Open Access

Open Journal of Semantic Web (OJSW)
Volume 1, Issue 2, 2014
http://www.ronpub.com/journals/ojsw
ISSN 2199-336X

Developing Knowledge Models of Social Media: A Case Study on LinkedIn

Jinwu Li [A], Vincent Wade [B], Melike Sah [C]

[A] School of Computer Science and Statistics, Trinity College Dublin, Dublin, Ireland, lij7@tcd.ie
[B C] Centre for Global Intelligent Content, School of Computer Science and Statistics, Trinity College Dublin, Dublin, Ireland, vincent.wade@scss.tcd.ie, sahm@scss.tcd.ie

Abstract

User Generated Content (UGC) exchanged [1] via large Social Network is considered a very important knowledge source about all aspects of the social engagements (e.g. interests, events, personal information, personal preferences, social experience, skills etc.). However this data is inherently unstructured or semi-structured. In this paper, we describe the results of a case study on LinkedIn Ireland public profiles. The study investigated how the available knowledge could be harvested from LinkedIn in a novel way by developing and applying a reusable knowledge model using linked open data vocabularies and semantic web. In addition, the paper discusses the crawling and data normalisation strategies that we developed, so that high quality metadata could be extracted from the LinkedIn public profiles. Apart from the search engine in LinkedIn.com itself, there are no well known publicly available endpoints that allow users to query knowledge concerning the interests of individuals on LinkedIn. In particular, we present a system that extracts and converts information from raw web pages of LinkedIn public profiles into a machine-readable, interoperable format using data mining and Semantic Web technologies. The outcomes of our research can be summarized as follows: (1) A reusable knowledge model which can represent LinkedIn public users and company profiles using linked data vocabularies and structured data, (2) a public SPARQL endpoint to access structured data about Irish industry and public profiles, (3) a scalable data crawling strategy and mashup based data normalisation approach. The proposed data mining and knowledge representation proposed in this paper are evaluated in four ways: (1) We evaluate metadata quality using automated techniques, such as data completeness and data linkage. (2) Data accuracy is evaluated via user studies. In particular, accuracy is evaluated by comparison of manually entered metadata fields and the metadata which was automatically extracted. (3) User perceived metadata quality is measured by asking users to rate the automatically extracted metadata in user studies. (4) Finally, the paper discusses how the extracted metadata suits for a user interface design. Overall, the evaluations show that the extracted metadata is of high quality and meets the requirements of a data visualisation user interface.

Type of Paper and Keywords

Communication: *LinkedIn, Ontology, SPARQL, Linked Open Data, Data Mining and Data Normalisation*

1 Introduction

Social media sites such as Facebook and Twitter are gradually changing how information is shared and exchanged around the world, and has become a new source of valuable knowledge. Users generate data with a feeling of reward, since they can gain recognition and attention from other users [1]. For researchers in Information Technology (IT) industry, UGC represents a new, inexpensive and fast way to obtain data which was barely impossible in the past.

LinkedIn.com, the world's largest profession network [1], contains a large amount of hidden career and country based industry information, but as yet not discovered or utilised. Unlike Facebook and Twitter, the study of LinkedIn.com is not getting as much attentions as it perhaps should.

Semantic Web, as believed, will be the "Next generation of knowledge representation and processing technology"[2]. It aims to extend the world of human readable Web documents to a new world of machine understandableand interoperable knowledge.

1.1 Motivations

The lack of existing research on LinkedIn.com provides an opportunity. In our work, we use LinkedIn Ireland as the study subject, and aim to develop a reusable, queryable knowledge model for LinkedIn public profiles. The importance of our work can be described in several aspects:

Firstly, obtaining full insights about Ireland industry, personal skills and professionals' education background is always important for a number of entities, e.g. government, semi state bodies, commercial operators. This can be easily scale to LinkedIn worldwide.

Secondly, it is a good complement for government statistics about industries and professionals. The effectiveness and timeliness of UGC can always guarantee that we receive the first hand data.

Thirdly, with our approach, an online public dataset will be provided so that everyone who interest in Ireland facts can query the public SPARQL endpoint with complex queries in order to receive information.

Finally, a user interface can be built on top of the knowledge model that we devloped in order to support complex queries about our dataset. Government, practitioners in Human Resources (HR), and job seekers are possible target audience. We briefly show an example user interface that uses the extracted knowledge from the LinkedIn public profiles.

1.2 Research Questions

We try to address the following questions in this paper:

1. Can we take advantages of Semantic Web technologies to build a knowledge model and generate a dataset from the publicly provided data by LinkedIn.com?

2. In order to achieve interoperability and common agreement, can we reuse existing ontologies and public vocabularies, and integrate them into our model?

3. How useful is the extracted data? Can we guarantee a level of metadata quality thus the extracted metadata will be useful?

4. Is it possible to have a working user interface that make use of our dataset and demonstrate some useful use cases?

1.3 Contributions

Reusable knowledge model from LinkedIn public profiles

We present a queryable, extendible knowledge graph (Figure 1) that capture the data relationship of LinkedIn.com public profiles and company profiles. It can be used by LinkedIn internally or by other researchers who are interested in UGC of LinkedIn.com.

Public online SPARQL endpoint for complex querying about Irish industry

We publish our dataset through a SPARQL endpoint at http://128.199.243.88/test/. It is a standard SPARQL endpoint that powered by 4store[3]. Our endpoint accept HTTP POST requests and support a number of RDF format, such as XML, JSON, plain and turtle. Anyone who is interested in discovering Irish industry and college facts can use this service. Here are some information and statistics about the dataset. The dataset was extracted from LinkedIn Ireland public profiles in August 2013. In total, we have 819,488 triples and 415,916 RDF links. In addition, the dataset contains 13,014 personal profile, 24,778 company profile and 15917 skill instances.

We also published the RDF dump of the extracted metadata [2] where interested researchers can directly access the dataset. Please note that user profiles were anonymized and any personal information was removed from the published metadata.

Scalable crawling strategy

Our crawling strategy is scalable to any numbers of LinkedIn public profiles. If we consider a profile is a node, since in the profile, LinkedIn will suggest 6 to 8 similar profiles (nodes), the graph is expanding very quickly (exponentially). Therefore, data mining practitioners (and other researchers) can make use of our strategy, as discussed in section 4, to download profiles in any subdomains of LinkedIn.

Data normalisation and mashup based city information extraction strategy

[1]According to their official website: `http://www.linkedin.com/about-us`

[2]`https://github.com/ljw7630/MasterDissertationProject/raw/master/result/archive.tar`

Based on our user study, the strategy is widely accepted by our survey participants, with average of 0.85 F-score and 4.089/5 user rating. The approach can be generalized to obtain city information by company names. Another approach is to use Google reverse Geocoding service [3]. However, even we do not have comparison result on hand, the provided results by reverse geocoding service seems to worse than using a country's yellow page database.

2 Related Work

2.1 Introduction

Our work focuses on developing knowledge model representations of UGC in the context of Semantic Web. Ideally, the data model should be general enough so that new knowledge can be inferred from the extracted data. Since we use RDF triples to represent data, SPARQL will be used as the query tool to answer questions.

In order to generate knowledge models from raw HyperText Markup Language (HTML) files of LinkedIn public profiles, a number of challenges are required to be addressed, such as data extraction, knowledge modelling, content integration and evaluation of the extracted data. In this section, we discuss related works in these fields in detail.

2.2 Data Extraction

[4] provides an up-to-date survey on web data extraction. In this paper, three common techniques for web data extraction is listed: 1. Tree-based approach: Analysis on Document Object Model (DOM) trees. 2. Web wrapper: Use procedures to seek and find the required data. 3. Machine learning approach: Using reasoning or other Artificial Intelligence (AI) techniques to find the data of interest. In addition, the paper provides a full list of famous applications that are being used in the real world. Since LinkedIn personal and company profiles are represented with few page templates, in our approach, a Web wrapper method will be used to extract data. This means that we do not need to learn many templates and a Wrapper approach is suitable in our context. Although LinkedIn pages do not contain structured knowledge, the formats are consistent and barely change. Even when profiles are incomplete, we can handle this in our Wrapper approach.

[5] presents a framework that exploits Web documents using a "Tree Alignment Algorithm", in which they build trees iteratively and try to find record boundaries and repeating patterns. Then, they build "conceptual graphs" to represent domain knowledge. Finally they map the conceptual structure to the extracted data items. Since the conceptual graph is directly mapped to a database schema, this approach can reduce the time of converting the extracted content to database records. The Tree Alignment Algorithm here could be very useful in our work, since we are also trying to extract data of interest from semi-structure LinkedIn profile files. However, as far as the author can tell, their mapping approach might be not scalable, as it requires manual creation of a "conceptual graph", which makes this approach no better than using pure "Regular Expression" approach. Nevertheless, we can learn from the "mapping" process and adopt it. In our case, Levenshtein distance (Edit distance) or Cosine similarity (Vector space model) could be used to classify terms and correct typos.

Finally, there are many systems for data extraction and semantic annotation, such as GATE [6], UIMA [7], PEARL [8], APACHE STANBOL [9], and CODA [10]. The common aspect of all these systems is that they provide a strong engineering infrastructure and dedicated languages to manage the extraction and annotation processes (e.g. GATE's JAPE and UIMA's RUTA). In our context, since the LinkedIn pages have a very specific layout, a Wrapper approach was tested. However, in future we can investigate how we can benefit from the existing systems in order to improve our work.

2.3 Knowledge Modelling

We can think of an Ontology is a collection of terms that defines the concepts and relationships of an area [4]. It is the cornerstone of the Semantic Web; by publishing ontologies and combining them together, the web of knowledge will finally be constructed.

[11] mainly focus on the strategy of building simple Ontologies for social networks. A tripartite model is suggested, specifically an Actor-Concept-Instance model. The paper demonstrates the applicability of the model using two examples. The paper also shows how the ontology is emerged based on the model and how it is extended to support Ontology Extraction from Web Pages. However, this approach mainly about Community Ontology Construction, as LinkedIn public profiles has no or very limit connection information. In our approach, we will try to enhance linkage/mapping to other datasets in order to increase interoperability of our model. For example, we have inter-linkage to: DBpedia (for general information), Academic Institution Internal Structure Ontology (AIISO) (for academic skills, courses) and Friend of A Friend (FOAF) (for person information). In

[3] `http://developers.google.com/maps/documentation/javascript/geocoding?csw=1#ReverseGeocoding`

[4] `http://www.w3.org/standards/semanticweb/ontology`

particular, we focus on reusing linked open data vocabularies and linking to existing objects whereas possible, which is a good practice in ontology engineering.

[12] focuses on extracting information from Artificial Intelligence related conferences and workshops by building an Ontology for AI. Again, it constructs domain concept knowledge from nested tags. For example, in HTML, <h1>means a more general term than <h2>, so an instance of <h2>is a subclass of an instance of <h1>. Then in the optimisation process, they perform“ontology pruning and union” to handle concept duplication. However, this strategy might result in wrong classification. To summarise, this approach is very useful provided that the user knows the concepts in the web pages for hierarchical classification. It could not be generalized for other loose structured websites.

In this work, our goal is to build an Ontology for LinkedIn public profiles using automated process. The reasons for doing that are, firstly, Linkedin.com is one of the main knowledge sources for professional information. People publish their education, skills, experiences on the site and we expect these kinds of information can answer complex questions. For example, decision maker may want to track the trending of an industry by looking at the number of employees and the number of new startups in the specific area. Secondly, we choose Semantic Web because we want to link the knowledge into the Linked Open Data (LOD) to maximise the usability of our data. In addition, the interoperability feature provided by RDF can lead to flexible use of triples (Again, in this case, data can drive the application developments)

2.4 Content Integration and Classification

One of the major problems in Information Extraction (IE), especially in social media information extraction is the variety of the similar words. For example, in LinkedIn.com, a user can claim himself as ”Graduated from Trinity College Dublin”, meanwhile, another user will say she is studying at ”TCD”. When we build an ontology and try to link our data to LOD, we really have to be very careful about declaiming a term more than once. A false positive result is also not acceptable, in a way that we might misclassify address ”Dublin” in ”Dublin Core” as the capital of Ireland. So finding ways to clean up the data and classify them correctly are considered two complex tasks in Information Extraction.

