Deliverable D2.6 LinkedTV Framework for Generating Video Enrichments with Annotations

Tomáš Kliegr, Jan Bouchner, Barbora Červenková, Milan Dojchinovski, Jaroslav Kuchař / UEP
José Luis Redonda Garcia, Raphaël Troncy / EURECOM
Jan Thomsen / CONDAT
Dorothea Tsatsou, Georgios Lazaridis, Pantelis Ieronimakis, Vasileios Mezaris / CERTH

17/10/2014

Work Package 2: Linking hypervideo to Web content

LinkedTV

Television Linked To The Web

Integrated Project (IP)

FP7-ICT-2011-7. Information and Communication Technologies

Grant Agreement Number 287911

Dissemination level	PU
Contractual date of delivery	30/09/2014
Actual date of delivery	17/10/2014
Deliverable number	D2.6
Deliverable name	LinkedTV Framework for Generating Video Enrichments with Annotations
File	`D2.6.tex`
Nature	Report
Status & version	Released & v1.0
Number of pages	46
WP contributing to the deliverable	2
Task responsible	UEP
Other contributors	EURECOM, CONDAT, CERTH
Author(s)	Tomáš Kliegr, Jan Bouchner, Barbora Červenková, Milan Dojchinovski, Jaroslav Kuchař / UEP José Luis Redonda Garcia, Raphaël Troncy / EURECOM Jan Thomsen / CONDAT Dorothea Tsatsou, Georgios Lazaridis, Pantelis Ieronimakis, Vasileios Mezaris / CERTH
Reviewer	Michiel Hildebrand / CWI
EC Project Officer	Thomas Kuepper
Keywords	Web mining, Video Enrichment, Linked Media, Crawling, Information Retrieval

Abstract (for dissemination)	This deliverable describes the final LinkedTV framework that provides a set of possible enrichment resources for seed video content using techniques such as text and web mining, information extraction and information retrieval technologies. The enrichment content is obtained from four type of sources: a) by crawling and indexing web sites described in a white list specified by the content partners, b) by querying the API or SPARQL endpoint of the Europeana digital library network which is publicly exposed, c) by querying multiple social networking APIs, d) by hyperlinking to other parts of TV programs within the same collection using a Solr index. This deliverable also describes an additional content annotation functionality, namely labelling enrichment (as well as seed) content with thematic topics, as well as the process of exposing content annotations to this module and to the filtering services of LinkedTV's personalization workflow. We illustrate the enrichment workflow for the two main scenarios of LinkedTV which have lead to the development of the LinkedCulture and LinkedNews applications, which respectively use the TVEnricher and TVNewsEnricher enrichment services. The original title of this deliverable from the DoW was *Advanced concept labelling by complementary Web mining*.

History

Table 1: History of the document

Date	Version	Name	Comment
19/06/2014	v0.1	Kliegr, UEP	Deliverable structure
23/07/2014	v0.2	Troncy, EURECOM	Update TOC structure
1/08/2014	v0.3	Tsatsou, CERTH	CERTH first input, chapter 5 title change
4/08/2014	v0.4	Kliegr, UEP	Chapter 3 - first input
7/08/2014	v0.5	Kliegr, UEP	Introduction, MES
16/08/2014	v0.6	Kliegr, UEP	Scenario example in MES, dashboard
22/08/2014	v0.6.1	Kliegr, UEP	THD plugin subsection
26/08/2014	v0.6.2	Kliegr, UEP	MES section extension
26/08/2014	v0.7	Thomsen, Condat	Chapter 2
12/09/2014	v0.8	Kliegr, UEP	IRAPI Evaluation, Summary, Abstract
15/09/2014	v0.81	Tsatsou, CERTH	Chapter 5
15/09/2014	v0.9	Redondo, EURECOM	Chapter 4
05/10/2014	v0.91	Troncy, EURECOM	Ready for QA
08/10/2014	v0.92	Troncy, EURECOM	Fix all typos found during QA
08/10/2014	v0.93	Kliegr, UEP	Intro & Section 3 revision based on QA
08/10/2014	v0.94	Troncy, EURECOM	Section 4 revision based on QA
08/10/2014	v0.95	Tsatsou, CERTH	Section 5 & Summary revision based on QA
17/10/2014	v0.96	Troncy, EURECOM	Final version addressing Lyndon's comments
17/10/2014	v1.0	Kliegr, UEP	Final editing changes

0 Table of Content

1 Introduction

This deliverable provides a review of the progress made on the web mining components developed within the LinkedTV consortium to identify related multimedia content and web documents that could be used for enriching seed videos. The suggested enrichment content can be further curated in the LinkedTV editor tool [6]. Furthermore, it is provided to WP4 which aims to apply a personalization layer on top of these suggestions, and to WP3 which will practically display the additional information in the rich hypervideo LinkedTV player [7].

1.1 Scenario examples

In this deliverable, we will illustrate the output of each enrichment component using examples from the RBB News Show broadcasted on April 4th, 2013[1] and from the Tussen Kunst and Kitsch episode 640 entitled "Museum Martena"[2] [9].

The RBB news show contains multiple news items. The one used for the running example is a news item covering the start of the asparagus[3] season in Germany. The complete news show program lasts nearly 30 min. It contains three news story highlights: "Explanation of the Brandenburg's finance minister", "Search for the first asparagus", and "Star placing". The asparagus headline news item starts at 8:39, with the news presenter quickly introducing the topic. A short report from the field follows. After only 25 seconds, the story is interrupted with a news story covering a discovery of a bomb in Oranienburg. The broadcast returns to asparagus after the breaking news finishes one minute later, at 10:17. The main part of the story is presented, taking two minutes and thirty seconds. The reporter develops the story, which is illustrated by a sequence of shots from an asparagus field and processing factory. The story is finished by an interview with a asparagus grower, with a short afterword of the reporter.

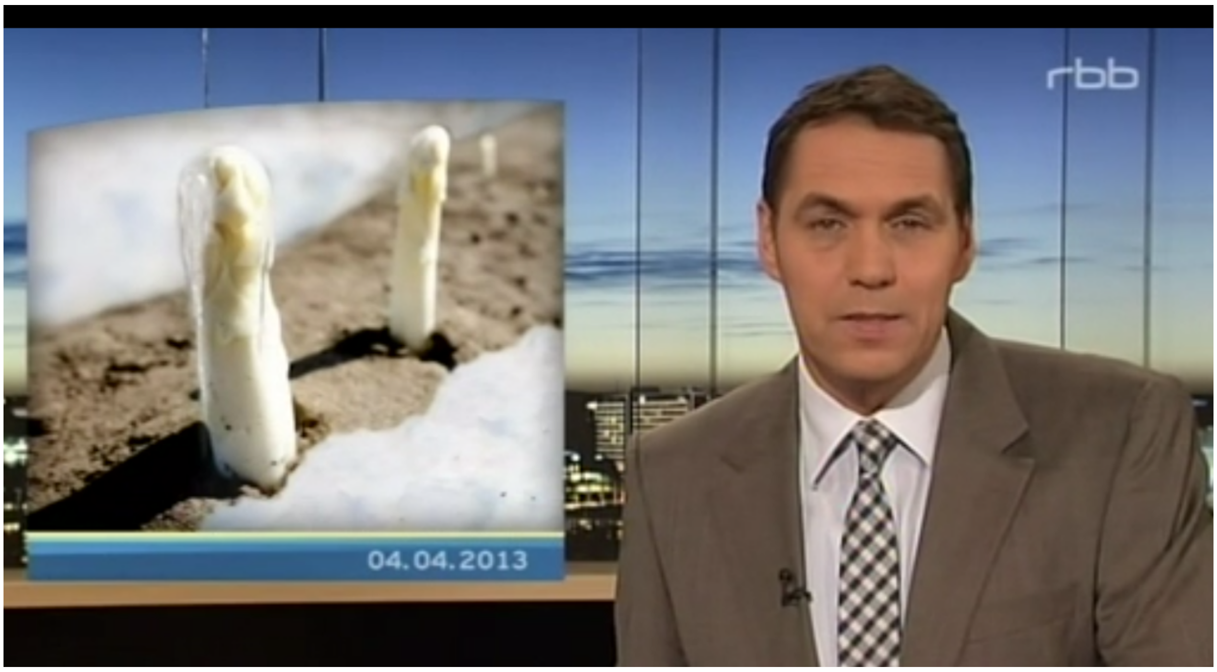


Figure 1: Screenshot of a news item on asparagus season from RBB News Show from April 4th, 2013.

Technically, the story is presented in two video fragments, the first lasting from 8:39 to 9:05, and the second from 10:17 to 12:50. A screenshot from the start of the second fragment of the Spargel news show is given on Figure 1. The RBB content partner also provided a subtitle file for this news show. There are in total 54 subtitles blocks associated with these video fragments. According to analysis performed

[1] http://data.linkedtv.eu/media/cbff5a80-a5b6-11e2-951c-f8bdfd0abfbd

[2] http://data.linkedtv.eu/media/8a8187f2-3fc8-cb54-0140-7dccd76f0001

[3] Spargel in German

by the content providers in WP6, the main query terms that could be used to retrieve enrichment content always include “Spargel”, and can be multi-keyword expressions such as: “Spargel and Brandenburg and Kälte”, “ Spargel and Brandenburg”, “Spargel and Kälte”, “Spargel and Ernte”, “Spargel and Saison”, “ Spargel and Frühling” [9].

The Tussen Kunst and Kitsch episode 640 “Museum Martena” contains seven chapters: Introduction, Horse painting, Silver tea jar, Bronze horse statue, Eisenloeffel clock, Jan Blom painting and 19th century broche. Within this deliverable, the episodes *Silver tea jar* and *Horse painting* are used for evaluation. For example, the synopsis of the silver tea jar chapter (00:35:58 – 00:39:44) as provided by the Netherlands Institute for Sound and Vision is as follows (Figure 2):

Silver tea jar is from the 18th century, from the Dutch province of Friesland. Tea was very expensive in those days, even more than the silver itself. The master sign of the silversmith on the tea jar belongs to an unnamed man who worked in the Fries city of Sneek after 1752. The silver tea jar is worth 8000 €.

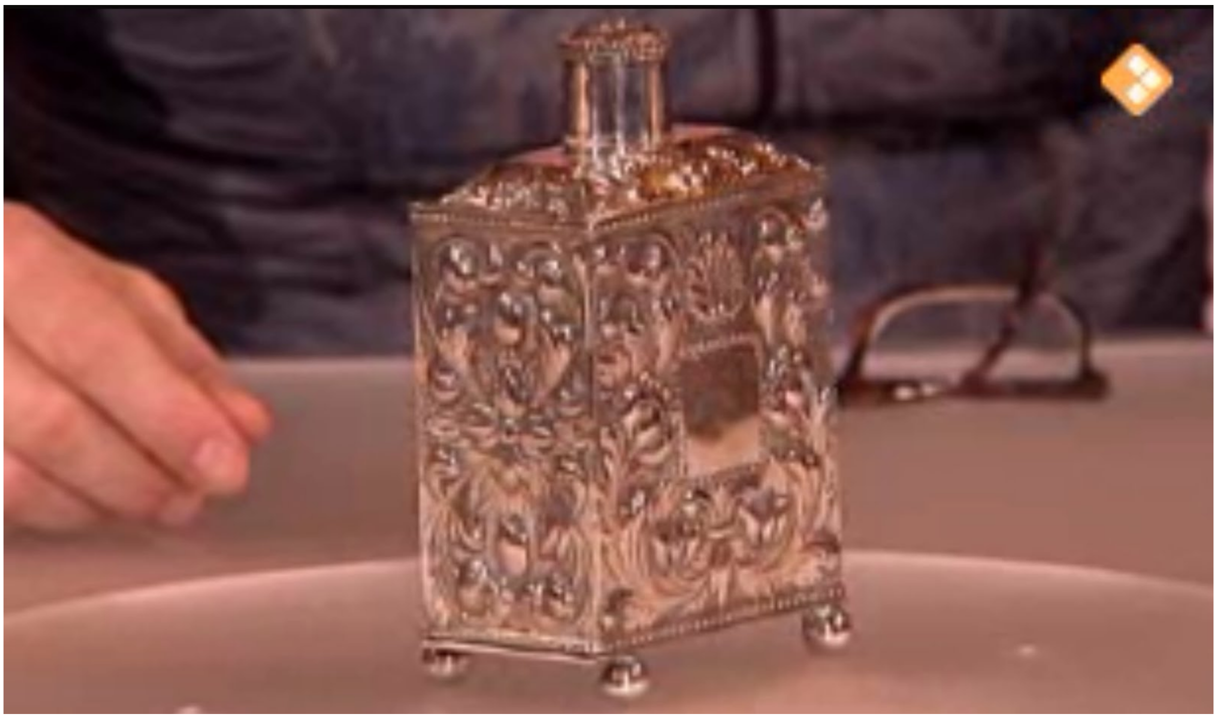


Figure 2: Screenshot of the silver tea jar object from the Museum Martena episode of Tussen Kunst and Kitsch.

According to analysis performed by the content providers in WP6, the main query terms that could be used to retrieve enrichment content are multi-keyword expressions such as: “Nelleke van der Krogt”, “Hessel van Martena”, “Museum Martena”, “fries zilver”, “friesland”, “thee voc” and “zilversmid friesland” [9].

1.2 Deliverable Structure

Following the architecture provided in the Deliverable D2.3 [2], enrichment content is retrieved in two fundamental ways: a) by crawling and indexing web documents and b) by querying public APIs and endpoints. Within the scope of this deliverable, we also describe how the enriched content is semantically annotated with entities, and how we use entity expansion to get the broader context of a story in order to propose more accurate enrichment. We first described the overall workflow and the connections to the other work packages via the LinkedTV platform interface in Chapter 2.

The enrichment tools developed and reported in this deliverable feature a range of technologies and algorithms that fall under the web mining discipline as defined in [27]. The **IRAPI Module** presented in the Chapter 3 is a web mining component, which features web crawling, information retrieval and web search algorithms. The system is based on the leading open source frameworks Apache Nutch and Solr, which have been extended with extractors for several types of multimedia content. The IRAPI module has been first introduced in D2.5 [4] under the name *Unstructured Search Module*. In this deliverable, we focus on specific enhancements. In particular, in addition to providing custom-built wrappers for several focus web sites to support scenario realization, we have developed an experimental pseudo-probabilistic extraction model called *Metadata Extraction Service*, which combines multiple extraction evidence to identify the media objects of interest and their metadata.

The LinkedTV enrichment framework described in Chapter 4 exposes two main services:

– **TVEnricher** developed for the LinkedCulture application, that makes use of the IRAPI, MediaCollector, Europeana and TV2Lucene modules;

– **TVNewsEnricher** developed for the LinkedNews application, that makes use of the EntityExpansion service, the IRAPI and TV2Lucene module, and Google CSE technologies.

Those services take the route of retrieval of enrichment content through querying of some selected APIs. It also aggregates the output of individual services, including the IRAPI Module, to a common format. For the LinkedTV application, it is not sufficient to extract the enrichment content from a specific site or even a list of sites. The scenarios require to propose enrichments from a broader variety of sources including social networks, media web sites and encyclopedia.

The approach taken in WP2 builds upon the proven web mining techniques such as web crawling, indexing, wrapper design, and information integration, and further extends them by providing the semantic annotation of the enrichment content. The multimedia objects and web pages retrieved are further annotated with multiple entities, i.e. Linked Open Data concepts, which help to disambiguate the associated semantics to WP4 services. These entities come from existing knowledge bases, the largest of which are DBpedia and YAGO. However, since the coverage of German and Dutch DBpedia version is limited and the corresponding YAGO version is non existent, which would hamper the annotation with entities of the LinkedTV scenario content, we have developed our own entity-type knowledge base for the two scenario languages (in addition to English). The **Linked Hypernyms Dataset** (LHD) contains nearly 5 million entity type assignments that were generated from the respective Wikipedias using a text mining algorithm. LHD dataset substantially complements the entity-type assignments available in the German and Dutch DBpedias. This dataset and the way it is used to annotate enrichment content is covered in the Section 3.3.

These annotations (as well as annotations of the original seed TV content as described in the Deliverable D2.3, chapter 3 [2]) are exposed into the **LUMO ontology** as described in Chapter 5 for two purposes: a) to facilitate use of the semantic description of the content by WP4 services within its homogeneous and lightweight concept space (the LUMO ontology), thus bridge WP2 and WP4 services and b) to provide further content annotations with topics from the LUMO ontology, as also described in Chapter 5, and thus enable semantic thematic categorization of content.

The types coming from the LHD dataset are used to generate annotations in the **Targeted Hypernym Discovery** tool, and are subsequently syndicated with type annotations assigned by other entity classifiers in the **NERD** framework. Together, TVEnricher/TVNewsEnricher and IRAPI provide a substantial input to the LinkedTV enrichment pipeline by identifying relevant web resources (webpages, images, videos, podcasts) on a large variety of web sites and social networking services that are potentially relevant for the *seed content* (the video item being enriched).

