Entity Oriented Search and Exploration for Cultural Heritage Collections

The EU Cultura Project

David Carmel, Naama Zwerdling, Sivan Yogev
IBM Research – Haifa Lab, Haifa 31905, Israel
{carmel,naamaz,sivany}@il.ibm.com

ABSTRACT

In this paper we describe an entity oriented search and exploration system that we are developing for the EU *Cultura* project.

Categories and Subject Descriptors: H.3.3 [Information storage and retrieval]: Information search and retrieval
General Terms: Algorithms
Keywords: Entity Oriented Search, Exploration, Cultural Heritage

1. INTRODUCTION

Recent large-scale digitization initiatives have made many important cultural heritage collections available on-line. This makes them accessible to the global research community and interested public for the first time. However, the full value of these heritage treasures is not being realized. After digitization, these collections are typically monolithic, difficult to explore and can contain text which is of variable quality in terms of language, spelling, punctuation and consistency of terminology and naming. As a result, they often fail to attract and sustain broad user engagement and so have only limited communities of interest.

The objective of the *Cultivating Understanding and Research Through Adaptivity* (*Cultura*) project, funded by the 7th Framework Programme of the European Commission, is motivated by the desire to provide a fundamental change in the way digital cultural heritage is experienced and analyzed by communities of interested individuals. These communities include a broad spectrum of experienced and novice users with a wide variety of knowledge, domain expertise, and interests. *Cultura* will enable researchers to conduct complex and labor intensive tasks with much greater ease, and will support deeper and more thorough interrogations of historical content collections than is traditionally achievable.

To validate the environment which will be developed by *Cultura*, two major artifacts have been selected - the 1641 Depositions held in Trinity College Dublin (See examples in Figure 1), and the IPSA Digital Herbal Archive held in the University of Padua. The 1641 Depositions are seventeenth-century manuscripts that comprise 4,000 (or 20,000 pages) depositions of witness statements, examinations and associated materials, in which Protestant men and women of all classes and from all over Ireland told of their experiences following the outbreak of the rebellion by the Catholic Irish in October 1641. The Imaginum Patavinae Scientiae Archivum (IPSA) collection is a digital archive of illuminated herbals manuscripts dating from the 14th century with Latin language commentaries. The two collections that have been selected for use display characteristics that are indicative of digital humanities collections. These characteristics include: rich morphology (noisy text, inconsistent spelling and grammar, archaic language); complexity (transcribed text, digital images, missing meta-data); and multiple entity types.

Figure 1: The 1641 Depositions: Handwritten Manuscripts and Associated Imagery

The *Cultura* research project tends to pioneer the development and fusion of (1) Natural Language Processing techniques, which normalize ambiguities in noisy historical texts; (2) Entity and relationship extraction, to identify the key individuals, events, dates and other entities and relationships within unstructured text; (3) Entity oriented search and exploration over the collection, entities, and relationships; (4) Multi-model adaptivity to support the dynamic reconciliation of multiple dimensions of personalization; (5) Social network analysis of users engaging with the content, to facilitate engagement and interaction between researchers and users and also to promote active contribution to the knowledge relating to the set of resources.

2. ENTITY ORIENTED SEARCH AND EXPLORATION

The *Cultura* environment provides a challenging testbed for experimenting with new search and exploration technologies. In this paper we focus on *entity oriented search (EoS)* over the 1641 collection that we are developing for this project. Our approach is based on a combination of an *expressive query language*, *faceted search*, and *entity relationship (ER) graph navigation*. Users can express their

Copyright is held by the International World Wide Web Conference Committee (IW3C2). Distribution of these papers is limited to classroom use, and personal use by others.
WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

initial information need using a wide range of queries, spanning from simple free-text queries to more structured constraints over entity properties and their relationships with other entities. This allows utilization of the search system to query entities and their relationships by both expert users and end-users who are neither familiar with the query language nor the data model.

The output of the search system is a ranked list of entities that match the user query. Similarly to traditional faceted search systems, the system provides a distribution of retrieved entities over the facets they belong to, including facets of various entity types, properties, and related entities. The user in turn can exploit such facets to focus the search on a specific entity type or attribute, or to explore another direction by navigating to another related entity in the ER graph.

Whenever the user chooses to either filter the current result set based on some facet or to follow related entities using a relationship facet, the user query is automatically refined to reflect her choice using structured query predicates, while releasing the user from the need of understanding the underlying query language. On each iterative query step, similarly to traditional faceted search systems, the system provides the user with a useful report on facet distributions, revealing the number of sub-results expected for each facet choice.

