www.forgetit-project.eu

ForgetIT

Concise Preservation by Combining Managed Forgetting and Contextualized Remembering

Grant Agreement No. 600826

Deliverable D4.3

Work-package	WP4: Information Consolidation and Concentration
Deliverable	D4.3:Information analysis, consolidation and concentration techniques, and evaluation - Second release
Deliverable Leader	CERTH
Quality Assessor	Walter Allasia (EURIX)
Dissemination level	PU
Delivery date in Annex I	31-01-2015 (M24)
Actual delivery date	31-01-2015 (M24)
Revisions	0
Status	Final
Keywords	text summarization, text condensation, word redundancy removal, semantic enrichment, feature extraction, concept detection, image quality, face clustering, image clustering, image summarization, near duplicate detection

Revision History

Version	Major changes	Authors
0.1	Toc Draft	CERTH
0.2	Initial input	All partners
0.3	First update after first intra-WP review	All partners
0.4	First draft for internal review	CERTH
1	Final version after QA review	CERTH

List of Authors

Partner Acronym	Authors
CERTH	V. Solachidis, O. Papadopoulou, K. Apostolidis, A. Ioannidou, V. Mezaris
USFD	M. Greenwood
DFKI	B. Eldesouky, H. Maus

Table of Contents

Glossary

AP	Average Precision. 32
BoW	Bag-of-Word. 31
CD	Concept detector. 52
CL	Can not Link. 43
DR	Dimensionality Reduction. 50
EM	Expectation-Maximization. 51
EMD	Earth Movers Distance. 44
GC	Geometric Coding. 53
GMM	Gaussian Mixture Models. 50
GPS	Global Positioning System. 11, 50
HSV	Hue Saturation Value. 42
LBP	Local binary pattern. 44
LDA	Linear Discriminant Analysis. 50
LOD	Linked Open Data. 24
MAP	Mean Average Precision. 36
MDS	Multidimensional scaling. 50
MXinfAP	Mean Extended Inferred Average Precision. 36
NDD	Near duplicate detection. 50
NLP	Natural Language Processing. 24
NMI	Normalized Mutual Information. 54
ORB	Oriented FAST and Rotated BRIEF. 31
PCA	Principal Component Analysis. 50, 52
PHP	Personal Home Page. 28

PIMO	Personal Information Manager. 23
RGB	Red Green Blue. 31
ROI	Region of interest. 48
SIFT	Scale Invariant Feature Transform. 52
SIN	Semantic Indexing. 36
SURF	Speeded Up Robust Features. 31
SVM	Support Vector Machine. 31
TD	Target Dimensionality. 55
TF.IDF	Term Frequency - Inverse Document Frequency. 19, 35
UTC	Coordinated Universal Time. 51
VLAD	Vector of Locally Aggregated Descriptors. 31, 52
WASP	Web Application Service Provider. 28
YODIE	Yet another Open Data Information Extraction system. 30

Executive summary

The goal of Work Package 4 is to develop methods that enable the condensation of content, both textual and multimedia. This is intended to support the gradual forgetting approach, where content is presented with varying levels of details with the passage of time or ceasing importance.

In this document we present the second release of the ForgetIT techniques for textual and multimedia information analysis, consolidation and condensation, building on the previous work on this topic that was reported in D4.1 [D4.1, 2013] and D4.2 [D4.2, 2014] and taking into account the user requirements as presented in personal and organizational preservation deliverables (D10.1 [D10.1, 2014], D9.1 [D9.1, 2013] and D9.2 [D9.2, 2014]) as well as in the integration ones (D8.1 [D8.1, 2013], D8.2 [D8.2, 2014] and D8.3 [D8.3, 2014]). Results of the evaluation of the developed techniques are also reported.

The problems addressed and the methods that are presented in this deliverable can be classified into text and image classes as follows:

Text processing methods:

- *Linguistically Motivated Text Simplification*, which simplifies sentences by removing words or phrases which are not required.
- *Single document Summarization*, that performs a more extensive reduction in document length than Text Simplification, inevitably removing some of the contained information.
- *Text analysis for semantic text composition*, aiming at improving the process of text composition by building on the semantics of the text being composed rather than its syntactic features.
- *Easy Integration of Text Services using GATE [Cunningham et al., 2011] WASP* (Web Application Service Provider), that allows the rapid deployment and integration of other existing text processing applications in ForgetIT.

Image processing methods:

- *Visual concept detection and online training*, which improves the concept detection by employing more advanced image representations and an automatic technique of collecting positive data from the Web for training concept detectors, making the method capable of assembling training corpora for arbitrary concepts automatically without any manual assistance.
- *Face / person clustering in image collections*, that detects faces and groups them in clusters.
- *Near-duplicate detection and time synchronization for multi-user collection summarization*, which uses methods such as clustering that considers time and GPS in-

formation, near duplicate detection and time synchronization in order to summarize image collections gathered from many users and related events.

