Deliverable D4.5 Content and Concept Filter v2

Daniel Stein, Rüdiger Klein, Manuel Kober, Jingquan Xie / Fraunhofer
Dorothea Tsatsou / CERTH
Tomáš Kliegr, Jaroslav Kuchař / UEP
Daniel Ockeloen, Pieter van Leeuwen / Noterik

13/11/2013

Work Package 4: Contextualisation and Personalisation

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	30/09/2013
Actual date of delivery	13/11/2013
Deliverable number	D4.5
Deliverable name	Content and Concept Filter v2
File	`content_conceptFilter_D4-5.tex`
Nature	Report
Status & version	Final & V1.2
Number of pages	31
WP contributing to the deliverable	4
Task responsible	Fraunhofer
Other contributors	
Author(s)	Daniel Stein, Rüdiger Klein, Manuel Kober, Jingquan Xie / Fraunhofer Dorothea Tsatsou / CERTH Tomáš Kliegr, Jaroslav Kuchař / UEP Daniel Ockeloen, Pieter van Leeuwen / Noterik
Reviewer	Matei Mancas / UMons Jan Thomsen / Condat
EC Project Officer	Thomas Küpper
Keywords	Content filtering, concept filtering, implicit preferences, explicit preferences, topic segmentation/labeling quality
Abstract (for dissemination)	Content annotation and enrichment within LinkedTV produces arbitrarily large amounts of quality links to the web, which on the one hand shows the potential of the involved algorithms, but on the other hand can be overwhelming for a single user if not filtered and prioritized beforehand. In this deliverable, we present our approaches to rank and filter these links based on a user's interest. We offer solutions for implicitly learned interests, and for explicitely given preferences, by exploiting the user-centered ontologies as defined in Deliverable D 4.4. Further, we explore ranking mechanisms directly based on the entities derived in the annotation and enrichment process. Finally, we offer quantitative and qualitative experiments and assessments on data drawn from the news broadcast scenario.

0 Content

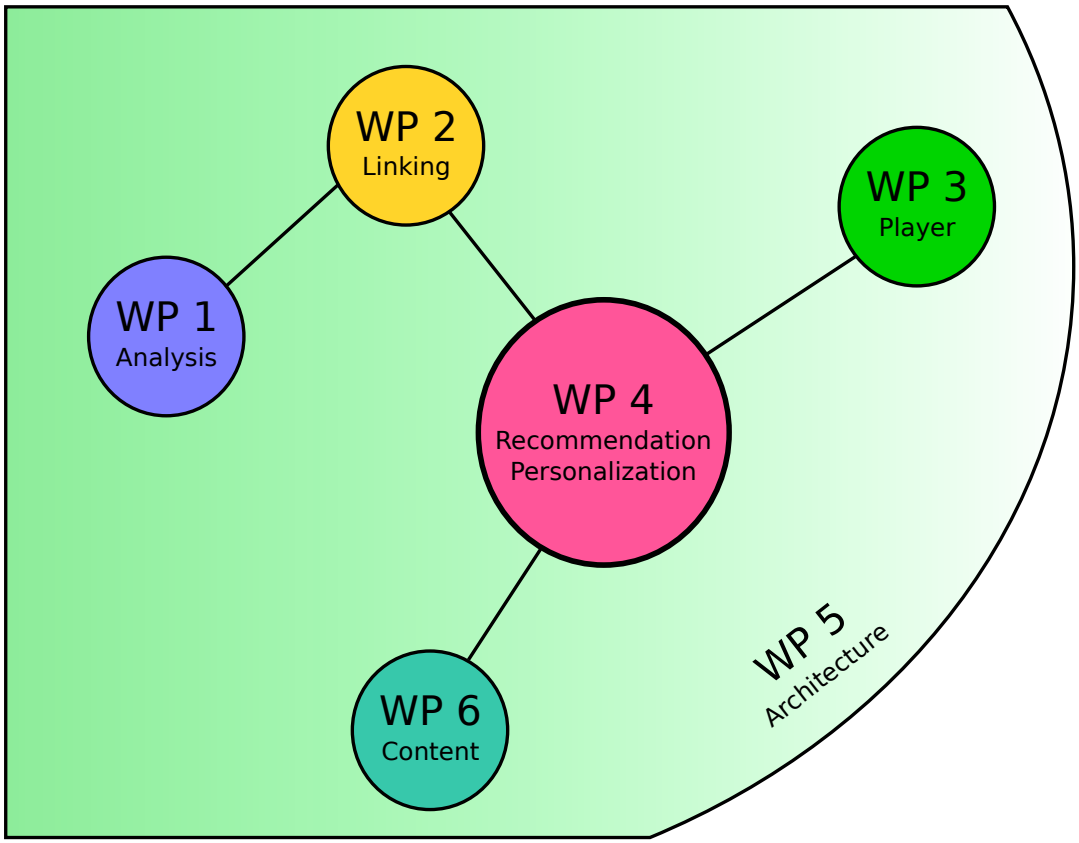

Figure 1: General architecture of work packages within LinkedTV

1 Introduction

Content annotation and enrichment within LinkedTV produces arbitrarily large amounts of quality links to the web, which on the one hand shows the potential of the involved algorithms, but on the other hand can be overwhelming for a single user if not filtered and priorized beforehand. Based on the user's semantic profiles, learned and captured both implicitly (by monitoring and aggregating his behavior) and explicitely (by allowing to specify concrete interests), we are able to personalize and contextualize the content that will be offered.

The work package on personalization and contextualization arguably has the highest interdependence with other work packages, as it maintains semantic user modeling based on the content annotation – as part of the first work package (WP 1) – and the enrichment output (WP 2). The scenarios as defined in WP 6 emphasize the use cases and demands from an archetype point-of-view. The filtered content has to be communicated to the LinkedTV media player (WP 3), whereas user actions and changes in the user's context need to be communicated back into our services and tools. Last but not least, the tools provided by this work package need to be embedded into the LinkedTV platform (WP 5). See Figure 1 for a graphical representation of this general architecture.

The core position of this work package within the LinkedTV architecture poses risks and opportunities. The risks include that our algorithms have had access to real-life data only at a later stage of the project. Not only need the videos be annotated for their content and then enriched in a common pipeline, but the amount of data processed also needs sufficient size in order to evaluate the filtering properly; a first small batch of proof-of-concept enriched hyperlinks from seed videos do not yet allow for quantitative filtering results. Further, the scenario descriptions establishing the archetypes need to be aligned to the vast possibilities and opportunities that a personalization and contextualization service can offer to the user's viewing experience. Last-not-least, this work package arguably has the highest demands in privacy concerns, since here sensible user data is aggregated and processed.

The opportunities include that many different experimental branches and approaches to personalization can be implemented and evaluated under ideal lab conditions, so that once all LinkedTV data hoses are connected we can choose from a large set of algorithms and services. While it is true that

lab conditions often do not generalize well to real-life conditions, the annotation and the enrichment work packages already have seen remarkable improvements over their baseline models; further, we are communicating the necessities of our algorithms back to the respective work packages so that crucial aspects can be improved for a stronger overall work-flow.

Whereas deliverable D 4.4 "User profile and contextual adaptation" describes how we build and maintain semantic user models, this deliverable D 4.5 focuses on the aspect of content and concept filtering based on these models. Similar to D 4.4, we allow for individual approaches to content filtering, in order to exploit various strength and weaknesses. These approaches are: filtering based on semantic annotations without relations, relational filtering including semantic relations, Description Logic Programming (DLP) based filtering, and filtering based on learned preference rules.

While year 2 within the LinkedTV project has witnessed remarkable work-flow integration and improvements in content annotation and enrichment, we are now at a stage where we can proceed to work on the whole pipeline, exploit synergies in our approaches and merge the most promising technologies into singular assets. A first quality assessment of the data derived from raw data is included in this deliverable.

This document is structured as follows: we first embed the work presented here into the general LinkedTV work-flow and review the internal structure of this work package (Section 2). Then, we describe our three approaches for content filtering (Section 3). We offer qualitative and quantitative experiments in Section 4 and conclude this deliverable in Section 5.

1.1 History of this document

The description of work foresees two deliverables of this work package to be submitted simultaneously, i.e., both this deliverable and D 4.4. Internal review raised awareness towards different vocabulary and inconsistencies with the described work-flow. It was thus decided to include major revisions into this document in order to strengthen the readability of this deliverable in conjunction with the other LinkedTV contributions. Note that this is mainly a matter of LinkedTV glossary and internal structuring, and that the strength of the technical contribution was unaffected. See Table 1 for an overview of the versioning.

Table 1: History of this document

Date	Version	Name	Comment
2013/08/26	v 0.1	Rüdiger Klein	1st draft
2013/09/10	v 0.2	Dorothea Tsatsou Tomas Kliegr	Contributions from CERTH and UEP
2013/09/23	v 0.9	Rüdiger Klein	Final draft paper ready for review
2013/10/07	v 1.0	Rüdiger Klein	Changes according to QA
2013/10/08	v 1.01	Martha Merzbach	Layout changes
2013/10/20	v 1.1	Rüdiger Klein	Changes according to scientific coordinator
2013/11/04	v 1.2	Daniel Stein	Heavy adaptation to D 4.4

1.2 Related deliverables

As stated above, this work package on personalization and contextualization is at the core of the overall LinkedTV work-flow. This is reflected by the comparably large amount of related documents that affect this deliverable.

Strongly related are:

- **D 4.3**, where year 1's efforts towards content and concept filtering within LinkedTV have been described. This deliverable builds upon the methods and findings described there.
- **D 4.4**, where the semantic user models are described, and where the process of capturing and aggregating the user preferences, both implicitly and explicitely, is described in details. It is these user models that form the basis for filtering as described in this deliverable.
- **D 2.4**, where the connection between the reference knowledge base and the content annotation is described. Both D 4.4 and this deliverable work with this reference knowledge base and are thus directly dependent on the ties and qualities with the content annotation.

Other deliverables that have connections and interdependencies with this deliverable include:

- **D6.1**, which describes the use case scenarios. It is upon this common vision of archetypes and their expectations that we have to tailor our approaches and solutions.
- **D3.6**, which describes the LinkedTV media player. The filtered and priorized enrichment will have to be communicated to the player, whereas the player maintains the user database and the logging in; it also aggregates and communicates changes in the contextual environment of the user.
- **D5.4**, which describes the LinkedTV platform. It elaborates on the integration and architecture backbone which embeds the tools presented in this deliverable.
- **D7.4**, which describes the demonstrators and the APIs of the services, and thus include the tools presented here.

1.3 Privacy

The project partners involved in this work package are quite aware that within their work proper treatment and respect for privacy, ethics and intrusiveness is a key requirement. In year 2 we have kept experiments internal, without any real user involvement. Since this is to change in year 3, we will (a) use hashes of the user id for anonymity and (b) use secure and encrypted connections whenever sensible data is transferred between remote services. In the current work-flow, the transmitted data is secured via the HTTPS protocol. The f-PocketKRHyper (Section 3.2) has very limited need for data storage and can thus be stored locally on the client. The homogeneous approach of LSF (Section 3.1) allows us to describe user interest models in a very concise form as well, and such compact UIM can be built and maintained on a local computer under full control of the user. Only the active user model consisting of a subset of all user interests has to be submitted to the filtering engine on the media host and will be anonymized via hashes. Similarly, the In-Beat Recommender (Section 3.3) only uses anonymized hashes of the user so that the aggregated data can not be traced back to its owner.

Any test user involved in upcoming evaluations will be given a full explanation of the purpose, the scope and the longevity of his content. Should any access to the services from outside continue to exist for a longer period than a self-contained evaluation time frame (i.e., longer than one week), any user is required to opt-in in regular intervals (e.g., every month) were he has to restate his permit of the data usage.