[13] introduces a widely used Open Source Search Engine: Lucene. We can simply take the advantage of the built-in keyword search feature to help us identify similar words in the content we extracted.

2.5 Evaluation of the Extracted Data

As everyone can publish their data on the Internet, the evaluation of the data quality becomes a very important aspect in Ontology building. People cannot reuse the data with bad quality, so publishing the data without quality assurance will significantly reduce the value and the reusability of the data. Therefore, we evaluated some metrics and a data assessment framework:

[14] proposes useful metrics to automatically evaluate the quality of the metadata. This is particularly important, since metadata quality assessment by human evaluators is a very time-consuming task and it does not scale. Therefore, we used these metrics to automatically assess overall metadata quality. In particular, we focused on data completeness and data linkage metrics. Metadata completeness measures how complete is the extracted metadata given the ideal metadata representation of a record. For example, in an ideal LinkedIn profile representation, a person record contains all possible metadata fields such as personal information, skills set, education background, work experiences, etc. In large datasets like our dataset, completeness metric could give an overall big picture about how complete is the person or company profiles. Hence, it can provide insights about the usefulness of the extracted metadata.

Moreover, we used data accuracy metric to measure how correct is the extracted metadata compared to the ground truth metadata provided by users. In particular, we utilized user studies and ask subjects to manually enter metadata values. Then, we take users answers as a ground truth and compare with the automatically extracted metadata, which is a common practice in data mining [15]. Using the ground truth, precision, recall and f-measure metrics can be utilized for data accuracy analysis [16].

Furthermore, using user studies, subjects can evaluate user perceived metadata quality by rating (i.e. scoring) the automatically extracted metadata. Finally, conformance to expectations is a way to test whether the schema meets the requirement of use cases, and supports arbitrary complex queries. Because our dataset will be used by another project: “Leveraging Power of Social Media and Data Visualisation”, we can evaluate the dataset by looking at whether the data is complied with the user and visualization requirements.

3 System Design

3.1 Requirements

In order to answer the research questions and produce public dataset at the end, the system should be capable to:

1. Use a knowledge model to describe how the data should be stored (in RDF format).

2. Download LinkedIn personal public profiles and company public profiles, where the data should be in HTML format.

3. Extract data from the raw HTML files.

4. Normalise the metadata to provide consistent and structured output, which means the system should be able to correct dirty data. Providing high quality metadata is an important requirement for the user interface (UI) that will use SPARQL queries to search and reason over the extracted metadata.

5. Convert the data into RDF triples, and store it in a publicly accessible triplestore.

Additionally, the system also has a number of non-functional requirements that needs to be fulfilled:

1. **Crawling Performance and Parsing Performance**
   The system should be able to download and parse enough number of profiles in a reasonable time. Because timeliness is the nature of UGC, if it takes too long to perform this, we lose the chance of getting first hand information.

2. **Query Performance**
   The system should be able to respond to the user interface queries in a reasonable time in order to make sure that the UI is usable.

3. **Metadata Fitness to the UI**
   The system should be able to extract the key metadata fields that is required by the visualization UI.

3.2 Knowledge Model

As discussed in the previous section, we decide to use Semantic Web technologies for knowledge modelling. The first step we need to perform is that we need to come out with an Ontology that can reflect the actual knowledge of LinkedIn personal profiles and company profiles. After investigating sample profiles from LinkedIn, we generated an ontology, called LinkedIn Ontology, by reusing existing concepts from the linked open data vocabularies whereas possible. Figure 1 presents a conceptual overview of the LinkedIn Ontology. The ontology is accessible online [5].

[5]https://github.com/ljw7630/MasterDissertationProject/raw/master/result/archive.tar

As shown in the figure, the model can be divided into three core concepts: Person, Education and Organisation. In LinkedIn personal profiles, a person might have current living city, skills, work position, job title, start date and end date of the position. In addition, a person might have education background, such as college name, major, degree and start year and end year of the college.

For a LinkedIn company profile, it might contain headquarters, company type (public, private own, etc.), industry type (e.g. IT) and company size.

Our knowledge model links the personal profiles with company profiles using the“position” concept. The whole graph is linked so that we can perform arbitrary queries. For example, we can discover the relationships between the education background and organisation through relations between person and position concepts.

One key point to note is Semantic Web is built around the idea of triples, which means an expression has the structure of “subject”, “predicate” and “object”. In the graph, the names in circle are “Class”, the link between two “Classes” is call “Property” (predicate), it is used to link an instance of one Class to an instance of another Class.

Moreover, we worked in cooperation with the data visualisation project for constructing the knowledge model. In particular, we specified 3 different end-user scenarios (as we explain in detail in section 5.4) and we work on the knowledge model to support these scenarios. According to discussions, we appropriately revised the knowledge model. In particular, an action research methodology was used with several cycles of Plan, Action, Observe and Reflect [17].

Reusing existing ontologies: Ontology reuse is an important concept in Ontology Engineering. According to [18], it increases the quality of the application, achieves interoperability, improves cost in ontology development and helps applications agree on the domain concepts. One mission of Semantic Web is to achieve data interoperability. If two applications cannot understand the meaning or the semantics of data, then these two applications cannot communicate. Therefore, we need to try our best to reuse existing, well known Ontologies so that other Semantic Web application can at least partially understand our domain, hence reduce the needs of Ontology mapping.

As illustrated in Figure 1, we partially reuse the following vocabularies or ontologies: Simple Knowledge Organization System (SKOS), AIISO, DBpedia, FOAF. However, the reasons for reusing them are varies:

1. Simple Knowledge Organization System (SKOS) [6]
   Just as its name suggest, SKOS is used for “knowl-

[6]http://www.w3.org/2004/02/skos/

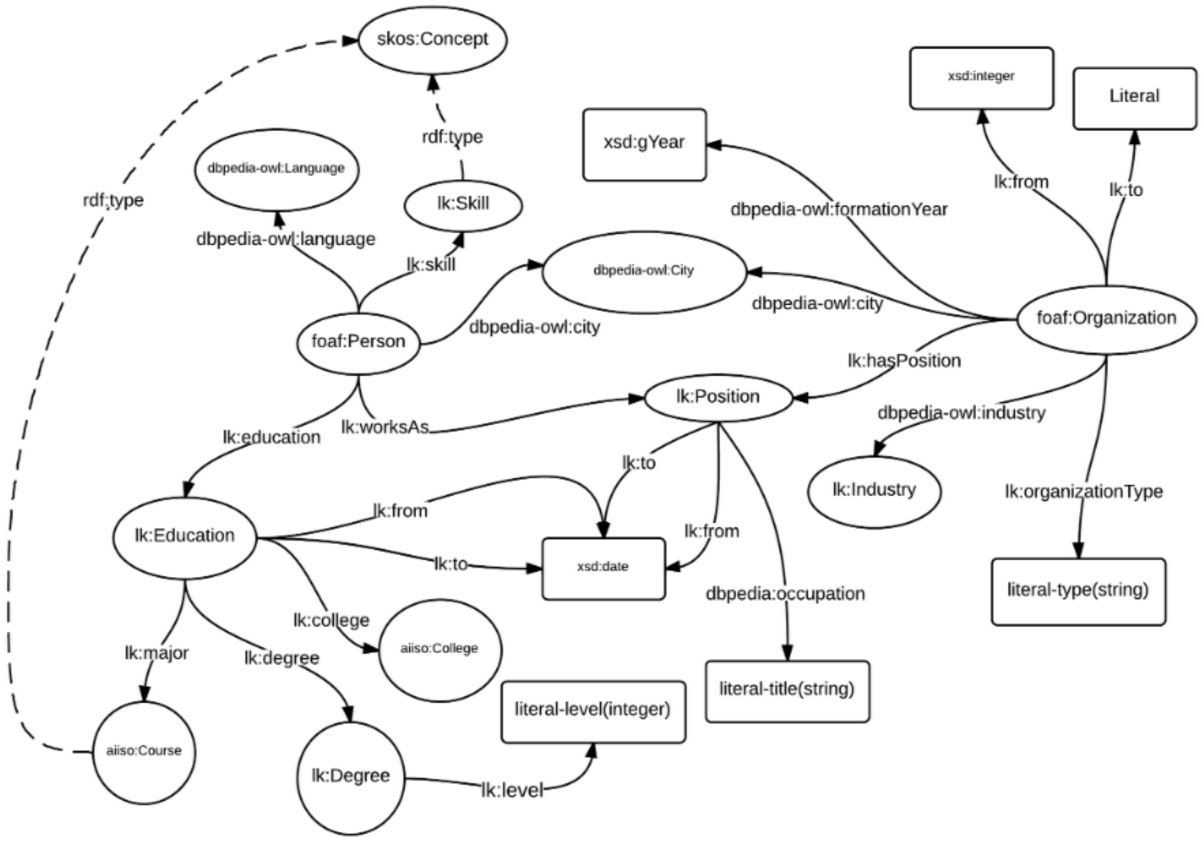


Figure 1: Conceptual knowledge model of the LinkedIn ontology

edge organisation", such as "structured vocabularies" and "classification schemes"[19]. We use SKOS representation to define academic disciplines (major) and relationships between them. For example, one LinkedIn user studies "Computer Science", another one studies"Artificial Intelligence". If we can define, in a SKOS manner, say "Computer Science" is a "broader" term of "Artificial Intelligence", then our system can handle the discipline hierarchy. It is also good for our endpoint UI that users our dataset. Since the hierarchy can identify both general and specific concepts. Thus, with SPARQL queries, users can reason and query inferred knowledge. To achieve this, we use DBpedia academic disciplines and convert to SKOS using skos:broader relationship. For instance, every academic discipline is represented as skos:Concept and it is connected to super or sub-concepts using skos:broader relationship.

2. AIISO [7]
   We make use of AIISO class definition to define our education concept. As the Ontology is used to describe the internal state of an Academic Institution, it has definitions about college, course (we also called major or discipline), and degree. Thus we do not need to create our own concept about education but reused these concepts instead.

3. DBpedia [8]
   DBpedia is the Semantic Web version of Wikipedia, and it is considered to be the nucleus of the Web of Data[20]. The more terms we can reuse from DBpedia, the more interoperability we can obtain. We use both class definitions and property definitions from DBpedia ontology: City (Class), city (property), industry (property), language (property), Language (Class).

4. Friend of a Friend (FOAF) [9]
   FOAF is another widely used ontology, which describes and links data in social networks[21]. We use foaf:Person to define our LinkedIn user and foaf:Organization to define the company in LinkedIn work experience.

[7]`http://vocab.org/aiiso/schema`

[8]`http://dbpedia.org/About`

[9]`http://www.foaf-project.org/`

DBpedia, FOAF and AIISO form the backbone of the developed knowledge model. Our LinkedIn ontology and the used ontologies are available online [10].

3.3 System architecture

According the requirements, we design the system as shown in Figure 2. In particular LinkedIn.com does not provide Application programming interface (API) for downloading their public profiles. Instead, we use Google search results to achieve this. A brief system flow is given as follows, where more details are discussed in the next section:

1. Use Google search engine to query user profiles from LinkedIn Ireland.
2. Receive the response from Google, and download the html files base on the Uniform resource locator (URL) in response content.
3. Call the parser to parse the HTML files and get the fields that will be used, as shown in the Figure 1.
4. Then the extracted data is sent to a data normalisation module (Lucene search engine in this case), where the data is cleaned up and normalised.
5. Finally we build the RDF triples using the normalised data and store them into our triplestore.