Consider for example the following search task over the 1641 collection: Sir Phelim O'Neill was one of the leaders of the Irish rebellion, and as such, his name appears repeatedly throughout the depositions and is mentioned in a variety of contexts. He is often accused of being directly involved in events happening during the rebellion. An expert researcher may wish to identify the full extent of the allegations made against him by the deponents. The completion of such a task through the manual analysis of manuscript resources would take months, if not years, and therefore the full extent of his actual involvement has never been accurately identified.

Cultura will enable the user to identify where, and in what context, Sir Phelim O'Neill is mentioned, as well as the nature of the allegations made against him. The noisy text is normalized to address variation in spelling and name ambiguity (e.g. Sir Phelim O'Neill, Phelin, felim, FFelim, O'Neill, Neil, Onell), thus, all occurrences of Sir Phelim O'Neill will be detected. Other entities that are related to him (individuals, locations, events) will be associated with him, thus enabling effective exploration and navigation throughout the collection. In addition, related information to retrieved entities will be exposed to give the searcher the historical context, including biographical information on the individuals involved, and background detail on the locations and events involved.

3. SYSTEM DESCRIPTION

In the following we describe the EoS system developed for the *Cultura* environment. We briefly describe the content analysis component and mostly focus on the search and exploration component of our system.

Content Analysis and Entity Extraction

Figure 2 illustrates the data flow of the system during indexing. At first, the (digitized) content of the depositions is tokenized, normalized, and entities and relationships are extracted through a *UIMA* pipeline (*http://uima.apache.org*),

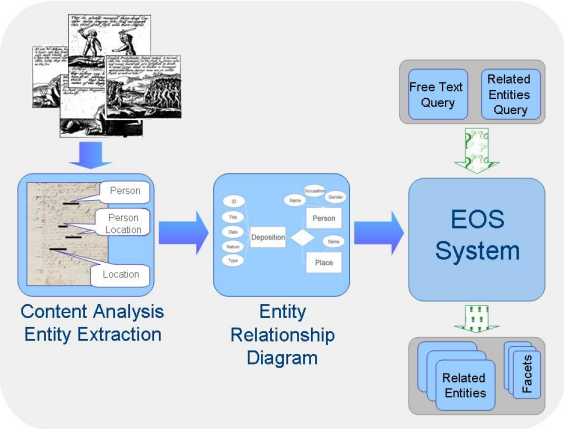

Figure 2: System data flow

using IBM *LanguageWare*, a natural language processing (NLP) technology that comprises a range of NLP functions, including text segmentation and tokenization, normalization, entity and relationship extraction, semantic analysis and disambiguation.

The 1641 collection is innately noisy, containing non-standard spelling, poor punctuation and obsolete grammar and word forms. We apply *LanguageWare* with specific tailored ancient English dictionaries for text tokenization and normalization, to remove issues of spelling, grammar, and punctuation. Once the content has been normalised, further analysis is conducted to perform named entity extraction and identifying attributes and relationships between entities. The extracted entities from the 1641 collection are individuals, events, and locations, along with their offset within the depositions, attributes (e.g. person's origin, occupation, religion), and interrelationships.

Entity Relationship Diagram

Our model is based on a simplified version of the conceptual entity-relationship model. In this model, an entity e is defined as any object or "thing" that can be uniquely identified and distinguished from other entities. Each entity has a type $e.type$ used for its classification. Each entity has a set of one to many attributes $e.a$ used to describe the entity's properties. Each attribute a has a name $a.name$, a type $a.type$ and a value $a.val$. An entity's key $e.key$ is further defined over its attributes, consisting of some minimal attribute subset that can be used to uniquely identify the entity. Each entity must have at least one key.

A relationship r captures an association between two or more entities, termed relationship members. Each relationship has a type $r.type$ and is uniquely identified by the combination of its entity member keys and attributes, denoted $r.key$. Each relationship entity member $r.e$ may have a role $r.e.role$ that captures the role this entity "plays" in the association. A relationship may further have zero to many attributes which can be used to describe its context.

Figure 3 depicts the *entity relationship digram (ERD)* for the 1641 collection. The rectangles represent the entities of the system (depositions, people, events, and locations), along with their associated attributes (eclipses), where the underlined attribute stands for the entity's key. The attributes of a deposition are unique id, title, content, and

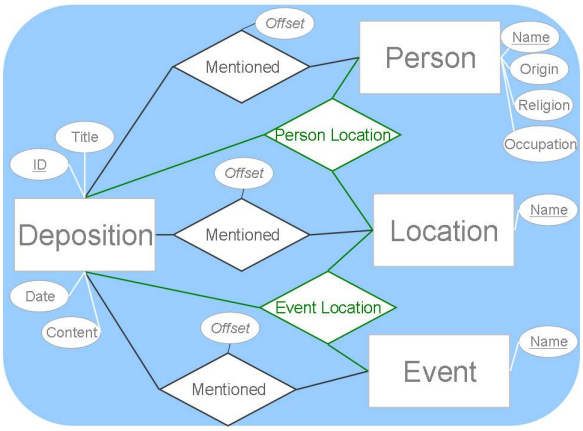

Figure 3: ERD for the 1641 collection

date. A person entity consists of a name and might have other attributes like occupation, origin, and religion. Events and locations are currently described by name only.