Software implementations for all these methods, in several cases updating the preliminary ones presented in D4.2 [D4.2, 2014], are presented as part of this deliverable.

1 Introduction

This deliverable presents the second release of the ForgetIT text and visual information analysis techniques for condensation and summarization. It is based on the state-of-the-art and requirement analysis that was reported in deliverable D4.1 [D4.1, 2013], and the first set of ForgetIT analysis techniques that were developed, evaluated and implemented as ForgetIT components in D4.2 [D4.2, 2014]. It contains improvements and extensions of the methods presented in D4.2 as well as new methods performing information analysis, consolidation and condensation.

In Section 2, the positioning of the WP4 components described in this document in the overall framework of ForgetIT is illustrated.

Starting with text analysis, in Section 3 a method that performs Linguistically Motivated Text Simplification through subtle simplifications, without altering the main point of the sentence, is presented. The theory of this method was presented in D4.2 [D4.2, 2014], while in this document evaluation results are reported as well its updated implementation as a web service, accompanied with an interactive demo.

Single document summarization that is presented in Section 4, achieves more extensive reduction than the aforementioned method, providing a summarization that can be used as a brief document overview.

The ForgetIT Semantic Text Editor, whose development started during the first year of the project (a first version of it was presented in D4.2) reduces the user's mental load during text composition. In this document, in Section 5, in order to further decrease the required user intervention, the knowledge based relation extraction used as part of this Text editor has been improved. Furthermore, coreference resolution is applied as a pre-processing step, in order to increase the number of entities used for relation extraction.

In Section 6 the GATE WASP (Web Application Service Provider) is presented, which allows the rapid deployment and integration of GATE text processing applications as web services.

Moving on to image analysis, in D4.2 a method for feature extraction and concept detection in image collections was presented. This method is extended in Section 7 in two directions. Firstly, we experiment with a more elaborate feature encoding process, which outperforms the one used in the previous version. Secondly, we propose a method for automatic acquisition of Web data which serve as training examples for visual concept detection, thus alleviating the need for manually-generated training corpora.

Furthermore, a face detection method employing multiple detectors and extra validations based on facial features existence was presented in the previous WP4 deliverable. In Section 8, the validation techniques that are part of our face detection method are extended by utilizing skin color and information about the number of detectors that detected the candidate facial region. Also, a face/person clustering method is proposed, using not only the face features but also other information such as costume and hair similarity.

Section 9 extends our previous experimentation with simple clustering algorithms in multiple directions. Initially, we extend our evaluation experiments using dimensionality reduction on the feature vectors which serve as input to the clustering algorithms. Then, clustering is extended by employing more data dimensions, such as time and GPS coordinates, in addition to visual information. A near duplicate detection method that can detect images portraying the same scene is developed and used in order to enrich our summarization method. Finally, a multi-user collection time synchronization method is presented, which temporally aligns image collections of the same event captured by different devices prior to clustering the photos into sub-events, thus enabling effective multi-user photo collection summarization.

Finally, in Section 10, the document conclusions are summarized and the Performance Indicators of WP4 are assessed and a description of how they are fulfilled is presented.

1.1 Target Audience

Although this document is quite technical, it could be read from multiple audiences having different backgrounds, since it is structured in such a way that each section targets different audiences. The first subsection of Sections 3 to 9, entitled “Problem statement”, describes shortly the method and targets a broad audience. It defines the problem to be solved and discusses how solving this problem supports the project’s scenarios.

The next subsection, entitled “ForgetIT approach”, contains the technical description of the presented method and targets more specialized technical audience. In this subsection, the methods’ algorithm details are presented.

Then, the “Experimental evaluation and comparison” subsection follows, which describes the method evaluation, presenting the employed dataset, the evaluation procedure, the measures that were adopted and the evaluation results.

Finally, the “Software implementation” subsection describes briefly the software implementation of each method and includes usage instructions and examples. This subsection is most useful for technical partners of the project (e.g. members of WP3, WP6, WP8, WP9, WP10) that want to integrate and use WP4 methods.

2 The Big Picture

The WP4 methods and component described in this deliverable are the constituent components of the two master components of the ForgetIT architecture that WP4 contributes to, the *Extractor* and the *Condensator*. These are parts of the Middleware layer of the ForgetIT architecture Fig. 1.

The *Extractor* takes as input the original media items (e.g. a text, a collection of texts, or a collection of images) and extracts information that is potentially useful not only for the subsequent execution of the *Condensator*, but also for other components or functionalities of the overall ForgetIT system (e.g. search).