3.4 Design Decisions

Several design decisions are discussed as follows:

Firstly, we decide to separate the profile downloader module from the profile parser module. This means that first we run the module and download enough profiles. Then, we call the parser module. The reason behind this is downloading profiles consumes large amounts of Google and LinkedIn resources, which implies that our connection can be switched off at any time by the remote server. Therefore we do not want to perform parsing together with downloading, since we do not want to spend extra effort in network problem handing. Another point is that our final result will be in RDF format, hence if we perform parsing together with downloading, extra storage and data structuring is required to store the intermediate results.

Secondly, a simple database is required to keep track of which url is downloaded, which profile is parsed and had been converted into RDF format. Because we are handling a very large amount of profiles, and the correctness of the parser has to be adjusted during parsing, thus it is unrealistic to assume that our program can parse and convert all the data with one-click. For example, if an unhandled exception occurred and stop the program, we can start parsing from the remaining profiles if we have a database that keeps track of the status.

Thirdly, just as the first reason we highlighted, the parser module, data normalisation module and RDF conversion module will form a pipeline. It means that the output from the previous module will be fed as an input to the next module. The reason is, we do not have to store the intermediate results from the previous two modules.

4 IMPLEMENTATION

4.1 Programming Languages and Corresponding Libraries

4.1.1 Python and Its Libraries

As discussed in the previous section, we used Python as the main programming language. The following Python libraries were used in our work:

- **Beautiful Soup**
  A easy-to-use library for parsing HTML documents. It provides both flexibility and performance for parsing HTML or XML files.
- **RDFLib**
  RDFLib is a library that allows Python to manipulate RDF files. It has RDF parser, serializer and also SPARQL 1.1 implementation. This library is all we need for RDF format conversions. In addition, we can also test our data with its built-in SPARQL query API.

4.1.2 Java and Lucene Text Search Engine

For data normalisation, the final decision is to use Lucene text search engine. According to [22], Lucene is a simple and powerful API for full-text indexing and searching. In our work, we took advantage of its keyword search and fuzzy search (by using Edit Distance[23]).

Another option is pylucene. It is the Python wrapper for Lucene text search engine. However, the project seems to only working on some particular version of Java Virtual Machine (JVM), which means that it is difficult to migrate to arbitrary machine.

Therefore, we use Java to implement our data normalisation engine and use socket to communicate with our Python modules. The process of the socket communication in shown in Figure 2.

[10] `https://github.com/ljw7630/MasterDissertationProject/raw/master/result/archive.tar`

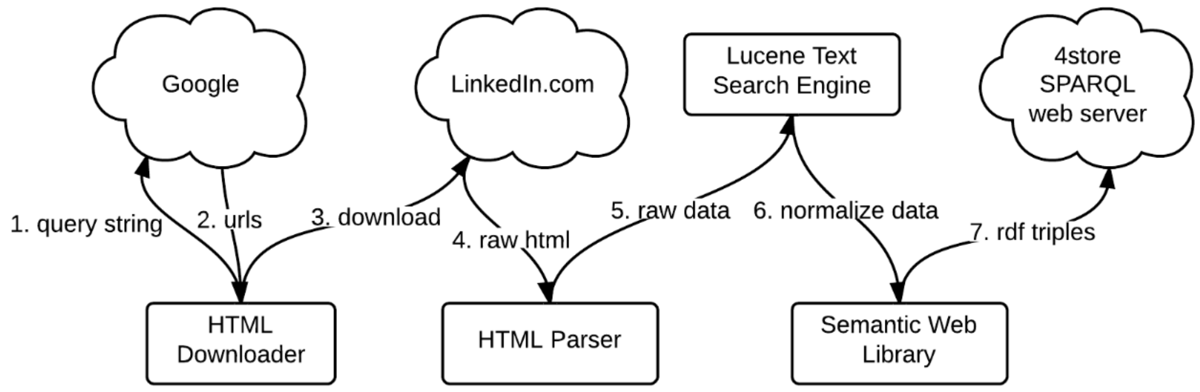


Figure 2: System Architecture

4.1.3 System Environment

The technologies and libraries that are used in our work are all open source. In order to utilise the command line and built-in tools, UNIX-like system is used in development, however our code can run on Windows machine. It requires Python2.7 and Java1.7.

The production environment is Amazon EC2 64 bit Ubuntu12.04, Intel Xeon Central processing unit E5-2650 2.00GHZ, 4G memory.

4.2 Crawling Strategy and Metadata Extraction Process

4.2.1 Crawling

In LinkedIn API documentation [11], there is no method for one to download public profile without OAuth Authentication. One point to note here is we will not disclose LinkedIn users personal information and we are downloading the public profiles. Therefore, we are legally valid to perform this action. In addition, all the extracted metadata is made anonymous such as we gave randomly generated URIs for profiles and clear all personal information (e.g. names, websites, emails, etc.). The extracted information is then used for analyzing job statistics in Ireland.

Inspired by [24], we decide to combine google keyword search and google site operator to get the profile urls. The google query is constituted by two parts: the keyword part and the query domain part. In order to receive more query results for the profile download module, we decide to use common Irish names as keywords. For the domain part, after carefully investigating same samples, we decide to use "http://ie.linkedin.com/pub/" as the site operator. The url means: "public profile directory for LinkedIn Ireland". Therefore, the final query string is: "Name site:http://ie.linkedin.com/pub/".

[11]LinkedIn Developers' Documentation: `https://developer.linkedin.com/apis`

In addition, every downloaded LinkedIn personal public profile has a section called "Viewers of this profile also viewed...", in which LinkedIn will suggest around six to eight similar users to the viewer. Therefore, one downloaded profile can link to 6 to 8 profiles, which we can perform this again and again. Even conflicts might happen, ideally we can download most of the profiles in LinkedIn Ireland.

4.2.2 Mashed up based City Information Extraction

As we mentioned before, the extracted metadata is utilized by a visualization UI. In particular, the UI uses city information to create statistics and compare various job title, skill, education trends across Ireland. Therefore, we need city information for every profile, if possible. Without the city information, our triplestore cannot reflect facts of interest. Although LinkedIn users can specify current city, our observations showed that some profiles do not have city information. Therefore, we come out with a strategy to automatically infer this information using a mash-up based approach. The strategy constitutes of three cases as follows:

1. Case 1: If the person's current living city is shown on the profile, we use our HTML parser to obtain it.

2. Case 2: If the information is not in the profile, we can sort the companies that this person had worked in, in reverse chronological order. For each of the

company, we query goldenpages.ie [12], which contains information about most of the companies. We obtain the first returned city as the person's current living city, and also set all return cities as the company's located city. (As a large amount of companies have offices in different cities.)

3. Case 3: If the person has no work experience or the company he worked in doesn't register on goldenpages.ie, we get the city base on his education experience, in reverse chronological order, again.

4.2.3 Metadata Extraction and Data Normalisation

As mention in Python and Its Libraries, we use BeautifulSoup to extract metadata from raw HTML files. Figure 1 shows the fields of interests. Abstract Data Types (ADTs) are created to store intermediate data. We need to perform data normalisation on the metadata we extracted. Normalisation is vital in this work since UGC always contain noisy information (error or synonyms). For example, one user might have a skill call "JavaScript", another user might have "JS" skill, we need to handle these synonyms or typos.

We use Lucene search engine to handle this problem. Basically, we are looking for ground truth from Web resources. Then we compare our extracted metadata with the ground truth strings to get the most likely string representation of the metadata. For example, a user who has education in "Trinity College Dublin, Ireland" will be likely to classify to "Trinity College Dublin" as it is the most similar string. To achieve this data normalisation, first we require a publicly available dataset to reprsent the ground truth. After investigating available resources, we decided to use Research Gate Topics [13] as a ground truth for skill set. In particular, Research Gate contains diverse list of skill topics. Similarly, for degree normalisation, we used "British degree abbreviation" [14] from Wikipedia to define shorthands for each degree we encounter. We use "Universities in Ireland" [15] to define the university names, thus typos or name variances on Irish university names can be handled by our Lucene engine keyword search. Finally, we used"List of academic disciplines" [16] from Wikipedia to define our subject hierarchy. In particular, this hierarchy is converted to RDF using SKOS as we explained in the previous section. Using the disciplines hierarchy, applications who use our dataset can reason the relationships between subjects.

5 Evaluations

It is very important to evaluate our work after the development.In this section, we explain evaluations of our system in detail. In particular, we perform 5 different types of evaluations: Automated metadata quality analysis, metadata accuracy, user perceived metadata quality, metadata fitness analysis to a user interface and system performance. All of these evaluations are important since; without high quality data, our dataset cannot be trusted by users; without high performance, our approach cannot be accepted by other developers and data is not queryable a front-end UI.

5.1 Automatic Metadata Quality Analysis

[14] suggests a number of automatic quality metrics in order to assess metadata. In large datasets like our case, it is not feasible to assess metadata quality manually since manual evaluation is labour intensive, time consuming and does not scale. Such automatic analysis can provide some feedback to assess metadata quality[16]. In our approach, we use data completeness and data linkage metrics. Data completeness is used to measure how complete is the profiles given the ideal representation. It is an important statistics we can obtain from LinkedIn.com as people are always interested in how complete are the profiles in general. It is also a hint for future research since we can quickly identify sparse data fields and intensive data fields.

5.1.1 Metadata Completeness

We present the completeness of profiles in LinkedIn.com.

Definition: [14] defines data completeness as: A degree of metadata contains all information required to have ideal presentation. To obtain it we can simple count the number of fields that contain the metadata and divided it by the total number of instances:

$$C = \frac{\sum_{i=1}^{N} F(i)}{N} \qquad (1)$$

In Equation 1, F(i) is 1 when the field has complete metadata and 0 when the field is empty. N is the total number of instances.

Table 1 shows the total number of personal profiles, company profiles and total number of skills in all of the profiles. These numbers are base numbers that will be used to calculate the percentages in the following tables.

[12]http://www.goldenpages.ie/

[13]http://www.researchgate.net/topics/A/ to http://www.researchgate.net/topics/Z/

[14]http://en.wikipedia.org/wiki/British_degree_abbreviations

[15]http://www.4icu.org/ie/

[16]http://en.wikipedia.org/wiki/List_of_academic_disciplines_and_sub-disciplines

Total number of public personal profiles	13014
Total number of company profiles	24778
Total number of skills	15917

Table 1: Total number of personal profiles, company profiles and skills

Public personal profile completeness: Many people do not fill their complete work experiences and education backgrounds into LinkedIn. Therefore, in our 13014 randomly downloaded and selected profiles, we can observe an overview of the percentages of people that filled these sections.

	Number	Percentage (of personal profiles)
Profiles that have work experiences	11501	88.4%
Profiles that have education	9913	77%
Profiles that have skills	10511	80%
Profiles that have city information	10158	78%
Profiles that have academic degree information	5230	40.2%
Profiles that have college major information	7825	60.1%

Table 2: Personal profile completeness

	Number	Percentage (of company profiles)
Company profiles that have industry type	11868	47.9%
Company profiles that have organisation type	11351	45.8%
Company profiles that have company size info	11343	45.8%

Table 3: Company profile completeness

In Table 2, the N in Equation 1 is the total number of public profiles, and F(i) is each field, i.e. work experience, education, skill, city, degree and major. It is observed that the percentage of profiles that have degree and major information are relatively low. It implies that people usually skip filling in these information into their profiles. The degree information is the lowest, that is because our Lucene text search engine does not accept any string that cannot be classified, in this case the normalized result is an empty string. Then our RDF converter just skip this triple. In order to improve this, we need to have more degree abbreviations and full names to cover every possible degree in universities worldwide, which can be a future work to investigate existing resources for this purpose.

