Combining Linked data and Data Mining Techniques to improve Clustering of Large Scale Media Repositories: A case study with BBC

Ross Fenning
British Broadcasting Corporation (BBC)
MediaCityUK
Salford, United Kingdom
ross.fenning@bbc.co.uk

ABSTRACT

Media companies produce ever larger numbers of articles, videos, podcasts, games, commonly collectively known as "content". A successful content-producing organisation not only has to develop systems to aid producing and publishing content, but there are also demands to engineer effective mechanisms to aid consumers in finding that content. Linked Data technologies provide data enrichment beyond attributes and keywords explicitly available within content data or metadata. In our work we experiment with a plausible mapping of an RDF graph representing a content item to an attribute set suitable for data mining. With such a mapping, we have explored the application of machine learning, particularly unsupervised learning, across an organisation's whole content corpus. We have created an innovative pipeline of linked data and data mining that allows clustering of media content items on the fly. We have evaluated such system in the context of website content produced by the British Broadcasting Corporation (BBC) and present the optimum combination of RDF graphs and data mining with qualitative and quantitative results.

1. INTRODUCTION

Media companies produce ever larger numbers of articles, videos, podcasts, games, etc. – commonly collectively known as "content". A successful content-producing website not only has to develop systems to aid producing and publishing that content, but there are also demands to engineer effective mechanisms to aid consumers in finding that content.

Approaches to aid discovery of content used in industry include providing a text-based search, hierarchical categorisation (and thus navigation thereof) and even more tailored recommended content based on past behaviour or content enjoyed by friends (or sometimes simply other consumers who share your preferences).

1.1 Problem

In order to build systems that operate across the full corpus of their content, organisations face several problems:

- Large organisations can have content across multiple content management systems, in differing formats and data models.
- Many content items are in fairly opaque formats, e.g. video content may be stored as audio-visual binary data with minimal, textual metadata to display on a containing web page.
- Content is being published continuously, which means any search or discovery system needs to keep up with content as it is published and process it into the appropriate data structures. Any analytical process that operates over all content (e.g. machine learning) may need to be run periodically or in an incremental fashion.

1.2 Hypothesis

The following hypotheses are proposed for gaining new insights about an organisation's diverse corpus of content:

- Research and software tools around the concept of *Linked Data* can aid us in rapidly acquiring a broad view (perhaps at the expense of depth) of an organisation's content whilst also providing a platform for simple enrichment of that content's metadata.
- We can establish at least a naïve mapping of an RDF graph representing a content item to an attribute set suitable for data mining. With such a mapping, we can explore applying machine learning across an organisation's whole content corpus.
- Linked Data and Semantic Web *ontologies* and models available can provide data enrichment beyond attributes and keywords explicitly avaiable within content data or metadata.
- Many content-producers currently enrich their web pages with small amounts of semantic metadata to provide better presentation of that content as it is shared on social media. This enables simple collection of a full breadth of content with significantly less effort than direct integration with content management systems (i.e. Enterprise Integration).

2. BACKGROUND

Data mining activities such as machine learning rely on structuring data as *feature sets*[**?**] – a set or vector of properties or attributes that describe a single entity. The process of *feature extraction* generates such feature sets from raw data and is a necessary early phase for many machine learning activities.

The RDF graph is a powerful model for metadata based on representing knowledge as a set of subject-predicate-object *triples*. The query language, SPARQL, gives us a way to query the RDF graph structure using a declarative pattern and return a set of all variable bindings that satisfy that pattern.

Notably, Kiefer, Bernstein and Locher[**?**] proposed a novel approach called SPARQL-ML – an extension to the SPARQL[**?**] query language with new keywords to facilitate both generating and applying models. This means that the system capable of parsing and running standard queries must also run machine learning algorithms.

Their work involved developing an extension to the SPARQL query engine for *Apache Jena*[1] that integrates with systems such as *Weka*[2]. A more suitable software application for enterprise use might focus solely on converting RDF graphs into a neutral data structure that can plug into arbitrary data mining algorithms.

An approach of generating attributes for a given resource was proposed by Paulheim and Fürnkranz[**?**]. They defined specific SPARQL queries and provided case study evidence for the effectiveness of each strategy.

Their work focused on starting with relational-style data (e.g. from a relational database) and using *Linked Open Data* to identify entities within literal values in those relations and generated attributes from SPARQL queries over those entities.