The relationships between entities are represented by a diamond shape. When a person, event, or a location is mentioned within a deposition, a *Mentioned* relationship between the two will be formed with a relationship attribute of the entity's offset within the deposition's content. The entities' offset is kept to enable measuring the relationship strength based on proximity between two entities. We assume that the closer the two within the content, the more related they are.

Whenever a relationship between two entities $(e1, e2)$ is extracted from deposition d, the relationship formed would be of the triplet $(d, e1, e2)$, along with the offset location of the two entities within the deposition's content as its attribute. In the current implementation, *LanguageWare* only identifies Person-Location and Event-Location relationships within the deposition content. Further relationship extraction rules are planned to be added in the future.

Indexing

We use an extended faceted search solution for indexing and searching entity-relationship data [2]. This implementation generalizes the dual entity data representation previously proposed by Amitay et al. in the context of social search [1]. Each entity is dually represented as a searchable document and as a category of all other entities it relates to. Thus, the categories of retrieved entities enable browsing over the ER graph.

However, this dual representation supports only binary relationships. Yogev et al. [2] presented a generalized solution which enables to capture arbitrary n-ary relationships by representing each relationship instance in the system as a *category set* that contains the categories of all participating entities. This representation is required when searching for specific relationship instances, for example, the set of depositions where a specific person was mention in relation with a specific location, a set that cannot be identified by binary relations only.

Each entity in the model is represented by a logical document which consists of three main building blocks, namely, *Fields*, *Categories*, and *Category sets*. The document's fields represent the entity attributes. The document's categories represent all entities which are related to the corresponding entity with at least one relationship. Any relationship in which the entity participates is represented by a category set that contains all categories of participating entities. This dual representation allows efficient search for entities, either through their attributes as well as through their relationships with others.

3.1 Query model

A query over ER data can be used to discover entities based on patterns of interest, either by directly querying entities, their relationships, or both. A query result is a list of entities, possibly of various types, ranked according to their "relevance" to the query.

Formally, a query q is a collection of one to many predicates. Each predicate describes some entity pattern, and the result of each predicate is expected to be a set of entities that "match" the specified pattern. Different predicates can be combined using the `AND`, `OR`, and `NOT` logical operators to construct a more complex query predicate. Three types of query predicates are supported, namely, entity attribute predicates, free-text predicates, and entity relationship predicates.

Entity attribute predicates

An entity attribute predicate allows to query entities based on their type and attributes. The basic form of such predicate is: `entityType.attributeName:attributeValue`

`entityType` defines a specific entity type or `*` for all possible types; `attributeName` defines a specific attribute name or `*` for all possible names; `attributeValue` defines a pattern of the attribute values of interest or `*` for all possible values. `attributeValue` can define either a specific value, or a range of possible values denoted as `[lowerBound TO upperBound]`, specifying a lower bound and upper bound. The following is an example for an entity attribute query, looking for all persons whose name is "Phelim O'Neill":

```
Person.name:"Phelim O'Neill"
```

Free-text predicates

As already mentioned, issuing structured queries such as the aforementioned entity attribute queries requires some level of knowledge regarding structure. However, such knowledge is usually attained only by system experts, while the majority of users cannot translate the information need into structured queries. Free-text query predicates treat entities as text documents and therefore query the entities' content regardless of their structure.

To efficiently support free-text queries, a special searchable field is added to the document entity. The text value of this field is the concatenation of all attribute values represented as string literals. Some attributes may contain data which is irrelevant for free text search (e.g. the entity id). Therefore, the model allows defining for each attribute whether its string representation should be available for free-text search, and the text value of the searchable field concatenates only those attributes' values.

The following example which demonstrates a free-text predicate combined with restriction on the required entity type, returns all depositions mentioning "Phelim O'Neill":

```
Deposition.*:* AND "Phelim O'Neill"
```

It is worth noting that `Deposition.*:*` can be expressed by `Deposition.*` or `Deposition.`.

Relationship predicates

A relationship query predicate can be used to retrieve entities based on relationships they participate in. For that, a relationship predicate allows to define the entity relationship participation pattern, including the type of relationship, its relationship member patterns and their roles, and relationship attributes that limit the search according to some context. The result of a relationship query predicate is a set of entities that participate in the relationship subject to the relationship specified pattern.