Company profile completeness Not every company is registered in LinkedIn company pages in order to have a company profile. If a company in a person's work experience is registered in LinkedIn.com, there is a hyperlink that links the company name to the complete company profile. If the company is not registered, there will be not such hyperlink. So we can easily draw a conclusion from Table 3, around 46% of companies in Ireland is registered in LinkedIn.com. Please note that N in Equation 1 for Table 3 is the total number of companies in our dataset. F(i) is each field, i.e. industry type, organisation type and the size of the company.

5.1.2 Data Linkage

Another automatic quality assessment metric is data linkage of a dataset [14]. The definition of RDF data linkage is:

$$average\_linkage = \frac{total\_number\_of\_links}{total\_number\_of\_objects}.$$

It is a measurement of how "sparse" of the RDF data is. Generally, high linkage means high correlation between objects.

Total number of objects	160251
Total number of links	415916
Average linkage	2.595

Table 4: Data Linkage: Total and Ave rage

As we can see in Table 4, in average, every object has 2.5 number of links to other objects, which is reasonable good since all metadata is obtained from manually entered UGC.

5.2 Metadata Accuracy

One way to assess accuracy of metadata is to introduce volunteers to manually extract the data out from the HTML files so that we can compare the automatically extracted metadata with the manually provided metadata [16]. By calculating prediction precisions, recalls, f-measures, we can know how well our parser and our data normalization is.

Evaluation setup We recruited 10 users, divided them into 5 groups, so each group have 2 participants. Each group of users will view the same 10 randomly selected profiles. They were asked to manually extract city, work experiences (including company names, job titles, job start dates and job end dates) and education backgrounds (including college names, majors, degrees, college start dates, college end dates). To achieve this, we created an online user interface. Figure 3 shows a screenshot of the user interface that asking users to transfer data from a LinkedIn profile to our evaluation system.

Results After the user evaluation, we take user input as the ground truth. We use string matching to compare the manually entered metadata with the automatically extracted metadata. If the string matching return false, we manually examine the data and decide whether the extracted data is correct.

The metrics here we use are precision, recall and f-measure.

$$precision = \frac{correctly\_predicted}{predicted} \tag{2}$$

$$recall = \frac{correctly\_predicted}{total} \tag{3}$$

$$f - measure = \frac{2 * precision * recall}{precision + recall} \tag{4}$$

The meanings of these metrics can be explained as follows[25]: Precision, or confidence, measures how good we are predicting; Recall, or sensitivity is a measure of the proportion we correctly predicted over the total dataset. F-measure, or F-score is designed to capture both precision and recall. In order to obtain high F-score, precision and recall must be high.

Table 5 shows that we can extract city information reasonably well. Even some lazy volunteers did not fill in city information, we still achieve an average of 0.85 F-score. This particularly important, since city information is vital for the user interface design and for user experience.

We receive very high score in company name field according to Table 6. The reason for that is participants normally copy and paste the company name to fill in our survey forms and the exact matching is very high.

For the job title field in Table 7, the result is similar to the company name field. Users normally copy and paste the text without any term generalisation (e.g. change product manager to manager as it is more general). This is what our parser does as well. Thus the average score is very high with 0.93.

User	Precision	Recall	F-Measure
User 1	0.8889	0.8889	0.8889
User 2	0.7500	0.6667	0.7059
User 3	0.8571	0.6667	0.7500
User 4	0.8571	0.6667	0.7500
User 5	0.8889	0.8889	0.8889
User 6	0.8889	0.8889	0.8889
User 7	1	1	1
User 8	1	1	1
User 9	1	0.7778	0.8750
User 10	0.8750	0.7778	0.8235
Average	0.9006	0.8222	0.8571

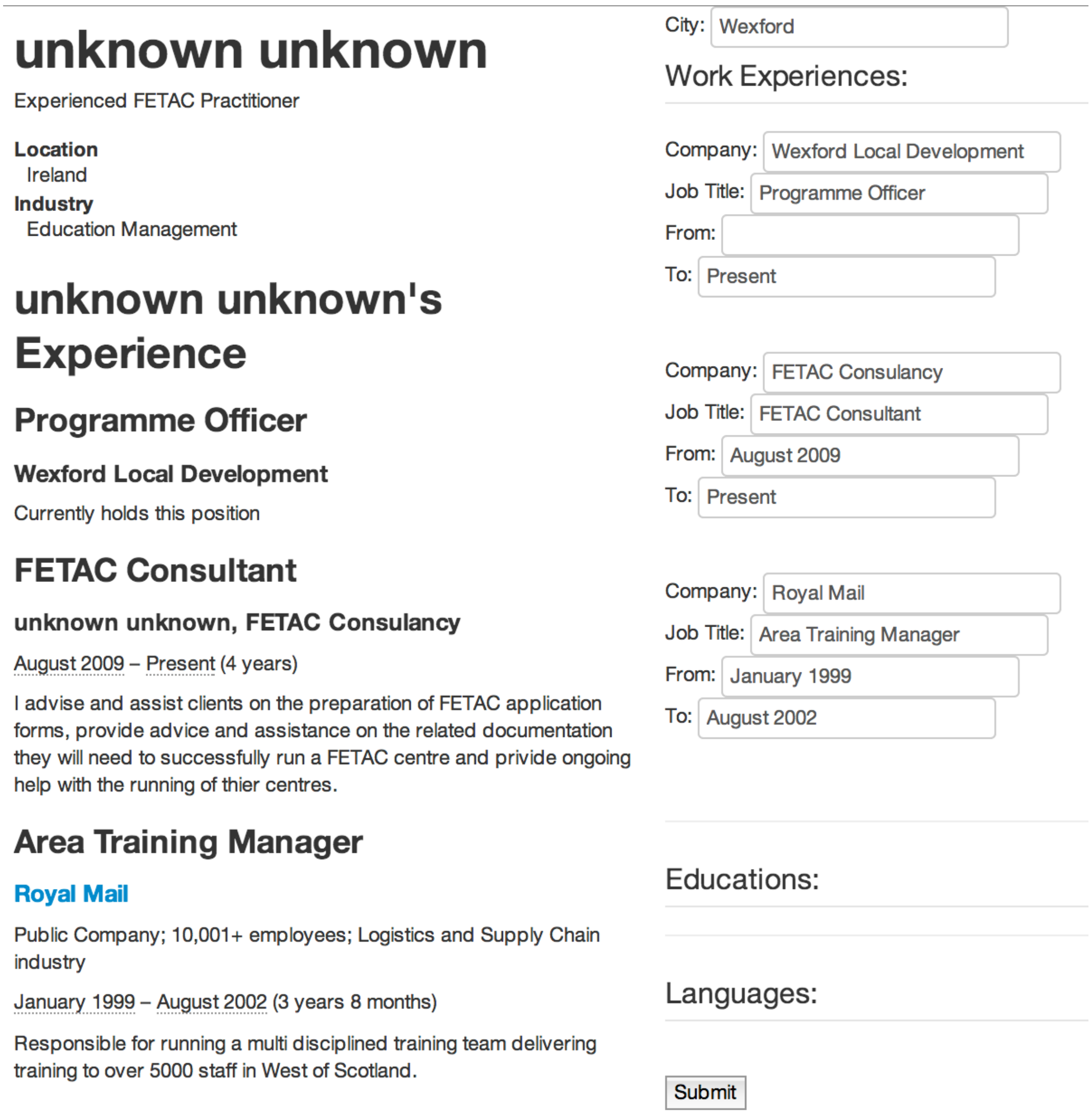
Table 5: Precision, recall and f-measure scores for city information

User	Precision	Recall	F-Measure
User 1	0.8947	0.8947	0.8947
User 2	0.9737	0.9737	0.9737
User 3	0.9600	0.9600	0.9600
User 4	0.8400	0.8400	0.8400
User 5	1	1	1
User 6	0.9556	0.9556	0.9556
User 7	1	1	1
User 8	1	1	1
User 9	0.9333	0.9333	0.9333
User 10	0.9556	0.9556	0.9556
Average	0.9513	0.9513	0.9513

Table 6: Precision, recall and f-measure scores for company information

User	Precision	Recall	F-Measure
User 1	0.8947	0.8947	0.8947
User 2	0.9474	0.9474	0.9474
User 3	0.9600	0.9600	0.9600
User 4	0.9200	0.9200	0.9200
User 5	0.9333	0.9333	0.9333
User 6	0.8667	0.8667	0.8667
User 7	0.9545	0.9545	0.9545
User 8	1	1	1
User 9	0.8889	0.8889	0.8889
User 10	0.9556	0.9556	0.9556
Average	0.9321	0.9321	0.9321

Table 7: Precision, recall and f-measure scores for job title information



unknown unknown

Experienced FETAC Practitioner

Location
Ireland

Industry
Education Management

unknown unknown's Experience

Programme Officer

Wexford Local Development

Currently holds this position

FETAC Consultant

unknown unknown, FETAC Consulancy

August 2009 – Present (4 years)

I advise and assist clients on the preparation of FETAC application forms, provide advice and assistance on the related documentation they will need to successfully run a FETAC centre and privide ongoing help with the running of thier centres.

Area Training Manager

Royal Mail

Public Company; 10,001+ employees; Logistics and Supply Chain industry

January 1999 – August 2002 (3 years 8 months)

Responsible for running a multi disciplined training team delivering training to over 5000 staff in West of Scotland.

City:	Wexford

Work Experiences:

Company:	Wexford Local Development
Job Title:	Programme Officer
From:	
To:	Present

Company:	FETAC Consulancy
Job Title:	FETAC Consultant
From:	August 2009
To:	Present

Company:	Royal Mail
Job Title:	Area Training Manager
From:	January 1999
To:	August 2002

Educations:

Languages:

Figure 3: Online user evaluation website: Users manually extract metadata from the randomly selected and anonymized LinkedIn profiles

Table 8 and Table 9 illustrate our prediction on start date and end date. The precision is high since LinkedIn always use same datetime pattern to represent the start date and the end date. The recall is low is because some profiles do not have the these fields.

The previous four tables (Table 6, Table 7, Table 8 and Table 9) illustrate that our parsed results for work experiences (company name, job title, job start date and job end date) are reasonably well with the average F-score greater than 0.9. We can summarise that the parser works well for these fields. The reason for such high results in both company name and job title is, we did not perform data normalization in these two fields. Basically, volunteers copy and paste these information to our survey form, and this is what our parser does as well. Since strings are fully matched, the scores are high.

We receive high score for college name field(Table 10). The explanation for high score is

User	Precision	Recall	F-Measure
User 1	1	0.8684	0.9296
User 2	1	0.8947	0.9444
User 3	1	0.8400	0.9130
User 4	1	0.8400	0.9130
User 5	1	0.8889	0.9412
User 6	1	0.8889	0.9412
User 7	1	0.7727	0.8718
User 8	1	0.7727	0.8718
User 9	1	0.9556	0.9773
User 10	1	0.9556	0.9773
Average	1	0.8678	0.9281

Table 8: Precision, recall and f-measure scores for experience start date information

User	Precision	Recall	F-Measure
User 1	1	0.8684	0.9296
User 2	1	0.8684	0.9296
User 3	1	0.8000	0.8889
User 4	1	0.8000	0.8889
User 5	1	0.9556	0.9773
User 6	1	0.9556	0.9773
User 7	1	0.8182	0.9000
User 8	1	0.7727	0.8718
User 9	1	0.9778	0.9888
User 10	1	0.9778	0.9888
Average	1	0.8794	0.9341

Table 9: Precision, recall and f-measure scores for experience end date information

User	Precision	Recall	F-Measure
User 1	1	1	1
User 2	0.9286	0.9286	0.9286
User 3	1	1	1
User 4	1	1	1
User 5	0.7778	0.7778	0.7778
User 6	0.9444	0.9444	0.9444
User 7	0.8462	0.8462	0.8462
User 8	0.9231	0.9231	0.9231
User 9	0.6538	0.6538	0.6538
User 10	0.9231	0.9231	0.9231
Average	0.8997	0.8997	0.8997

Table 10: Precision, recall and f-measure scores for college information

the same as company name field, users just copy and paste the college name from the profile. However, one difference is that we also use our data normalization tool to classify college name to our ground truth college name. Even with the data normalization, we receive high metadata accuracy for the college name. This shows that data normalization works reasonably well. Because in our implementation of college name normalisation, if we could not find any similar string, we create a new entry in our search engine database and assume it is a new college name.