The *Condensator* gets as input the *Extractor's* output and when necessary also the original media items, in order to generate a summary of the target data (or a subset of these media items). The *Condensator* performs further text and image analysis tasks whose results are specific to the condensation process. The final output of the *Condensator* is the condensed (i.e., summarized) media items or collections, or pointers to them.

The *Extractor's* and *Condensator's* output is fed to other ForgetIT components such as the *Contextualizer* and the *Collector/Archiver*.

2.1 Extractor

In this deliverable several subcomponents for text and image analysis are presented. Some of them are extensions and improvements of the ones introduced in D4.2 [D4.2, 2014], where others are new. A list of *Extractor* subcomponents is illustrated in Fig. 2. In this figure the reader is also able to view the subcomponents of the first release which have been presented in D4.2 [D4.2, 2014] and see which are extensions and improvements and which are new.

2.2 Condensator

The subcomponents of the *Condensator* are presented in Fig. 3. They are, similarly to the *Extractor's* subcomponents,extensions and improvements of the ones introduced in D4.2.

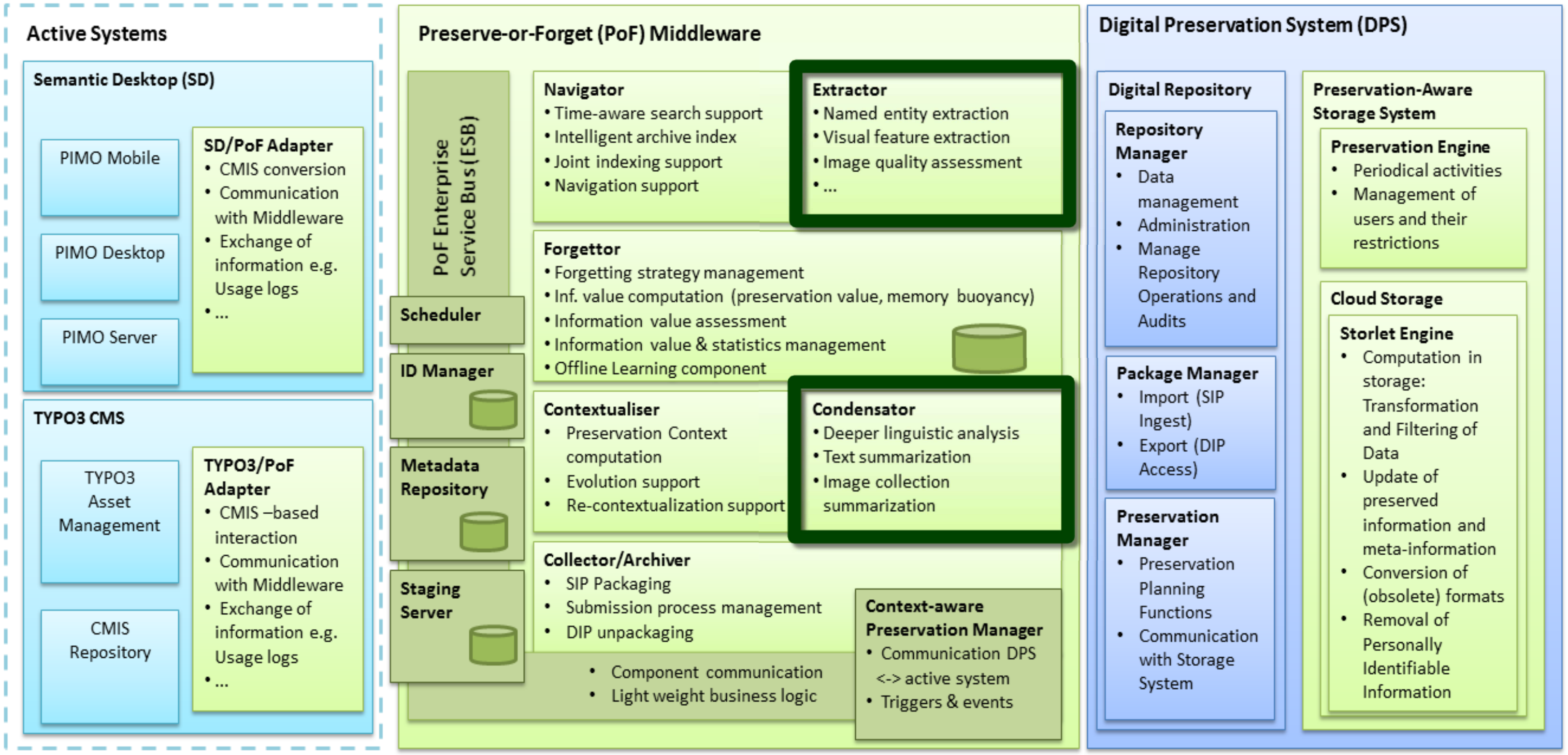

Active Systems
Semantic Desktop (SD)
PIMO Mobile
PIMO Desktop
PIMO Server
SD/PoF Adapter
• CMIS conversion
• Communication with Middleware
• Exchange of information e.g. Usage logs
• ...
TYPO3 CMS
TYPO3 Asset Management
CMIS Repository
TYPO3/PoF Adapter
• CMIS –based interaction
• Communication with Middleware
• Exchange of information e.g. Usage logs
• ...
Preserve-or-Forget (PoF) Middleware
PoF Enterprise Service Bus (ESB)
Scheduler
ID Manager
Metadata Repository
Staging Server
Navigator
• Time-aware search support
• Intelligent archive index
• Joint indexing support
• Navigation support
Extractor
• Named entity extraction
• Visual feature extraction
• Image quality assessment
• ...
Forgettor
• Forgetting strategy management