2 Personalization and Contextualization Work-flow

In this section, we will review any technology and vocabulary that is needed for the understanding of the content and concept filtering presented throughout this deliverable. To avoid redundancy, we keep these descriptions to a bare minimum, and refer the interested reader to the in-depth deliverables as listed in Section 1.2. Consequently, in Section 2.1, we give a short overview of all components and services within this work package that are not the core contribution of this deliverable, but rather belong to D 4.4. In Section 2.2, we will introduce the role of the components directly related to concept and content filtering, on a bird's eye perspective.

2.1 User Profile and Contextual Adaptation

To enable personalization and contextualization, we need to have a good estimation of a users interest and preferences.

First of all, we need to decide whether we want to capture the user interests *implicitly* or *explictly*, or by using a combination of both. Implicit user interests are captured via tracking user actions in the real-world (via a camera, e.g., Kinect) and his interaction with the media player. In the LinkedTV work-flow, real-world tracking is conducted via the Attention/Context tracker (D 4.4, Section 5.1.1), and passed on as an event into the media player. Together with user interaction directly related to the player, this information is then passed into GAIN (D 4.4, Section 3.1). Next, we need to tie these informations into a representation of the user preferences, where we face the following sub-challenges:

(i) First, we need a semantic annotation of media fragments, i.e., a set of semantic expressions approximating the media fragment content with reasonable quality (in terms of precision and recall, but also in terms of type coverage).

(ii) The second sub-challenge faced is a knowledge base describing our "knowledge about the world", i.e., those aspects not directly mentioned in the media fragment but describing its content in a broader sense (e.g., roles certain people play in the world, which organizations they belong to, architects of a building, locations, time periods, . . .).

(iii) The third sub-challenge is employing an ontology reflecting the user interests in order to capture the semantics of both the annotations and the knowledge base.

(iv) Finally, the concrete user interests have to be formulated as semantic expressions using these ontologies.

The semantically enriched annotation described in (i) is the main responsibility of WP 2. However, the demands of this work package 4 is constantly back-propagated to their respective WP 2 services, and the quality of the enrichment output needs to be guarded carefully in order to estimate whether erroneous output of this work package is due to a propagation of enrichment and/or annotation mistakes or whether there is room for improvement in our own algorithms. In general, relevance and confidence scores passed on in WP 2 should be taken into account. In this section, we treat the output of WP 2 as a black-box and take it as-is, and refer to deliverable D 2.4.

The semantic web as described in (ii) can rely on existing and well-established web sources either broad like DBPedia[1], Freebase[2], or YAGO2[Hoffart et al., 2013]; or more focused like Geonames[3] or the Internet Movie Database (IMDB)[4]. Current output of WP 2 supports DBPedia, Wikipedia, Freebase and Yago. In Section 4.2, we will investigate the current quality of a reasonable sized, enriched data set. We also introduced our own built knowledge base called LUMOPedia, which has been described in D 4.4, Section 6.1.1.

The ontology that reflects the user interests (iii) is introduced via the LUMO approach. Currently, there are two research track in this WP 4 that model this key element. On the one hand, LUMO-v1 (D 4.4, Section 2.1) is designed for convergence between implicit user feedback and its semantic interpretation. On the other hand, LUMO-rel (D 4.4, Section 6.1) has a better convergence with explicit preferences, i.e., preferences expressed manually by the user. LUMO-v1 and LUMO-rel have been extensively discussed in D 4.4. Both ontologies are tightly related and their merging will be a major effort in year 3.

[1] dbpedia.org
[2] freebase.org
[3] geonames.org
[4] imdb.com

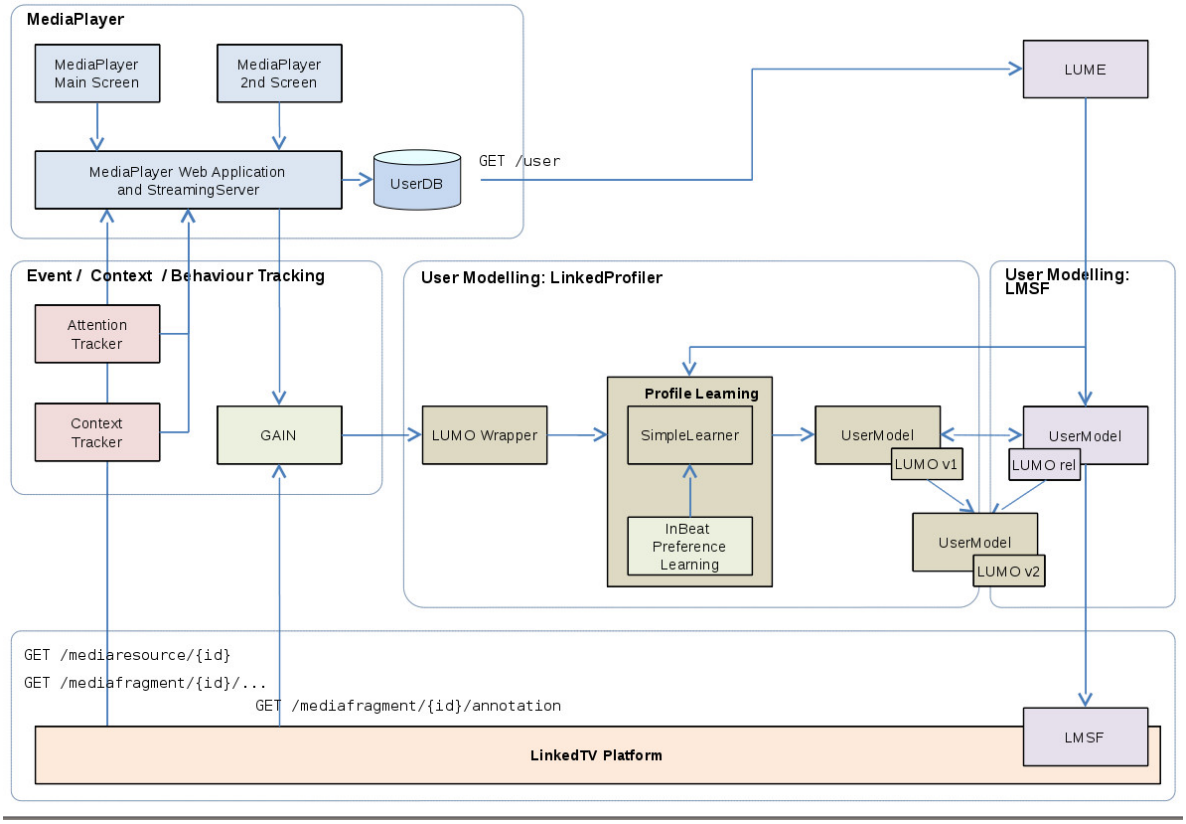

Figure 2: Work-flow of the user and context capturing and the user modeling within LinkedTV

Capturing the user interests (iv) is then part of the LinkedProfiler (D 4.4, Section 4), for implicit learning. Here, information aggregated by GAIN is processed further, by several sub-processes, namely LUMO Wrapper (D 4.4, Section 4.1), ProfileLearning (D 4.4, Section 4.2), SimpleLearner (D 4.4, Section 4.2.1) and EasyMiner (D 4.4, Section 4.2.2), to produce a user model that can be used for content and concept filtering.

Explicit user preferences can be captured in a graphical user interface as provided by LUME (D 4.4, Section 6.2), but basically needs the same information as implicit user interest capturing.

See Figure 2 for an architecture overview.

2.2 Content and concept filtering

In this subsection, we will introduce the main filtering techniques that are applied within LinkedTV from a bird's-eye view (cf. Figure 3). In Section 3, we will describe the tools and services in detail.

LSF, as a technical implementation of the LinkedTV Semantic Filtering, aims to provide an efficient and scalable system for content and concept filtering, based on user models. While in principal capable of working with arbitrary OWL-based ontologies and their derived user models, LSF's research focus has been focusing on the user models created in the LUMO-rel ontology, i.e., work with explicitly given user interests. See Section 3.1 for details on the implementation.

f-PocketKRHyper aims in principle at the same goals but has a much stronger focus on implicit learning, consequently relying on the LUMO-v1 ontology. It builds upon DLP based filtering and is an extension of the Pocket KRHyper mobile reasoner [Kleemann, 2006]. See Section 3.2 for more details.

A special role plays the In-Beat Recommender as part of the EasyMiner solution, which was briefly described in D 4.4, Section 4.2.2.1. Since part of its components are also able to rank a list of enriched content, its functionality is also part of this deliverable, and will be further explained in Section 3.3 below.

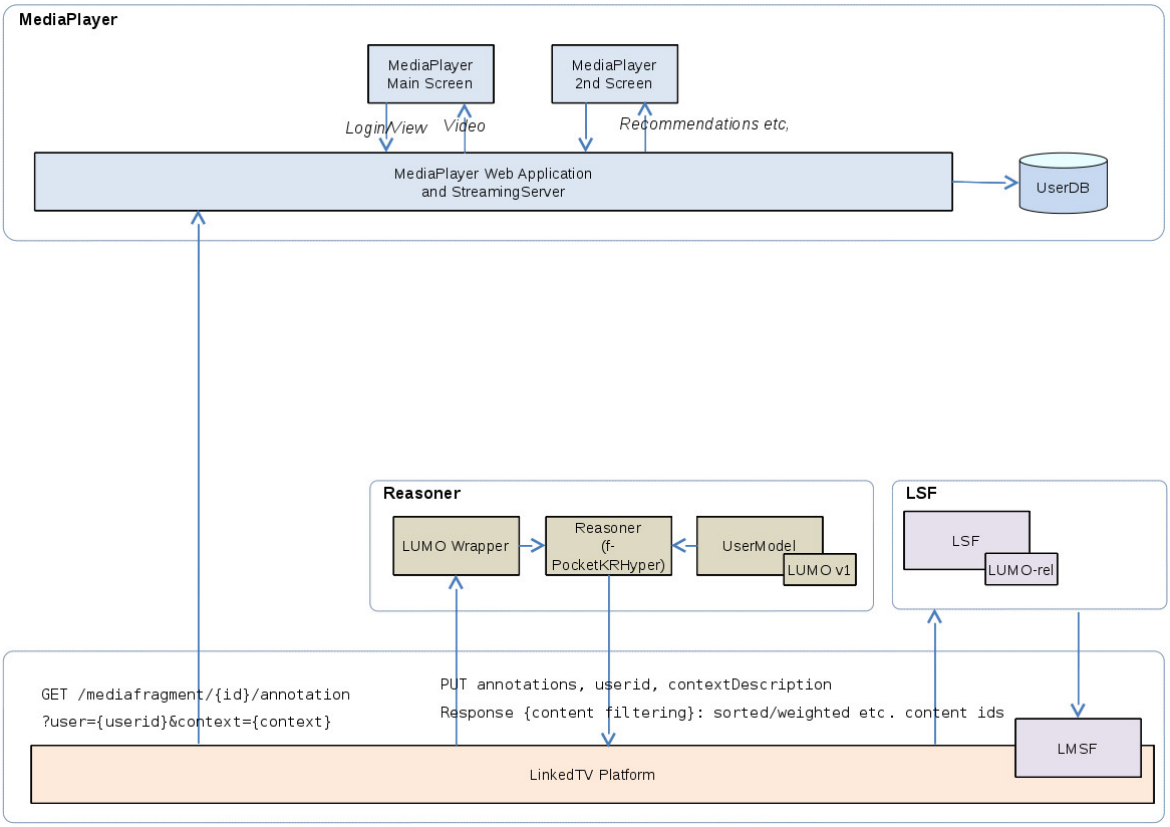

Figure 3: Work-flow of the content filtering and adaptation

3 Content and Concept Filtering

In this section, we will describe the three filtering approaches that we maintain within LinkedTV. The approaches all have their focus on other aspects, but their ranking can be combined for an overall filtering. Currently though, the following subsections describe parallel solutions, whereas their merging will be a major effort in year 3 (cf. Section 5).