User	Precision	Recall	F-Measure
User 1	0.8571	0.8571	0.8571
User 2	0.8571	0.8571	0.8571
User 3	0.5714	0.5000	0.5333
User 4	0.4286	0.3750	0.4000
User 5	0.8571	0.6667	0.7500
User 6	0.8571	0.6667	0.7500
User 7	0.6667	0.6154	0.6400
User 8	0.7500	0.6923	0.7200
User 9	0.7083	0.6538	0.6800
User 10	0.4000	0.2308	0.2927
Average	0.6954	0.6115	0.6480

Table 11: Precision, recall and f-measure scores for major information

According to Table 11, our metadata accuracy scores for major field are lower that other fields. If we look at the low scores and high scores carefully, we can see that they come in pairs. This means that the user input data is consistent, and in some groups of profiles, our data normalization fail to normalise degree information correctly. The reason for this is people do some data cleaning in their minds thus the major information they fill in is cleaned and well known. But our system did not perform any language processing, so the result of a simple copy and paste approach is different from the result that is generated by human mind.

User	Precision	Recall	F-Measure
User 1	1	1	1
User 2	1	1	1
User 3	1	0.8750	0.9333
User 4	1	0.7500	0.8571
User 5	0.9444	0.9444	0.9444
User 6	1	0.8889	0.9412
User 7	0.9167	0.8462	0.8800
User 8	0.8333	0.7692	0.8000
User 9	1	0.8846	0.9388
User 10	1	0.9615	0.9804
Average	0.9694	0.8920	0.9275

Table 12: Precision, recall and f-measure scores for degree information

We obtain very high data accuracy for degree information as shown in Table 12. This shows that we can properly clean degree information and classify it reasonably well. This is because of the extensive degree information we use from DBpedia for data normalisation. In particular, DBpedia provides rich data about degree information, which proves to be working well.

User	Precision	Recall	F-Measure
User 1	0	0	0
User 2	0	0	0
User 3	1	0.1250	0.2222
User 4	1	0.1250	0.2222
User 5	0	0	0
User 6	0	0	0
User 7	1	0.3077	0.4706
User 8	1	0.2308	0.3750
User 9	1	0.5000	0.6667
User 10	1	0.7308	0.8444
Average	0.6000	0.2019	0.2801

Table 13: Precision, recall and f-measure scores for education start date information

User	Precision	Recall	F-Measure
User 1	0	0	0
User 2	0	0	0
User 3	1	0.1250	0.2222
User 4	1	0.2500	0.4000
User 5	0	0	0
User 6	0	0	0
User 7	1	0.3077	0.4706
User 8	1	0.2308	0.3750
User 9	1	0.5000	0.6667
User 10	1	0.7308	0.8444
Average	0.6000	0.2144	0.2979

Table 14: Precision, recall and f-measure scores for education end date information

The data accuracy scores for college start date (Table 13) and college end date (Table 14) did not perform well. This evaluation helps us discover a problem in extracting these two fields. Since our system makes a wrong assumption about the format of the start date and end date(as 'yyyy-mm-dd'), we could not capture the fact that the start date and end date format in education background is 'yyyy'. This finding also prove that performing user studies is very important in system evaluation.

Discussion of Results In summary, Figure 4, summarizes the overall average precision, recall and f-measure for all fields. It can be seen that city, company names, job titles, work experience start and end date, college names and degree information have high average scores. In particular, results showed that our data normalization approach for degree and college information classification works well with an average f-measure scores of 0.92 and and 0.89 respectively. In addition, our mash-up based city information extrcation strategy performs well with an average f-measure score of 0.85. However, for major, we received an average f-score slightly bigger than 0.6, which means we cannot reasonably classify the major names. In the future, we will investigate to improve this such as we might use natural language processing and college subjects database to obtain better results. Finally, with the feedback, we need to fix the problem in extracting college start date and college end date, which involves updating the parser.

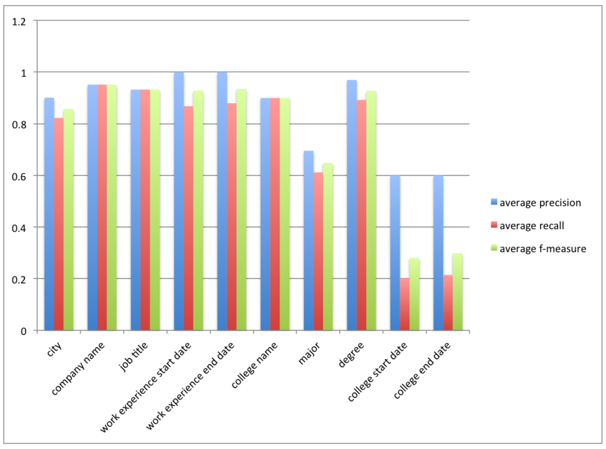


Figure 4: Summary: average precion, recall, f-measure for all fields

5.3 User Perceived Metadata Quality

Apart from metadata accuracy, we also collect user perceived metadata quality ratings for each field and users' overall rating for each profile. To achieve this, we created a user interface as shown in Figure 5. After, the user manually enters metadata fields about a profile, then we show the automatically extracted metadata along with the manually entered metadata. Subsequently, we ask users to rate the automatically extracted metadata quality for each entry. Ratings are from 1 to 5, where score 5 is the highest. To summarize, the idea of user generated rating is to measure the users' opinions about the metadata quality. Since sometimes users may not be happy with the normalised and cleaned metadata. In our opinion, user generated ratings is complementing with the precision and recall metrics.

In the following text, we only show the average rating for each field. From these ratings, we want to obtain consistent results about how well is our parser and data normalisation strategies.

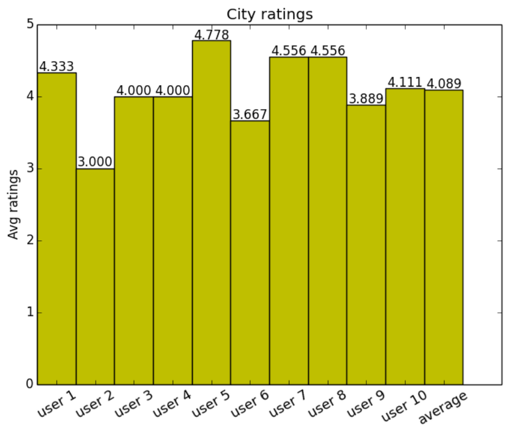


Figure 6: User average rating for city

Figure 6 shows that city information extraction strategy works reasonably well. However, in some cases, users' average ratings are lower. The reason for this is our city extraction strategy is setting the first returned city from all possible cities as the person's current living city. Sometimes profiles may contain very little information for us to guess where the person is actually located. Some users simply did not like our result when they see "A person work for Tesco Ireland is now living in Limerick" as they think there is too few information to predict the correct city.

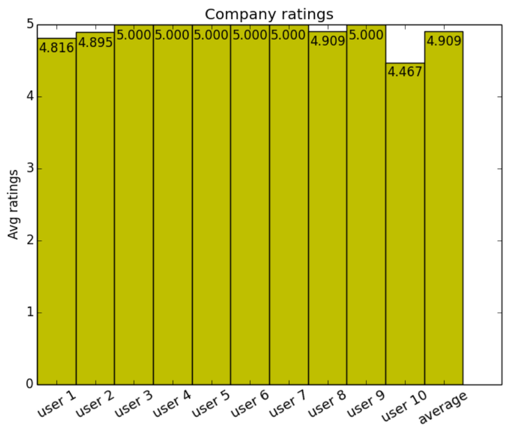


Figure 7: User average rating for company name

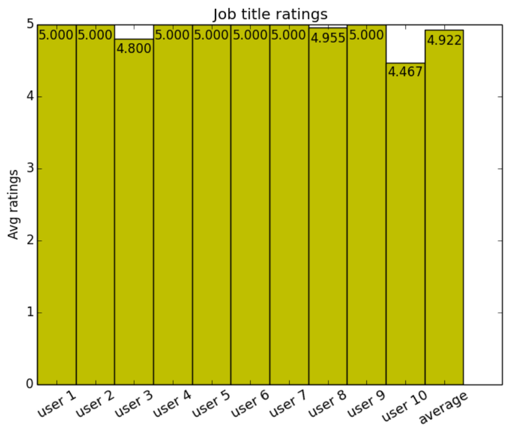


Figure 8: User average rating for job title

Figure 7 and Figure 8 presents user perceived metadata quality ratings for company name and job title, which shows that we obtain consistent results with Table 6 and Table 7. As we discussed earlier, a simple copy and paste approach is accepted by our participants.

