper presents a multi-phase method which attempts to automatically extract information about researchers and their affiliation from CFP documents and place it in a relational database. The proposed method integrates evidence from both content (named entities) and layout (punctuation and column-based structures). It first identifies regions of a document likely to contain researcher and organisation information. Since the named entities in such regions are unlikely to be classified with 100% accuracy, the method then detects patterns, which are subsequently used to correct missing and inaccurate entity classifications. Next, the approach uses multiple methods to attempt to identify entities and link them in the database. Feedback loops at various points aim to increase the likelihood that information is accurately extracted. The system also has a post-processing phase, which runs in the background, and uses relationships between entities to attempt to detect and correct remaining entity identification anomalies.

2. RELATED WORK

Several recent information extraction projects have considered CFP data. Jeong and Kim [9] developed an ontology (SEDE) to accurately represent all of the information related to academic events (not just PC member affiliation). They concluded that fully-automatic extraction methods were not

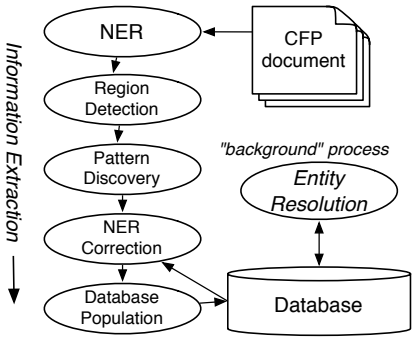

Figure 2: System Architecture

effective in transferring data from CFPs into their ontology and used a semi-automated extraction process. Correia et al. [3] developed a system (AllCall) which aims to extract information about conferences (especially title, dates, location and topics). Their goal was to use the conference information to drive a web-site that researchers could use to discover information about conferences of interest to them.

A critical aspect of any IE system is entity resolution, and this problem has been studied extensively in the literature. For example, Yosef et al. [11] describe AIDA, a system that aims to resolve mentions of entities in tables and text using a range of graph-based methods incorporating features such as *prominence*. Another approach, in [7], uses clustering in an RDF database to identify/resolve entities. Our approach is different to the above in that we split the entity resolution process across two phases: we first use the text data and information from the database to try to accurately identify entities during information extraction, and then use relationships between entities over the populated database in order to resolve any remaining inaccuracies in the data.

3. SYSTEM ARCHITECTURE

As shown in Figure 2, the system contains two phases: Information Extraction (IE) and Entity Resolution (ER).

3.1 Target Schema

The target schema for the database includes tables for the main entity classes (People, Organisations, and Locations) as well as tables for relationships among these. The schema used in this work is part of a larger schema which also includes tables for Events, Projects, Topics, etc. Figure 3 gives further details of the tables. The `People` table contains de-

```
People(id, name, email, ...)
Groups(id, name, type, parent, ...)
Locations(id, name, type, parent, ...)
Affiliations(person, organisation, role, ...)
OrgLocations(organisation, location, ...)
NameWords(word, type, id)
Aliases(name, type, id)
```

Figure 3: Part of Target Schema

tails about one individual; the figure shows a minimal subset of these attributes. `Groups` is used to represent organisations along with other ad hoc groups such as project teams, SIGs, etc. It identifies the type of group and also allows for hierarchies (the `parent` attribute), such as a department within a faculty within a university. The `Locations` table holds information about places at a wide range of granularities (e.g. country, state, city, suburb) and provides for containment (the `parent` attribute).

The `Affiliations` table provides a link between an individual and an organisation they are associated with. `OrgLocations` allows for an organisation to have a presence in several different physical locations. The `NameWords` table is used to hold individual words from entity names, to assist with recognising variations on names. Each tuple in this table links a word to a specific entity in a specific table; a word appears in multiple tuples if it is used in multiple entities. `Aliases` stores alternative names for entities, if those names could not be easily derived from the full name (e.g. "Alex Liu" for "Shih-Hsi Liu"); the `type` attribute specifies which entity table and the `id` attribute determines a tuple in that table.

3.2 Information Extraction

As noted in Figure 2, the Information Extraction process is composed of five distinct sub-parts.