Throughout this section, we will denote $\exists$ for "some", "exists", $\forall$ for "forall", $\sqcup$ and $\vee$ for "or", $\sqcap$ and $\wedge$ for "and", $\sqsubseteq$ for "implies" and $\bot$ for "bottom".

3.1 LSF

The core function of LSF is the ranking and filtering of media fragment annotations in a semantic way, with special focus on explicit user feedback. We proceed to describe the work-flow in detail. In the following, we mainly refer to the LUMO-rel ontology where the main focus was on, but LSF works with LUMO-v1 as well.

3.1.1 Work-flow

LSF takes as input an active user interest model (aUIM) and a set of enriched content (enriched media fragment annotations; eMFA). The aUIM is the active part of the user model to be used for filtering at a given moment. All other contexts are "inactive" and are not considered in the filtering process. Currently, the aUIM is selected by the user. In the future, user observation techniques and preference learning will be used to find out which parts of the user models have to be used in a given situation. An eMFA corresponds to a media fragment (MF) which is a part of a complete video and describes a specific topic. Each fragment (shot, scene) contains enriched hyperlinks on top of (automatic) content annotation, but also describes additional information that goes beyond the directly visible entities: the concepts they belong to, related entities, etc.

LSF outputs normalized rankings for these enriched contents (i.e., between 0 and 1) by incorporating knowledge from the LUMO-rel ontology and LUMO-Pedia knowledge base. Further, LSF provides a

graphical user interface for research purposes that directly shows the output of the ranking.

A detailed discussion about user interest models is given in D 4.4, which is why we keep the aUIM description short. In principle, an aUIM contains a set of user interests, and each user interest can be associated with a set of constraints, e.g., "politician *leads Germany*". The enriched content consists of textual representations of entities or concepts within a media fragment, e.g., "politician", "Obama" or "Berlin", which are for example captured via named entity recognition, and which are further enriched by mappings to knowledge bases such as DBPedia. These entities are then adapted and aligned with LUMO-rel and LUMO-Pedia, which serves as the common information backbone in LSF for semantic matching and ranking. The objective of LUMO-Pedia is to construct a comprehensive knowledge based for common daily knowledge in a structure way and it must be aligned to the LUMO-rel ontology. Similar work is the Freebase and Yago dataset. Note that these datasets are not compatible to the LUMO-rel ontology; actually most of the data in LUMO-Pedia is and will be imported from these datasets.

Bundling these informations, the actual graph matching algorithm then performs a comparison of the two graphs as spanned by the user models and the enrichment, aggregates the ranking and outputs the final ranking score.

3.1.2 Reasoning

To find out if a MF fulfill a user interest means to find out if this user interest (UI) is semantically entailed by the eMFAs of this video and the LUMO-rel ontology. Depending on the concrete shape of the used LUMO-rel ontology, the eMFA, and user interests the reasoning techniques used to determine entailment can be kept simple and efficient.

3.1.2.1 Simple Filtering The Simple Filtering approach is used for those cases where the eMFA do not contain any relational information. They just contain named and unnamed entities (ne_i) with their LUMO-rel types, and they may contain LUMO-rel concepts c_j and LUMO-rel topics t_n:

$$\text{eMFA} = \{\mathsf{ne}_1, \mathsf{ne}_2, \ldots, \mathsf{ne}_m, \mathsf{c}_1, \ldots, \mathsf{c}_k, \mathsf{t}_1, \ldots, \mathsf{t}_l\} \tag{1}$$

Accordingly, we restrict user interests u_j to the same representational means (where all entities are named ones from LUMO-Pedia). We denote a sub-topic relationship between two topics t_i and t_j as $t_i \leq t_j$, a sub-concept relationship between two concepts c_i and c_j as $c_i \sqsubseteq c_j$ and the instance relation between a named entity e and a concept a as $e : a$.

The filtering algorithm handles the following equation:

- A user interest u_j is entailed by the eMFA of a video/MF if each of the expressions in the UI is entailed by the eMFA (Eqn. 2)
- A named entity ne_i is entailed by the eMFA if the eMFA contains this named entity (Eqn. 3)
- A topic t_i is entailed by the eMFA if it contains this topic or one of its sub-topics in the LUMO ontology (Eqn. 4)
- A concept c_i is entailed by the eMFA if it contains this concept, one of its LUMO sub-concepts, or a (named) entity belonging to this concept (Eqn. 5)

$$eMFA \models_{\text{LUMO-rel}} u_j \quad \text{iff} \quad \forall e_j \in u_j : \text{eMFA} \models_{\text{LUMO-rel}} e_j \tag{2}$$
$$eMFA \models_{\text{LUMO-rel}} \mathsf{ne}_i \quad \text{iff} \quad \mathsf{ne}_i \in \text{eMFA} \tag{3}$$
$$eMFA \models_{\text{LUMO-rel}} t_i \quad \text{iff} \quad \exists t_j \in \text{eMFA} \wedge (t_j = t_i \vee t_j \leq t_i) \tag{4}$$
$$eMFA \models_{\text{LUMO-rel}} \mathsf{c}_i \quad \text{iff} \quad \exists e_j \in \text{eMFA} \wedge (e_j = c_j \vee e_j \sqsubseteq c_i \vee e_j : c_i) \tag{5}$$

The equations can be checked quite efficiently, which enables fast filtering.

3.1.2.2 Semantic Filtering with Relational Expressions Now let us assume that the user interests as well as the eMFA contain relational expressions as outlined above. Denoting a_i as LUMO-rel concepts/topics or LUMO-Pedia named entities, and rel_i as LUMO-rel relations, the eMFA then has the shape:

$$\text{eMFA} = \left\{a_1, \text{rel}_j(a_1,a_j), \text{rel}_k(a_1,a_k), \ldots, a_n, \text{rel}_l(a_k,a_l), \text{rel}_m(a_k,a_m), c_1, \ldots, c_j, t_1, \ldots, t_x\right\}, \quad (6)$$

i.e., a set of (named) entities with relations to other entities, as well as a set of concepts and topics. Accordingly, we restrict user interests to the same representational means (where all concrete entities are named ones from LUMO-Pedia):

$$u_i = a_i \sqcap \exists \text{rel}_j.a_j \sqcap \ldots \sqcap \exists \text{rel}_n.a_n\,. \quad (7)$$

The relational filtering further includes Eqns. 2–5, and further a relational expression $a_i.\exists \text{rel}_j.a_j$ in the UI, which is entailed by the eMFA if the eMFA contains an expression $\text{rel}_j(X,Y)$ and X entails the anchor term a_i and Y the a_j:

$$eMFA \models_{\text{LUMO-rel}} \exists \text{rel}_j.a_j \quad \text{iff} \quad \text{rel}(X,Y) \in \text{eMFA} \wedge X \models_{\text{LUMO-rel}} a_i \wedge Ye_j \models_{\text{LUMO-rel}} a_j \quad (8)$$

This reasoning can straightforwardly be extended towards more complex user interests (with conjunctions and disjunctions, relational concatenations, data properties and constraints on data in user interests, etc.). The current version is mainly determined by the restrictions we have on the user interest and the MF annotation side, and by efficiency considerations. The more complex the expressions are the more complex is the reasoning.

3.1.2.3 Weights and ranking In order to allow users to express their interests with varying degrees we assign weights to user interests. Currently, we are using real numbers in the interval [0,1] in order to assign weights to user interests. Additionally, we can explicitly express "dislike" – with the meaning that every MF matching with a "dislike" user interest is filtered out – regardless how well it matches with other UI.

The weights are used to rank all those MF which fulfill one of the user interests in a user model: each MF fulfilling a user interest is weighted with the highest weight of a matching user interest. In this way, the set of all filtered MF can be ranked according to the weighted user interests. The filtering procedure decides if a MF fits any of the user interests. The weights of user interests and MF annotation elements decide about the degree of interest a user may have in a MF. There can be different ways to determine this degree of interest. We propose one which takes user interest weights and MF weights into account in a simple weight computation:

For each user interest u_j in the active UIM we compute if it is entailed by the eMFA. If it is entailed we determine the supporting eMFA (eMFA+) as a minimal set of those eMFA elements which are needed for this entailment.

$$\forall u_j \in \text{aUIM} : \text{if} \quad \text{eMFA} \models_{\text{LUMO}} u_j$$
$$\text{eMFA}^+(u_j) \subseteq \text{eMFA with eMFA}^+(u_j) \models_{\text{LUMO}} u_j$$

Each supporting eMFA$^+$ will be assigned a weight which is the average weight $w_0(e_j)$ of its elements $w_0(\text{eMFA}^+(u_j))$ multiplied with the weight of the matching user interest $w(u_j)$:

$$w_0\left(\text{eMFA}^+(u_j)\right) = \frac{1}{N} * \sum w_o(e_j) \qquad \forall e_j \in \text{eMFA}^+(u_j) \qquad \text{and} \quad N = ||eMFA^+(u_j)||$$
$$w(\text{eMFA}^+(u_j), u_j) = w_0\left(\text{eMFA}^+(u_j)\right) * w(u_j)$$

Each eMFA may entail a user interest u_j in more than one way. Each MF is weighted with the maximum weight its eMFA gained in the weighting procedure of all active user interests u_j:

$$w(\text{MF}) = \max\left(\left\{w(\text{eMFA}^+(u_j), u_j) | u_j \in \text{aUIM}\right\}\right)$$

Depending on the user interface the highest ranked MF can be presented to the user. The user may select one of them to watch it. Currently, MF annotations are not weighted. Assigning weights to them can help to represent MF content more precisely because things shown in a shot or scene have different importance in the context of the video (in an interview the interviewed person is normally more important than the journalist or the microphone).

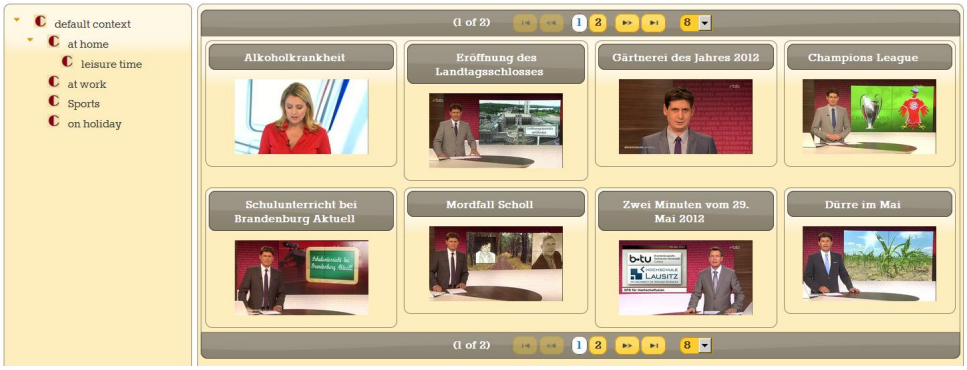

Figure 4: User interface in LSF for the top ranked videos

3.1.2.4 Related content filtering If a user watches a video this user may be interested in related information: about people just shown in the video, objects, locations, etc. Similar information may be also contained in other videos ("related videos") or in other Web content (like Wikipedia, Tumblr, pages found by a (semantic) search engine, or other sources). How much a user is interested in various kinds of related information is described in the aUIM. A user will be supported in managing the related content by ranking it according to this active user interest model.