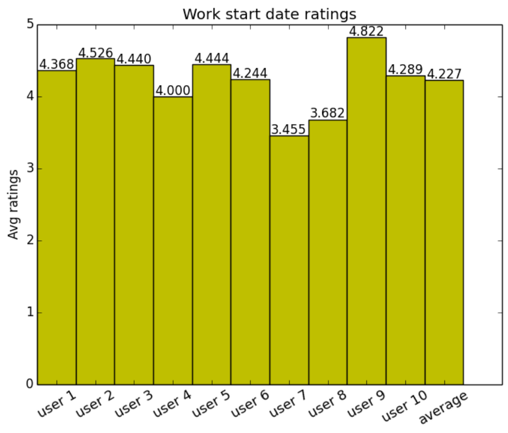
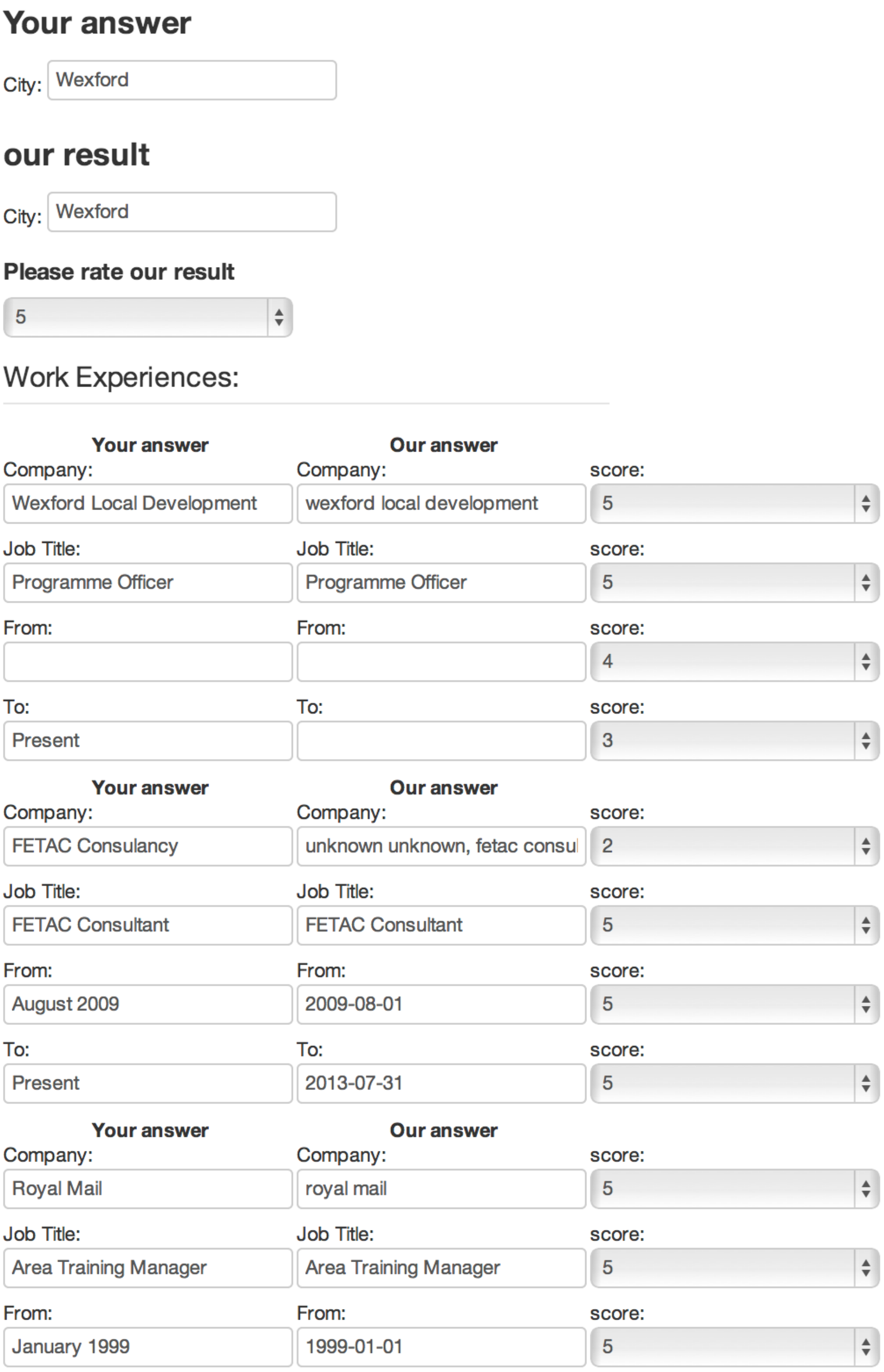


Figure 9: User average rating for experience start date

For the start date ratings (Figure 9) and end date ratings (Figure 10), we receive lower user satisfaction comparing to what we obtain at the previous section (Table 8 and Table 9). The reason for this is, some volunteers was not happy that we convert the datetime format from 'MM yyyy' to 'yyyy-mm-01'. What we are doing here is we assume all the start date and end date of a work experience is on the first day of the month. We explained the reason to volunteers during the evaluation, as we need this format to match the date literal definition in XML Schema[26].



Your answer

City: Wexford

our result

City: Wexford

Please rate our result

5

Work Experiences:

Your answer	Our answer	
Company: Wexford Local Development	Company: wexford local development	score: 5
Job Title: Programme Officer	Job Title: Programme Officer	score: 5
From:	From:	score: 4
To: Present	To:	score: 3

Your answer	Our answer	
Company: FETAC Consulancy	Company: unknown unknown, fetac consul	score: 2
Job Title: FETAC Consultant	Job Title: FETAC Consultant	score: 5
From: August 2009	From: 2009-08-01	score: 5
To: Present	To: 2013-07-31	score: 5

Your answer	Our answer	
Company: Royal Mail	Company: royal mail	score: 5
Job Title: Area Training Manager	Job Title: Area Training Manager	score: 5
From: January 1999	From: 1999-01-01	score: 5

Figure 5: Online user evaluation website: Asking users to compare the automatically extracted metadata with the manually entered metadata

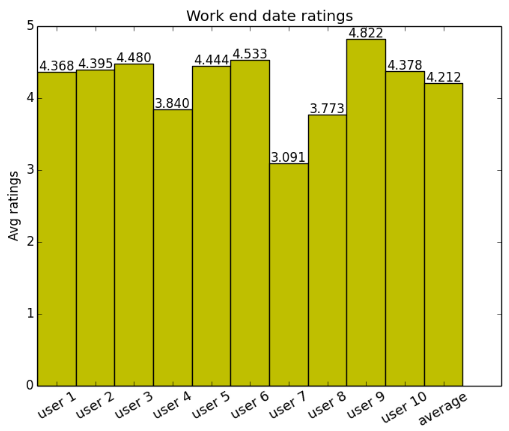


Figure 10: User average rating for experience end date

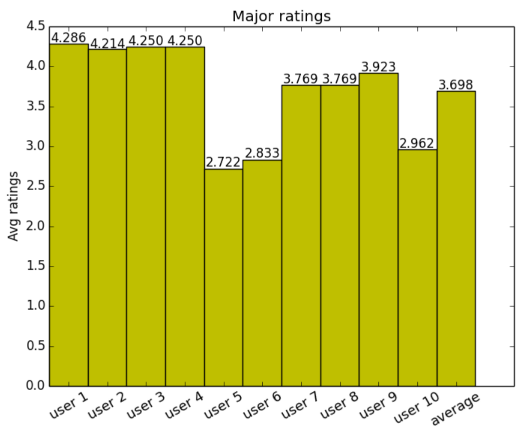


Figure 12: User average rating for major

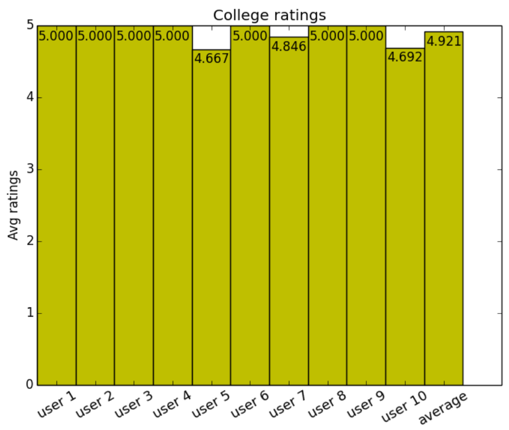


Figure 11: User average rating for college

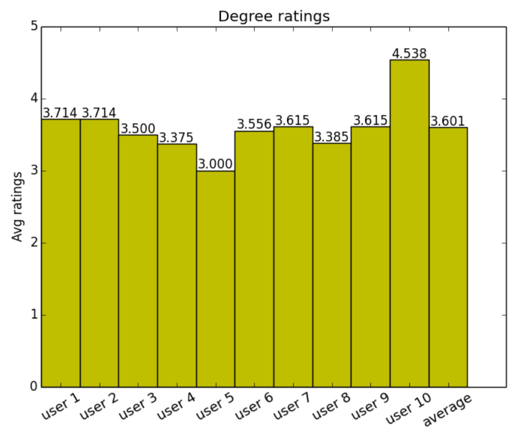


Figure 13: User average rating for degree

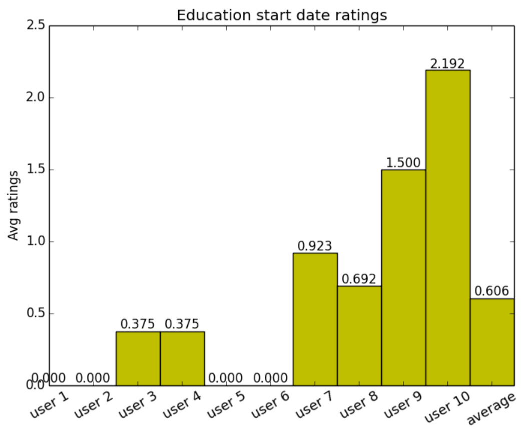


Figure 14: User average rating for education start date

Similar to Figure 7, our user perceived average metadata quality rating for college names (Figure 11) is widely accepted. It is a success, even we use our data normalisation module to classify college names. The reason is when we encounter a new unknown college name, we add it to our Lucene text search engine database instead of leaving the college name empty.

(Figure 12) shows user perceived metadata quality ratings for the major field. It can be seen that quality ratings varies for different users. In particular, every two users were viewing the same 10 randomly selected profiles. User 5 and User 6 specifically rated lower than the average since our parser did not perform well in those 10 profiles. Notice that the user ratings for major field is actually higher than the scores we obtain in Table11. This is may be because participants are happy to know that our system is not that intelligent as human mind and

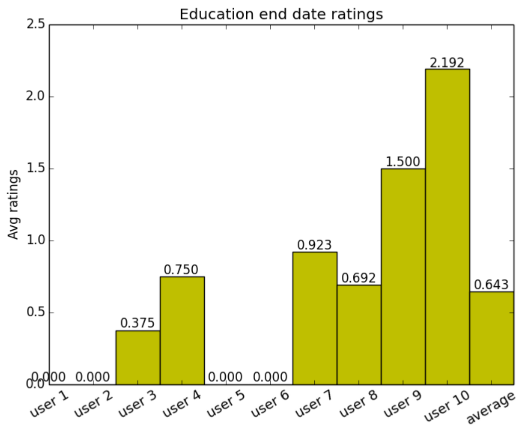


Figure 15: User average rating for education end date

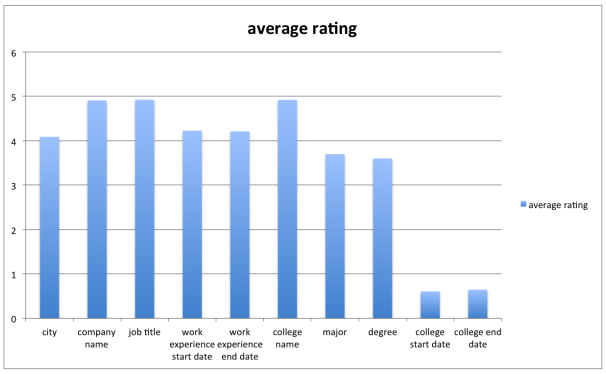


Figure 16: Summary: average user ratings for all fields

fail to clean the major field correctly. So participants rate their satisfaction scores high.

We receive lower user ratings for the degree field (Figure 13) comparing to Table 12. This is because the data completeness for degree field is low. For every profile that has no degree field, we set the score to 3, which is the average score by default.

As mentioned in the previous section, our parser failed to capture the fact that education start date and education end date is using 'yyyy' pattern in representing the datetime. Therefore, most of our volunteers just set the rating to 0 since there was no available information (Figure 14 and Figure 15). This problem can be simply fixed by changing the date format in parsing.

Discussion of Results In summary, Figure 16 shows average user ratings for all fields. As we can be seen, users are quite satisfied with our results in city, company name, job title, work experience start date and end date. In addition, users are reasonably satisfied with our result in major and degree fields, which can be further improved in future. Finally, users are not satisfied with the empty college start and end date, which can be fixed as we mentioned above.

5.4 Metadata Fitness

Metadata fitness measures how well our knowledge model and our dataset match the requirements of the upper layer user interface, which is a data visualisation interface that uses the metadata extracted by our work. By analysing how useful is the extracted metadata to support real life user interfaces, we can assess metadata fitness. Therefore, we collected feedbacks from the developer of the Data Visualisation project.

First, we briefly explain the data visualisation project. The project provides a search interface that accesses our dataset using the online SPARQL endpoint. The interface is designed to support three different scenarios based on various user personas, such as government officers, human resources personnel and job seekers. In Figure 17 system architecture and metadata requirements of the visualization interface is shown. In the following text, we summarize each scenario as follows:

Scenario 1, Government
The first scenario is that the interface should reflect the needs of Government officers. A government officer might be interested to search about employment statistics in a particular city or compare job statistics of various cities according to job title, industry type, academic degree or company size. Thus, the user interface requires the following metadata fields: City, industry type, academic degree and company size. Figure 18 shows a screen shot of the user interface that supports government officers.