Named Entity Recognition (NER). The NER module splits the document into tokens, and then determines whether each token or token-group (phrase) belongs to any of a set of predefined categories. In our work, the interesting categories are People (P), Organization (O) and Location (L). Besides these entity categories, some layout features (e.g. newline, bracket, comma) are also identified, for later use in determining patterns.

There are several available tools for NER such as the Stanford Named Entity Recognizer [4], LingPipe [2], Apache OpenNLP [8] and Gate [1]. Since these tools have complementary strength, one possiblity is to use all of these systems and combine the results in some way. The approach we adopted, however, was to select one system, determined by performance against the following criteria: NER recall (proportion of entity words classified as entities), NER precision (proportion of classified words that are correctly classified), parsing speed, and well-maintained (active user and developer communities).

In a comparison using a sample of CFP documents, the Stanford Named Entity Recognizer performed best overall. Most importantly, Stanford NER has very high accuracy in recognizing names of people, which is very useful for our system.

Region Detection. The Region Detection module aims to identify where the useful data groups (e.g. (name, organization, location) triples) are located within a document. This allows subsequent components to focus on just this data and ignore irrelevant text. The region detection algorithm uses a sliding window and detects short sequences of tokens that contain a significant proportion of entities; it then merges adjacent short regions into larger high-density regions (HDRs).

After the process, regions containing many named entities (P,O,L) will be identified. However some of these regions may be "noise", incorrectly detected because because a group of tokens has been wrongly labeled as P/O/L by the NER system. The next module detects frequently occurring pat-

terns (or lack thereof) and uses this to identify noisy regions and then ignore them.

Pattern Discovery. The main job for the pattern discovery module is to find the most common pattern in the detected HDRs and use this pattern in extraction module. It can also help to identify the noisy HDRs.

In CFP documents, authors frequently write Program Committee lists using a common pattern such as "P, O" or "P (O,L)". However, there are occasional exceptions within a given PC region (e.g. a person written with 'O,L" rather than just "O"). These may be genuine exceptions or may be caused by misclassification by the NER. Despite such problems, the NER can generally recognize enough tokens correctly that valid patterns can be detected. Once a pattern is determined, it can be used to help correct misclassified tokens in the column.

We only consider patterns beginning with a "P" token. Invariably, the format of PC lists in CFPs start with a person.

An n-gram approach is used to detect the most frequent pattern in three steps: use different-sized sliding windows (4-gram to 10-gram) to generate multiple possible patterns (choose the most frequent pattern for each window size). Patterns are scored by taking layout features (e.g. position of newlines) into account, and the highest-scoring pattern is chosen. Scoring is based on the following: if a pattern contains all of O, L and a newline character ('\n'), it gets the highest score. Finally, the pattern is applied to the region to see what proportion of potential records it detects. If the most frequent pattern matches only a small amount of the region, we consider the region to be noisy and ignore it. The threshold for acceptance as a non-noisy region is $\theta = 17\%$; this value is based on tests on a sample of 100 documents.

NER Correction. This module aims to improve the accuracy of detecting and classifying named entities. It uses the pattern discovered in the previous step to detect anomalies and attempts to correct them via reference to entity tables in the database.

The first step in NER correction is to determine record boundaries. The pattern for the region is clearly important here, but we also make use of delimiters (such as line boundaries), since these are frequently used in CFPs to delimit records. Detecting the start of records can be done relatively accurately since (a) we assume that records start with a person's name and (b) the Stanford NER can detect such names with high precision. In the vast majority of CFPs, each record occurs by itself on a single line, and so detecting the end of records can also be done accurately.

Each record contains a set of tokens which were classified in the NER stage; we now aim to group the tokens into fields, which will form the basis of the extraction step. In the best case, this token sequence exactly matches the pattern and we can precisely identify the fields and their types. In a typical case, only some of the tokens have been classified, and some may have been incorrectly classified, and overall the pattern is not precisely matched using all tokens in the record. To handle such cases, we exploit the fact that we are dealing with a relatively small number of tokens and form token subsequences which we attempt to match against the database (using the `NameWords` table) to try to group tokens until they fit the pattern. Since this largely ignores the original NER classification, we can overcome both missed and incorrect classifications. We also also exploit any delimiters to assist in identifying likely field boundaries.