The entity enrichment as constituted by the NERD framework, and its LinkedTV developed THD service and **LHD** datasets, are used to semantically describe the seed as well as enrichment content. The annotations over the seed content descriptions are used already in WP2 to construct queries for the enrichment content. Finally, the semantically annotated seed and enrichment content is saved to the platform for subsequent use in personalization and by the LinkedTV player.

The services developed by WP2 can be called by the LinkedTV Platform, or by specific clients that are able to consume metadata modeled using the LinkedTV core ontology such as the LinkedTV Editor Tool [6] or the LinkedTV player [7].

2 Enrichment Workflow and Platform Integration

The enrichment step is the final step of the metadata aggregation process for LinkedTV media resources which includes the preceding steps of *video analysis*, *serialization*, *entity recognition* and *annotation* [8]. In the context of the enrichment step, it is important to note that this step relies on the results of the preceding steps. The more data sources related to a media resource (such as subtitle files, TV Anytime metadata files or results of the video analysis) have been processed, the more data is available which can be enriched. What exactly these enrichments consist of, from which data they are derived and how these enrichments themselves work will be described in the following chapters. This chapter explains how the enrichment process is integrated into the LinkedTV Platform workflow and interfaces. Basically, this consists of three main parts: a) adding enrichments to the repository, b) triggering subsequent actions if defined and c) making the enrichments available for client applications via the Platform REST API. These three parts will be described in the following sections in more detail.

2.1 Triggering enrichment and storing the results

Although the enrichment process can be based on single annotation types, within the LinkedTV workflow, this step is generally triggered on the basis of the processing of the complete media resource. The precondition for triggering the enrichment process is that the media resource is in state ANNOTATED, i.e. the results of the serialization and entity recognition process are stored as media fragment annotations within the RDF repository[4]. Technically, the enrichments are also just new annotations which are attached to a different motivation (`oa.motivatedBy=Linking`)[5].

The enrichment process is triggered by calling the service named *TVEnricher* (Chapter 4). The TVEnricher service is hosted by LinkedTV partner EURECOM at `http://linkedtv.eurecom.fr/tvenricher`. It serves as a single service endpoint but, in itself, covers different sub-services which are not called by the Platform directly, following the cascading integration approach described in the Deliverable D5.4 [5]. Although the necessary precondition for the enrichment process is the state ANNOTATED of the respective media resource, reaching that state is not the only event on which the enrichment process can be called. In general, we can distinguish the following three enrichment strategies:

1. *onAnnotated*: the enrichment is triggered directly when the media resource is in state ANNOTATED
2. *onScheduledUpdate*: the enrichment is triggered by scheduling it, e.g. on a daily or weekly basis; since enrichment results change very quickly over the time, a scheduled update mechanism ensures that the preprocessed enrichments include the most recent ones. The *onScheduledUpdate* strategy should include two sub-strategies *add* and *replace*, where *add* just adds new enrichments and *replace* always discards previous ones and only stores the most recent ones.
3. *onDemand*: the enrichment process is triggered on demand by a LinkedTV client application directly at playout time, or shortly before playout time with caching and optional further processing, e.g. for personalization or categorization. This of course ensures that the enrichments consist of the most up-to-date results. Within the onDemand strategy it does of course not make sense to store the results in the repository.

Currently, the Platform itself supports the *onAnnotated* and the *onScheduledUpdate_Add* strategies while the *onScheduledUpdate_Replace* and the *onDemand* strategies will also be supported by the Platform in future releases.

2.2 Integration into the LinkedTV Service Interface

For the integration of LinkedTV services such as analysis, annotation or personalization, the LinkedTV Platform exposes a special Service Integration Layer with a dedicated RESTful Service Interface[6]. Basically, the service interface provides proxy services to the different distributed LinkedTV services together with all checks and actions necessary for the processing within the LinkedTV workflow. The LinkedTV Service Interface for the enrichment service is depicted in the Table 2.

[4]Six steps are currently defined in the LinkedTV workflow, namely: IMPORTED, TRANSFERRED, ANALYSED, ANNOTATED, ENRICHED and CURATED.

[5]For a complete description of the nature of enrichments, see the Deliverable D2.4 [3].

[6]See the Deliverable D5.6 [8]

Table 2: LinkedTV Service Interface for the enrichment service

URI	`http://services.linkedtv.eu/mediaresource/:uuid/enrichment`
Checks (Preconditions)	`http://api.linkedtv.eu//mediaresource/:uuid` EXISTS `http://api.linkedtv.eu/mediaresource/:uuid/state` >= ANNOTATED
Calls	(service specific parameters apply)
Parameters	store=none|add|replace action=start|getResult|delete
Actions	If store = add: store enrichment results in the repository If store = replace: delete previous enrichments and add new ones If store = none: directly return results Add URL for `http://editortool.linkedtv.eu/:publisher/:uuid` Send notification to clients (in future releases)
Postcondition	`http://api.linkedtv.eu/mediaresource/:uuid/state` = ENRICHED (if CURATED this state remains)

2.3 Getting enrichment results: Integration into the LinkedTV Data Interface

The LinkedTV Data Layer includes all persistent LinkedTV data, in particular the RDF Repository, but also a SQL Database for non-RDF data, a Solr-Index and a document-oriented NoSQL Database (MongoDB), currently only for internal usage. The Data Layer is exposed via the LinkedTV Data Interface at `http://data.linkedtv.eu` (RDF data) and `http://api.linkedtv.eu` (SQL) data. The Data Interface is a LDA[7] compliant REST API[8] implemented using the ELDA Framework[9].

The integration of the access to the enrichments consists of the following REST call defined in ELDA (the base URL is `http://data.linkedtv.eu`):

Prefixes:

```
PREFIX oa:        <http://www.w3.org/ns/oa#> .
PREFIX ma:        <http://www.w3.org/ns/ma-ont#> .
PREFIX linkedtv:      <http://data.linkedtv.eu/ontologies/core#> .
```

In LinkedTV, we support four type of granularity for attaching enrichments, namely: (1) an entire media resource, (2) a media fragment, (3) a chapter and (4) a shot. We define below the corresponding API calls that enable to retrieve a set of enrichments depending on the granularity.

(1) Endpoint: `http://data.linkedtv.eu/mediaresource/:id/enrichment`
Corresponding SPARQL Query:

```
SELECT ?item
WHERE
  ?mediafragment ma:isFragmentOf ?mediaresource .
  ?mediafragment a ma:MediaFragment .
  ?annotation a oa:Annotation .
  ?annotation oa:hasBody ?item .
  ?annotation oa:hasTarget ?mediafragment .
  ?annotation oa:motivatedBy oa:linking .
```

(2) Endpoint: `http://data.linkedtv.eu/mediafragment/:id/enrichment`
Corresponding SPARQL Query:

```
SELECT ?item
WHERE
  ?mediafragment a ma:MediaFragment .
  ?item a oa:Annotation .
  ?item oa:hasTarget ?mediafragment .
  ?item oa:motivatedBy oa:linking .
```

(3) Endpoint: `http://data.linkedtv.eu/chapter/:id/enrichment`
Corresponding SPARQL Query:

```
SELECT ?item
WHERE
  ?mediafragment a ma:MediaFragment .
  ?mediafragment ma:isFragmentOf ?mediaresource .
  ?annotation a oa:Annotation .
  ?annotation oa:hasBody ?chapter .
  ?chapter a linkedtv:Chapter .
  ?annotation oa:hasTarget ?mediafragment .
  ?annotation oa:motivatedBy oa:linking .
```

[7]LDA: Linked Data API, `https://code.google.com/p/linked-data-api/wiki/Specification`
[8]For a description of the current state of the LinkedTV Data Interface REST API, see D5.6 [8]
[9]`https://github.com/epimorphics/elda`

(4) Endpoint: `http://data.linkedtv.eu/shot/:id/enrichment`
Corresponding SPARQL Query:

```
SELECT ?item
WHERE
  ?mediafragment a ma:MediaFragment .
  ?mediafragment ma:isFragmentOf ?mediaresource .
  ?annotation a oa:Annotation .
  ?annotation oa:hasBody ?shot .
  ?chapter a linkedtv:Shot .
  ?annotation oa:hasTarget ?mediafragment .
  ?annotation oa:motivatedBy oa:linking .
```

3 IRAPI Module

The IRAPI Module (also called Unstructured Search Module or USM) serves within the LinkedTV project for retrieval of enrichment content from a curated list of web sites. It is a WP2 component ensuring the crawling and indexing of web sites that have been put by the LinkedTV content partners on the crawling "white list", which ensures that only results from credible web sites are returned. The system is implemented as a set of plugins for the Apache Nutch crawling framework. Apache Nutch is the leading open source crawling technology, however, the LinkedTV plugins extend its functionality to media crawling and indexing. The overall architecture of the system has been described in D2.5 [4], this section focuses on the following web mining processes:

1. extraction of candidate enrichment content and the corresponding metadata from web pages,
2. annotation of enrichment content with entities.

The main advance in the extraction process covered in this section is the release of the Metadata Extraction Service (MES), which extracts supplementary metadata information for an identified media object. The web page source code is analyzed using a probabilistic approach. MES has been released along with a focused crawler that embeds it to complement the standard flat crawling approach and custom-built metadata extraction wrappers introduced in D2.5.

The annotation of enrichment content with entities is a new process that was put in place according to the specification in D2.5. The enrichment content is performed via a dedicated application (THD Annotation Service), which is invoked from IRAPI. This section covers not only how this annotation service is used within IRAPI, but also briefly covers the advances in the text-mining approach used to generate the underlying *Linked Hypernyms Dataset* knowledge base from Wikipedia article texts (more details can be found in the referenced papers).

Finally, this section describes the Dashboard feature of the IRAPI, which provides an overview of how many media objects (videos, podcasts and images) have been identified in the crawled web pages. This component also gives the content partners the ability to control the white list.

This section is organized as follows. In Section 3.1, we describe the Metadata Extraction Service (MES) update to the IRAPI. Section 3.2 describes the focused crawler. Section 3.3 describes how the Targeted Hypernym Discovery (THD) algorithm is used within the IRAPI Module to semantically annotate the enrichment content. Finally, Section 3.4 presents the dashboard. Section 3.5 presents the evaluations.

3.1 Metadata Extraction Service

The primary goal of the Metadata Extraction Service (MES) is to identify and extract textual metadata describing media objects embedded in web pages. The relevant media objects include videos, podcasts and images. The textual metadata includes text in web pages containing the titles or descriptions of the metadata objects.

The output of MES are records having the following structure:

– media object identifier such as URL (1 occurrence)

– title (0-1 occurrences)

– description (0-3 occurrences)

– date (0-1 occurrence)

Clients of MES may specify existing media objects in the form of annotations on documents on the input of the MES, which then translates these annotations to candidate occurrences of media objects for extraction, and attempts to surround them with metadata.

The algorithmic approach taken in MES is covered in Section 3.1.1, the specific extraction model used is detailed in Section 3.1.2. The way the output of the MES service has been integrated with the previously developed Nutch plugin for media metadata extraction is described in Section 3.1.3. As part of the MES update, the IRAPI Module has been extended with a focused video crawler, which is introduced in Section 3.2.

3.1.1 Approach

To extract metadata describing media objects in web pages, we adopted the approach of *extraction ontologies* (EO). The method was first introduced by Embley [14] and it consists in augmenting a domain ontology with extraction knowledge that enables automatic identification and extraction of references to ontology concepts in text.

In their basic form, extraction ontologies define the concepts, the instances of which are to be extracted, in the sense of various attributes, their allowed values as well as higher level (e.g. cardinality or mutual dependency) constraints. Extraction ontologies are assumed to be hand-crafted based on observation of a sample of resources but are often suitable for intra-domain reuse. They have been primarily applied to the extraction of records consisting of textual attribute-values like product descriptions from heterogeneous HTML web pages, but are also applicable to other formats of text documents.

A key benefit is that extraction ontologies provide immediate semantics to the extracted data, alleviating the need for subsequent mapping of extracted data to a domain ontology. At the same time, they allow for rapid start of the actual extraction process, as even a very simple extraction ontology is likely to cover a sensible part of target data and generate meaningful feedback for its own redesign; several iterations are of course needed to obtain results in sufficient quality.

3.1.1.1 Extraction ontology engine An extraction engine named *Ex* [24] developed at UEP's Department of Knowledge Engineering, extends the original method of extraction ontologies in several aspects:

– Defines a formal XML-based Extraction Ontology Language (EOL).

– Uses a pseudo-probabilistic model to combine multiple extraction evidence to estimate probabilities of potentially extractable objects.

– Allows for definition of regular expression patterns at the level of whole words, characters and HTML element tags, with possible references to pre-existing annotations, occurrences of other potential attribute values and matches of other patterns.

– Can extract structured records as well as standalone attribute values.

– Allows for the specification of integrity constraints or axioms that impose restrictions on both the values of extracted attributes and on the content of extracted structured records, e.g. by allowing or restricting certain combinations of attribute values that comprise a valid record. In addition, all constraints can be treated as fuzzy by adding probabilistic parameters.

– Uses a scripting language (JavaScript) to enable scripting during the extraction process; primarily used to define axioms and integrity constraints.

– Supports using and training machine-learning classifiers or sequence labelers to aid the extraction process once enough training data becomes available.

For the purpose of extracting metadata about media objects found in web pages, the Extraction Ontology Language and the underlying system were extended in the following ways.

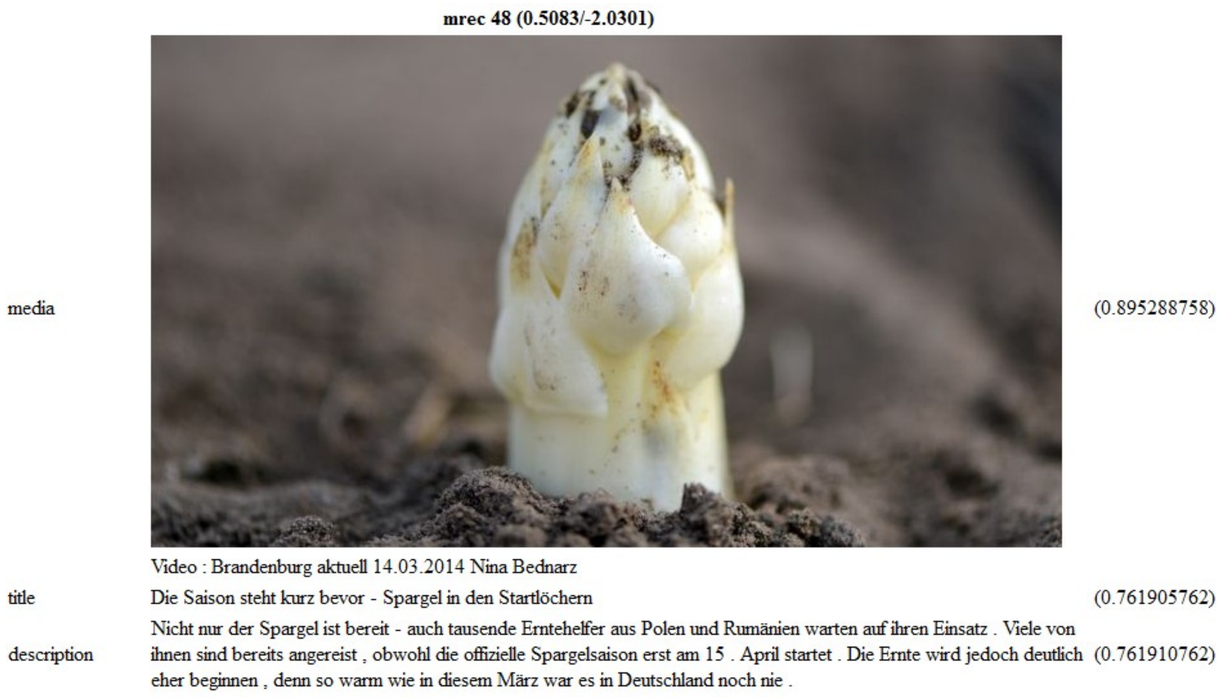
– Capability was added to extract non-textual attribute values corresponding to HTML sub-trees that represent whole sections of web pages including formatting.

– Support for extraction patterns based on XPath queries was added in combination with token-level regular expression patterns.

3.1.2 Extraction model

In this section we describe the extraction ontology (extraction model) that has been developed for the purpose of extracting textual metadata surrounding occurrences of various media objects in a web page.