Scenario 2, Company's human resource department
The user interface also supports daily activities of HR. For example, searching for people in a certain city with a particular skills set, work experience, degree or expertise. The user interface requires the following fields: City, degree, skill, work experience and start date.

Scenario 3, Job seekers and college students In this scenario, we assume users use the interface to search for jobs as well as learn about job trends in general. The interface requires the following metadata fields: City, degree, skill and position.

To satisfy the needs of the visualisation interface, we extracted city information using a novel strategy. In addition, we extracted industry type, company size, degree information, skills, work experience, start/end date and

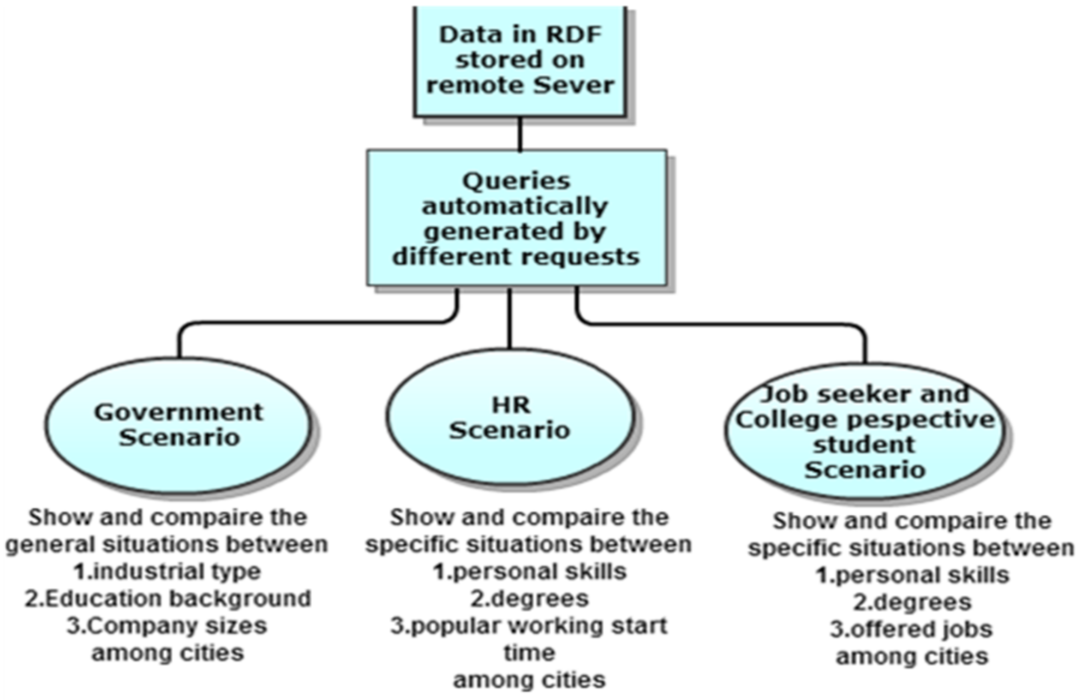

Data in RDF stored on remote Sever
Queries automatically generated by different requests
Government Scenario
HR Scenario
Job seeker and College pespective student Scenario
Show and compaire the general situations between 1.industrial type 2.Education background 3.Company sizes among cities
Show and compaire the specific situations between 1.personal skills 2.degrees 3.popular working start time among cities
Show and compaire the specific situations between 1.personal skills 2.degrees 3.offered jobs among cities

Figure 17: System architecture and metadata requirements of the data visualisation interface

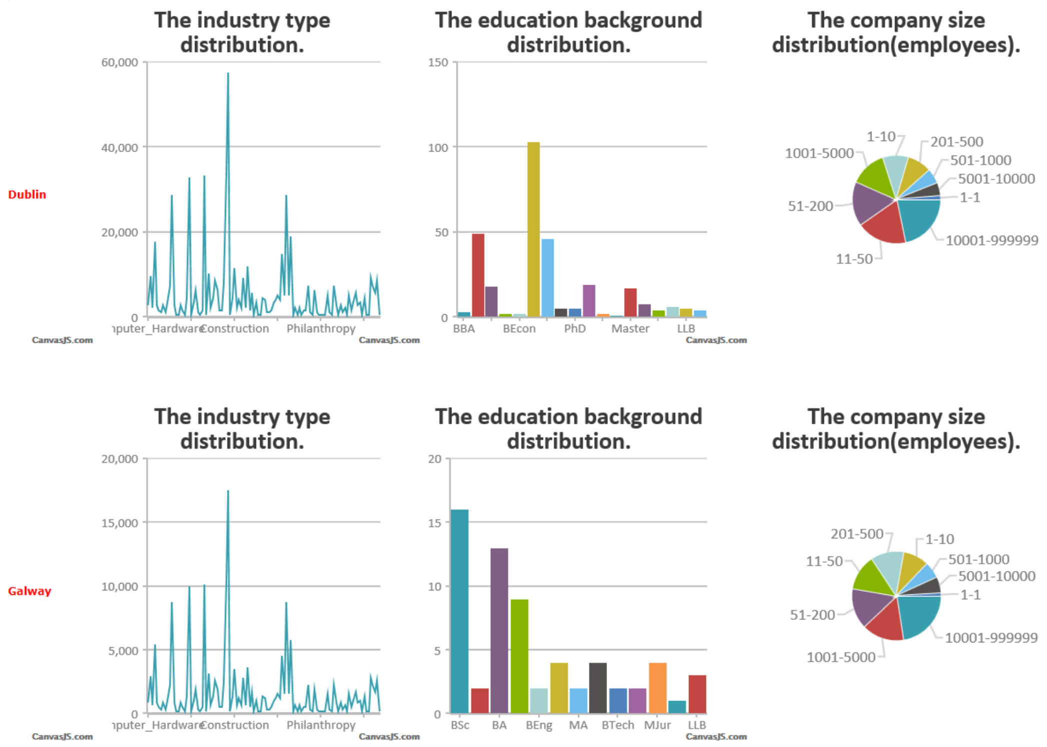


Figure 18: Screen shot of the visualisation interface; comparing industry types, education background and company sizes across cities

position information from the profiles. However, during the development and evaluation, we found some drawbacks of the extracted metadata that makes querying and development of the user interface difficult as we discuss below:

1. We do not have semantic information to group similar job titles together. This is important as we want to return more correct results as well as keep the query as simple as possible. For example, when one queries "software engineer" similar terms such as "application developer, Java software engineer" should be returned since these job titles have no significant difference between them if we want to compare jobs in IT over another industry. Performing such classification is difficult as we discuss in future work section in the next section.

2. Company names may have aliases. One example is "Oracle" and "Oracle EMEA". Since we cannot find company names database as ground truth, our system cannot handle this problem. This issue is very similar to the previous job titles problem since we need resources to accurately normalise data.

3. As SPARQL is very limit function in datetime manipulation, the start date and end date approach in our model is not working really well. For example, if one user is looking for "How many people had been working in a company for more then 5 years?". The query is very difficult to write so it would be easier if our model have "year between" field that address this requirement.

The details of possible solution of drawbacks will be discussed in the future work section.

5.5 System Performance

In System Environment section, we list our software and hardware details. Here we want to illustrate the performance of some critical modules, to provide more comprehensive details of the system. One important point to note is that the performance measurements do not include database accessing and file serialisation and other miscellaneous, therefore, in the production environment, the system performance could be worse.

5.5.1 Parsing performance

We run our parser 10 times, each time it parses 100 randomly selected profiles. The average time spending on parsing 100 profiles is: 18.27 seconds.

5.5.2 Normalising and converting performance

We run our RDF converter 10 times, each time in try to normalise the data in 100 profiles and convert it into RDF triples, the average time spending on this is: 345.53 seconds.

5.5.3 Query performance

We tried several queries with varying level of complexity in order to assess query performance. Notice that the complexity means the complexity of the query structure, it does not mean the complexity of the SPARQL engine in processing it.

A query that only returns subject, predicate and object:

```
select * where {?subject ?predicate ?
    object. }
```

Query 1: A query that return all subjects, predicates and objects

The time spend on this query is: 13.179s, and it returned 819488 rows.

A query that searches subject, predicate and object then summing up the subject:

```
select (count(?subject) as ?total)
    where {
        ?subject ?predicate ?object.
}
```

Query 2: A query that return all subjects, predicates, objects and count subjects

The time spend on this query is: 1.799s, and it returned 1 rows.

Notice that this query (Query 2) is 7 times faster that the previous query (Query 1) even though the complexity is higher. We do not know the implementation of 4store, but one possible explaination is that the COUNT method is optimised thus the program does not really need to read all rows to receive the actually result.

A query that defines two relationships:

```
select * where {
        ?person a foaf:Person;
        lk:skill ?skill.
}
```

Query 3: A query that defines two relationships

The time spend on this query is: 1.795s, and it returned 196187 rows.

It is an interesting result because receiving graph patterns (Query 3) is actually quicker than the first query, which only ask for all triples. One possible answer is the 4store SPARQL engine has special optimisation on graph matching.

A query that defines two relationships with group by and count:

```
select ?city (count(?person) as ?
   pCount) where {
      ?person a foaf:Person;
      dbpedia-owl:city ?city .
} group by ?city
```

Query 4: A query that defines two relationships and use group by and count

The time spend on this query is: 0.353s, and it returned 19 rows.

This time, data aggregation query (Query 4 is actually quicker than simple "print all" query (Query 1) and query with simple graph matching (Query 3). This result demonstrates 4store implementation of data aggregation has very high performance.

A query that defines two relationships with group by, and count and order by:

```
select ?skill (count(?skill) as ?
   sCount) where{
      ?p a foaf:Person;
      lk:skill ?skill.
} group by ?skill order by desc(?
   sCount)
```

Query 5: A query that defines two relationships and use group by, count and order by

The time spend on this query is: 2.204s, and it returned 16185 rows.

This query (Query 5) takes more than 2 seconds to execute. This is because to generate the correct result, the SPARQL engine has to split all the skill into groups, sum them up and finally sort the results.

The performance seems to be reasonable if we are not trying to receive all raw triples from the server. However, the performance can be even better if we have a better ware (our production environment is Amazon EC2 64 bit Ubuntu12.04, Intel Xeon Central processing unit E5-2650 2.00GHZ, 4G memory). Currently, we do not have enough budget to get a high performance CPU instance on Amazon EC2. Figure 19 shows a comparison of query performances between our production server and our development server. The parameters of our development machine are: MacBook Pro OSX 10.8 Mountain Lion, Intel core i7 2.7GHz CPU, 8GB 1600MHz DDR3 memory.

Notice that if we run our query on our development machine, the time spend on receiving all triples is around

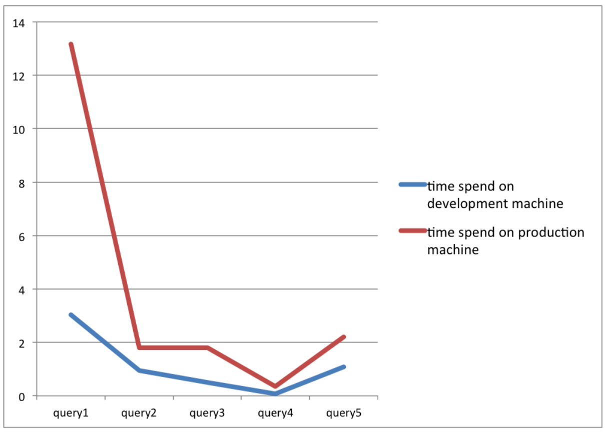


Figure 19: Query performance of all 5 queries on development machine and production machine

3 second, which is 4 times faster than our production server.