Database Population. This module extracts data from the records and stores it into the database. This requires us to relate the field values identified in the previous step to entities in the database (i.e. we need to establish a database identifier (either new or existing) for each field). The previous steps give us reasonable confidence that a particular string is e.g. a person's name, but attempting to discover which entity in the database this corresponds to throws up some new challenges. We describe a number of these below, and discuss how we attempt to deal with them. Note also that even if we do not achieve 100% correct identification at this stage, once more data becomes available in the database, we may be able to correct some of these errors; this is the task of the Entity Resolution process which we describe later.

The entity tables in the database are not initially empty, but are "seeded" with the following data: the `Locations` table was populated from the GeoNames database [10] and contains a large number of place names, including alternative spellings (which are placed in the `Aliases` table); the `Groups` table was populated from a list of university names and locations from the Universities Worldwide database [5]; the `People` table was populated using a list of names from the DBLP database [6].

First, we to try to match the named entity against a record stored in the database. We use the following strategies:

- For locations, we use string matching on the names in the `Locations` table; if no match is found at this stage, we attempt to match the string in the `Aliases` table; we also use string similarity measures if no exact match is found (to cater for e.g. spelling mistakes).
- For people, we attempt the following matching strategies: string match on `People` table; string match on `Aliases` table; string match based on permutations of the words in the name (e.g. "Wei Wang" vs "Wang Wei"); matching based on individual words; matching based on forming initials from given names.
- For organisations, we attempt the following matching strategies: string match on `Groups` table; string match on `Aliases` table; string matching after removing stopwords; matching based on individual words; using the `abbreviation.com` web service.

The other important operation in the Database Population phase is to set up relationships between entities. This is only possible if we have two uniquely identified entities (in each case, either a unique match with a known entity, or a new entity added because there was no match). A (person,organization) pair adds a new tuple to the `Affiliations` table. An (organization,location) pair adds a new tuple to the `OrgLocations` table.

If two known organizations are uniquely identified in one record, we assume that the first is a sub-unit of the second and set up a parent-child relationship between entries in the `Organizations` table. If a person's affiliation is involved, we associate the person to the sub-unit. If two apparently separate unknown organizations are identified in one record, we combine them into a single organization name and store that as a new entity in the database.

3.3 Entity Resolution

The Entity Resolution phase aims to resolve the problems of precise entity identification and relationship finding left from the information extraction phase. This phase is executed periodically, after a significant amount of data has been stored in the database, and attempts to repair some common types of error. The erroneous data in the database can be categorized as follows:

1. Two `People` entries for one person
2. Two `Groups` entries for one organization
3. Missing relationships between organizations
4. Missing affiliations
5. Incorrect data

For the first two cases, we use two well-known string-similarity methods: Levenshtein distance and Jaccard similarity.

Resolving Person Entities. The system checks pair-wise similarity between every name in the `People` table, and uses the Levenshtein string distance (any distance less than 2) to find candidate pairs that might refer to the same individual. Once such pairs are found, any other information about the entities (e.g. affiliations) is checked. If the weight of evidence suggest that these two entities are the same person, they are merged by putting one name in the `Aliases` table, linking it to the other name as an alternative name, and transferring all relations of the former name to the latter one.

Resolving Organization Entities. The system searches the database for pairs of organizations satisfying the following: they have an affiliation with two people who have the same name, they are located in the same city or country. Such pairs are treated as candidates for being synonyms.If two entities are deemed to be the same organization, one is added to the `Aliases` table and all of its relationships are transferred to the other.