The model consists of a single class titled "MediaRecord" which encapsulates a single occurrence of a *media* object, exactly one occurrence of its textual *title* found nearby, and optional multiple occurrences of further textual *descriptions* and mentions of *dates* occurring near the extracted media object. The media object is extracted in the form of a fragment (subtree) of the analyzed HTML document (e.g. the corresponding <img>, <video> or <embed> tag along with its contents).



mrec 48 (0.5083/-2.0301)

media		(0.895288758)
	Video : Brandenburg aktuell 14.03.2014 Nina Bednarz	
title	Die Saison steht kurz bevor - Spargel in den Startlöchern	(0.761905762)
description	Nicht nur der Spargel ist bereit - auch tausende Erntehelfer aus Polen und Rumänien warten auf ihren Einsatz . Viele von ihnen sind bereits angereist , obwohl die offizielle Spargelsaison erst am 15 . April startet . Die Ernte wird jedoch deutlich eher beginnen , denn so warm wie in diesem März war es in Deutschland noch nie .	(0.761910762)

Figure 3: Labeled output of Ex showing a single extracted media record with a single title and a description extracted for a news picture.

3.1.2.1 Extraction Evidence The features used by the current extraction ontology model are limited to those available in the HTML source that appears on the input of the system.

To extract media objects, the system uses the following extraction features:

– Annotations inserted into the analyzed document as a product of preceding analysis by various parsers (preceding Nutch plugins). When available, these annotations are translated by the model to highly confident candidates for extracted media objects.

– Several XPath patterns aiming to extract occurrences of typical HTML subtrees that may represent relevant images, videos or podcasts.

To extract candidate title and description texts, the model uses token- and HTML tag-level patterns that aim to model:

– HTML tag types that often contain image captions,

– word counts (length of text) typical for titles and descriptions,

– proximity to media objects.

Furthermore, token and character level regular expressions are used for identifying more structured attibute values like dates.

To link candidate titles and descriptions to media objects, the model uses:

– proximity in words of document

– proximity in formatting elements

– similarity of candidate text to other metadata known for the media like the alt attribute

Sample output of this extraction ontology model for a part of a news HTML page is shown in Fig. 3, showing extraction confidences for the whole record in the top part of the figure (the first number in brackets) and confidences for the individual attribute values to the right. The shown labeled format serves for visual inspection of results only, the output of the system used for further processing uses a format shown below in Listing 1. There may be multiple records (instance elements) per analyzed page.

Listing 1: XML output of MES showing a single extracted record

```
<instance id="1" class="mrec" p="0.5083" a="ex">
  <lab att="mrec.media" ins="1" p="0.8953" a="ex">
    <IMG alt="Spargel (Quelle: dpa)" height="398" width="708" pagefrequency="1"
         src="spargel_files/size708x398.jpg" title="Spargel (Quelle: dpa)" />
    Video : Brandenburg aktuell 14.03.2014 Nina Bednarz
  </lab>
  <lab att="mrec.title" ins="1" p="0.7619" a="ex">
    Die Saison steht kurz bevor - Spargel in den Startl?chern
  </lab>
  <lab att="mrec.description" ins="1" p="0.7619" a="ex">
    Nicht nur der Spargel ist bereit - auch tausende Erntehelfer aus Polen
    und Rum?nien warten auf ihren Einsatz. Viele von ihnen sind bereits angereist,
    obwohl die offizielle Spargelsaison erst am 15. April startet. Die Ernte wird
    jedoch deutlich eher beginnen, denn so warm wie in diesem M?rz war es
    in Deutschland noch nie.
  </lab>
</instance>
```

3.1.2.2 Example excerpt from the extraction model Listing 2 shows a simplified version of the utilized extraction ontology model expressed in *EOL*.

Listing 2: Excerpt of a simplified extraction model expressed in the Extraction Ontology Language

```
<class id="MediaRecord" prune="0.05">

  <pattern id="media_near_title" type="pattern" cover="0.3">
    ^ ( ($title <tok/>{0,10} $media) | ($media <tok/>{0,10} $title) ) $
  </pattern>

  <pattern id="media_next_to_title" type="pattern" cover="0.1">
    ^ ( ($title $media) | ($media $title) ) $
  </pattern>

  <pattern cover="0.05" type="format"> no_crossed_block_tags </pattern>

  <attribute id="media" type="xml" card="1" prior="0.01" eng="1">
  <value>
      <pattern type="pattern" p="0.9">
        ($videourl | $podcasturl | $pictureurl)
      </pattern>
    <pattern id="xp1" type="xpath" p="0.6">
      //*[local-name()='img' or local-name()='embed' or local-name()='video']
    </pattern>
  </value>
  </attribute>

  <attribute id="title" type="text" card="1" prior="0.01" eng="0.5" prune="0.1">
    <pattern id="words1"> <tok/> </pattern>
    <pattern id="words2"> <tok/>{2,5} </pattern>
    <pattern id="words3"> <tok/>{6,35} </pattern>

    <value>
      <or>
        <pattern p="0.6"> <tag name="caption|h1|h2|h3"> <pattern ref="words1"/> </tag> </pattern>
        <pattern p="0.7"> <tag name="caption|h1|h2|h3"> <pattern ref="words2"/> </tag> </pattern>
        <pattern p="0.8"> <tag name="caption|h1|h2|h3"> <pattern ref="words3"/> </tag> </pattern>
      </or>
      <length><distribution min="1" max="30" /></length>
    </value>

  </attribute>

  <attribute id="description" type="text" card="0-3" prior="0.01" eng="0.7" prune="0.1">
    <value>
      <pattern type="xpath" p="0.8">
        //*[(local-name()='p' or local-name()='div')]
      </pattern>

      <length><distribution min="5" max="500" /></length>

      <pattern cover="1" type="format"> has_one_parent </pattern>
      <pattern cover="1" type="format"> fits_in_parent </pattern>
      <pattern cover="1" type="format"> no_crossed_inline_tags </pattern>
      <pattern cover="1" type="format"> no_crossed_block_tags </pattern>
    </value>
  </attribute>

</class>
```

3.1.3 Integration of MES into the IRAPI Module

The IRAPI already contains a Nutch module, which is responsible for identification of metadata. This module has been designed to wrap frequently occurring design patterns to embed media objects on white listed web sites. In general, the hand-crafted wrapper (described in D2.5 section 3.3.1) provides better results than MES if the web site layout falls within one of the supported design patterns. In some cases, the probabilistic MES system can provide complementary output to metadata extraction with the wrapper approach. The integration problem is aggravated by the probabilistic nature of the MES output, which can provide multiple candidates for a given attribute (e.g. title).

To take advantage of MES and the hand-crafted wrappers simultaneously, we have taken the following approach. The metadata identified by the hand-crafted wrappers are saved to the original index fields (title and description), the output of the new MES module is saved into a separate field. All of the fields are used for retrieval. While MES supports also other media types, the initial release focuses on video, which is the most significant enrichment media type and and at the same time the most difficult one to extract metadata from due to the variety of ways videos are embedded into web pages.

The MES is deployed for metadata extraction within the focused video crawler see the next Section 3.2.

3.2 Focused video crawler

The MES has been incorporated into the focused video crawler module. The purpose of the focused video crawler is to index documents that are relevant to queries issued to IRAPI.

The focused video crawler is based on the assumption that for each scenario, there are several high priority web sites which can be assumed to contain multiple relevant results for a significant portion of enrichment queries. The web sites covered by the focused video crawler are RBB Mediathek and ARD Mediathek (RBB use case) and avro.nl (SV usecase). These web sites are too large to crawl exhaustively, which results in IRAPI retrieving only a portion of relevant content. Additionally, the search results can omit recently added items. To address this issue, the focused crawler wraps the video facet search facility which these large web sites offer. Using the on site search, the crawler identifies web pages embedding video that are relevant to the query issued to IRAPI. These are crawled in a priority queue, which indexes them typically within minutes of the original user query.

Indexing the on-site search results, as opposed to the “proxy” approach, has several advantages both in terms of performance and the overall quality of the results:

– The response time is not susceptible to delays in response of the individual on-site search facilities.

– The THD annotations can be created at the indexing stage (as opposed to time-consuming on-demand computation).

– Allows to sort the results according to a globally valid relevance measure[10] across all web sites.

The focused video crawler is a separate service, which is invoked each time the IRAPI component of IRAPI receives a query. The system first checks the query against the history: if the same query was issued in a predefined history window, it is believed that up-to-date results are already in the index. Otherwise, the supported on-site search interfaces are queried using the video facet, and the top N results (where N is pre-specified parameter) are saved to the index along with THD annotation and MES extraction results.

3.3 Annotating Enrichments with Entities

The enrichment content crawled and indexed with the IRAPI Module is enriched with semantic annotations. This has been implemented to comply with the WP 4 requirement to have enrichment content annotated with entities, to allow matching with the seed content, which is also semantically annotated.

Sections 3.3.1 introduces the Targeted Hypernym Discovery algorithm, which is used to perform the enrichment, and the Linked Hypernyms Dataset; a knowledge base created with this algorithm. The THD Annotation Service is described in Section 3.3.2. The IRAPI module which wraps this service to perform the annotation of enrichment content is described in Section 3.3.3.

[10] The same retrieval algorithm is used to rank content indexed from on all web sites.

3.3.1 THD and Linked Hypernyms Dataset

THD [21] is a text-mining algorithm which uses pattern-based matching (GATE JAPE grammars) to extract type information from Wikipedia article pages and resolves it to a DBpedia concept. The Linked Hypernyms Dataset (LHD)[11] is a database of entity-type pairs which have been pre-computed by the algorithm from a Wikipedia (DBpedia) snapshot.

The advances in LHD from the previous version, described in D2.3:

- Extraction grammars for German and Dutch
- Automated generation process
- The texts of articles for analysis are retrieved from DBpedia (rather than by parsing a Wikipedia dump as in the previous version).
- The types are assigned from DBpedia 3.9
- Type inference algorithm for increased coverage with DBpedia Ontology types

The first release of the LHD dataset including German and Dutch was generated from 2012 Wikipedia snapshot and contained 2.8 million entities. The second release in summer 2014, increased the number of entities covered to 4.8 million. Details on the latest developments can be found in [22, 23]. Paper [22] focuses on the statistical ontology alignment algorithm used to increase the coverage with DBpedia Ontology types, while [23] describes the LHD generation framework.

3.3.2 Targeted Hypernym Discovery Annotation Service

THD Annotation Service[12] performs the THD annotation of submitted plain text content [11]. The service was developed specifically for LinkedTV purposes to provide annotation of German and Dutch content, which was unsupported by related services in the beginning of the project.

A distinct advantage of THD Annotation Service, which fosters the fusion of its results with results of other services in the NERD framework, is the fact that it uses its own precomputed database of types – the *Linked Hypernyms Dataset* (LHD). Optionally, THD returns also types from YAGO and DBpedia knowledge bases. A unique feature of the tool is the ability to obtain the type from live Wikipedia, which allows to return type for entities that had their Wikipedia page setup only recently, and have not yet been included into the LHD/YAGO/DBPedia knowledge bases.

Selected advances in the THD from the previous version, described in D2.3:

- multiple filtering options, e.g. to return entity types only in `dbpedia.org/resource` or `dbpedia.org/ontology` namespace.
- the results from the processing can be serialized in the NLP Interchange Format (NIF) 2.0 and JSON-LD Linked Data based format
- Cross-Origin Resource Sharing (CORS) protocol which enables third-part applications to make client-side cross-origin requests.
- The knowledge base was updated to LHD 2.3.9

More details on the THD Annotation Service can be found in [11].

3.3.3 THD Annotation plugin for Nutch

THD annotation plugin for Nutch is a plugin which is used to enrich document in the index with semantic annotations. The annotation is performed using the Targeted Hypernym Discovery web service, which was introduced in Deliverable D2.3.

The IRAPI Module holds slightly different set of fields for individual media types (video, audio, podcast, text). The plugin submits the (plain text) content of selected fields for each media type for analysis to the THD web service. Since the THD API requires the information about the language of a given text, the `LanguageIdentifierPlugin`[13], which is distributed with Nutch but not included by default, is configured to run before the THD plugin.

[11] `http://ner.vse.cz/datasets/linkedhypernyms/`
[12] `http://ner.vse.cz/thd`, also available at `http://entityclassifier.eu`
[13] `http://wiki.apache.org/nutch/LanguageIdentifierPlugin`

The THD web service returns a JSON-formatted list of entities and their types (see Listing 3), which have been identified in the text. This output is further processed by the Nutch THD plugin to a compact representation (see Listing 4) and then the annotation is saved to the index.

Listing 3: JSON format for "Charles Bridge"

```
[ { "endOffset" : 18,
    "entityType" : "named entity",
    "startOffset" : 4,
    "types" : [ { "confidence" : { "bounds" : null,
              "type" : "extraction",
              "value" : null
            },
          "entityLabel" : "Charles Bridge",
          "entityURI" : "http://dbpedia.org/resource/Charles_Bridge",
          "provenance" : "dbpedia",
          "typeLabel" : "Infrastructure",
          "typeURI" : "http://dbpedia.org/ontology/Infrastructure"
        },
        { "confidence" : { "bounds" : null,
              "type" : "extraction",
              "value" : null
            },
          "entityLabel" : "Charles Bridge",
          "entityURI" : "http://dbpedia.org/resource/Charles_Bridge",
          "provenance" : "dbpedia",
          "typeLabel" : "ArchitecturalStructure",
          "typeURI" : "http://dbpedia.org/ontology/ArchitecturalStructure"
        },
        { "confidence" : { "bounds" : null,
              "type" : "extraction",
              "value" : null
            },
          "entityLabel" : "Charles Bridge",
          "entityURI" : "http://dbpedia.org/resource/Charles_Bridge",
          "provenance" : "dbpedia",
          "typeLabel" : "Place",
          "typeURI" : "http://dbpedia.org/ontology/Place"
        }
      ],
    "underlyingString" : "Charles Bridge"
  } ]
```

The primary reason for using the compact representation is higher efficiency (saving space in the index). The field holding thd annotations can prospectively be used also in retrieval. For the latter purpose, a custom tokenization for the field was implemented. For example the annotation "http://dbpedia.org/resource/Italy" produces the token ”italy”.

Listing 4: compact format for "Charles Bridge"

```
http://yago-knowledge.org/resource/Charles_Bridge=http://yago-knowledge.org/resource/
    wikicategory_Bridges_in_the_Czech_Republic,http://yago-knowledge.org/resource/
    wordnet_bridge_102898711,http://yago-knowledge.org/resource/
    wikicategory_Bridges_completed_in_1402;http://dbpedia.org/resource/Charles_Bridge=http://dbpedia
    .org/ontology/Infrastructure,http://dbpedia.org/ontology/ArchitecturalStructure,http://dbpedia.
    org/ontology/Place,http://dbpedia.org/ontology/RouteOfTransportation,http://dbpedia.org/ontology
    /Place,http://dbpedia.org/ontology/Infrastructure,http://dbpedia.org/ontology/Bridge,http://
    dbpedia.org/ontology/Bridge,http://dbpedia.org/ontology/RouteOfTransportation,http://dbpedia.org
    /ontology/ArchitecturalStructure
```

The compact representation uses the following constructs:

– different entities with {space}

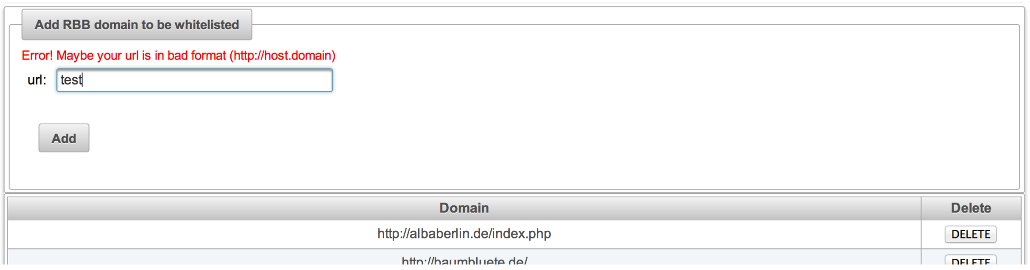
– same entity but different resource : ";"

– types related to entity : ","

– entity to type: "="

The retrieval component of the IRAPI Module (IRAPI) can re-serialize the THD annotation back to a JSON output format.

Finally, the THD annotation process is run at crawling time. In case the THD annotation service is not available, the crawler skips the annotation process and indicates this in the index. When document with a missing annotation is encountered, then the IRAPI component performs on demand enrichment and in addition to returning the result it also saves the annotation to the index. The same mechanism can be invoked when THD annotations have not been stored for a given field.


Add RBB domain to be whitelisted

Error! Maybe your url is in bad format (http://host.domain)

url: test

Add

Domain	Delete
http://albaberlin.de/index.php	DELETE
http://baumbluete.de/	DELETE

Figure 4: Whitelist Administration with two options for adding and removing URL.

3.4 Monitoring the Crawling and Indexing of White Listed Web Sites with a Dashboard

A dashboard is a simple user interface which helps to monitor results of the crawling and indexing processes. The main purpose of the dashboard is to offer detailed and up-to-date statistics for the data stored in the index. The dashboard distinguishes between the types of documents stored (web page, image, video, podcast). Data which are shown in the dashboard are retrieved directly from the index using appropriate queries, therefore the dashboard displays “live” index status. There are several other functionalities. It is possible to monitor the server availability, to check the last modification date of the index, to determine the number of stored documents, to filter and to sort results or make several data exports.