The weights are computed as described above. For each eMFA element e_j we compute the weight w.r.t. the active user interest model aUIM:

$$\forall e_i \in \text{eMFA}: \quad \text{if} \quad \exists u_j \quad \text{s.t.} \quad e_i \in \text{eMFA}^+(u_j) \quad \text{then} \quad w(e_i,u_j) = w_0(e_j) * w(u_j)$$

where $w_0(e_j)$ is the user independent weight of the eMFA element it got in the annotation procedure as indication of its importance in the context of the media fragment. Those annotation elements which are not contained in any supporting $\text{eMFA}^+(u_i)$ will get the weight 0.

Depending on the user interface the highest ranked annotation elements can be presented to the user. The user may select one of them to get related information. Videos/MF related to the selected annotation element as well as other Web pages (like Wikipedia, Tumblr, etc.).

3.1.3 Frontend

For research purposes, LSF provides extensive functionality for testing and demonstrating its core ranking functionalities. The modules of the front end are: user authentification, top ranked videos, related shots and videos, and related web information.

3.1.3.1 User authentification The LSF front-end is session-based. In order to use the system, an authentification is needed. This is done in the session-scoped managed bean. A servlet filter is created to check for each request if the request is authorized, if not, the user is redirected to the login page. For the RESTful web services, the authentication mechanism is different since no graphical user interface is needed there. The authentication information is embedded in the HTTP header. Each RESTful client has its own API key. The API key is needed for each HTTP request.

3.1.3.2 Top ranked video The "top ranked video" runs as a back-end service and invokes the matching module and applies it to all users, all contexts and all video annotations to generate a rank list of videos for each user and each context. The results are stored in the database for later fast querying. This module is transaction-sensitive since concurrent access can happen and the results should be isolated to each request. The component is implemented in the EJB layer. See Figure 4 for a screenshot.

3.1.3.3 Related Shots and Videos In LSF, each video is structured hierarchically in two layers: the video as a whole and a list of shots which are temporal segmentations of the video. The shots are generated automatically by WP 2. In the LSF front-end, the shots of each video are displayed directly below the video. The current active shot is highlighted with green rectangles. In the front-end, the current active shot and the video annotations belonging to that shot are automatically synchronized as the video

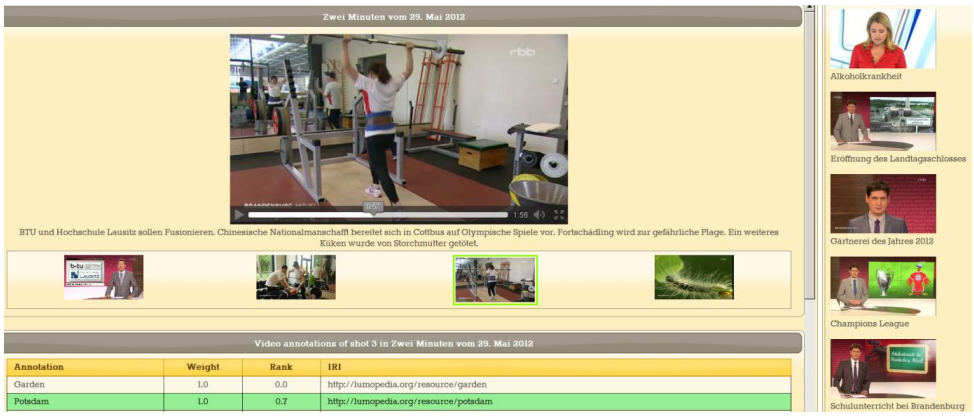

Figure 5: User interface in LSF for the related shots and videos

plays. On the side, the user can jump to a specific shot by clicking on the poster image of that shot. Once the user has clicked on a specific annotation or one of its enrichments, the related shots will be updated on the right. A related shot is chosen based on the following criteria:

- Its rank must be above zero for the current active context
- It must contain the selected annotation
- If the user clicks on a related shot, the video which owns this shot will be loaded and LSF seeks to the start of the shot and starts to play the video.

The “related shots and videos” module is implemented in both the EJB layer and the JSF layer. See Figure 5 for a screenshot.

3.1.3.4 Related Web Information Besides of the related shots, if a user clicks an annotation, related web information is loaded. Currently only Wikipedia and Flickr data are fetched from Internet. Other related web information can however be provided in future versions. The Wikipedia article is fetched based on the title of the respective Wiki page, i.e., if the title is the same as the annotation, then the first 200 words are extracted and displayed as summary, including a link to the Wikipedia page. The Flickr images are fetched based on the image tags given by the Flickr user, i.e. if the tag list contains the video annotation then it will be returned by Flickr. Only the first 16 images are displayed. This is however configurable.

The “related web information” module is coded in the JSF layer and executed on the client side. It is basically developed on top of the Popcorn.js HTML5 framework10. The Javascript code is generated automatically by the JSF Facelet template and sent as a response to the web client. The core part of the template is as follows:

```
<script type="text/javascript">
lsf = {
initPopcorn : function() {
lsf.popcorn = Popcorn("#mainVideo");
var p = lsf.popcorn;
var x = #{model.codeShot};
},
loadweb : function(label, wikiurl) {
lsf.popcorn.flickr({
tags: label,
numberofimages  : "20",
height : "100",
width: "100",
target: "formSecondScreen\:tabWeb\:flickr"
});
```

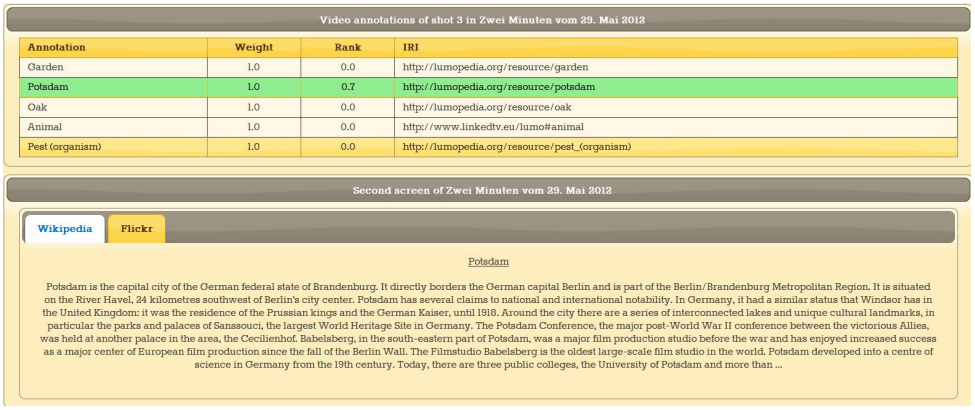

Figure 6: User interface in LSF for the related web sites

```
if (wikiurl==’null’) {
lsf.popcorn.footnote({
text:’no wiki info found’,
target: "formSecondScreen\:tabWeb\:wikipedia"
});
} else {
lsf.popcorn.wikipedia({
src: wikiurl,
title  : label,
target: "formSecondScreen\:tabWeb\:wikipedia"
});
}
},
pause : function() {
lsf.popcorn.pause();
},
seek : function(seekto) {
lsf.popcorn.play(seekto);
}
};
$(lsf.initPopcorn());
</script>
```

See Figure 6 for a screenshot.

3.1.4 Implementation details

LSF is a typical distributed three-tier web application: the front-end, the middle layer business logics, and the database backend. The main technical infrastructure is based on Java Enterprise Edition (JEE) eco-system.

- The front-end is mainly developed with Java Server Faces (JSF) with some ad-hoc HTML5 code (JavaScript, CSS3, etc.)
- The middle business layer is based on the transactional Enterprise Java Beans (EJB)
- The backend layer is based on the database management system (DBMS) JavaDB (formally known as Derby). The Object-Relation Mapping (ORM) is done with Java Persistence API (JPA)
- The RESTful web services for the integration with other software components are implemented with JAX-RS specifically Jersey Framework
- And the Java application server which acts as the container is the Glassfish Open Source Edition.

3.1.4.1 RESTful web services In order to support the inter-operability between software components, LSF provides two RESTful web services to access the semantic matching module and the related video module.

In order to use the web services, the user must apply a key from Fraunhofer at first. After you get the key, set this key in the HTTP header as the value of "X-Authentication". Since RESTful is stateless, i.e. not session-based, you must attach this key-value pair in each HTTP request header. The result of the web services can be formatted in XML or JSON. The default one is XML. To change the format produced by the web services, please specify another key-value pair in the HTTP header. To request the result as JSON, use the following key-value pair in HTTP header Accept : application/json; To request the result as XML, use the following key-value pair in HTTP header Accept : application/xml or just remove this key-value pair in the header since XML is the default format.

For error handling, the service consumer should check the response code of the result. If anything is wrong, for example, the given user name or context name does not exist, then the response code would be HTTP 400, indicating a `BAD_REQUEST`. And the content type would to set to `TEXT_HTML` with some human readable error messages.

The following two tables give more details about the RESTful web services provided in LSF:

Description	Retrieve the ranking
URL	`/service/rank/{username}/{ctxname}/{annotation}/{weight}`
HTTP method	GET
Response content type	XML, JSON
Authentication	*key required*
Sample response	`<matchResult><weight>0.2</weight></matchResult>`

Description	Retrieve the related videos
URL	`/service/related_video/{username}/{ctxname}/{video_id}`
HTTP method	GET
Response content type	XML, JSON
Authentication	*key required*

Sample response	`<videos>`
	` <video>`
	`  <description>`
	`Der in alkoholischen Getraenken enthaltene Alkohol wird als Ethanol bezeichnet. Im Verlauf koennen sich Beschaffung und Konsum von Alkohol zum lebensbestimmenden Inhalt entwickeln.`
	`  </description>`
	`  <id>6</id>`
	`  <title>Alkoholkrankheit</title>`
	`</video>`
	` <video>`
	`...`
	`</videos>`

3.2 f-PocketKRHyper: Semantic Reasoning Post-Filtering

The filtering described above is based on user interests and MF annotations described as sets of semantic tags or as relational expressions. If we need more expressive means to represent user interests a more expressive modeling and reasoning approach is needed.

As mentioned in D 4.3, a semantic reasoner, namely f-PocketKRHyper, is being used for concept and content post-filtering. The algorithmic foundation of f-PocketKRHyper lies in the crisp reasoner it has extended: the Pocket KRHyper mobile reasoner [Kleemann, 2006]. Thus it is a first-order model generating reasoner implementing the hyper-tableaux calculus [Baumgartner, 1998]. Its expressivity lies within the tractable DLP fragment [Grosof et al., 2003].

f-PocketKRHyper has extended Pocket KRHyper to fuzziness and has made improvements on the original implementation efficiency-wise and with respect to disjunction handling. Since its first extension, the original J2ME implementation was transformed back to JavaSE, while maintaining the original implementation's principles of a lightweight and efficient algorithm, capable of performing reasoning services

in limited resource devices. More details about the algorithmic foundation and basis of f-PocketKRHyper can be retrieved in D4.3.[5]

3.2.1 Semantic matchmaking and filtering

Match-making by the reasoner takes as input the semantic knowledge base for each problem, expressed in a variant of the KRSS[6] ontological notation, i.e., in the case of match-making that is the merge of

- the reference knowledge base (LUMO, cf. also D 4.4),
- the user profile (cf. D 4.4), and
- the content annotation for a single content item from the list of candidate items that are available (cf. D 2.4) .

A manufactured concept – for demonstration purposes here called UIC (after User Interests Concept) – is included in the profile, for which it applies:

$$\exists \text{hasInterest}.(A \sqcup B \sqcup C \sqcup \ldots) \sqsubseteq \text{UIC} \tag{9}$$

where A,B,C are concepts that denote user interests. These concepts can either be LUMO classes, individuals instantiating a LUMO class, or complex concepts, i.e., logical expressions, for example:

$$\exists R.(D \sqcap E) \sqsubseteq C \tag{10}$$

where R is a LUMO property, D and E are LUMO classes.