For a large content-producer, there is a more general problem where many content items do not have a relational representation and the content source is a body of text or even a raw HTML page. However, the feature generation from Paulheim and F urnkranz proves to be a promising strategy given we can acquire an RDF graph model for content items in the first place.

3. DESIGN

An experiemental system was designed and built around the following high-level requirements:

1. gathering (meta)data about all of an organisations content items;
2. further enriching that metadata with information not explicitly present in the content item itself; and
3. applying machine learning to that content metadata to gain new insights about that content.

The system was also designed with the end goal of being able to present users with alternative content items deemed similar to any given item they are currently ready or watching. This provided a focus to ensure some industrial application of the experiment and also a means by which to evaluate the results in that results can be scored on how they perform at driving such a website feature.

Figure 1 shows a high-level view of a "Content Miner" software application. A set of "notifier" applications can be created to connect different content production and management systems to the data mining system such that it is notified when new content is created. The notification need only be an *IRI* that uniquely identifies that content item.

[1]https://jena.apache.org/

[2]http://www.cs.waikato.ac.nz/ml/weka/

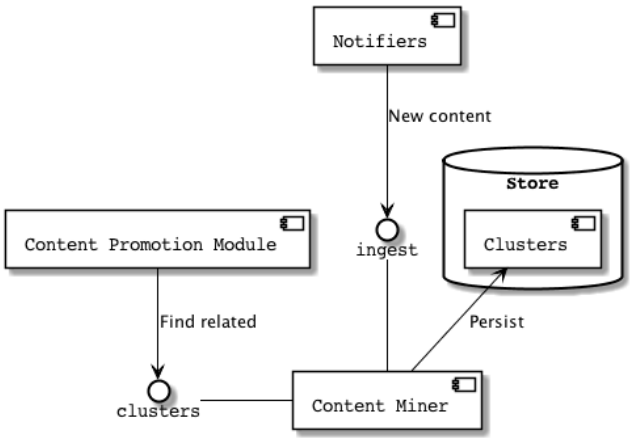

Figure 1: High-level component diagram with interfaces for each use case

Once the application has meaningful clusters of content, we can imagine another system that can query this miner application to provide some view of related content given some initial content item.

3.1 Data Pipeline

The overall content miner system conflates the two jobs of generating feature sets suitable for data mining and also the data mining process itself. Thus a core subsystem in the overall system is a conceptual data pipeline whose input is a URI or IRI identifying a content item published on an organisation's website and the output is feature sets ready for applying machine learning.

This subsystem was conceptually modelled as a data "pipeline" that takes an IRI identifying a piece of web content at the source end and produces feature sets at the other end. In this pipeline, the three primary data structures employed are the *IRI*[3] (in many cases the public URL for the content's primary page is sufficient), the *feature set* (modelled as a schema-less set of key-value pairs) and the *RDF graph* as an intermediate structure for manipulating along the pipeline.

The data pipeline was then designed inductively, introducing potential methods for creating RDF graphs from web content and also enriching those graphs.

```
{
  "@id": "http://example.com/entity/1",
  "@graph": []
}
```

Listing 1 : Identity graph for a content item in JSON-LD sy

With the knowledge only of a content item's IRI, we are arguably only able to produce an empty named RDF graph. Such a graph for an example IRI `http://example.com/entity/` is illustrated in JSON-LD syntax in Listings 1. This provides our base case in the inductive design.

Paulheim and Fürnkranz[**?**] described a number of SPARQL queries for generating feature sets from RDF data, inspired a simple query such as that shown in Listings 2. This query generates a boolean `true` value for any properties that

[3]http://tools.ietf.org/html/rfc3987

match and implies `false` for those that do not and we then flatten this into a simple feature format: `?p_?o=true`, e.g. `rdf:type_schema:Article=true`

```
SELECT ?p ?o
WHERE { ?iri ?p ?o . }
FILTER isIRI(?o)
```

Listing 2 : Generates field `content_?p_?v` with value `true`

Note that this ignores triples whose objects are literals, for which case the a analogous strategy as shown in listings **??** was employed to produce features of the form: `?p=?v`.

```
SELECT ?p ?v
WHERE { ?iri ?p ?v . }
FILTER isLiteral(?o)
```

Listing 3 : Generates field `content_?p_?v` with value `true`

Finally, an additional layer of indirection was created

4. IMPLEMENTATION

5. ANALYSIS AND EVALAUTION

6. EVALUATION

7. CONCLUSIONS