3.4.1 Implementation

The statistics displayed by the dashboard are generated using Lucene queries, separate queries are issues for individual media media types stored in index, the returned results are processed and counts of media types are used to populate data tables and charts. Whitelist administration (adding or removing of URL) is realized by remote modifying configuration files, which are part of the Nutch crawling process.

– Seed file which is the main file to add URL to be processed.

– Regex-urlfilter to restrict URL by regular expression pattern.

– Whitelist-urlfilter to indicate which type of whitelisted URL is stored in index.

3.4.2 Functionality

3.4.2.1 Whitelist Administration The dashboard allows to manage both whitelists[14]. The user first picks up the whitelist, and then there he is presented with two options.

– Add URL

– Delete URL

This is shown in Figure 4.

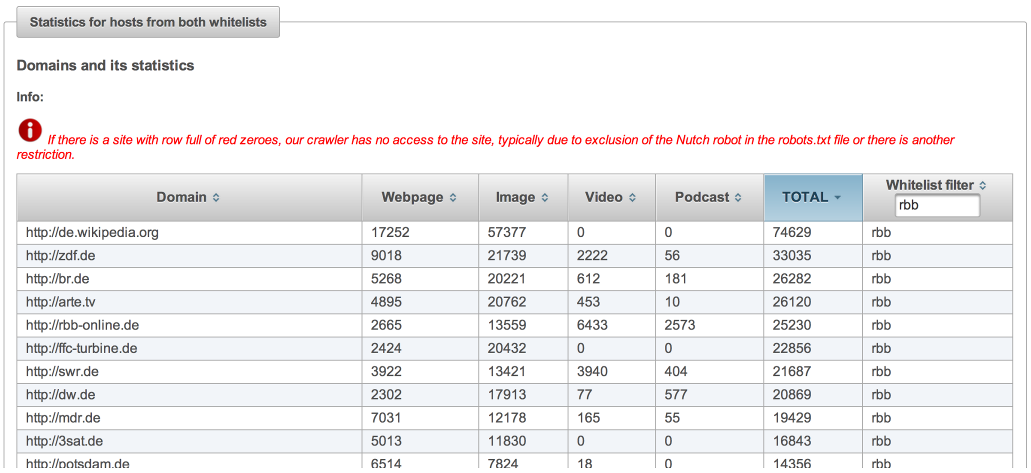
3.4.2.2 Index Statistics in Data Table Displaying index statistics is main function of the dashboard. Crawling and indexing statistics are stored in *sortable* data table which supports filtering by whitelist.
Example data table is shown in Figure 5.

3.4.2.3 Data Export The data table with index statistics supports data export into various formats. Supported data formats are CSV (Listing 5) and XML (Listing 6).

Listing 5: Example of a structure of data exported to CSV format

```
"http://3sat.de","5013","11830","0","0","16843","rbb"
"http://berlin-recycling-volleys.de","573","639","0","0","1212","rbb"
"http://br.de","5268","20221","612","181","26282","rbb"
```

[14] RBB and S&V



Statistics for hosts from both whitelists

Domains and its statistics

Info:

If there is a site with row full of red zeroes, our crawler has no access to the site, typically due to exclusion of the Nutch robot in the robots.txt file or there is another restriction.

Domain	Webpage	Image	Video	Podcast	TOTAL	Whitelist filter rbb
http://de.wikipedia.org	17252	57377	0	0	74629	rbb
http://zdf.de	9018	21739	2222	56	33035	rbb
http://br.de	5268	20221	612	181	26282	rbb
http://arte.tv	4895	20762	453	10	26120	rbb
http://rbb-online.de	2665	13559	6433	2573	25230	rbb
http://ffc-turbine.de	2424	20432	0	0	22856	rbb
http://swr.de	3922	13421	3940	404	21687	rbb
http://dw.de	2302	17913	77	577	20869	rbb
http://mdr.de	7031	12178	165	55	19429	rbb
http://3sat.de	5013	11830	0	0	16843	rbb
http://potsdam.de	6514	7824	18	0	14356	rbb

Figure 5: Sortable data table with statistics of index.

Listing 6: Example of a structure of data exported to XML format

```
<?xml version="1.0"?>
<displayDomains>
  <item>
    <domain>http://3sat.de</domain>
    <webpage>5013</webpage>
    <image>11830</image>
    <video>0</video>
    <podcast>0</podcast>
    <total>16843</total>
    <whitelist filter>rbb</whitelist filter>
  </item>
  <item>
    <domain>http://albaberlin.de</domain>
    <webpage>178</webpage>
    <image>626</image>
    <video>0</video>
    <podcast>0</podcast>
    <total>804</total>
    <whitelist filter>rbb</whitelist filter>
  </item>
<displayDomains>
```

3.4.3 Index statistics

The statistics listed below illustrate collected media types in the index of IRAPI Module (to 11/09/2014).

– Webpage: 422055

– Video: 43300

– Image: 740779

– Audio: 23818

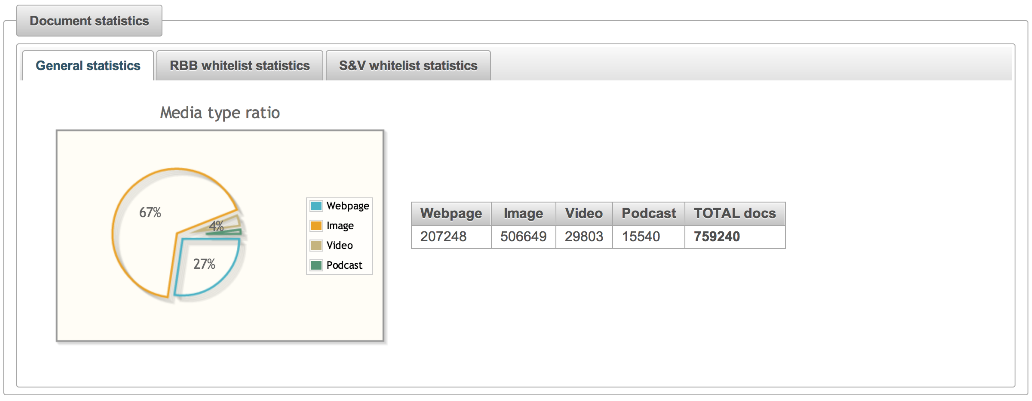
A brief overview of the index statistics is also given as a pie chart (see Figure 6).

There are several web sites which do not have any media crawled and indexed. These web sites are indicated by "red zero" column values in the row dedicated to the web site.

If there is a web site with row full of zeroes, it means that crawler has no access to the site, typically due to exclusion of the Nutch robot in the *robots.txt* file or due to another restriction from the web site provider.

3.5 Evaluation

This section presented advancements in two major processes: information retrieval and entity enrichment. The retrieval and entity linking algorithms developed and researched within LinkedTV are annually being evaluated in two international benchmarking competitions: the NIST Text Analysis Conference (Entity Linking Track) and MediaEval (Search&Hyperlinking task). English content is analyzed in both



Webpage	Image	Video	Podcast	TOTAL docs
207248	506649	29803	15540	**759240**

Figure 6: Pie chart with statistics for individual media types.

these contests. Additionally, smaller scale evaluation of the algorithms deployed into the WP2 pipeline has been performed on the LinkedTV scenario content (Dutch and German).

This section presents the following evaluations:

– To evaluate the retrieval performance we used a set of queries for RBB and S&V content. The total number of documents evaluated was 206 and the number of queries 32 (total for all media types). D2.7, the final WP2 deliverable, will include the report on participation in the comprehensive MediaEval'14 search task.[15]

– To evaluate the entity enrichment, two evaluations were performed:

  ◦ THD evaluation in the NIST TAC'13 contest[16], which evaluates whether the entities are correctly linked with their representation in the reference knowledge base (i.e., Wikipedia or DBpedia). The TAC'14 evaluation, which will be included in D2.7, newly covers also coarse-grained type evaluation.

  ◦ THD evaluation on LinkedTV data. THD was evaluated on large scale within TAC'13, and since there is no such benchmark for German and Dutch, the LinkedTV scenario languages, we complemented it with small scale evaluation on scenario content (152 entities in total).

The evaluation of the document retrieval on LinkedTV scenario content is presented in Section 3.5.1. Section 3.5.2 presents the evaluation of the THD entity annotation service.

3.5.1 Retrieval evaluation

This section presents evaluation of the retrieval performance of the IRAPI Module on Linked news and Linked culture content described in Section 1.1. For LinkedNews, seven queries specified by WP6 were subject to evaluation, for LinkedCulture, six queries were specified. For each query, a list of relevant document types was indicated.

For each of the queries, a list of maximum 10 documents per media type was retrieved using the IRAPI interface. The relevance of the search results was assessed by the respective content partner: 1 when the document could be considered as relevant for the editor curating the enrichment content for the seed video, 0 otherwise.

The relevance judgments provided by the content partners were processed to generate the following metrics for each document type:

– relevant@n - number of relevant documents in the first n results.

– precision@n - the proportion of the top-n documents that are relevant.

– success rate@n - percentage of queries for which there was at least one relevant result in top n.

– distinct@n - number of distinct domains the top n results come from.

[15]`http://www.multimediaeval.org/mediaeval2014`
[16]`http://www.nist.gov/tac/2013/KBP/EntityLinking/index.html`

Only documents with IRAPI relevance higher than 0.1 were considered. Evaluation results for the Linked News scenario are listed on Table 3 and results for the Linked Culture on Table 4. For each scenario, the average of the metrics value across all queries is listed.

Table 3: Linked News – IRAPI Evaluation results.

	queries	**relevant@10**	**precision@10**	**success@10**	**distinct@10**
image	6	4.17	0.86	1.00	2.33
webpage	7	4.60	0.69	1.00	4.17
video	6	5.20	0.80	1.00	1.67
podcast	5	1.00	1.00	1.00	1.00

Table 4: Linked Culture – IRAPI Evaluation results.

	queries	**relevant@10**	**precision@10**	**success@10**	**distinct@10**
image	3	5.0	0.56	0.33	2.0
webpage	7	3.85	0.39	1.0	2.86
video	2	0.0	0.0	0.0	0.0
podcast	2	2.0	1.0	0.5	1.0

For the Linked News scenario, the evaluation results indicate that IRAPI has been successful in obtaining at least one relevant result for all of the queries across all media types. The precision of the result set was also judged as high with 0.80 to 1.00 for all media types except webpage. About 95% of the results in top ten for video queries were retrieved using the Metadata Search Engine focused crawler, which shows the efficacy of this new IRAPI subsystem. The podcast media type has the smallest number of hits, only one podcast was retrieved for each of the queries. However, in the multimedia setup this media type is possibly of the lowest importance.

For the Linked Culture scenario, the evaluation result shows that that IRAPI has been successful in obtaining at least one relevant result for all of the queries on the webpage media type. For the remaining media types, the success rate was lower, and for video, no relevant document was retrieved. All results in top ten for video queries were retrieved using the Metadata Search Engine focused crawler. A detailed analysis of the failure to provide a suitable enrichment video shows that this can be attributed to the fact that the queries posed were either too specific ("zilversmid friesland"), which yielded only one incorrect hit, or too generic ("Nelleke van der Krogt"). For the latter query, although full 10 videos were retrieved, all were judged as irrelevant. The reason given is that these videos are presented by Nelleke van der Krogt (TV presenter), but they are not about her.

3.5.2 Entity annotation evaluation

Evaluation on TAC 2013

In this section we report on the results from the participation of the UEP team in the English entity linking task at the TAC KBP 2013. The task of the challenge was to link entity mentions in a document corpora to entities in a reference knowledge base. If the entity was not present in the reference knowledge base, a new entity node in the knowledge base had to be created. Each participation team was given a a set of $2,190$ queries consisting of a `query-id`, `doc-id`, `name` (name mention of the entity) and a `begin` and `end` index of the entity in the document. The system performing the entity linking had provided information about the `query-id` and the `kb-link` (or NIL identifier, if the entity was not present in the KB) and a `confidence score`. Each participation team could submit maximum up to 9 runs.

We evaluated various modifications and combinations of a Most-Frequent-Sense (MFS) based linking, a Entity Co-occurrence based linking (ECC), and a Explicit Semantic Analysis (ESA) based linking. We employed two of our Wikipedia-based NER systems, the Entityclassifier.eu and the SemiTags. Additionally, two Lucene-based entity linking systems were developed.

For the competition we submitted 9 submissions in total, from which 5 used the textual context of the entities, and 4 submissions did not. Below we provide brief description of the Most-Frequent-Sense based linking method, the Entity-Co-occurrence based linking method and the Explicit Semantic Analysis based linking method. We also provide brief description of each submitted run.

Most-Frequent-Sense based linking. This method does not consider the context around the entity mention, but it only relies on the surface form of the mentioned entity. In this approach the entity is linked with the most-frequent-sense entity found in the reference knowledge base. To this end, we employed the Wikipedia Search API to realize the MFS based entity linking.

Entity-Co-occurrence based linking. This method aims at capturing relations between entities occurring in the document rather than their textual representation. In our case we measure number of paragraphs where the two candidates occur in the same paragraph in our knowledge base (Wikipedia). We are searching for the best combination of candidates (possible meanings) for individual surface forms in an analysed text, where individual paragraphs represent the context.

Explicit Semantic Analysis based linking. ESA [16] has been initially introduced as method for measuring semantic similarity of two texts. In our case we adopted ESA for the entity linking task. We compute similarity between the text around the entity mention and text description of each entity candidate from the reference knowledge base. As description of the entity candidates we used the first paragraph from the Wikipedia page describing the entity.

Run #1. This run relies purely on the MFS approach. In this run each entity mention was considered as an entity, so the entity spotting step was not performed. This run uses the Wikipedia Search API the realize the MFS base linking. The article with the highest rank in the result list was considered as the correct entity.

Run #2. This run also relies on the MFS linking approach. Compared to the previous run, in this run we used the Entitiyclassifier.eu NER system to perform the linking. In this run each entity mention was submitted to the NER system. The system decides whether the string represents an entity. In a positive scenario the system disambiguates the entity by running a Wikipedia search on the API for the local English Wikipedia mirror.

Run #3. This run, same as the previous two runs, relies on the MFS linking approach, but instead of the Wikipedia Search API it uses the Lucen index created for an English Wikipedia snapshot, as of 18/9/2012. Each entity mention was searched in the index and the first returned result was considered as the correct entity.

Run #4. This is a merged submission of the previous three MFS linking approaches.

Run #5. This run relies on the ECC method used for entity linking which is used in the SemiTags NER system. For each query we sub- mitted 800 characters long text to the NER sys tem. Submitted text consists of the entity and 400 characters preceding and following the entity mention. The NER system recognizes entities in the text, links those entities with the reference knowledge base.

Run #6. This run relies on the ESA entity linking method. In this method each entity mention is considered as an entity. For each query, top five results returned by Wikipedia Search API are used as entity candidates. Next, the first paragraph of each of these candidate articles is retrieved. Finally, the ESA method is used to compute the similarity of the first paragraph of each entity candidate with the entity context text. After computing the similarity between each first entity paragraph and the entity context text, the entity (first Wikipedia article paragraph) with the highest similarity score is considered as correct and it was linked with the entity in the reference KB.

Run #7. This is a merged submission of the ESA and the ECC linking methods.

Run #8. This run combines the MFS and the ESA entity linking approaches.

Run #9. This run is a merged submission of four individual submissions. The conflicts were resolved by assigning priority to each individual submission. The highest priority was given to the run #2 (MFS with Entityclassifier.eu NER), followed by run #5 (SemiTags NER), run #6 (ESA) and run #1 (MFS baseline).

In Table 5 we provide the overall performance achieved of each individual run.

Table 5: Overall performance of all the runs.

Id	μAVG	B^3 P	B^3 R	B^3 F1	B^{3+} P	B^{3+} R	B^{3+} F1
run #1	**0.737**	0.870	**0.596**	**0.707**	**0.653**	**0.483**	**0.555**
run #2	0.686	0.821	0.515	0.633	0.568	0.387	0.461
run #3	0.727	0.844	0.593	0.696	0.632	0.471	0.540
run #4	0.733	0.835	0.570	0.678	0.622	0.461	0.530
run #5	0.611	**0.912**	0.428	0.582	0.558	0.292	0.383
run #6	0.625	0.896	0.500	0.642	0.562	0.358	0.437
run #7	0.658	0.901	0.499	0.642	0.604	0.373	0.462
run #8	0.717	0.887	0.546	0.676	0.640	0.433	0.517
run #9	0.704	0.850	0.580	0.690	0.610	0.461	0.525

The results show that in overall, the MFS linking method (cf. run #1) performed the best, achieving 0.707 B-cubed F1 score, 0.555 B-cubed+ F1 score and highest B- cubed+ precision score 0.653. Since this entity linking method is deployed in THD, the competition outcome justifies this design choice. In the TAC *English Entity Linking Evaluation* task this algorithm performed at median F1 measure (overall) [12]. This result can be considered as a success given the highly competitive nature of the contest and the fact that many competing systems were research prototypes that can take virtually unlimited time to perform the computation, while THD is a publicly available web service with response time within seconds on moderately sized documents.