Similarly, another axiom exists for disinterest, implying a manufactured concept UDC (after User Disinterest Concept), for which it holds that it is disjoint with UIC, such that:

$$\text{UIC} \sqcap \text{UDC} \sqsubseteq \bot \tag{11}$$

UIC and UDC being disjoint, enabling the reasoner to refute a match whenever UDC is satisfied (see below), even if UIC is also satisfied.

3.2.1.1 Content Filtering The purpose of match-making (i.e. content filtering) is to determine whether UIC is satisfied by a given content item. For this purpose, the reasoner runs once per content item in a list of candidate content items. If UIC is satisfied then the candidate matches the user profile and accordingly an entry of the match is stored internally until all candidates are examined.

The match degree is defined as the degree of the fuzzy entailment of UIC[7]. If UIC is not satisfied then no match is made and the content item is disregarded. If UIC is satisfied but also UDC is satisfied, the reasoner will result in a refutation, since the two concepts are disjoint. This serves for rejecting content that the user dislikes, even if at some level it contains interesting material. Figure 7 describes the flowchart of the f-PocketKRHyper match-making process.

3.2.1.2 Toy Example Let us assume a user from Berlin who is interested in sport events, but only those that are in the user's local vicinity. The user also bares an interest in politics in general. However, he has shown specific disinterest in Barack Obama whom he dislikes, by rejecting news items about him, either regarding politics or personal news stories. The user model would read:

$$\begin{aligned} \exists \text{hasInterest}.(0.8 \bullet \text{Rule}_1 \sqcup 0.6 \bullet \text{Politics}) &\sqsubseteq \text{UIC} \\ \text{Location}(\text{Berlin}) \sqcap \text{Sports\_Event} &\sqsubseteq \text{Rule}_1 \\ \exists \text{hasInterest}.(0.7 \bullet \text{Politician}(\text{Barack Obama})) &\sqsubseteq \text{UDC} \\ \text{UIC} \sqcap \text{UDC} &\sqsubseteq \bot \end{aligned}$$

We also know from LUMO that:

$$\text{Politician} \sqsubseteq \forall \text{hasTopic.Politics} \tag{12}$$

Now let us consider 3 candidate content items:

[5] The semantics of the reasoning services supported by f-PocketKRHyper can be found in `mklab.iti.gr/linkedtv/files/F-pocketKRHyper_semantics.pdf`.

[6] `dl.kr.org/dl97/krss.ps`

[7] based on the fuzzy semantics of `mklab.iti.gr/linkedtv/files/F-pocketKRHyper_semantics.pdf`

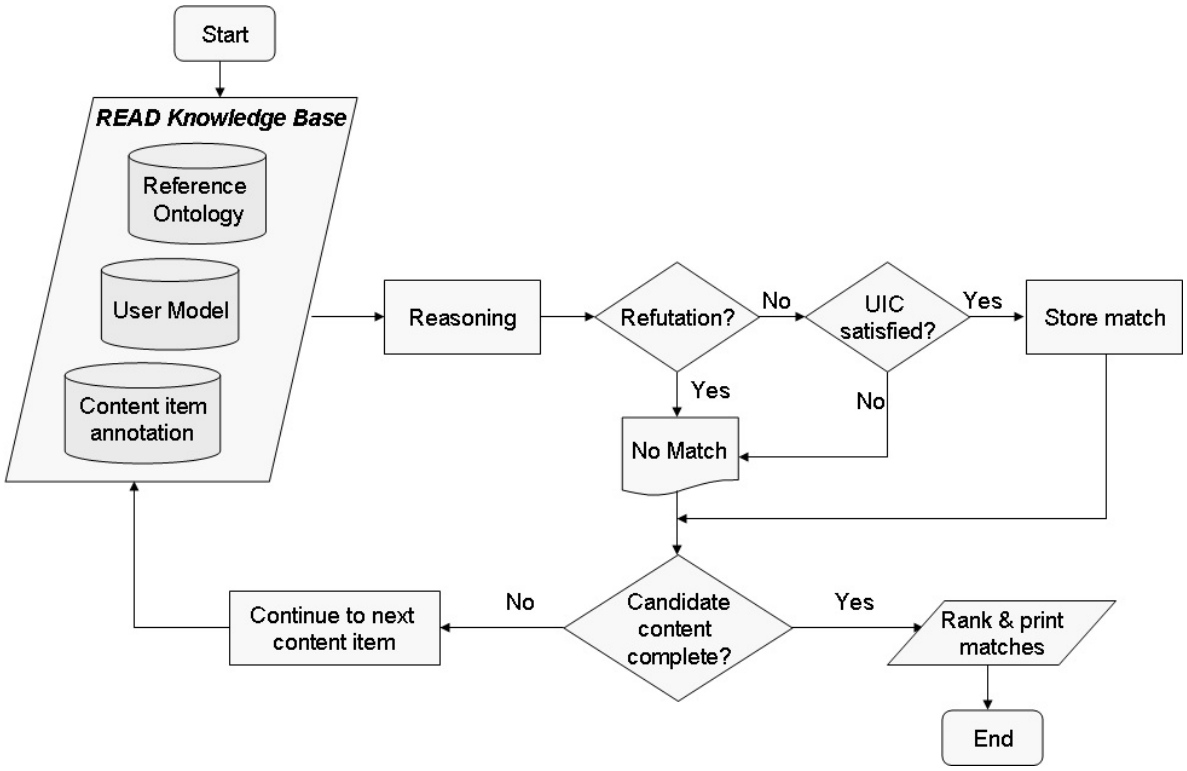

Figure 7: The f-pocketKRHyper match-making algorithm flowchart.

(A) , which includes a news story about the UEFA Champions League Final that takes place in London. Its semantic description would read:

$$\text{Sports\_Event}(\text{UEFA\_Champions\_League\_Final}) \geq 0.8$$
$$\text{Location}(\text{London}) \geq 0.7$$

(B) , which includes a news story about the Bundesliga Final that takes place in Berlin. Its semantic description would read:

$$\text{Sports\_Event}(\text{Bundesliga\_Final}) \geq 0.8$$
$$\text{Location}(\text{Berlin}) \geq 0.7$$

(C) , which includes a news story about Barrack Obama. Its semantic description would read:

$$\text{Politician}(\text{Barrack\_Obama}) \geq 0.7$$

Note:

– In statements of the form <Degree>•<Concept>, which apply only to profile concepts, <Degree> denotes the weight by which the user likes or dislikes the certain <Concept>.

– In statements of the form <Concept> $\geq$ <Degree>, which apply only to facts existent in the content annotation, <Degree> denotes the confidence degree by which the <Concept> is recognized to semantically describe the specific content item.

The reasoner now checks for each content item if UIC is satisfied without any refutation occurring:

Item	Inference steps	R*	S*	M*
A	Sports_Event satisfied by fact Sports_Event(UEFA_Champions_League_Final) $\geq$ 0.8 Location(Berlin) not satisfied. Therefore Location(Berlin) AND Sports_Event not satisfied. Therefore $Rule_1$ not satisfied. Therefore neither of 0.8•$Rule_1$ OR 0.6•Politics are satisfied. Therefore UIC not satisfied.	NO	NO	NO
B	Sports_Event satisfied by fact Sports_Event(Bundesliga_Final) $\geq$ 0.8 Location(Berlin) satisfied by fact Location(Berlin) $\geq$ 0.7. Therefore Location(Berlin) AND Sports_Event satisfied by 0.7. Therefore Rule1 satisfied by 0.7. Therefore one of 0.8•$Rule_1$ OR 0.6•Politics, i.e, 0.8•$Rule_1$ is satisfied. Therefore UIC is satisfied by 0.56.	NO	YES	YES (0.56)
C	Politics satisfied by fact Politician(Barrack_Obama) 0.7. Therefore one of 0.8•$Rule_1$ OR 0.6 • Politics, i.e, 0.6 •Politics is satisfied. Therefore UIC is satisfied by 0.42. ALSO UDC is satisfied by 0.7•Politician(Barack Obama) $\geq$ 0.7 by 0.49. SO UIC DISJOINT UDC is satisfied, Therefore a refutation is found.	YES	YES	NO

* (R=Refute, S=UIC satisfied, M=Match)

Refer to `mklab.iti.gr/linkedtv/files/F-pocketKRHyper_semantics.pdf` for an explanation of how fuzzy assertions are handled, therefore how degrees are derived.

3.2.2 Concept Filtering

Concept filtering is achieved by propagation of user interests across the reference knowledge (i.e. LUMO) concept space. This is achieved by instantiating all user interests (concepts that imply the UIC concept in the user profile), constructing pseudo-instances of the maximum confidence degree (i.e. 1.0) where needed.

3.2.2.1 Example In the previous toy example, we assume that all primitive concepts are instantiated, such as:

$$\begin{aligned} \text{Politics}(a) &\geq 1.0 \\ \text{Location}(\text{Berlin}) &\geq 1.0 \\ \text{Sports\_Event}(b) &\geq 1.0 \end{aligned}$$

The concepts that subsume the user interests along the taxonomical or non-taxonomical relations in the reference ontology constitute the set of interesting concepts to the user and are presented as the set of listed concepts, along with the degree of interest which is dependent on the weight of preference that the interests which are subsumed by them have in the user profile.

3.2.3 RESTful web services

LinkedTV content and concept filtering services via semantic reasoner are supported by a REST API, currently hosted on a CERTH-ITI server[8].

[8]base URI: `160.40.50.224:8182/api/` until integrated onto the LinkedTV platform

3.2.3.1 Content filtering Invokes and returns the results of the process of filtering several content items based on a given user profile, i.e. a list of content items, ranked according to the estimated preference of the given user.

Description	Content filtering
URL	/api/content_filtering?uid={uid}
Response content type	application/json

Sample response	{
	"[{"Degree":"0.6723999999999999",
	"URL":"www.example.org",
	"cid":"silbermond_seed_content_chapter"}]",
	"[{"Degree":"0.6723999999999999",
	"URL":"www.example.org",
	"cid":"stilbruch_20120322_silber_m"}]",
	"[{"Degree":"0.6723999999999999",
	"URL":"www.example.org",
	"cid":"silbermondclubconzert"}]",
	"[{"Degree":"0.6051599999999999",
	"URL":"www.example.org",
	"cid":"radioberlin_silbermond"}]",
	"[{"Degree":"0.6051599999999999",
	"URL":"www.example.org",
	"cid":"www_silbermond_de"}]",
	"[{"Degree":"0.6051599999999999",
	"URL":"www.example.org",
	"cid":"radioberlin_silbermond_eine"}]"
	}

3.2.3.2 Concept filtering Invokes and returns the results of the process of retrieving all concepts of interest to a user based on a given profile, i.e., an unranked list of concepts from a dedicated concept space onto which the given user's interests are propagated.