Resolving Organization Relationships. We noted in the the section on database population that "organisation" fields in affiliation records may sometimes contain two organisation names, where one refers to a sub-unit of the other (e.g. "Department of Computing, University of Glasgow"). Other records contain a combination of two separate organizations (e.g. "UNSW and NICTA"). These are handled as follows:

- If one part of a "double organisation" phrase exists as an independent entity in the database (e.g. "University of Glasgow"), and if the other component contains one of a set of sub-unit keywords (such as "school", "department", "group"), then we treat the sub-unit as an entity in its own right and link it to the parent entity.
- If a "double organisation" phrase contains two organizations that already exist as entities in their own right (e.g. "UNSW" and "NICTA"), the system removes the record for the incorrectly combined organizations and adds affiliations for both of the separate organisations.

	t_p	f_n	f_p	Prec	Recall	F-measure
RecordsNum.	629	125	32	.095	0.83	0.89

Table 1: Evaluation of Information Extraction

4. RESULTS

The described system is currently being implemented and tested. More in-depth evaluations on every phase of the system are being carried out. Here, we report on our preliminary results on the information extraction phase (as a whole). We have tested the system on 100 emails (consisting 754 distinct affiliation records) received from various mailing lists. These announcements contain duplicates and near-duplicates, which helps us evaluate if our system can recognize repeated records. We use the well-known IR measures:

$$Precision = \frac{t_p}{t_p + f_P}, \qquad Recall = \frac{t_p}{t_p + f_n} \tag{1}$$

$$F - measure = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{2}$$

where, t_p is the number of correct records extracted, f_n is the number of missing records and f_p is the number of unexpected extracted records.

Table 1 shows the summary of the results. We made the following observations on the results:

- It is possible that in a region, multiple patterns exist. Since the system chooses only one pattern for a region, all of the potential records following different patterns will be interpreted incorrectly or ignored. We believe allocating weights to each pattern in a region can be a solution for further improvement.
- The NER system does not identify sufficient entities: the techniques described in Section 3.2 may not have sufficient data to infer high density regions. Considering more layout attribute may be helpful.

5. CONCLUSION AND FUTURE WORK

We have develop a system that can process plain-text Calls-for-Papers documents to: (a) discover regions containing information about researcher affiliations (relating a person to the organisation they work for); (b) identify patterns of entity types in those regions (i.e. record structures); (c) extract values from these records and use these values to populate a relational schema. In order to achieve these results, the system uses a combination of relatively simple text processing techniques and the Stanford NER system. Our preliminary experiments indicate that even this simple approach can be very effective. As discussed the proposed system is part of a bigger research data management system. We only evaluated our system on sample plain text documents to see the potential improvement of the system, and it would require future work to adopt our system to be applied to structured text such as HTML.

6. REFERENCES

[1] Kaling Bontcheva, Hamish Cunningham, Diana Maynard, Valentin Tablan, and Horacio Saggion. Developing reusable and robust language processing components for information systems using gate. In *Database and Expert Systems Applications, 2002. Proceedings. 13th International Workshop on*, pages 223–227. IEEE, 2002.

[2] Bob Carpenter and Breck Baldwin. Natural language processing with LingPipe 4, draft edition, 2011.

[3] Fábio L Correia, Rui FS Amaro, Luís Sarmento, and Rosaldo JF Rossetti. Allcall: An automated call for paper information extractor. In *5th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–4. IEEE, 2010.

[4] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

[5] Klaus Förster. Universities worldwide. http://univ.cc/world.php.

[6] Michael Ley. DBLP: computer science bibliography. http://www.informatik.uni-trier.de/ ley/db/.

[7] Steven N Minton, Craig A Knoblock, Raymond Liuzzi, Sofus A Macskassy, Peter LaMonica, and Kane See. Monitoring entities in an uncertain world: Entity resolution and referential integrity. In *Twenty-Third IAAI Conference*, 2011.

[8] Thomas Morton, Joern Kottmann, Jason Baldridge, and Gann Bierner. OpenNLP: a Java-based NLP toolkit, 2005.

[9] Senator Jeong and Hong-Gee Kim. SEDE: An ontology for scholarly event description. *Journal of Information Science*, 36(2):209–227, 2010.

[10] Mark Wick and Christophe Boutreux. Geonames. http://www.geonames.org/.

[11] Mohamed Amir Yosef, Johannes Hoffart, Ilaria Bordino, Marc Spaniol, and Gerhard Weikum. Aida: An online tool for accurate disambiguation of named entities in text and tables. *Proceedings of the VLDB Endowment*, 4(12), 2011.