Evaluation on LinkedTV scenarios

In this experiment we evaluated the performance of the THD entity linking performance on LinkedTV scenario data.

The evaluation was performed on a subtitles datasets provided by the Netherlands Institute for Sound and Vision (S&V) and the German national broadcaster Rundfunk Berlin-Brandenburg (RBB). From the S&V dataset we processed the subtitles for the *Horse Painting* scene, which is part of the *Museum Martena* episode. This is a 3 minute long episode. From the RBB dataset we also processed subtitles from a 3 minute long episode about asparagus.

Using our NER tool Entityclassifier.eu (THD) we performed entity extraction and linking over the subtitles. Each entity mention was linked to a corresponding resource in DBpedia. In this experiment we evaluated the correctness of the entity linking. The annotators were asked to assess whether the entities are linked to the correct Wikipedia article. The following types of relevance judgments were collected:

– *correct* - the Wikipedia article exclusively describes the entity.

– *incorrect* - the Wikipedia article does not describe the entity.

– *disambiguation page* - the Wikipedia page is a disambiguation page and does not describe the entity.

Note that for German and Dutch written texts THD links the entities to Wikipedia articles from the German and Dutch Wikipedias, respectively. However, if an article does not exist in the German or Dutch Wikipedia, the tool links to the English Wikipedia. For this evaluation, we considered disambiguation to the English Wikipedia as correct as long as there was a semantic match. Also, if there was a misspelling in the original text which caused an incorrect article to be linked, such an entity was excluded from the evaluation.

The results summarized in Table 6 indicate that the results of entity linking for both German and Dutch have similar precision, but they differ in the types of errors made. For Dutch, a third of the entities is assigned a disambiguation page instead of a specific sense. Work towards detecting and overcoming the disambiguation page problem constitutes the largest potential for improvement.

Once the entity has been linked to a correct DBpedia resource, the accuracy of the type assignment depends on the accuracy of the knowledge base used to obtain the type from. This is either DBpedia [1], YAGO [20] or the LHD dataset [22]. The accuracy of types in the LinkedTV supported LHD dataset for German and Dutch, the scenario languages, has been reported e.g. in [23].

Table 6: Evaluation results on LinkedTV data.

Dataset	Correct	Incorrect	Disambiguation page	Entities
RBB	57.75%	25.35%	16.90%	71
S&V	44.44%	22.22%	33.33%	81

For the full set of fields please refer to our paper [13]. The annotation process was performed by two annotators, and for spurious cases a third annotator resolved the conflict. The enriched News dataset contains 588 entities, from which 580 have assigned CoNLL type, 367 DBpedia Ontology type and 440 a Wikipedia URL.

To facilitate straightforward use of the newly created News dataset for evaluation of NER systems, we have developed a GATE Evaluation Framework. It consists of plugin to load the News dataset into

GATE, and an ontology type alignment plugin which performs alignment of the type provided by a third-party NER system and a type found in the DBpedia Ontology. We evaluated THD system using the developed GATE evaluation framework and the enriched News dataset. The results form the evaluation are reported in Table 7.

Table 7: Evaluation results for the THD NER system on the News benchmark dataset.

	Precision (strict/lenient)	**Recall** (strict/lenient)	**F1.0 score** (strict/lenient)
Entity recognition	0.69/0.78	0.33/0.38	0.45/0.51
Entity linking	0.37/0.41	0.18/0.20	0.24/0.27
Entity classification	0.69/0.78	0.33/0.38	0.45/0.51

4 LinkedTV Enrichment Framework

We advocate the adoption of the Linked Media principles, where video segments are annotated with structured information and linked to other video segments. A new generation of innovative video services intend to use those semantic descriptions and media fragments for providing the users a novel experience where television content and web information are seamlessly interconnected.

In this chapter, we first describe the **TVEnricher** service that includes a number of modules that each provides enrichments according to a source or a dimension (Section 4.1). This service is particularly adapted to the LinkedCulture scenario [18].

Relying on subtitles to extract named entities that can be used to index fragments of a program is a common method. However, this approach is limited to what is being said in a program and written in a subtitle, therefore lacking a broader context. Furthermore, this type of index is restricted to a flat list of entities. In the context of the LinkedNews demonstrator, we explored another approach where we combine the power of non-structured documents with structured data coming from DBpedia to generate a much richer, context aware metadata of a TV program. We demonstrate that we can harvest a rich context by expanding an initial set of named entities detected in a TV fragment. We described this specific **TVNewsEnricher** service in the Section 4.2 [17, 19].

Finally, for both the LinkedCulture and the LinkedNews scenarios, there is a requirement for hyperlinking fragments of a particular chapter or fragment of a program with other fragments of the same type of program but generally broadcasted on a different date (e.g. another RBB News Show program or another Tussen Kunst and Kitsch episode). We describe a specific module called **TV2Lucene** that considers all programs that have been processed by the LinkedTV platform as one big collection from which hyperlinks between parts of videos can be derived according to similarities in their descriptions (Section 4.3).

4.1 TVEnricher: getting enrichments for the LinkedCulture scenario

We first describe two important modules of this service, namely the Europeana module (Section 4.1.1) and the Media Collector module (Section 4.1.2) that provides an interface to the IRAPI component described in the Chapter 3. Then, we detail the specific API of TVEnricher (Section 4.1.3). The REST calls that are defined are typically used by the platform or by a client such as the Editor Tool.

4.1.1 Getting enrichments from Europeana

The **Europeana Module** is a component developed by EURECOM and integrated in the **Media Collector** component which is itself integrated into the **TV Enricher** service. This module enables to make search queries against the Europeana SPARQL endpoint, where information is available in RDF following the EDM ontology[17].

The SPARQL query being submitted to the Europeana's endpoint takes as input a string (typically, the label of an entity), and searches for items where this label is mentioned in the title, in the description, or in the creator fields.

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX edm: <http://www.europeana.eu/schemas/edm/>
PREFIX ore: <http://www.openarchives.org/ore/terms/>
PREFIX dct: <http://purl.org/dc/terms/>
SELECT ?item ?mediaurl ?poster ?date1 ?date2 ?description ?publisher1 ?publisher2
WHERE {
  ?proxy ore:proxyFor ?item;
         dc:title ?title;
         dc:creator ?creator;
         dc:description ?description;
         dc:type \ type \ ;
    OPTIONAL {
      ?proxy ore:proxyIn ?provider.
      ?provider edm:object ?mediaurl.
    }
    OPTIONAL {
      ?proxy ore:proxyIn ?providerAux.
      ?aggregation ore:aggregate ?providerAux.
      ?aggregation edm:preview ?poster.
    }
    OPTIONAL {?proxy dc:date ?date1.}
    OPTIONAL {?proxy dct:created ?date2.}
    OPTIONAL {?proxy dc:publisher ?publisher1.}
```

[17] http://pro.europeana.eu/edm-documentation