Description	Concept filtering
URL	`/api/concept_filtering?uid={uid}`
Response content type	application/json

Sample response	{
	[{Degree":1.00,"Concept":"location"}],
	[{Degree":0.82,"Concept":"agent"}],
	[{Degree":1.00,"Concept":"intangible"}],
	[{Degree":0.82,"Concept":"music_organization"}],
	[{Degree":0.76,"Concept":"AND sports_club user_location"}],
	[{Degree":1.00,"Concept":"person"}],
	[{Degree":1.00,"Concept":"sports_club"}],
	[{Degree":0.68,"Concept":"lifestyle_leisure"}],
	[{Degree":0.68,"Concept":"leisure"}],
	[{Degree":1.00,"Concept":"user_dimensions"}],
	[{Degree":1.00,"Concept":"sports_event"}],
	[{Degree":0.68,"Concept":"recreation"}],
	[{Degree":1.00,"Concept":"context_dimensions"}],
	[{Degree":0.51,"Concept":"AND olympics athlete"}],
	[{Degree":1.00,"Concept":"spatio-temporal"}],
	[{Degree":1.00,"Concept":"athlete"}],
	[{Degree":0.68,"Concept":"crafting"}],
	[{Degree":1.00,"Concept":"sports_agent"}],
	[{Degree":0.82,"Concept":"band"}],
	[{Degree":1.00,"Concept":"olympics"}],
	[{Degree":0.68,"Concept":"hobby"}],
	[{Degree":1.00,"Concept":"event"}],
	[{Degree":0.68,"Concept":"topics"}],
	[{Degree":1.00,"Concept":"sports_organization"}],
	[{Degree":1.00,"Concept":"domain_dependent_dimensions"}],
	[{Degree":0.54,"Concept":"weather"}],
	[{Degree":1.00,"Concept":"user_location"}],
	[{Degree":0.82,"Concept":"organization"}],
	}

3.2.4 Future extensions

Additional extensions planned within LinkedTV include the extension of the reasoner's expressivity to cover the entirety of the OWL 2 RL3 fragment (an extension of the DLP fragment to OWL 2 expressivity) and fuzzy role assertions, and possibly also the inclusion of conditionals to accommodate contextualised profile handling (cf. D 4.4). On the semantic filtering level, concept filtering will be extended with an option of a degree decay factor along each propagation step of the interests along the reference knowledge base.

3.3 In-Beat: Matching Preference Rules with Content

Preference rules learned with EasyMiner via the In-Beat Preference Learner (see D 4.4, Section 4.2.2) can be directly used to rank relevant enrichment content according to user preferences, in addition to further processing of these rules by SimpleLearner. In this section, we introduce In-Beat Recommender, an experimental recommender, which serves for direct recommendation based on preference rules learned with EasyMiner and stored in the Rule Store. As a pilot solution, In-Beat is not intended for inclusion into the WP 2 work-flow. There are no LinkedTV dependencies on In-Beat. For this reason, In-Beat might be also used as a test-bed for multiple recommender algorithms, in addition to rule-based recommendation, such as, but not necessarily, the UTA method drafted in D 4.2.

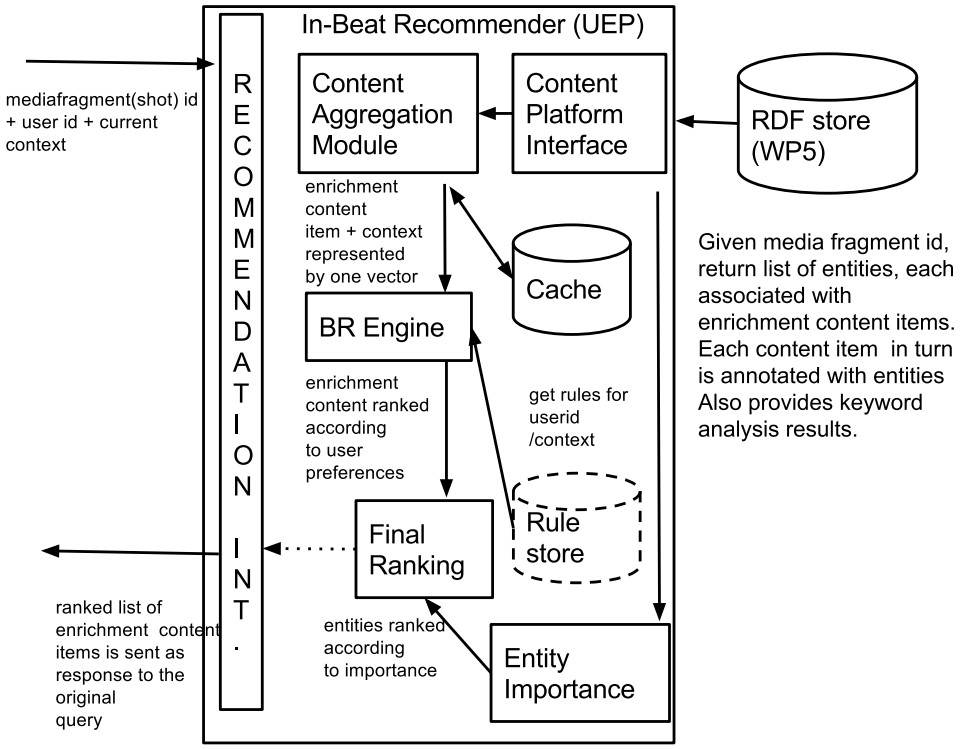

Figure 8: Architecture and work-flow of the In-Beat Recommender

3.3.1 Work-flow

Since the enrichment content is grouped by entities recognized in the corresponding shot of the seed media content, the ranking process can be approached in three stages:

1. Determine the importance of entity to the user
2. For given entity, re-rank the enrichment content
3. Combine score for entity and enrichment content into one value

The work-flow of the work-in-progress “Interest Beat” (In-Beat) Recommender Engine is described on Figure 8.

3.3.2 Components

The Interest Beat (In-Beat) recommender consists of several components described below. Recommendation Interface module obtains requests for recommendation, which comprises the user identification, identification of the currently playing media fragment (seed media fragment), and description of user context. As a response, the module returns a ranked list of enrichment content. Content aggregation module requests the enrichment content for the seed media fragment id from the platform. Consequently, the content aggregation module requests the annotations (entities) for each enrichment item. The Content Aggregation Module then transforms this semantic description of the enrichment item to a single vector, attaching also the contextual representation. This leads to a creation of a vector with same features as the vectors used to describe the seed content, which are used for rule learning (ref. to D 4.4, Section 3.2), lest of course, the scalar value of interest. BR Engine module finds the rules matching the seed content vector, aggregates their conclusions, and returns a predicted single value of interest. Entity Importance module uses the WP1 Keyword Analysis to rank entities according to importance. Refer to D 1.2, Section 6 for more details. Final Ranking module combines the estimated user interest in the individual enrichment content item produced by the BR Engine with the importance of the entity, for which the enrichment item was found.

3.3.3 Interface

The In-Beat recommender responds to requests for enrichment content made by the LinkedTV player with a ranked list of enrichment content. There are three interfaces described on Figure 8. Recommendation Interface obtains a JSON-formatted request for recommendation, which contains the user id, media fragment id, and current context. The format for the request has not yet been fixed. In-Beat responds with a JSON-formatted set of enrichment results. The response below is only illustrative. This response provides a ranked list of content. Each enrichment content item is assigned to the entity to which it relates to. Additional information required to visualize these recommendations can be obtained from the Platform, using the object identifier.

```
[{
  entity: "dbpedia:Football",
  url: "http: //www.rbb-online.de/sport/fussball/index.html"
},
{
  entity: "dbpedia:North_Korea",
  url: "http: //www.rbb-online.de/rbbaktuell/archiv/20130826_2145/turbine-potsdam-nordkorea.html"
},
{
  entity: "dbpedia:North_Korea",
  url: "http: //www.rbb-online.de/rbbaktuell/archiv/20130826_2145/asylheim-im-innenausschuss.html"
}]
```

Content Platform Interface uses the media fragment (shot) identifier to obtain enrichment content stored in the platform, and subsequently, the Content Platform Interface will for each enrichment content item request its annotation (entities). Entity Importance module uses the Keyword Analysis provided by WP1. The results of Keyword Analysis can be either obtained through the Content Platform Interface from the platform, or from local cache for personalized keywords (see D 1.2, Section 6.4.2).

3.3.4 State of development

The recommender system described in this section is a work-in-progress. Some of the components described in Section 3.3.2 have already been developed within GAIN, and need only be adapted for the use within the recommender (Content Aggregation module), other systems are in development (BR Engine module). The development of the foreseen Entity Importance module has not yet been started. Before its deployment to the system, a small feasibility study will be performed assessing its contribution to the quality of recommendation. The development of the Final Ranking module is contingent on the deployment of the Entity Importance module.

4 Experiments

In this section, we conduct qualitative and quantitative experiments based on the technology described above and in D 4.4. In Section 4.1, we start by showing that the algorithms work as intended, by employing a manually curated set of enrichment information. These experiments use LSF as the underlying content filter mechanism. In Section 4.2, we investigate the quality of the enrichment information when obtaining the data from the LinkedTV work-flow, and where we expect erroneous data.

For the experiments, we work with videos from the local German news show "Brandenburg Aktuell", taken from Public Service Broadcaster Rundfunk Berlin-Brandenburg (RBB),[9] which correlate to the scenario "news broadcast" in D 6.2.

4.1 Proof-of-concept: Content Filtering

In order to demonstrate the core functionalities of the filtering system, we have manually created some video annotations for 42 RBB videos. By assigning the enrichment information that we would hope for, we obtain a total list of 300 enrichment informations. For example, a 2 minute video with mixed information contains informations about (i) a reform on the point system for bad driving, which is located in Flensburg, Germany, (ii) a ply for mercy from an abductor of a bartender's son, and (iii) a new military transportation system to be located in Lusatia, Germany, was assigned with the following types from LUMO-Pedia:

(i) Police, Transport, Point_system_(driving), Flensburg, Driver's_license,

(ii) Kidnapping, Probation, Arrest, Punishment, Judge,

(iii) Airbus, Lusatia, Military, Aerodrome

Next, we modeled the archetypes Nina and Peter for the news show scenario (D 6.2), using the LUMO user preference editor. Here, it became apparent that the scenario description is quite specialized on certain seed videos but it is harder to generalize over Nina's general preferences that reflect her interest in a larger video selection. We thus deduced additional preferences that would fit her personality and included these into her profile. Nina, living in a hipster quarter in Pretslau, Berlin, should have a strong interest in localized context. Further, we decided that she is not interested in sports in general but has a knack on soccer. Treating each of these 42 videos as possible links from an (assumed) seed video, we could then employ the front-end functionality of LSF for a first proof-of-concept (Figure 9). The shown videos match our expectations.

4.2 Quality Assessment: Topic Segmentation and Topic Recognition

As seen above, both user modeling and content filtering relies heavily on the quality of the content annotation and enrichment pipelines conducted in other work packages. This means that we operate on top of possible annotation errors, named entity recognition errors, wrongly assigned types and erroneous mapping into the ontology. In the previous subsection we bypassed this challenge by working with a manually annotated and curated subset. In this subsection, we want to investigate the quality of the topic segmentation and the topic labeling.

Topic segmentation is crucial for implicit personalization of the links; if we register user activity and use this feedback for learning his/her preference, the time segment of the current topic of the video should be known properly. Topic labeling is crucial for assigning proper filtering techniques as described in Section 3, since they rely on type accuracy.

So, rather than working on prepared, clean data, we conduct these techniques on automatically generated output. We employ and combine multi-modal media analysis, namely: visual shot segmentation, automatic speech recognition, keyword extraction, named entity recognition, DBPedia enrichment, and the mapping into LUMO. Rather than treating all these steps as a black-box, we offer manual evaluation of each of these steps, in order to understand the nature of possible error sources.

[9] www.rbb-online.de

Figure 9: Visualization of video ranking within LSF, based on Nina's archetype as defined in the news show scenario. Content of the videos (line-by-line, from left to right): car accident, news overview, soccer scandal, organ donors, Berlin airport, women soccer championship, Berlin airport II, bridges on freeway.