Discussion of Results: We can see that for complex group by, count and order by query, our production server takes about 2 seconds to run. One important factor that result in slow queries is the system hardware. Because we test the performance in production environment, the machine is an Amazon EC2 m1.medium instance, which is relatively low comparing to current standard server. We also illustrate the performance issue can be potentially alleviated by upgrading to a higher Central processing unit (CPU). With these query times, our server should be able to respond to public queries.

6 Lessons Learned

Search Engine and People Networks for Crawling: In the case when the content from a domain is not directly available, search engine queries can be used as a seed for crawling. Our experiments showed that utilizing a search engine approach to start crawling and using LinkedIn connected profiles to download more profiles works really well, since it can exponentially scale. This approach can be simply applied to information extraction approaches in other social networks such as Facebook. One drawback is that dealing with remote network requests take considerable time, since we process thousands of profiles. In addition, there is a possibility that the crawler program can be thought as a security treat and it is necessary to place appropriate delays between remote server requests.

Template-Based Information Extraction from Semi-Structured Content: While working on LinkedIn, we noticed that templates for public person profiles and company profiles have a very structured layout and they hardly change. Thus, writing template-based scripts for information extraction was straight forward. It is fair to say that in large semi-structured knowledge bases, corporate sites or wikis that have very structured layout, template-based information extraction can be easily applied. A similar template-based information extraction and RDF conversion is applied to Wikipedia pages, which resulted in DBpedia knowledge base [20].

Modular Design: One of the key design lessons that we have learned is separating downloader, parser, data normalisation and RDF conversion modules. This is essential since in each stage several unexpected problems may occur. Thus, by separating the modules, each problem can be handled and solved separately without affecting the other modules. Modular design is useful for other information extraction approaches since workflow of information can be more effectively controlled.

Keeping Status Track and Intermediate Results during Parsing and Information Extraction: We realize that this is a good engineering practice. Our experiences showed that an unhandled exception may occur and stop the parsing, which happened several times during the development. In this case, if the status is not known and intermediate results are not stored, and then we lose all the extracted knowledge and require re-starting the whole process again (which is not acceptable). This is particularly frustrating when dealing with thousands of profiles. However, when track record is kept and intermediate results are stored, we can simply re-start from the remaining profiles. Therefore, in similar approaches, we strongly suggest to design a data workflow and store the status.

Data Cleansing and Normalisation. Our experiences with working the visualisation project showed that data cleansing/normalisation is very important for the usability of the user interface. In particular, efficiency and usefulness of the extracted data decreases when it contains duplicated instances and when the data is not normalised. In future work, we plan to carry out more work in data normalisation, thus the quality of the metadata can be improved further.

Knowledge Modelling and Supporting the User Interface Requirements. During the development of the knowledge model, we worked in cooperation with the visualisation project to support the user interface design. We learned that thinking in mind the end-users of the data is very important, which affects the way the knowledge is modelled. In addition, knowledge modelling is an iterative process and involves several cycles and design, assessment and reflection.

7 Conclusions and Future Work

Semantic Web is one of the best technologies that allow structured knowledge to be shared and processed automatically. In this work, we took LinkedIn.com public profiles as a research subject and investigated the possibility of converting semi-structure profiles into a RDF knowledge graph using existing linked data vocabularies. As a result, we created a knowledge graph that can express the semantics behind the profiles and the relationship between the nodes. Our work and the upper layer user interface have proven that Semantic Web technologies can be used to extract useful semantics from LinkedIn.com. In addition, we published the extracted knowledge through a public SPARQL endpoint (http://128.199.243.88/test/) that allows everyone to query job statistics in LinkedIn Ireland. Furthermore, extensive evaluations showed that the extracted metadata is high quality.

In future work, we will investigate how company names and job titles can be classified. One possible solution is to re-use exiting ontologies or datasets as a ground truth. Fortunately, "European Skills/Competences, qualifications and Occupations" (ESCO) taxonomy [17] was recently released, which can be used to classify job titles as well as skills. In particular, ESCO categorises skills, competences, qualifications and occupations in a standard way. However, possible combinations of company names are infinite, thus it can be very challenging to find such a comprehensive dataset. Alternatively, we might look for a machine learning approach; first we can extract large amounts of metadata from the profiles, then create a game that ask volunteers to manually link the data with the same meaning. Subsequently, with the validated dataset, we can easily apply different machine learning algorithms for classification. Furthermore, it would be very interesting to work with LinkedIn data engineering team and integrate our work to LinkedIn.com.

References

[1] J. Krumm, N. Davies, and C. Narayanaswami, "User-generated content," *Pervasive Computing, IEEE*, vol. 7, no. 4, pp. 10–11, 2008.

[2] N. Shadbolt, W. Hall, and T. Berners-Lee, "The semantic web revisited," *Intelligent Systems, IEEE*, vol. 21, no. 3, pp. 96–101, Jan.-Feb.

[17] `http://www.cedefop.europa.eu/EN/news/22025.aspx`

[3] S. Harris, N. Lamb, and N. Shadbolt, "4store: The design and implementation of a clustered rdf store," in *5th International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS2009)*, 2009, pp. 94–109.

[4] E. Ferrara, P. De Meo, G. Fiumara, and R. Baumgartner, "Web data extraction, applications and techniques: A survey," *arXiv preprint arXiv:1207.0246*, 2012.

[5] A. Hemnani and S. Bressan, "Extracting information from semi-structured web documents," *Advances in Object-Oriented Information Systems*, pp. 389–396, 2002.

[6] H. Cunningham, "GATE, a General Architecture for Text Engineering," *Computers and the Humanities*, vol. 36, pp. 223–254, 2002.

[7] D. Ferrucci and A. Lally, "Uima: An architectural approach to unstructured information processing in the corporate research environment," *Nat. Lang. Eng.*, vol. 10, no. 3-4, pp. 327–348, Sep. 2004. [Online]. Available: http://dx.doi.org/10.1017/S1351324904003523

[8] M. T. Pazienza, A. Stellato, and A. Turbati, "Pearl: Projection of annotations rule language, a language for projecting (uima) annotations over rdf knowledge bases." in *LREC*, N. Calzolari, K. Choukri, T. Declerck, M. U. Dogan, B. Maegaard, J. Mariani, J. Odijk, and S. Piperidis, Eds. European Language Resources Association (ELRA), 2012, pp. 3828–3835. [Online]. Available: http://dblp.uni-trier.de/db/conf/lrec/lrec2012.html#PazienzaST12

[9] V. Damjanovic, T. Kurz, R. Westenthaler, W. Behrendt, A. Gruber, and S. Schaffert, "Semantic enhancement: Key to massive and heterogeneous data pools," in *ERK*, 2011.

[10] M. Fiorelli, M. T. Pazienza, S. Petruzza, O. Stellato, and A. Turbati, "Computer-aided ontology development: an integrated environment," in *New Challenges for NLP Frameworks held jointly with LREC2010*, 2010.

[11] P. Mika, "Ontologies are us: A unified model of social networks and semantics," *Web Semant.*, vol. 5, no. 1, pp. 5–15, Mar. 2007. [Online]. Available: http://dx.doi.org/10.1016/j.websem.2006.11.002

[12] S. Wang, Y. Zeng, and N. Zhong, "Ontology extraction and integration from semi-structured data," *Active Media Technology*, pp. 39–48, 2011.

[13] E. Hatcher, O. Gospodnetic, and M. McCandless, "Lucene in action," 2004.

[14] X. Ochoa and E. Duval, "Quality metrics for learning object metadata," in *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2006*, E. Pearson and P. Bohman, Eds. Chesapeake, VA: AACE, June 2006, pp. 1004–1011. [Online]. Available: http://www.editlib.org/p/23127

[15] M. Sah and V. Wade, "Automatic metadata mining from multilingual enterprise content," *Journal of Web Semantics*, vol. 11, pp. 41–62, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.websem.2011.11.001

[16] ——, "Automatic metadata extraction from multilingual enterprise content," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, ser. CIKM '10. New York, NY, USA: ACM, 2010, pp. 1665–1668. [Online]. Available: http://doi.acm.org/10.1145/1871437.1871699

[17] R. OBrian, "An overview of the methodological approach of action research." [Online]. Available: http://www.web.net/~robrien/papers/arfinal.html

[18] E. Simperl, "Reusing ontologies on the semantic web: A feasibility study," *Data & Knowledge Engineering*, vol. 68, no. 10, pp. 905 – 925, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0169023X0900007X

[19] A. Miles and J. R. Pérez-Agüera, "Skos: Simple knowledge organisation for the web," *Cataloging & Classification Quarterly*, vol. 43, no. 3-4, pp. 69–83, 2007.

[20] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," *The Semantic Web*, pp. 722–735, 2007.

[21] J. Golbeck and M. Rothstein, "Linking social networks on the web with foaf: A semantic web case study." in *AAAI*, vol. 8, 2008, pp. 1138–1143.

[22] M. McCandless, E. Hatcher, and O. Gospodnetic, *Lucene in Action, Second Edition: Covers Apache Lucene 3.0*. Greenwich, CT, USA: Manning Publications Co., 2010.

[23] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*, 2nd ed. Cambridge, MA, USA: MIT Press, 2001.

[24] Y. Li, Y. Shi, X. Fan, and M. Bhavsar, "Careergalaxy a planner for future," 2012.

[25] D. Powers, "Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

[26] P. Biron, A. Malhotra, W. W. W. Consortium *et al.*, "Xml schema part 2: Datatypes," *World Wide Web Consortium Recommendation REC-xmlschema-2-20041028*, 2004.

AUTHOR BIOGRAPHIES

Jinwu Li is a Software Engineer at Tencent, Inc. He has M.Sc. degree in Computer Science from Trinity College Dublin (2013) and B.Sc. degree in Computer Science from Dublin Institute of Technology (2012). Li's research interests include Semantic Web and Distributed Systems.

Prof. Vincent Wade is Director of the CNGL Centre for Global Intelligent Content and Head of the Discipline of Intelligent Systems at the School of Computer Science and Statistics, Trinity College Dublin (TCD). He is also Academic Director (Founder) of Learnovate Centre, TCD. Prof. Wade graduated from University College Dublin with a B.Sc. (Hons) in Computer Science (1987) and received his M.Sc. and PhD postgraduate degrees in Computer Science from TCD. He holds the position of Associate Professor in the School of Computer Science and Statistics and in 2002 was awarded Fellowship of Trinity College for his contribution to research in the areas of knowledge management and adaptive technologies. He was also awarded the position of Visiting Scientist in the Center for Advanced Studies at IBM for his research in adaptive hypermedia and knowledge management (2005-2008). Prof. Wade is author of over 150 scientific papers in peer-reviewed research journals and international conferences and has received eight best paper awards for publications in IEEE, IFIP and AACE Conferences within the last nine years. Prof. Wades research interests focus on Knowledge Engineering research, in particular adaptive web systems, dynamic personalisation, adaptive management and control systems, and process management.

Dr. Melike Sah is a research fellow in Centre for Global Intelligent Content (CNGL) at School of Computer Science and Statistics, Trinity College Dublin, Ireland. Between November 2010 and October 2012, she hold Irish Research Council postdoctoral fellowship award for her research on personalized search and exploration on the Linked Open Data. Sah has a Ph.D. degree in computer science from the University of Southampton (2009). She also has M.Sc. (2005) and B.Sc. (2003) degrees in computer engineering from Eastern Mediterranean University, North Cyprus. Her research interests include Semantic Web, personalization, user modeling, information retrieval, personalized search, data mining and fuzzy systems. She has published peer-reviewed papers in international conferences and journals in her field.