```
    OPTIONAL {?proxy dc:source ?publisher2.}
    FILTER (REGEX(STR(?creator), \ query \, "i") &&
            REGEX(STR(?title), \ query \, "i") &&
            REGEX(STR(?description), \ query \, "i") )
}
LIMIT 10
```

This template query contains a few variables:

– query that should be replaced by the query string, e.g. “hessel van martena”

– type which should be replaced by the type of resource, e.g. “image” or “boekillustratie” or “prent”

This template query contains a AND between the different filters, i.e., results that will be returned contain the query string in the creator, title and description properties. This is therefore a very constrained query that can be relaxed when using the OR (||) boolean operator. The `dc:type` property appears also to be a constraint that can be relaxed.

4.1.2 MediaCollector: Getting enrichments from APIs

MediaCollector is a REST API that enables to search for images and videos using different settings depending on the type of search performed. Items returned are:

– Fresh media item shared on 12 social networks (Google+, MySpace, Facebook, TwitterNative, Twitter, Instagram, YouTube, FlickrVideos, Flickr, MobyPicture, TwitPic, Lockerz) in the last 7 days. Media items can also be restricted to particular social networks accounts and channels that are pre-selected by the content provider;

– Content from Europeana as retrieved by the Europeana module (Section 4.1.1);

– Content from a white list of resources from RBB and Sound and Vision as retrieved by the IRAPI module (Chapter 3).

Results are returned in a unified json format. The query pattern is: `http://linkedtv.eurecom.fr/api/mediacollector/search/TYPE/TERM` where:

– TYPE: one of the following

  ◦ RBB: returns results from the RBB white list including specific YouTube channels, arte+7 (arte replay that returns shows that were aired in the last 7 days) and results from the IRAPI module.

  ◦ SV: returns results from SV white list including specific YouTube channels and results from Europeana.

  ◦ freshMedia: returns fresh media items from social platforms.

  ◦ combined: combines fresh results and white list results.

– TERM: the search term

4.1.3 TVEnricher API

We describe in the following sub-sections the various API calls available for the TVEnricher service. As a design principle, we found that there are information needs that require enrichments attached to various fragments granularity and that this it is not sufficient to get enrichments for each single entity. For example, in the LinkedCulture application, a chapter corresponds to a single artwork. The TVEnricher service enables to select the granularity of the media fragments to attach enrichment. It has also a logic to find out what entities should be used to trigger such an enrichment.

4.1.3.1 TVEnricher for a Media Resource (generally called by the platform) This REST call creates a new media resource inside TVEnricher with the specified parameters, and automatically launches the enrichment process over it. It also allows to modify those parameters in the case the media resource already existed.

POST `http://linkedtv.eurecom.fr/tvenricher/api/mediaresource/UUID?endpoint=ENDPOINT&graph=GRAPH&namespace=NAMESPACE&granularity=MF_GRANULARITY&broadcaster=BROADCASTER` where:

– UUID = ID of a valid media resource inside ENDPOINT // [mandatory]

– endpoint = URL of the SPARQL endpoint where the media fragments annotations are available. // [optional, by default `http://data.linkedtv.eu/sparql`]

– namespace = base URI for creating the enrichment RDF instances // [optional, by default `http://data.linkedtv.eu/`]

– broadcaster = token indicating the company authoring the multimedia content // [optional, values: [SV, RBB], by default SV]

– granularity = the level of granularity of the media fragment where the enrichment content will be attached to. // [optional, values: [Chapter, Shot], by default Shot]

– graph = the RDF graph inside ENDPOINT where the media resource is located. [optional, e.g. `http://data.linkedtv.eu/graph/linkedtv`, by default ALL graphs in the ENDPOINT]

If the UUID corresponds to a media resource that does not exist in ENDPOINT, the enrichment will be registered in TVEnricher with a status value of `Error`. Every time a media resource parameter is changed, the enrichment is automatically reprocessed again.

4.1.3.2 TVEnricher On-demand (generally called by the Editor Tool) TVEnricher can enrich not only complete media resources but also be called (on-demand) using search terms. The following REST call generates a set of enrichments based either on a term or a list of terms (comma separated), and returns results serialized in JSON according to MediaCollector's output format.
GET `http://linkedtv.eurecom.fr/tvenricher/api/entity/enrichment/RBB?q=Obama` where:

– q = query string

– strategy = allows to select between different sets of enrichment sources [combined, freshMedia, Europeana, RBB, SV]

4.1.3.3 Getting the list of media resources processed by TVEnricher This API call enables to get the list of UUIDs corresponding to media resources that have been processed by TVEnricher so far:
GET `http://linkedtv.eurecom.fr/tvenricher/api/mediaresource/list`

4.1.3.4 Getting the enrichments which have been previously computed for a particular media resource This API call enables to get the enrichments of a particular media resource.
GET `http://linkedtv.eurecom.fr/tvenricher/api/mediaresource/UUID/enrichment` where:

– UUID = ID of a valid media resource inside ENDPOINT // [mandatory]

If a client tries to retrieve the enrichment of a media resource where STATE = `Error`, a 404 Not Found error code is returned as a response.

4.1.3.5 Getting the status of a particular media resource in TVEnricher This API call shows the status information for a particular media resource that has been or is been processed by TVEnricher. Apart from the parameters specified during the creation, it is also possible to check the STATE flag value, and some details about the serialization file in the case the enrichment process has already finished successfully.
GET `http://linkedtv.eurecom.fr/tvenricher/api/mediaresource/UUID` where:

– UUID = ID of a valid media resource inside ENDPOINT // [mandatory]

Some possible outputs depending on the current state of the enrichment are:

– Processing: The media resource is already on the REST service, but the enrichment process is still being executed. A later call will be needed to actually see when the enrichment is ready.

– Processed: The media resource has been successfully enriched, the property STATE changes to processed and a link to the RDF file is included in the json serialization under the field `enrichment`.

– Error: In case of non-existent media resource, problems during enrichment process, or exceptions during the RDF conversion.

4.2 TVNewsEnricher: getting enrichments for the LinkedNews scenario

In this section, we present an approach that generates complex structured annotations of a news event video by alleviating the lack of textual resources that limits the application of semantic extraction techniques. We first extend the initial set of descriptions about an event via Google searches and entity clustering. Secondly, we use an optimized pathfinding algorithm [10] implemented in the Everything is Connected Engine (EiCE). Applying these algorithms to Linked Data enables to resolve complex queries that involve the semantics of the relations between resources, discovering relevant resources and context-sensitive filtering resources. Each path between the resources discovered has a semantic meaning that can be traced back to the original configuration of the user and forms the basis of an explanation rather than a ranking. A major contribution in this approach is that it minimizes the size of the candidate pool of resources in order to optimize queries and increase the quality of the resulting paths [19, 17].

To reconstruct the semantic context associated with one particular news video, we extract the main concepts and entities from the subtitles and explain how they are related to each other. The complete processing workflow takes as input the textual transcript of a multimedia resource illustrating an event, as well as the start and end date for which that particular event is considered.

We assume that this event has a minimal presence and coverage on the Web to ensure that the subsequent data mining techniques can collect sufficient data to reconstruct the event's context. The output of the algorithm is a pair $Context_{Event1} = [\varepsilon, \rho]$ where ε is a list of named entities together with a numeric relevance score ($\varepsilon = \{E \times \mathbb{R}\}$, E being a set of named entities classified using the NERD ontology) and ρ being a set of predicates $[e_1, e_2, p]$, relating these entities ($e_1 \in E \vee e_2 \in E$).

Our hypothesis states that this representation of the events provides a sufficient source of information for satisfying the viewer's information needs and supports complex multimedia operations such as search and hyperlinking.

4.2.1 Named Entity Extraction

For each news item, we perform named-entity recognition over the corresponding subtitles using the NERD framework [30]. In our experiment, the language of the videos is English but NERD supports other languages. The output of this phase is a collection of entities annotated using the NERD Ontology, that comes with a first relevance score obtained from the extractors which have been used. This set includes a list of ranked entities that are explicitly mentioned in the video. Other entity based video annotation tools [26] stop at this point even when entities that can be relevant for the viewer in the context of the event are still missing. We tackle this problem by extending this first list of concepts via the entity expansion component.

4.2.2 Named Entity Expansion from Unstructured Resources

The set of entities obtained from a traditional named entity extraction operation is normally insufficient and incomplete for expressing the context of a news event. Sometimes, some entities mentioned in a particular document are not disambiguated because the textual clues surrounding the entity are not precise enough for the name entity extractor, while in other cases, they are simply not mentioned in the transcripts while being relevant for understanding the story. Disambiguating named entities is useful for providing users with basic information about the entities, e.g. by providing information from Wikipedia/DBPedia. We aim to use entities to find enrichments but we found out that the entities extracted in the subtitles are not the only ones users want to know something about. In some cases they want to know more about entities not explicitly mentioned. This justifies the development of the Entity Expansion service.

The named entity expansion operation relies on the idea of retrieving and analyzing additional documents from the Web where the same event is also described. By increasing the size of the set of documents to analyze, we increase the completeness of the context and the representativeness of the list of entities, reinforcing relevant entities and finding new ones that are potentially interesting inside the context of that news item.

The entire logic will further be described in the following subsections and mainly consist of (1) building an appropriate search query from the original set of entities, (2) retrieving additional documents about the same news event, and (3) analyzing them for providing a more complete and better ranked set of final entities, as illustrated in Figure 7.

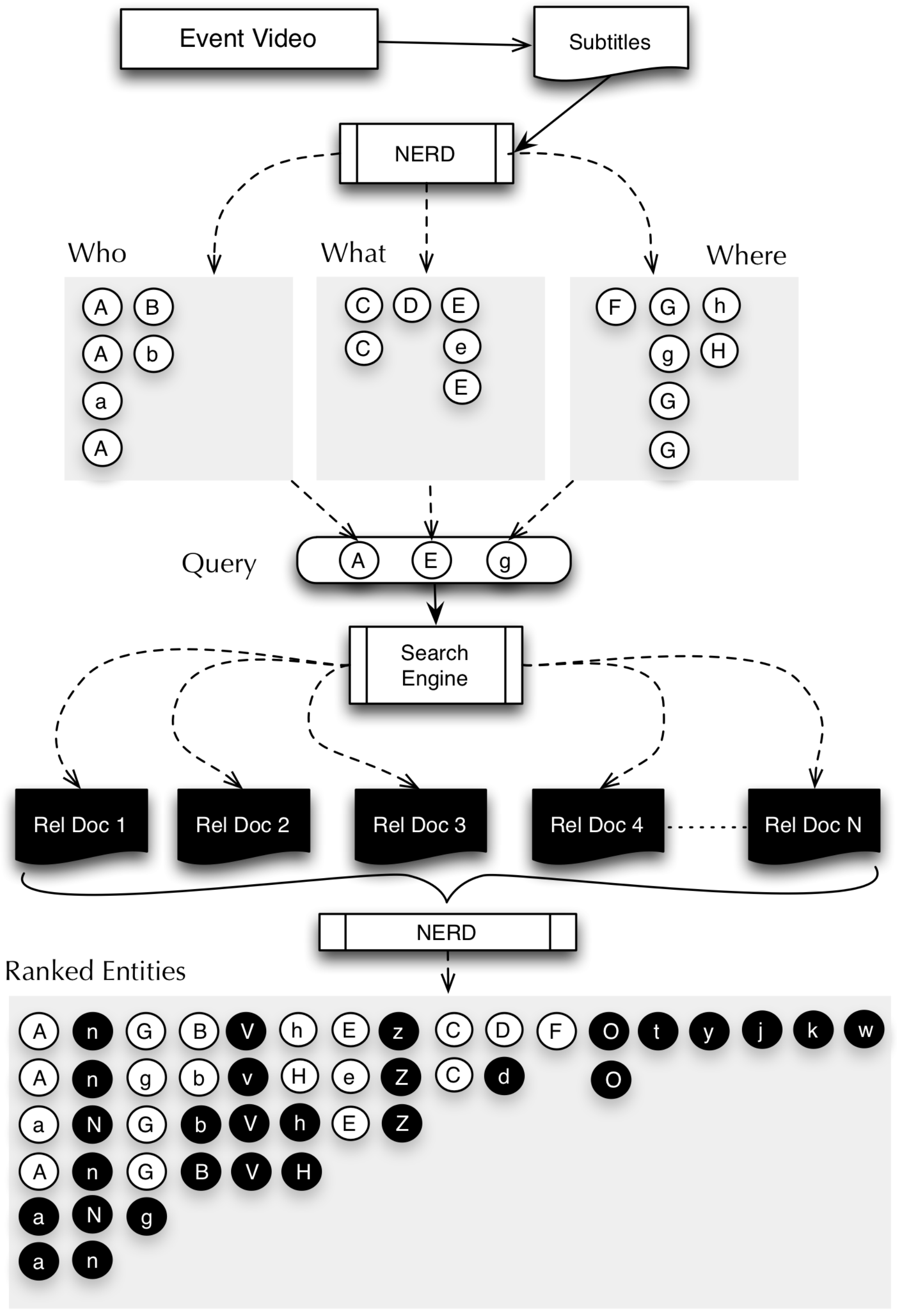


Figure 7: Schema of the Named Entity Expansion Algorithm

4.2.2.1 Query Generation The *Five W's* is a popular concept of information gathering in journalistic reporting. It captures the main aspects of a story: who, when, what, where, and why [25]. We try to represent the news item in terms of four of those five W's (who is involved in the event, where the event is taking place, what the event is about, and when it has happened) in order to generate a query that retrieves documents associated to the same event.

First, the original entities are mapped to the NERD Core ontology, which considers 10 main classes: Thing, Amount, Animal, Event, Function, Organization, Location, Person, Product and Time. From those ten different categories, we generalize to three classes: the Who from `nerd:Person` and `nerd:Organization`, the Where from `nerd:Location`, and the What from the rest of NERD types after discarding `nerd:Time` and `nerd:Amount`. The When or so-called temporal dimension does not need to be computed since it is considered to be provided by the video publisher.

After generating the three sets of entities, the next step consist in ranking them in relevance according to a weighted sum of two different dimensions: their frequency in the transcripts and their former relevance scores coming from the named entity extractors. We have defined the function *filterEntities(S)* for selecting the n entities inside the set of entities S whose relative relevance

$$R_{rel}(e_i,S) = R(e_i)/Avg(R(e_i)) \tag{1}$$

falls into the upper quarter of the interval

$$[max(R_{rel}(e_i,S)) - min(R_{rel}(e_i,S))] \tag{2}$$

The final query is a pair

$$\text{Query}_{Event} = [\text{textQuery}, t] \tag{3}$$

where *textQuery* is the result of concatenating the labels of the most relevant entities in the sets Who, What, Where in that particular order, and t the time period dimension. This query generation is depicted in the upper part of Figure 7.

4.2.2.2 Document Retrieval Once Query_{Event} is built out of the original set of named entities, it will be ready to be injected into a document search engine where additional descriptions about the news event can be found. In this situation, the kind of query generated in the previous step and the search engine chosen should be closely tied in order to maximize the quality of the obtained results. The different behavior of search engines make some alternatives more suitable than others for certain kinds of events. The way the resulting documents change in the search engines for a particular kind of event is a research question that will not be studied in this paper.

We rely on the Google Search REST API service[18] by launching a query with the text *textQuery*. Due to quota restrictions imposed by Google, the maximum number of documents that can be retrieved is set to 30.

Concerning the temporal dimension, we only keep the documents published in the time period $t + t_e$. We increase the original event period in t_e because documents concerning a news event are not always published during the time of the action is taking place but some hours or days after. The value of t_e depends on many factors such as the nature of the event itself (whether it is a brief appearance in a media, or part of a longer story with more repercussion) or the kind of documents the search engine is indexing (from very deep and elaborated documents that need time to be published, to short post quickly generated by users). Based on the simple assumption that the longer an event, the longer it is likely to generate buzzes, we approximated $t_e = t$ which means that we also consider document published during the course of an event.

The middle part of Figure 7 shows this process. The query is input in the search engine in order to retrieve other documents that report on the same event discussed in the original video. Those documents (colored in black in the Figure 7) will be further processed to increase the size of the collection and get additional insights about the news item.

4.2.2.3 Entity Clustering In this phase, the additional documents which have just been retrieved are now processed and analyzed in order to extend and re-rank the original set of entities and consequently get a better insight about the event. Since most of the resources retrieved are Web pages, HTML tags

[18] `http://ajax.googleapis.com/ajax/services/search/web?v=1.0`

and other annotations are removed, keeping only the main textual information. This plain text is then analyzed by the NERD framework in order to extract more named entities.

In order to calculate the frequency of a particular resource within the entire corpora, we group the different appearances of the same instance and check their cardinality. This is not a trivial task since the same entity can appear under different text labels, contain typos or have different disambiguation URL's pointing to the same resource. We performed a centroid-based clustering operation over the instances of the entities. We considered the centroid of a cluster as the entity with the most frequent disambiguation URL's that also have the most repeated labels. As distance metric for comparing pairs of entities, we applied strict string similarity over the URL's, and in case of mismatch, the Jaro-Winkler string distance [33] over the labels. The output of this phase is a list of clusters containing different instances of the same entity.

4.2.2.4 Entity Ranking The final step of the expansion consists of ranking the different named entities obtained so far. To create this ordered list, we assigned a score to every entity according to the following features: relative frequency in the transcripts of the event video; relative frequency over the additional document; and average relevance according to the named entity extractors. The three dimensions are combined via a weighted sum where the frequency in the video subtitles has a bigger impact, followed by the frequency on the searched documents and the relevance from the extractors. The final output of the entity expansion operation is a list of entities together with their ranking score and the frequency in both the main video and in the collected documents retrieved from the search engine.

Entities with a higher $relScore_i$ in the final classification are considered more representative for describing the context than the original entities. Furthermore, we observe that:

– The bigger the sample size, the better becomes the ranking. Entities appearing repeatedly in the additional documents will be promoted while those appearing rarely will be pushed back to the end of the list.

– Entities that originally have not been disambiguated can now have their corresponding URL if any of the similar instances appearing in the additional documents provide a link to a Web resource. The same occurs with incomplete or misspelled labels.

– Finally, some entities not spotted in the original transcripts but important in the context of the event are now included in the list of relevant items since they have been extracted from the collected documents.

4.2.3 Refining Event Context via DBpedia

Once the set of context relevant entities has been expanded, we will use the knowledge from structured sources to reinforce the important entities and finding relevant predicates between them.

4.2.3.1 Generating DBpedia paths Before we filter the relations between resources to reinforce important entities, the candidate resources to be included in relations are pre-ranked. They are pre-ranked according to "popularity" and "rarity" which are essential components in the original PageRank algorithm [29] and which is used to sort candidate related nodes in the EiCE. The implementation of the EiCE takes the relations into account by making use of the Jaccard coefficient to measure the dissimilarity and to assign random walks based weight, so that rare resources have a higher rank while guaranteeing that paths between resources prefer specific relations and not general ones [28].

We pass on the main start resource, the target one and some via points. Figure 8 shows the iterative process for generating the DBpedia paths. An initial state is computed in step 8a. There are low weights and high weights. Based on the weights of the links, each path through the vias is optimized, so a path with the lowest total weight will be selected first, until the vias are added to the exclude list. The path from start to end is forced through the given via points (8b). This leads to additional visited resource as via points (8c). They occur because each computed path starts both from the start and the target resources and goes through the via points. The resources where they converge to each other are considered as new via points. These additional via points are included in the paths and therefore marked as visited. This is to make sure that in the next iteration, paths will go around the visited via (8d). The next paths are being computed over and over (8e) until a threshold number of paths is found and the context is large enough or when it takes too long to compute the next path (out of range). The final set of optimized paths is used for the context expansion.

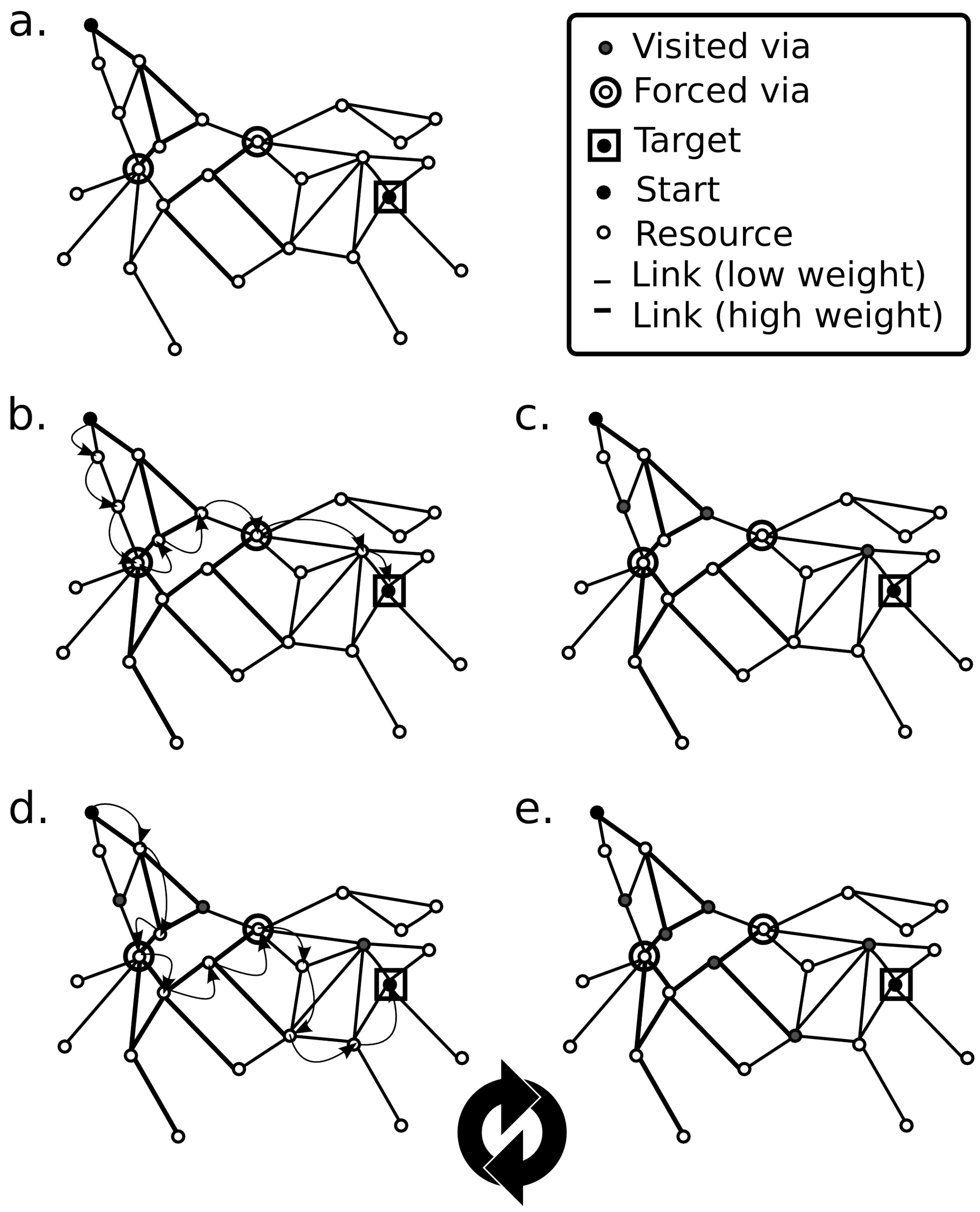


Figure 8: Pattern matching with multiple results using an iterative pathfinding process

4.2.3.2 Analyzing Context Relevant Paths In this step, the DPpedia path finding technique implemented in 4.2.3.1 is applied over the set of entities obtained via entity expansion. Since this list is too broad, we need to establish in first place a division between what we consider the *mainEntities*(entities whose relative scores fall under the higher 25% of the relevance range), and the *additionalEntities*(the rest of entities in $E_{Expansion}$).

Afterwards, we calculate paths between all the possible pairs inside the set*mainEntities*. Once all the possible paths have been retrieved, we perform various analysis for detecting which are the most frequent classes and predicates:

– We detect the most frequent nodes ($F_{nodes} = f_{max}(n_i)$ in $Paths(mainEntities)$).

– We find the most frequent properties ($F_{prop} = f_{max}(p_i)$ in $Paths(mainEntities)$). The edges (DBpedia properties) will determine which are the most relevant properties taking part in the context of this news item.

– We study the connectivity between nodes (Adjacency Matrix $M_{i,j}$ where the distance between $e+i$ and $e+j$ is the average length of the paths linking them ($m_{i,j} = Avg(lenght(Paths(e_i,e_j)))$)

The output of this phase is a re-ranked list of entities from the expansion entity set based on the paths found in DBpedia, the most important predicates and nodes inside the paths between pairs in $mainEntities$, and the adjacency matrix.

$$\{Entities_{Extension+DBpedia}, E_{Expansion}, F_{nodes}, F_{prop}, M_{i,j}\} \quad (4)$$

In the future, we plan to use the frequency measures about predicates available in F_{prop} in order to more precisely rank the named entities. By using the *commonalities* function (`http://demo.everythingisconnected.be/commonalities?between=Res1&and=Res2&allowed=property`) over the set of $mainEntities$ inside $Entities_{Extension+DBpedia}$ and for the properties with highest F_{prop}, we obtain the set of entities directly related with the original ones through the top predicates. Once more, the most frequent entities in Commonalities could promote the already existing entities in $Entities_{Extension+DBpedia}$.

4.2.4 TVNewsEnricher API

TVNewsEnricher is finally the API which is publicly exposed for enriching TV news programs with online articles and media from the Web following the process described in this section. This collection process is performed along five different dimensions: Opinion, OtherMedia, TimeLine, InDepth and geolocalized data from Twitter. The results of each dimension can be invoked via separate REST API calls as it will be further explained below. The service is available at: `http://linkedtv.eurecom.fr/newsenricher/api/`.

This service makes use of the search capabilities from Google CSE and Twitter API. It is intended to be plugged over the results obtained from the Name Entity Expansion service. However, further research is needed as how an appropriate set of entities (from the expanded set) should be selected for issuing queries in each of those dimensions. A live demo displaying the related documents found by this service is available at `http://linkedtv.project.cwi.nl/news/auto/`.

4.2.4.1 Opinion Dimension API call: GET `http://linkedtv.eurecom.fr/newsenricher/api/opinion?query=TERMS&startdate=START&enddate=END&cse=CSE&limit=50` where:

– TERMS: query string with + separated entity labels

– START: start date in the format YYYYMMDD, mandatory

– END: end date in the format YYYYMMDD, optional, by default, the current date

– limit: maximum number of documents to be retrieved, optional, by default 10

– CSE: ID of the Google custom search engine to be used for collecting related documents

The CSE defined for the RBB use case is: `008879027825390475756:xnwxm5pcj8w`

4.2.4.2 Other Media Dimension

API call: GET `http://linkedtv.eurecom.fr/newsenricher/api/othermedia?query=TERMS&startdate=START&enddate=END&cse=CSE&limit=50` where:

- TERMS: query string with + separated entity labels
- START: start date in the format YYYYMMDD, mandatory
- END: end date in the format YYYYMMDD, optional, by default, the current date
- limit: maximum number of documents to be retrieved, optional, by default 10
- CSE: ID of the Google custom search engine to be used for collecting related documents

The CSE defined for the RBB use case is: `008879027825390475756:jttjpihzlns`. When using the query terms `spargelsaison+sonne+beelitz+brandenburg`, the start date = 2013/04/01 and the end date = 2013/04/30, the following results are returned:

- Erster Spargel da! 18.4.13 (`http://www.berliner-kurier.de/brandenburg/edelgemuese-erster-spargel-da-,7169130,22523298.html`)
- Spargelzeit: Beelitzer Bückware 18.4.13 (`http://www.berliner-zeitung.de/brandenburg/spargelzeit-beelitzer-bueckware,10809312,22523960.html`)
- Wettervorhersage: Frühling bringt 20 Grad nach Berlin 10.4.13 (`http://www.morgenpost.de/berlin-aktuell/article115193221/Fruehling-bringt-20-Grad-nach-Berlin.html`)
- Saisonstart: Spargelernte beginnt trotz Kälte pünktlich 8.4.13 (`http://www.berliner-zeitung.de/berlin/saisonstart-spargelernte-beginnt-trotz-kaelte-puenktlich,10809148,22320852.html`)

4.2.4.3 Timeline Dimension

API call: GET `http://linkedtv.eurecom.fr/newsenricher/api/timeline?query=TERMS&startdate=START&enddate=END&cse=CSE&limit=50` where:

- TERMS: query string with + separated entity labels
- START: start date in the format YYYYMMDD, mandatory
- END: end date in the format YYYYMMDD, optional, by default, the current date
- limit: maximum number of documents to be retrieved, optional, by default 10
- CSE: ID of the Google custom search engine to be used for collecting related documents

4.2.4.4 In Depth Dimension

API call: GET `http://linkedtv.eurecom.fr/newsenricher/api/indepth?query=TERMS&startdate=START&enddate=END&cse=CSE&limit=50` where:

- TERMS: query string with + separated entity labels
- START: start date in the format YYYYMMDD, mandatory
- END: end date in the format YYYYMMDD, optional, by default, the current date
- limit: maximum number of documents to be retrieved, optional, by default 10
- CSE: ID of the Google custom search engine to be used for collecting related documents

4.2.4.5 Related Tweet Dimension

API call: GET `http://linkedtv.eurecom.fr/newsenricher/api/tweets?query=TERMS&startdate=START&enddate=END&lat=LAT&lon=LON&rad=RAD&limit=50` where:

- TERMS: query string with + separated entity labels
- START: start date in the format YYYYMMDD, mandatory
- END: end date in the format YYYYMMDD, optional, by default, the current date
- LAT: latitude in degrees for geolocalized tweets, optional
- LON: longitude in degrees for geolocalized tweets, optional
- RAD: radius in km, optional, by default 5 km
- limit: maximum number of documents to be retrieved, optional, by default 10

4.3 TV2Lucene: Hyperlinking to other programs analyzed via the LinkedTV platform

In the Deliverable D2.5 [4], we have described our participation in the MediaEval Search and Hyperlinking task held in 2013 which tackles the issue of information seeking in a video dataset [15]. The scenario proposed is that of a user looking for a known video segment, and who could be interested in watching related content proposed by the system from a closed collection. Hence, such a task is at the heart of LinkedTV: it is a way to test the LinkedTV framework for hyperlinking videos together on a TV dataset, in real-world conditions. In MediaEval, the dataset offered for this task contains 2323 videos from the BBC, amounting to 1697 hours of television content of all sort: news shows, talk shows, series, documentaries, etc. The idea of the TV2Lucene module is to apply this framework with content from the LinkedTV providers (LinkedNews and LinkedCulture scenarios). First, we describe how we index all metadata in a Solr index. Second, we describe the TVEnricher API call that enables to retrieve related media fragments from programs analyzed by the LinkedTV workflow. A dashboard monitoring the Solr index is available at `http://data.linkedtv.eu/solr/#/`.

4.3.1 Indexing all LinkedTV analysis results

For each program going through the LinkedTV workflow, the following metadata is indexed:

- at the program level:
  - provider (RBB, SV or UMONS)
  - title of the brand (general program) if available
  - synopsis of the brand (general program) if available
  - episode title
  - episode description (longer description found)
  - broadcaster
  - keywords given by tvanytime
  - videoId
- at the fragment level:
  - fragment id (composed of videoId + media fragment URI)
  - fragment type (so far, "scene" or "shot")
  - start time
  - end time
  - subtitle aligned with the fragment
  - visual concepts extracted by the WP1 analysis processes

We have therefore created three different indexes and we assume that the SV content will be in Dutch, the UMONS content will be in English and the RBB content will be in German. The indexes are available at:

- Index for RBB content: `http://data.linkedtv.eu/solr/#/RBBindex`
- Index for SV content: `http://data.linkedtv.eu/solr/#/SVindex`
- Index for UMONS content: `http://data.linkedtv.eu/solr/#/UMONSindex`

4.3.2 Getting enrichments from the Solr index

The TVEnricher service (Section 4.1 includes another API call that enables to get the enrichments of a media fragment based on the different Lucene indexes maintained by LinkedTV. The output is a list of LinkedTV media fragments URI's serialized in JSON.

GET `http://linkedtv.eurecom.fr/tvenricher/api/mediaresource/UUID/enrichment?t=1443.56,2447.4` where:

- UUID = ID of a valid media resource within the LinkedTV platform [mandatory]

- t = temporal references of the media fragment in the format "&t=start,end" where the timecodes are float numbers expressed in seconds
- index = [RBB, SV, UMONS] [Optional, by default RBB]

5 LUMO-based annotations and topic labelling

In the interest of bringing together heterogeneous information in the annotations of LinkedTV content under a common, user-centric vocabulary to render them more useful for WP4 services (as argued in D2.4, ch. 3), annotations of seed and enrichment content are exposed and mapped to WP4's LUMO[32][19] ontology via its mappings[20] vocabulary. Nevertheless, this exposure to LUMO offers more capabilities at the content annotation level, before the layer of personalised content delivery. A benefit of conveying automatically extracted annotations to LUMO is the ability to further annotate content with topics.

Without manual curation of content, automatic annotation tools can detect only the data that is directly present in the visual, audio and transcript context of a media items. For example, they can recognize a specific named entity and assign it with a corresponding type within a LOD vocabulary (e.g. recognize the entity "die Linke" and assign it with type "dbpedia-owl:PoliticalParty"). Consequently, types of people, locations, events, objects etc are the most commonly detected semantic information by automatic annotation tools.

From then on, annotation tools rely on the vocabularies used to type and disambiguate entities in order to retrieve additional information about the content. Vocabularies prominently used by WP2 annotation tools (e.g. DBPedia, YAGO), offer extensive domain coverage, solid hierarchies and connections at the named entity-to-type (aka instance-to-concept) level. In the example above, the recognized type can be expanded across the DBPedia hierarchy and thus additionally assign to entity "die Linke" the type "dbpedia-owl:Organisation", which is a superclass of "dbpedia-owl:PoliticalParty".

However, another useful information within the LinkedTV context would be the detection of the thematic categories that a content item falls under. Vocabularies used in WP2 lack such semantics at the schema level, or efficient connections with relevant other vocabularies that could be used to infer such information. Conversely, LUMO models such categories and also offers intelligent connections between topics and other kinds of concepts at the schema level[21].

Therefore, in the example above, the concept "Political Party" is connected to the concept "Politics" within LUMO, so via semantic inference the content item that mentions "die Linke" can be characterized as content that deals with "Politics". In conclusion, a LUMO-based topic labelling service can be used as a standalone annotation tool to assign topics to media resources/fragments or enrichments, thus offering the possibility of thematic clustering and/or categorization while delivering content to the end user (whether personalised or not).

5.1 Exposing content annotations to LUMO

This section discusses the advances in mappings between WP2 vocabularies and LUMO within year 3 of LinkedTV and presents the process of exposing WP2 annotations to LUMO. The results of this step are used as input to both the personalisation services of WP4 as well as the standalone topic detection service.

5.1.1 LUMO v2 mappings

The WP4 LUMO ontology has been extended and revised to an updated version 2 (See deliverable D4.6, chapter 3.1) and with it, the mappings[22] to WP2 vocabularies have been expanded to reflect the new vocabulary.

Furthermore, the mappings have been extended to three new vocabularies, on top of the most recent versions of the previously supported ones (namely DBPedia ontology 2014, schema.org, NERD v0.5, GUMO, IPTC news codes). Two of these newly mapped vocabularies were elected in order to support the full spectrum of advanced content annotations of WP2, while mapping to the third was deemed useful to the reuse of LUMO by the Semantic Web given the domain(s) it supports.

To this end, vocabularies mapped to LUMO now also include:

– The aforementioned *Linked Hypernyms Dataset* (LHD), therefore offering mappings to a dedicated set of DBPedia resources that are not included in the DBPedia schema but are deemed abstract enough to serve as types of entities, enabling optimal connection between LUMO and WP2's THD annotation tool,

[19]A detailed description of the ontology can be found in the deliverable D4.4, chapter 2
[20]A first detailed description of the mappings can be found in the deliverable D2.4
[21]See D4.4, chapter 2.1.2 for technical details and some examples
[22]As before, published under `http://data.linkedtv.eu/ontologies/lumo_mappings/`

– *YAGO2s[20]*, also to support the full THD spectrum of annotations, and

– the Arts & Architecture Thesaurus (AAT)[23] hierarchy, which was deemed as the most relevant and concise vocabulary to fully correspond to the LUMO-arts expansion[24] which covers LinkedTV's arts and cultural artefacts scenarios, thus enhancing LUMO's re-usability by the Semantic Web.

5.1.2 Converting annotations to LUMO

As mentioned in the introduction of this chapter, converting content annotations from the various LOD vocabularies used in LinkedTV to LUMO aims to (a) homogenize annotations under a uniform user-pertinent vocabulary and (b) provide advanced concept connections at the schema level.

The first point supports bringing forward only the information about the content that are meaningful from a user's perspective, thus bridging WP2 and WP4 services. At the same time it deduces the concept space into a lightweight and expressive vocabulary, thus enhancing the computational efficiency of inferencing services that use LUMO instead of more extensive vocabularies. The latter point offers additional annotation facilities, an example of which is topic labelling that will be presented in the following section.

Exposing annotations to LUMO involves retrieving the entities that comprise the semantic description of a candidate content item ("candidates" are defined depending on the task at hand). Such content items might be one or more media fragments or a set of enrichments attached to one or more media fragments. Then LinkedTV's LiFR reasoner[31] is employed to infer mappings between the types of the entities detected in the annotation and corresponding concepts of LUMO, using the mappings vocabulary. This functionality is supported by the LUMO wrapper tool described in D2.4.

Entities in WP2 annotation can be assigned to one or more types. Something worth noticing is that if an entity type in the annotation is not mapped to any LUMO concept, that type-entity pair will not be conveyed to the services that perform LUMO-based inferencing (i.e. the implicit personalisation and topic labelling services). Consequently, if an entity is entirely typed with classes that do not map to LUMO, this entity as a whole will not be passed to those services.

This was a design decision in modelling LUMO and its mappings, in which it is accepted that the concept space relevant to an end user, both personalisation-wise and categorisation-wise, lies only within the LUMO "world". However, the coverage of LUMO and its mappings is not narrow, as demonstrated below. The following example illustrates how an entity retrieved by the WP2 annotation tools is assigned with several types and mappings to all, from the more generic to the most specific, are retrieved.
An example of an entity in the annotation of a media item, the entity label marked in green and the recognized types for this entity marked in red:

```
<http://data.linkedtv.eu/entity/c0176517-4bc2-404b-bc32-b32777f05e31>
    a   dbpedia-owl:PoliticalParty , nerd:Organization , dbpedia-owl:Organisation ,
        linkedtv:Entity , dbpedia-owl:Agent ;
    rdfs:label "die Linke" ;
    linkedtv:hasConfidence "0.635045"^^xsd:float ;
    linkedtv:hasRelevance "0.406828"^^xsd:float ;
    dc:identifier "8134557" ;
    dc:source "textrazor" ;
    dc:type "DBpedia:Agent,Organisation,PoliticalParty;Freebase:/government/political_party,/
         business/employer,/organization/organization" ;
    owl:sameAs <http://de.wikipedia.org/wiki/Die_Linke> , <http://dbpedia.org/resource/Die_Linke> .
```

Retrieving mappings between entity types and LUMO concepts:

```
dbpedia#PoliticalParty = lumo#Political_party
nerd#Organization = lumo#Organization
dbpedia#Organisation = lumo#Organization
dbpedia#Agent = lumo#Agent
```

Retrieved LUMO concepts per entity are transcoded to a LUMO-based content profile, in the LiFR-supported syntax[25]. This content profile is the input of the topic labelling and the WP4 filtering services.

Extract of a LUMO-based profile. For simplification purposes, this example demonstrates the profile for one entity, whereas actually a content item will usually contain a number of entities and corresponding types.

[23] http://www.getty.edu/research/tools/vocabularies/aat/
[24] See deliverable D4.6, chapter 3.1
[25] http://mklab.iti.gr/project/LiFR#semantics

```
(IMPLIES (SOME hasContent inst_die_Linke ) c0176517-4bc2-404b-bc32-b32777f05e31)
(IMPLIES inst_die_Linke Political_party)
(IMPLIES inst_die_Linke Organization)
(IMPLIES inst_die_Linke Agent)
(INSTANCE die_Linke inst_die_Linke >= 0.635045)
```

5.2 LUMO-based Topic Labelling of Seed and Enrichment Content

The topics represented in LUMO v2 are, as in v1, heavily influenced by the IPTC news codes[26] and were extended to accommodate relations between newly introduced concepts and further revised in v2 in terms of semantics. In addition, some arts and artefacts related topics were added in the arts expansion based on the AAT hierarchy.

As aforementioned, thematic topics are connected with relevant other concepts (e.g. people types, events, object types etc) within LUMO via non-taxonomical relations, namely via the two object properties *hasTopic* and *hasSubtopic*. Since thematic classification of content is rarely achievable without manual (curator-based or crowdsourced) intervention, these relations allow to automatically label media items with thematic topics based on its contents.

In the example illustrated in the previous section for instance, the topic labelling service can infer that the content item that deals with the *Political_party* "die Linke" has to do with the topic *Politics* based on the concept's hierarchy and the property *hasTopic*.

The owl-xml representation of the semantic information within LUMO that lead to this conclusion consist of:

```
    <SubClassOf>
        <Class IRI="#Political_party"/>
        <Class IRI="#Political_movement"/>
    </SubClassOf>


    <SubClassOf>
        <Class IRI="#Political_movement"/>
        <Class IRI="#Political_organization"/>
    </SubClassOf>


    <SubClassOf>
        <Class IRI="#Political_organization"/>
        <ObjectAllValuesFrom>
            <ObjectProperty IRI="#hasTopic"/>
            <Class IRI="#Politics"/>
        </ObjectAllValuesFrom>
    </SubClassOf>
```

The inferencing steps in a more reader-friendly illustration consist of the following:

```
Political_party is a Political_movement.
Political_movement is a Political_organization.
ALL Political_organization(s) have as topic Politics.