4.2.1 Related work

For topic segmentation, one popular aspect commonly used is lexical cohesion (e.g. [Hearst, 1997, Stokes et al., 2004]). The general idea is to segment text into fixed-length blocks or into blocks containing sentences or paragraphs. In a next step, cohesion (by word repetitions, synonyms, related words, etc.) between these blocks is determined and areas with low cohesive strength are considered to indicate topic changes.

[Fuller et al., 2008] present a topic segmentation for pod-casts based on lexical cohesion calculated on Automatic Speech Recognition (ASR) transcripts. Instead of punctuation and paragraph information, which are not present in ASR transcripts, the authors use speech related information like time information from ASR and low energy points for segmentation into smaller blocks. [Guinaudeau et al., 2010] introduce an approach for topic segmentation incorporating confidences and likelihoods from the ASR process to calculate lexical cohesion for segmentation.

Still, most approaches are limited to one modality like text or speech only. We include an additional modality by using visual cues from shot segmentation replacing sentence and paragraph structures to determine smaller blocks. Furthermore, we consider lexical cohesion aspects beyond word repetitions by including linking information from Wikipedia in our approach.

4.2.2 Processing chain

In this experiment, we work with 1030 RBB videos which were collected over a time period of 5 months, and each day was already segmented by human editors from RBB, for an average of 6.8 segments per day. Note that these segments were not mono-thematic; roughly one-third contained short news summaries consisting of multiple topics.

For each video, we apply automatic speech recognition and, on top, keyword extraction and named entity extraction. Using visual shot segmentation as a fine-granular temporal segmentation, we assign the keywords and the NERs to these time segments and compute a distance score on them, which will be introduced in Section 4.2.3.1.

The following analysis and enrichment techniques are mainly described in D 1.3 and D 2.4. We proceed to summarize the main idea shortly.

Automatic Speech Recognition. We employ a speech recognition system that recently shifted from Julius [Lee et al., 2001] to the DNN paradigm, as implemented in KALDI [Povey et al., 2011]. The acoustic training data has been recently extended to roughly 900 hours of transcribed German broadcast and

Table 2: ASR performance on the various test sets, with new training data, the DNN decoder paradigm and SPSA tuning. Results given as Word Error Rate [%].

		DisCo		LinkedTV	
	dev	planned	spontaneous	planned	spontaneous
baseline (Julius) [Schneider et al., 2008]	30.2	26.4	33.5	27.0	52.5
new training data	29.6	24.0	31.1	26.4	50.0
+ SPSA [Stein et al., 2013]	27.7	22.6	28.4	24.5	45.6
DNN (KALDI)	23.9	18.4	22.6	21.2	37.6
+ SPSA	23.8	18.2	22.5	20.7	37.7

talk shows. The parameters of KALDI have been optimized using Simultaneous Perturbation Stochastic Approximation (SPSA) [Spall, 1992]. See Table 2 for a break down of influence regarding these factors, on the German Difficult Speech Corpus [Baum et al., 2010] as well as an LinkedTV in-house test set which is taken from the same domain as the processed videos.

Shot Segmentation. The temporal segmentation of the videos into shots is performed using the algorithm proposed in [Tsamoura et al., 2008]. This technique extracts visual features, namely color coherence, Macbeth color histogram and luminance center of gravity, and forms an appropriate feature vector per frame. Then, given a pair of neighboring (either successive or non-successive) frames of the video, the distances between their vectors are computed, composing distance vectors, that are finally evaluated using one or more SVM classifiers, resulting to the detection of both abrupt and gradual transitions between the shots of the video. Shot detection accuracy on this material is at 98.5% accuracy (269 correct shots out of 273 annotated in a sub-set of 30 minutes).

Keyword Extraction. Keyword extraction is based on [Tschöpel and Schneider, 2010], working mainly with the inverse document frequency (TF-IDF, see [Robertson and Walker, 1994]) paradigm and employing Snowball[10] as stemmer.

Named Entity Recognition. For NER, we employ the *NERD Client*, which is part of the NERD[11] framework. A multilingual entity extraction is performed over the video transcript and the output result is a collection of entities related to each video. Hence, the entities are classified using the core NERD Ontology v0.5[12] and serialized in JSON format, so they have to be translated by *tv2rdf* into a RDF representation. For German, NERD currently supports AlchemyAPI, THD, SemiTags and TextRazor.

4.2.3 Topic Segmentation

For tuning and evaluating our approach, we selected ASR transcripts of any two segments, with the goal of retrieving the seam shot of these segments by automatic means. We restricted these sets by only allowing segments taken from the same day, in order to ensure that the topics of these segments are distinct (especially important for topics with a high longevity in this period, such as the delay in Berlin's airport construction). Keyword extraction has been re-run on each of these double-segments. With an average of seven segments per day, all possible combinations of distinct segments gave 6830 testing instances. Of these, we used 5800 combinations as development set, and reserved 1030 combinations as test set.

4.2.3.1 Approach Our approach for topic segmentation combines visual cues from Shot Segmentation with information from ASR. In video production cuts, dissolves and wipes are used to visualize shifts in time or space including topic shifts in news productions. Furthermore, ASR provides the basis to determine lexical cohesion. First, we consider repetitions of extracted keywords in different shots. Second, we indirectly include associated words (e.g. generalizations, statistical associations, etc.) in a similar way by analyzing Wikipedia links and extract associated words for each of the extracted named entities. After combining both sources of information, shot boundaries in areas of low cohesive strength are considered to indicate topic transitions.

The best result on the dev set – 5-best recall showed 62% of the segmentation point to retrieve the exact seam point – was achieved by employing the following scoring:

– compute keyword splitting penalty (cohesive strength) per shot and per keyword, by taking the

[10] snowball.tartarus.org/
[11] nerd.eurecom.fr/
[12] nerd.eurecom.fr/ontology/nerd-v0.5.n3

Table 3: Evaluation of predicted break points, with $\pm$ 1 shot accuracy. Precision indicates that the current break point candidate is indeed a valid topic shift, recall indicates that the seam break point was found either by this shot or by one the higher ranked shots. The second set of experiments was conducted by leaving out all poly-thematic news summary segments.

	with short news		without short news	
	precision	recall	precision	recall
1-best	69.7	47.4	69.9	53.1
2-best	60.3	51.8	61.0	57.6
3-best	44.0	59.5	44.4	64.3
4-best	37.2	64.6	35.0	68.8
5-best	34.7	69.2	32.1	73.1

minimum number of keyword occurrences either on the left or the right hand-side of this shot, weighted with the TF-IDF relevance of each keyword

- compute a Wikipedia splitting penalty (cohesive strength) per shot and Wikipedia entry, by taking the minimum number of interlinked Wikipedia entry occurrences either on the left or the right hand-side of this shot, weighted with the (black-box) TextRazor relevance. In order to penalize generic links, we further divide this score by the tenth-part of the number of outgoing links of each Wiki entry
- add the penalties with an empirically set weight of the Wikipedia splitting penalty to 0.475
- smooth the joined penalty by applying a 5-window triangular filter bank, to emphasize concise penalty shifts
- compute the first-derivative
- determine the shot which has the highest absolute value indicating low cohesive strength

4.2.3.2 Evaluation A qualitative analysis of the shots which were preferred as split points rather than the segment seam produced interesting results, including: (a) split points whenever an interview started, because the interviewee employed different wording, (b) split points when a news point was commented on a higher level, and (c) many split points in between brief news summary sessions that were merged as one segments. One recurring peculiarity were weather reports, which normally were split well if the weather remained constant but received a split point in-between if the forecast predicted a change in the upcoming weather (e.g. “tomorrow the sun finally shines again, bringing warm temperatures to the north. . . ”).

For this work package, this means that we should not rely on editorial split points as offered by content providers for our algorithms. Especially for news summaries, a user might be interested only in a few sub-topics that are covered there, but in order to adress these preferences, we indeed need more fine-granular segmentation; we also should be aware whenever a media resource is poly-thematic.

Segmentation Results. The splitting points for (a) and (b) are quite useful in the LinkedTV context and thus not wrong in our context. For a better grasp of the segmentation quality, we hand-labeled “weak” topic segmentation shots within a segment that correspond to (a) and (b), in the remaining 1030 test combinations. This introduced on average 1.4 new split points for each file. Overall, the first-best guess has a precision of 69.7% to hit a correct topic break point within $\pm$ 1 shot accuracy, and a 47.4% recall to hit the actual seam point. See Table 3 for all results, and Figure 10 for an example.

News Summary Detection. A closer inspection of the results reveals that segment combination which do not include short news summaries roughly have the same precision but quite an improved recall (4–6% absolute), since the topic changes at the seam break points are less confusing. Knowing that a segment is poly-thematic when you see it thus seems like quite a valuable information; especially if we want to decide if it should be filtered or not. News summaries make up substantial 33.2% of the segments.

For each file, we extract the following features: the sum of the penalty in each shot, the sum of the active keywords in each shot, and the number of sign changes in the penalty derivate, each divided by the number of shots per file, respectively. Using a support vector machine with a linear kernel, we can classify these short news segments with an accuracy of 83.1% (average over 4-fold cross-validation). We can thus mark them appropriately and take special precautions when learning implicit

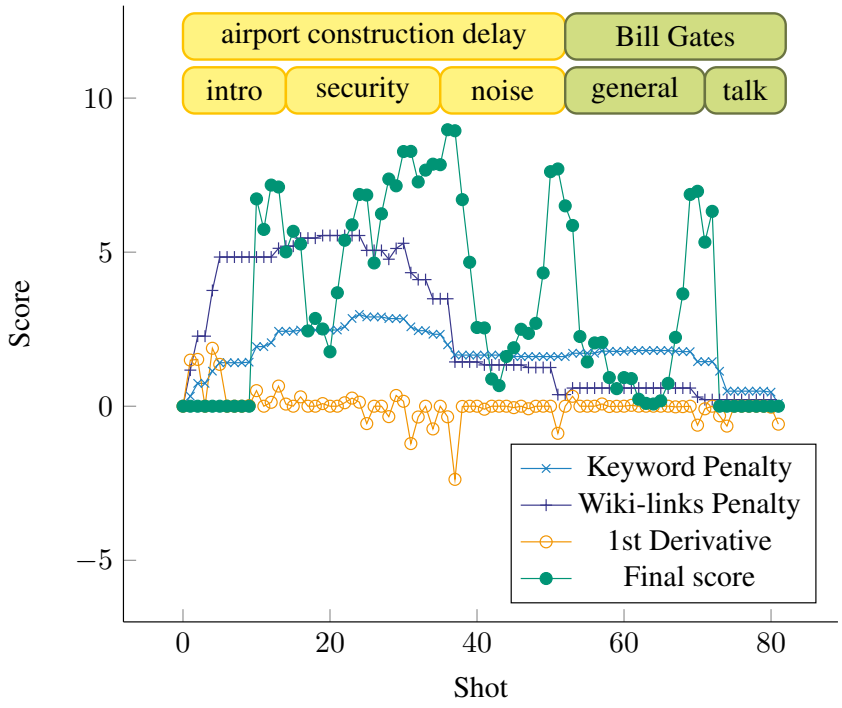

Figure 10: Example of the various scores for two joined segments, the first being on Berlin's airport construction delay, the second being on Bill Gates visiting Germany. The weaker topic changes (2 for Berlin, 1 for Gates) are marked accordingly.

user preferences and when we want to filter them. For example, a user might not be interested in in-depth coverage of Berlin's airport delay but still be interested in a short overview of this topic.

4.2.4 Topic Labeling

The segments detected were labeled with a plurality of concepts denoting news topics from the LUMO-v1 (D 4.4, Section 2.1). The process involves mapping types of extracted TextRazor entities to LUMO concepts and expanding them across the LUMO ontology via semantic reasoning. Filtering will take place on the basis of this mapping.