More formally, a relationship predicate is defined by the following expression:

```
relatType ( (WITH *) | (withExp (AND withExp)*))
```

`relatType` specifies a specific relationship type or `*` if any type should be considered. `withExp` further denotes an expression that takes the form: `WITH (relatMem | relatAtt)`. `WITH` is a special operator used to describe a single relationship entity member pattern (`relatMem`) or relationship attribute (`relatAtt`) pattern. `relatMem` may include any combination of free-text or entity attribute predicates.

The following query demonstrates the usage of the `WITH` expression within a relationship predicate, looking for locations that are related to a person named Phelim:

```
Location.  AND (Person-Location WITH Person.name:"Phelim").
```

Indirect relationship predicates

The previous example retrieves entities that are directly related to to the input entity. However, in many situations we are interested in indirectly related entities, for example those that are mentioned together with the queried entity. The following query retrieves all persons that are mentioned together with "Phelim O'Neill" in at least one deposition:

```
Person.  AND (Mentioned WITH (Deposition.  AND
  Mentioned WITH Person.name:sir_phelym))          (1)
```

The inner predicate retrieves all depositions mentioning the person "Phelim O'Neill"; the second WITH predicate retrieves all entities mentioned in those depositions; the outer Person predicate filters the result set to person entities only. Figure 4 shows the search results for this query, i.e., the most related persons to Phelim O'Neill.

The rich query model described above is mostly useful for expert users, not for novices. For a user who is not familiar with the query language or the data model, an interactive approach is preferred, where the information need of the user may be gradually covered by a series of query reformulations, starting from the user's original query, based on user selections.

As an illustrative example, lets consider again Query 1. The information need expressed by this query could be equivalently achieved using the following interactive query session

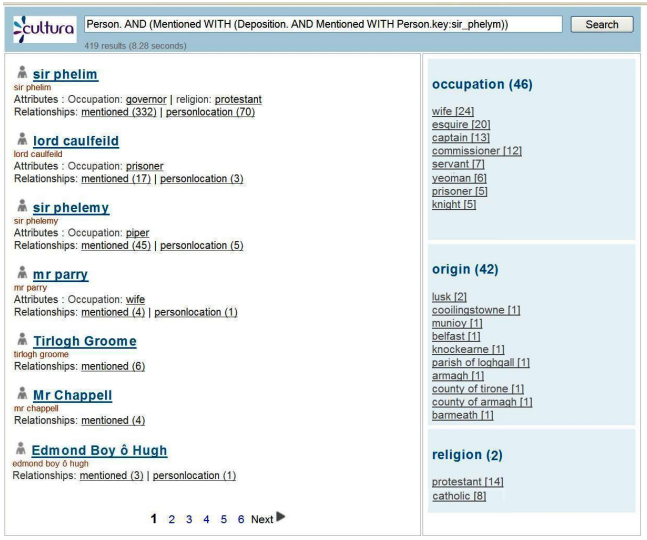

Figure 4: Search results for query 1: All persons related to (mentioned together with) "Phelim O'Neill"

having four steps:

```
q0 = "Phelim O'Neill"
//All entities containing this name are retrieved
//The user selects the Person facet
q1 = Person.  AND "Phelim O'Neill"
//All persons with this name are retrieved
/The user selects the governor "Sir Phelym"
q2 = Person.name:sir_phelym
//The user focuses on mentioning depositions
q3 = Deposition.  AND Mentioned WITH Person.name:sir_phelym
//The user focuses on mentioned persons in those depositions
q4 = Person.  AND (Mentioned WITH (Deposition.  AND
  Mentioned WITH Person.name:sir_phelym)))
```

4. SUMMARY

We are just at the beginning of exploring the capabilities of EoS for cultural heritage collections. The effectiveness of the search system strongly depends on the availability and quality of advanced entity extraction tools and the accuracy of discovered entities and their interrelationships. Our systems faces the same classical problems in the entity extraction field such as name disambiguation and name resolution which are even more complicated in noisy text. In addition, while a wide range of information needs can be answered using our interactive query language, there are still many queries which cannot be answered, for example queries which further impose constraints on the relationships between results. Advanced user interfaces for EoS should still be developed to ease interaction and exploration of the search results. Finally, entity ranking methodologies should be investigated for EoS in general, and for each collection in particular. We leave these extensions for future work.

5. REFERENCES

[1] E. Amitay, D. Carmel, N. Har'El, S. Ofek-Koifman, A. Soffer, S. Yogev, and N. Golbandi. Social search and discovery using a unified approach. In *Proceedings of Hypertext and Hypermedia*, pages 199–208. ACM, 2009.

[2] S. Yogev, H. Roitman, D. Carmel, and N. Zwerdling. Toward Expressive Exploratory Search Over Entity-Relationship Data. In *WWW*. ACM, 2012.