Therefore, a content item about a Political_party is also about Politics.
```

An example of a full content item is illustrated in Figure 9, for an RBB news clip describing a casting event for a new movie. The automatic annotation retrieves, through ASR analysis, information mentioned in the presentation of the reportage, such as the event itself (casting), the main participants (George Clooney), details about the contents of the movie (situated in WWII) etc. As observed in Figure 9, the resulting topics reflect the artistic aspect of the news item as well as the detected movie contents, but it also conveys given misses/mistakes in the annotation.

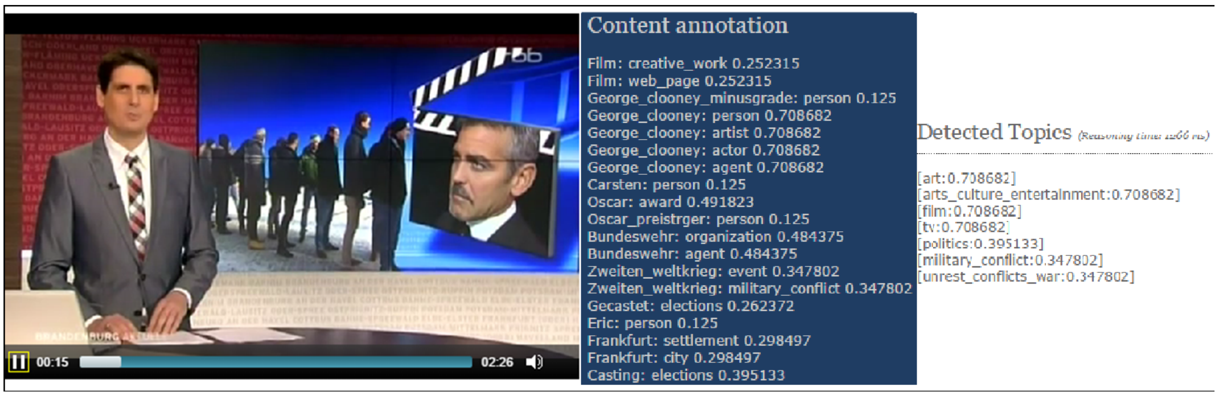
In this example, an entity type mis-recognized in the annotation ("Election" as the type for "Casting") reflects to the topic labelling, resulting to the mis-recognised topic "Politics":

```
<http://dbpedia.org/resource/Casting_(performing_arts)> <http://www.w3.org/1999/02/22-rdf-syntax-ns#
    type> <http://dbpedia.org/ontology/Election> .
```

5.2.1 Topic Labelling REST API

The topic labelling service is offered as standalone tool via a REST interface in CERTH servers which returns the LUMO topics detected for a single content item based on its annotation:

[26] www.iptc.org/site/NewsCodes/, a prominent classification schema for news topics



Content annotation

Film: creative_work 0.252315
Film: web_page 0.252315
George_clooney_minusgrade: person 0.125
George_clooney: person 0.708682
George_clooney: artist 0.708682
George_clooney: actor 0.708682
George_clooney: agent 0.708682
Carsten: person 0.125
Oscar: award 0.491823
Oscar_preistrger: person 0.125
Bundeswehr: organization 0.484375
Bundeswehr: agent 0.484375
Zweiten_weltkrieg: event 0.347802
Zweiten_weltkrieg: military_conflict 0.347802
Gecastet: elections 0.262372
Eric: person 0.125
Frankfurt: settlement 0.298497
Frankfurt: city 0.298497
Casting: elections 0.395133

Detected Topics (Reasoning time: [illegible] ms)

[art:0.708682]
[arts_culture_entertainment:0.708682]
[film:0.708682]
[tv:0.708682]
[politics:0.395133]
[military_conflict:0.347802]
[unrest_conflicts_war:0.347802]

Figure 9: Screenshot from the LiFR services demo web page, of a news item on a George Clooney movie dealing with WWII, the content annotation as mapped in LUMO and the resulting topics detected.

```
GET /api/topic_detection?cid={cid}

Description: GET /api/topic_detection?cid={cid}
HTTP method: GET
Content-Type: application/json
URI (open content version): http://multimedia.iti.gr/api/topic_detection?cid={cid}
URI (password-protected version for LinkedTV content): http://multimedia.iti.gr/api/topic_detection?
     cid={cid}&auth=true

Example of response:
 {
   [{"Time elapsed":"485 ms."}]
    "[{"politics":"0.76"}]"
    "[{"disaster":"0.451"}]"
    "[{"disaster_accident":"0.451"}]"
    "[{"natural_disaster":"0.451"}]"
}
```

5.3 Topic Labelling Evaluation

Three sets of evaluations took place for asserting the efficiency of topic labelling based on the LUMO ontology. The results are presented in Table 8 for the first two test cases and in Table 10 for the third one.

In the first use case, a controlled dataset was chosen, where 73 freely available media items (i.e. open-licensed videos, images, web pages) were selected online and manually annotated with LUMO concepts. In principal, the content was annotated with the kinds of concepts that can be automatically detected via LinkedTV's annotation services (i.e. they were not annotated with topics but with "things", such as people, locations etc). Such manual annotations are considered error-free, as opposed to automatic annotations which inevitably would carry some level of error.

Since topics are inferred from relevant concepts in the content annotation, errors in the annotation, lack of semantics modelled in LUMO, or lack of mappings between the annotation and LUMO are directly reflected at the topic labelling layer. Consequently, this controlled test case demonstrates (a) the level of completeness of LUMO at the pre-topic detection level, both in terms of concept space coverage and of mappings coverage, as well as (b) its completeness regarding the non-taxonomical relations between concepts and their topics and the overall efficiency of the topic labelling module.

Results show a very good performance validating the service's efficiency in all these aspects, with a room for improvement. Given however the fact that one of LUMO's main goals is to achieve an optimal trade-off between coverage and maintaining the ontology lightweight, the results are deemed satisfactory and not much further extension is to be pursued.

The second use case consisted of evaluating topic labels retrieved based on a large set of automatically annotated content. Out of a pool of 970 content items of RBB news segments, 50 items were selected randomly as the test set and annotated automatically. The automatic annotation of this particular dataset consisted only of processing the audio track of the videos through ASR analysis, therefore did not undergo the full WP2 processing pipeline (e.g. analyzing textual transcripts). In addition, a "golden standard" set of manual annotations of the same 50 items was prepared as ground truth. Although not undergone the full WP2 analysis, this experiment can still demonstrate the completeness and efficiency of the topic labelling module, especially looking at the improvement over the precision of the topics retrieved as compared to the precision of the automatic annotation (Table 9).

The low scores of the automatic annotations as compared to the ground truth annotations (Table 9), and their direct correspondence to the topic labelling scores for this test case, verify that the low performance of the topic labelling service is a direct consequence of the quality of annotations. In other words it is not due to significant misses in the topic labelling pipeline, i.e. lack of relevant semantics in the LUMO ontology, lack of relevant mappings in the LUMO mappings vocabulary, errors in the automatic creation of the test set's LUMO-based content profiles or errors in the LiFR-based semantic inferencing. In contrary, comparing the precision between the automatic annotation evaluation (0.29) and the precision of detected topics (0.33), we observe an improvement that presumably fortifies the decision to restrict affected services in the LUMO "world", as it appears that omission of information out of this "world" eliminated some metadata that were probably irrelevant to the content (potentially noise).

Table 8: *Average precision, recall, f-measure of topic labelling for the first two test cases.*

Topic Labelling	**Precision**	**Recall**	**F-Measure**
Use case 1: manual annotations	0.895136986	0.8865753	0.874246575
Use case 2: 50 RBB videos	0.337642002	0.437047619	0.348962268

Table 9: *Average precision, recall, f-measure of the automatic annotation of 50 RBB videos in comparison with the ground truth annotations.*

	Precision	**Recall**	**F-Measure**
Use case 2: 50 RBB videos	0.293180596	0.461643468	0.350653773

The third use case consisted of labelling chapters of the seed content of the year 3 scenarios. This consisted of a much smaller set than the previous test case: 4 chapters for the RBB scenario and 9 chapters for the S&V scenario. This content was annotated automatically, after having undergone the entire WP1-WP2 analysis pipeline. The curated annotations of the corresponding content in LinkedTV's editor tool were used as ground truth in the evaluation of this task. For this test case, the experiments were performed based on LUMO v2[27], before the completion of the arts-specific LUMO expansion (re the S&V scenario).

The results in Table 10 are reported separately for the RBB and S&V (TKK content) scenarios. In the RBB case, the results verify the good performance of the topic labelling module and as a reflection the good performance of the WP1-WP2 annotation tools. In the TKK case, given the previously concluded high quality of annotations and the fact that this experiment was performed based on LUMO v2 and not LUMO-arts, we can deduce that the poorer performance is a reflection of the lack of domain-specific semantics in the background knowledge (LUMO) that this scenario requires. With the creation of LUMO-arts, a next round of experiments is planned, where significant improvement is expected.

Table 10: *Average precision, recall, f-measure of topic labelling for the third test case, for the RBB and the TKK scenarios content respectively.*

Topic Labelling	**Precision**	**Recall**	**F-Measure**
Use case 3: RBB content	0.833333333	0.711111111	0.738461538
Use case 3: TKK content	0.520426487	0.474074074	0.475753567

At the time that these experiments were conducted, an adequate enrichment test set was not available, but evaluation of topic labelling for enrichments is now under way.

[27]The core LUMO ontology is more general and thus oriented towards the more generic news scenario

6 Summary

In this deliverable, we have presented a suite of tools that can enrich seed video programs while providing also annotations of those enrichments for further personalisation. As foreseen in D2.3, these tools make use of different resources available on the web: i) web sites from a curated white list for which tailored crawling and indexing processes have been developed; ii) APIs exposed by social networks and encyclopedia such as Europeana; and iii) public search services such as Google Custom Search Engine. The enrichment content is initially annotated using multiple ontologies, such as the DBpedia Ontology, YAGO, schema.org or the NERD ontology. To support the personalisation and contextualisation services developed within WP4, the annotations of seed and enrichment content are exposed and mapped to WP4's LUMO ontology via its mappings vocabulary.

Chapter 2 presented the developed infrastructure for enabling the automatic communication of the set of analysis tools developed by WP2 with the LinkedTV platform. The enrichment process is triggered by calling the *TVEnricher* service. Following the cascading integration approach, the TVEnricher serves as a single service end point but in itself covers the individual WP2 sub-services. The enrichment results are made available for client applications via the Platform REST API.

Chapter 3 describes the IRAPI Module, which is a WP2 component for crawling and indexing web sites from the white list curated by the content partners. This chapter describes the three major advances over the work reported in D2.5: the Metadata Extraction Service which combines probabilistic wrapper approach with focused crawling to improve the retrieval of video content, the annotation of enrichment content with entities using the THD service, and the dashboard utility for monitoring and controlling the white list.

Chapter 4 described the main LinkedTV enrichment services, namely TVEnricher for the LinkedCulture scenario and TVNewsEnricher for the LinkedNews scenario. The second service makes use of a so-called named entity expansion service that enables to get a wider context for a news story and therefore, enables to trigger much focused enrichments.

Chapter 5 presents the mappings between the vocabularies used in WP2 and the WP4 LUMO ontology and the process of conveying WP2 annotation to LUMO. The mappings have been extended and revised to reflect the updated version 2 of the WP4 ontology. The extension includes mappings to three new vocabularies, on top of the most recent versions of the previously supported ones, to support the full spectrum of content annotations provided by the WP2 entity enrichment services. Furthermore, it presents and evaluates an added annotation functionality, that of labelling enrichments (as well as media fragments of seed content) with topics from the LUMO ontology, which can be used for thematic categorization of the content.

The enrichment tools developed within WP2 were evaluated using content from the consortium partners and by participating at international benchmarking activities. The results from the NIST TAC 2013 entity linking contest are reported in this deliverable. The results of the MediaEval 2014 benchmark as well as the NIST TAC 2014 entity linking will be reported in the upcoming WP2 deliverable D2.7. Additional results on scenario content can be found in the WP6 deliverable D6.4.

In conclusion, WP2 offers a rich set of enrichment modules that can efficiently address the enrichment requirements both on document and entity level for each of the LinkedTV scenarios.

References

[1] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. DBpedia - A crystallization point for the Web of Data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154 – 165, 2009.

[2] LinkedTV consortium. Deliverable 2.3: Specification of web mining process for hypervideo concept identification, 2012.

[3] LinkedTV consortium. Deliverable 2.4: Annotation and retrieval module of media fragments, 2013.

[4] LinkedTV consortium. Deliverable 2.5: Specification of the linked media layer, 2013.

[5] LinkedTV consortium. Deliverable 5.4: Final linkedtv integrating platform, 2013.

[6] LinkedTV consortium. Deliverable 1.5: Linkedtv annotation tool - final release, 2014.

[7] LinkedTV consortium. Deliverable 3.7: Linkedtv user interfaces selected and refined - version 2, 2014.

[8] LinkedTV consortium. Deliverable 5.6: Final linkedtv end-to-end platform, 2014.

[9] LinkedTV consortium. Deliverable 6.4: Scenario demonstrators – version 2, 2014.

[10] Laurens De Vocht, Sam Coppens, Ruben Verborgh, Miel Vander Sande, Erik Mannens, and Rik Van de Walle. Discovering meaningful connections between resources in the Web of Data. In *6^{th} Internation Workshop on Linked Data on the Web (LDOW'13)*, 2013.

[11] Milan Dojchinovski and Tomáš Kliegr. Entityclassifier.eu: real-time classification of entities in text with Wikipedia. In *ECML'13*, pages 654–658. Springer, 2013.

[12] Milan Dojchinovski, Tomáš Kliegr, Ivo Lašek, and Ondřej Zamazal. Wikipedia search as effective entity linking algorithm. In *Text Analysis Conference (TAC) 2013 Proceedings*. NIST, 2013.

[13] Milan Dojchinovski and Tomás Kliegr. Datasets, gate evaluation framework for benchmarking wikipedia-based ner systems. In Sebastian Hellmann, Agata Filipowska, Caroline Barrière, Pablo N. Mendes, and Dimitris Kontokostas, editors, *NLP-DBPEDIA Workshop*, 2013.

[14] D. W. Embley, C. Tao, and D. W. Liddle. Automatically extracting ontologically specified data from html tables of unknown structure. In *ER 2002*, pages 322–337, London, UK, 2002.

[15] M. Eskevich, J Gareth J.F., S. Chen, R. Aly, and R. Ordelman. The Search and Hyperlinking Task at MediaEval 2013. In *MediaEval 2013 Workshop*, Barcelona, Spain, 2013.

[16] Evgeniy Gabrilovich and Shaul Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *20th International Joint Conference on Artifical intelligence (IJCAI'07)*, pages 1606–1611, Hyderabad, India, 2007.

[17] José Luis Redondo Garcia, Michiel Hildebrand, Lilia Perez Romero, and Raphaël Troncy. Augmenting TV Newscasts via Entity Expansion. In *11^{nd} Extended Semantic Web Conference (ESWC'14), Demos Track*, Anissaras, Crete, 2014.

[18] José Luis Redondo Garcia and Raphaël Troncy. Television meets the Web: a Multimedia Hypervideo Experience. In *12^{th} International Semantic Web Conference (ISWC'13), Doctoral Consortium Track*, Sydney, Australia, 2013.

[19] José Luis Redondo Garcia, Laurens De Vocht, Raphaël Troncy, Erik Mannens, and Rik Van de Walle. Describing and Contextualizing Events in TV News Show. In *2^{nd} International Workshop on Social News on the Web (SNOW'14)*, pages 759–764, Seoul, South Korea, 2014.

[20] Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, and Gerhard Weikum. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence*, 194:28–61, 2013.

[21] Tomáš Kliegr, Krishna Chandramouli, Jan Nemrava, Vojtěch Svátek, and Ebroul Izquierdo. Combining image captions and visual analysis for image concept classification. In *9th International Workshop on Multimedia Data Mining*, pages 8–17, Las Vegas, Nevada, 2008.

[22] Tomáš Kliegr and Ondřej Zamazal. Towards Linked Hypernyms Dataset 2.0: complementing DBpedia with hypernym discovery and statistical type inferrence. In *9th International Conference on Language Resources and Evaluation, LREC 2014*, 2014.

[23] Tomáš Kliegr, Václav Zeman, and Milan Dojchinovski. Linked hypernyms dataset - generation framework and use cases. In *3rd Workshop on Linked Data in Linguistics: Multilingual Knowledge Resources and Natural Language Processing, co-located with LREC 2014*, 2014.

[24] M. Labský and V. Svátek. Combining multiple sources of evidence in web information extraction. In *International Symposium on Methodologies for Intelligent Systems*, 2008.

[25] Li-Jia Li and Li Fei-Fei. What, where and who? classifying events by scene and object recognition. In *IEEE 11th International Conference on Computer Vision (ICCV'07)*, pages 1–8, 2007.

[26] Yunjia Li, Giuseppe Rizzo, José Luis Redondo Garcia, and Raphaël Troncy. Enriching media fragments with named entities for video classification. In *1st Worldwide Web Workshop on Linked Media (LiME'13)*, Rio de Janeiro, Brazil, 2013.

[27] Bing Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage, 2nd. ed. Data*. Springer, 2011.

[28] Joshua L Moore, Florian Steinke, and Volker Tresp. A novel metric for information retrieval in semantic networks. In *The Semantic Web: ESWC 2011 Workshops*, pages 65–79, 2012.

[29] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: bringing order to the web. Technical report, Stanford InfoLab, 1999.

[30] Giuseppe Rizzo and Raphaël Troncy. NERD: A Framework for Unifying Named Entity Recognition and Disambiguation Extraction Tools. In *13th Conference of the European Chapter of the Association for computational Linguistics (EACL'12)*, 2012.

[31] Dorothea Tsatsou, Stamatia Dasiopoulou, Ioannis Kompatsiaris, and Vasileios Mezaris. LiFR: A Lightweight Fuzzy DL Reasoner. In *11th ESWC 2014 (ESWC2014)*, May 2014.

[32] Dorothea Tsatsou and Vasileios Mezaris. LUMO: The LinkedTV User Model Ontology. In *11th ESWC 2014 (ESWC2014)*, May 2014.

[33] William E Winkler. Overview of record linkage and current research directions. In *Bureau of the Census*, 2006.