Enriching Entities with Types. Entities extracted by TextRazor were accompanied by types from DBPedia and Freebase[13]. However, types extracted by TextRazor (where existent) were deemed too generic in a first manual evaluation. Therefore, the entities recognized by NERD were enriched with more types from the Linked Open Data (LOD) cloud. The types were sourced from three knowledge bases: DBPedia [Bizer et al., 2009], YAGO 2s [Hoffart et al., 2013] and the Linked Hypernyms Dataset [Kliegr and Dojchinovski, 2013]. The input entities are identified using DBPedia URIs from the German DBPedia namespace. For DBPedia and the Linked Hypernyms dataset, which have both English and German subsets, the types were returned for both the original entity, as well as for its English version. The resulting file contained 56000 types, which is on average almost 10 types per entity.

The entity enrichment results show that the accuracy of types, where retrieved, is higher than TextRazor's, containing also more granular types rather (e.g. Politician rather than just Person).

Mapping Types to Dedicated Ontology. We mainly focus on LUMO's "Topics" sub-hierarchy, which is inspired by the IPTC news codes[14] and Wikipedia categories.

The subsumption service of the f-PocketKRHyper (Section 3.2) fuzzy semantic reasoner was devised to map the recognized types to LUMO via the LUMO mappings ontology. For each media item, the mapped LUMO types/concepts carry a degree that takes into account both the confidence of each entity recognized as well as the frequency by which a type occurs in the same news item for different entities.

[13] www.freebase.com/
[14] http://www.iptc.org/site/NewsCodes/

Table 4: Ranked accuracy of topic labels.

Rank	% of entities per threshold			
	None	> 0.1	> 0.2	> 0.3
1	30.85	29.30	18.87	12.96
2	14.22	14.06	15.09	18.52
3	11.16	9.77	14.15	18.52
4	11.60	11.33	9.43	3.70
5	32.17	35.55	42.45	46.30
Avg. rank	3.0	3.1	3.41	3.52

We compute the degree by:

$$\text{degree} = (\sum \text{conf}/\text{freq}) \cdot \frac{\text{freq}}{\max(\text{freq})}, \tag{13}$$

where $\sum conf$ is the sum of of all (black-box) confidence values recognized by TextRazor for the same type occurring for different entities within the same news item, freq is the frequency of the type for different entities in the news item and $\max(\text{freq})$ is the frequency of the most frequent of all types occurring in the particular news item.

From Types to Topics. With the use of LUMO's $hasTopic$ and $hasSubtopic$ object properties, for which it applies $hasSubtopic \sqsubseteq hasTopic$, and its respective universal quantification axioms, reasoning enables connection of concepts such as objects, events, agents, locations etc, which are most frequently detected as types, to their respective topics. The connecting axioms in LUMO are of the form $entityType \sqsubseteq \forall has(Sub)Topic.topic$, where $entityType$ is a LUMO concept and $topic$ is subsumed by the $Topics$ LUMO category/concept.

Again using f-PocketKRHyper, the concepts denoting types of entities per news item were expanded across the LUMO ontology to retrieve all LUMO concepts relevant to the initial types, through fuzzy inference. The result was a model of fuzzy LUMO concepts satisfying a news item by a $degree \in [0, 1]$. Of this model, only the concepts that are subsumed by the $Topics$ concept were used to label the segments.

Topic Labeling Results. 100 segments were manually labeled with concepts/topics from the LUMO $Topics$ sub-hierarchy and were compared against the automatically retrieved topic labels. The results reflect not only the topic labeling quality but carry forward the accuracy of the TextRazor entity extraction, the type recognition and enrichment, the mappings quality, down to the topic labeling.

The recall for evaluated items was 68%. For precision, the 5-best entries per news item were marked with an "appropriateness" score ranging from 1 to 5 (1:completely unrelated – 5:true and main topic). Table 4 shows the precision results within this ranking along various thresholds of topics' confidence degree.

While this is a good start, it also shows challenges that need to be addressed. For example, 5% recall loss can be tracked down to segments about "weather" which are not labelled correctly; in other words, even if we knew that a user is not interested in weather reports we could not filter out such links because we would be unaware of this topic (based on this data set). The reason for this is that, e.g, "Wetter" (German for "weather") is not recognized as an entity by textrazor; "Eis" (German for "Ice") is marked as food by DBPedia, but not as the weather phenomenon; "Schneefall" (German for "Snow") has no DBPedia types at all. So, although the mapping was correct and the ASR correctly produced the words, the semantic link to weather could not be retrieved with this current pipe.

In general, although each step introduced new error sources and propagated old ones, we were able to arrive at promising performance for the sub-tasks. What does this imply for content and concept filtering? Mainly, the above experiment emphasizes that this work package should not treat the input of the other work packages as black box input. For filtering, we need to be able to map the content of a linked media resource into our semantic user interest ontology, and constantly propagate back missing information to the annotation task and the enriching task. A good ASR performance can be measured in word error rates, and likewise a NER extraction can be measured in precision/recall; this information can be misleading, though – when the information of such an elemental concept such as "weather" is not found properly, user profiling and filtering will suffer in terms of accuracy. Propagated errors such as these can only be found and treated by extensive (cross-work-package) evaluation, which needs to be a major effort in year 3 (being well aware that the final evaluation of this work package's performance is the designated task T 4.4 which only starts at the beginning of year 4).

5 Conclusion

In this deliverable, we have elaborated on content and concept filtering. We have offered three parallel yet complementary solutions, namely LSF (Section 3.1) with a stronger focus on explicit user preferences, f-PocketKRHyper (Section 3.2) with a stronger focus on implicitly aggregated user interests, and In-Beat Recommender (Section 3.3) which offers a ranking directly based on the enriched annotation entities. Further, we provided qualitative and quantitative experiments and assessments on curated content and actual data as provided by the other work packages.

We thus have an established set of tools and services to draw upon when enhancing the user's viewing experience in the LinkedTV project. There is still a lot to do – pressing issues for the upcoming year include:

- **"Best of two worlds"**: merging ontologies. It has already been elaborated on in D 4.4 that the current status of parallel ontologies designed to capture user interests (namely LUMO-v1 and LUMO-rel) has many complementary merits which can be harvested and exceeded into a merged ontology to be named LUMO-v2. Once this new ontology is released, it will affect all the tools mentioned in this deliverable since they will need to be extended for support.
- **Ockham's razor**: exploiting synergies. Parallel solutions have different merits and flaws; joining these approaches will likely improve the overall system performance. However, if certain services do not add to the quality, we should focus on the more sophisticated approaches.
- **Dissemination**: stronger collaborative publications. As stated in the introduction, this work package was working under lab-conditions since it it is at the top of the processing pipeline. With the recent advances in the other work packages and a strong and stable overall work-flow currently being established, we look forward for collaborative experiments and dissemination.
- **Transparent privacy**: continue awareness. Privacy will continue to play a key role in such a user-sensitive working environment. With user evaluations on their way, every experiment needs to be planned carefully, and the concise communication to every tester (i) what data is captured, (ii) how it is stored, (iii) what it is used for and (iv) for how long we will maintain access to it, with (v) options to remove the consent at a later stage is non-negotiable.
- **Evaluation**: user-trials, larger data. Keeping privacy issues in mind, this work package looks forward to thorough evaluation on both large data and a larger set of test users.

Acknowledgment

The authors thank all partners in the LinkedTV consortium who contributed with their results, suggestions and their critical remarks to the research described in this paper.

6 References

[Baum et al., 2010] Baum, D., Schneider, D., Bardeli, R., Schwenninger, J., Samlowski, B., Winkler, T., and Köhler, J. (2010). DiSCo — A German Evaluation Corpus for Challenging Problems in the Broadcast Domain. In *Proc. LREC*, Valletta, Malta.

[Baumgartner, 1998] Baumgartner, P. (1998). Hyper tableaux – the next generation. In *Proc. International Conference on Automated Reasoning with Analytic Tableaux and Related Methods*, pages 60–76, Oosterwijk, The Netherlands.

[Bizer et al., 2009] Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., and Hellmann, S. (2009). DBpedia - a crystallization point for the web of data. *Web Semant.*, 7(3):154–165.

[Fuller et al., 2008] Fuller, M., Tsagkias, E., Newman, E., Besser, J., Larson, M., Jones, G., and de Rijke, M. (2008). Using term clouds to represent segment-level semantic content of podcasts. In *2nd SIGIR Workshop on Searching Spontaneous Conversational Speech (SSCS 2008)*, Singapore.

[Grosof et al., 2003] Grosof, B., Horrocks, I., Volz, R., and Decker, S. (2003). Description logic programs: Combining logic programs with description logic. In *Proc. International Conference on the World Wide Web (WWW 2003)*, Budapest, Hungary.

[Guinaudeau et al., 2010] Guinaudeau, C., Gravier, G., and Sbillot, P. (2010). Improving asr-based topic segmentation of tv programs with confidence measures and semantic relations. In Kobayashi, T., Hirose, K., and Nakamura, S., editors, *INTERSPEECH*, pages 1365–1368. ISCA.

[Hearst, 1997] Hearst, M. A. (1997). Texttiling: segmenting text into multi-paragraph subtopic passages. *Comput. Linguist.*, 23(1):33–64.

[Hoffart et al., 2013] Hoffart, J., Suchanek, F. M., Berberich, K., and Weikum, G. (2013). YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence*, 194:28–61.

[Kleemann, 2006] Kleemann, T. (2006). Towards mobile reasoning. In *Proc. Workshop on Knowledge Engineering and Software Engineering*, Bremen, Germany.

[Kliegr and Dojchinovski, 2013] Kliegr, T. and Dojchinovski, M. (2013). Linked hypernyms: Enriching DBpedia with Targeted Hypernym Discovery. Submitted.

[Lee et al., 2001] Lee, A., Kawahara, T., and Shikano, K. (2001). Julius – an Open Source Real-Time Large Vocabulary Recognition Engine. In *Proc. Eurospeech*, pages 1691–1694, Aalborg, Denmark.

[Povey et al., 2011] Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., and Vesely, K. (2011). The kaldi speech recognition toolkit. In *Proc. ASRU*. IEEE Signal Processing Society. IEEE Catalog No.: CFP11SRW-USB.

[Robertson and Walker, 1994] Robertson, S. E. and Walker, S. (1994). Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proc. SIGIR*, SIGIR '94, pages 232–241, New York, NY, USA. Springer-Verlag New York, Inc.

[Schneider et al., 2008] Schneider, D., Schon, J., and Eickeler, S. (2008). Towards Large Scale Vocabulary Independent Spoken Term Detection: Advances in the Fraunhofer IAIS Audiomining System. In *Proc. SIGIR*, Singapore.

[Spall, 1992] Spall, J. C. (1992). Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37:3.

[Stein et al., 2013] Stein, D., Schwenninger, J., and Stadtschnitzer, M. (2013). Improved speed and quality for automatic speech recognition using simultaneous perturbation stochastic approximation. In *Proc. Interspeech*, pages 1–4, Lyon, France.

[Stokes et al., 2004] Stokes, N., Carthy, J., and Smeaton, A. F. (2004). Select: A lexical cohesion based news story segmentation system.

[Tsamoura et al., 2008] Tsamoura, E., Mezaris, V., and Kompatsiaris, I. (2008). Gradual transition detection using color coherence and other criteria in a video shot meta-segmentation framework. In *Proc. Image Processing*, pages 45 –48.

[Tschöpel and Schneider, 2010] Tschöpel, S. and Schneider, D. (2010). A lightweight keyword and tag-cloud retrieval algorithm for automatic speech recognition transcripts. In *Proc. Interspeech*.