• Inf. value computation (preservation value, memory buoyancy)
• Information value assessment
• Information value & statistics management
• Offline Learning component
Contextualiser
• Preservation Context computation
• Evolution support
• Re-contextualization support
Condensator
• Deeper linguistic analysis
• Text summarization
• Image collection summarization
Collector/Archiver
• SIP Packaging
• Submission process management
• DIP unpackaging
• Component communication
• Light weight business logic
Context-aware Preservation Manager
• Communication DPS <-> active system
• Triggers & events
Digital Preservation System (DPS)
Digital Repository
Repository Manager
• Data management
• Administration
• Manage Repository Operations and Audits
Package Manager
• Import (SIP Ingest)
• Export (DIP Access)
Preservation Manager
• Preservation Planning Functions
• Communication with Storage System
Preservation-Aware Storage System
Preservation Engine
• Periodical activities
• Management of users and their restrictions
Cloud Storage
Storlet Engine
• Computation in storage: Transformation and Filtering of Data
• Update of preserved information and meta-information
• Conversion of (obsolete) formats
• Removal of Personally Identifiable Information

Figure 1: The *Extractor* and *Condensator* components an their positioning in the Middleware layer of the ForgetIT architecture

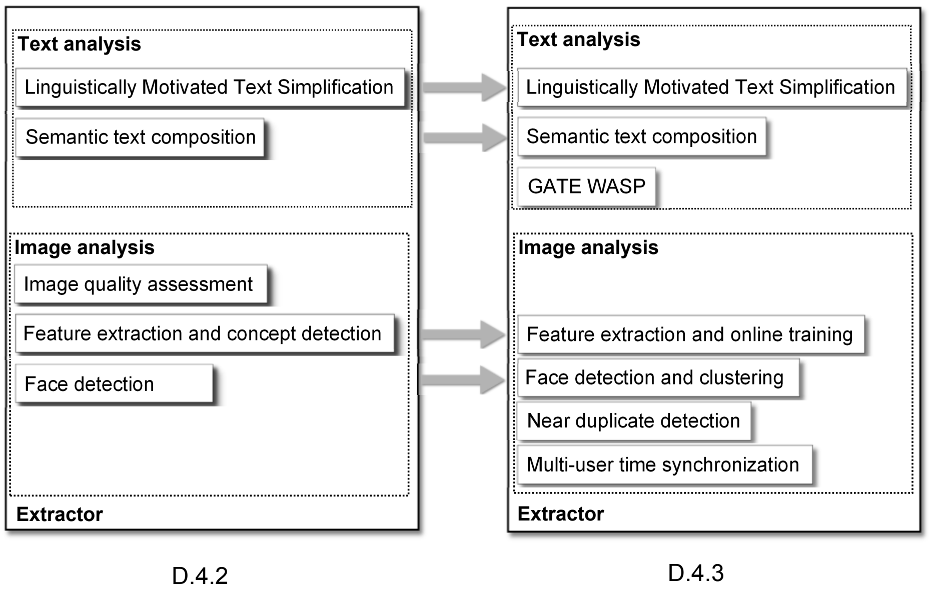

Text analysis
Linguistically Motivated Text Simplification
Semantic text composition
Image analysis
Image quality assessment
Feature extraction and concept detection
Face detection
Extractor
D.4.2
Text analysis
Linguistically Motivated Text Simplification
Semantic text composition
GATE WASP
Image analysis
Feature extraction and online training
Face detection and clustering
Near duplicate detection
Multi-user time synchronization
Extractor
D.4.3

Figure 2: The *Extractor's* subcomponents that were presented in D4.2 (left) and the new or updated subcomponents that are presented in the current deliverable (right)

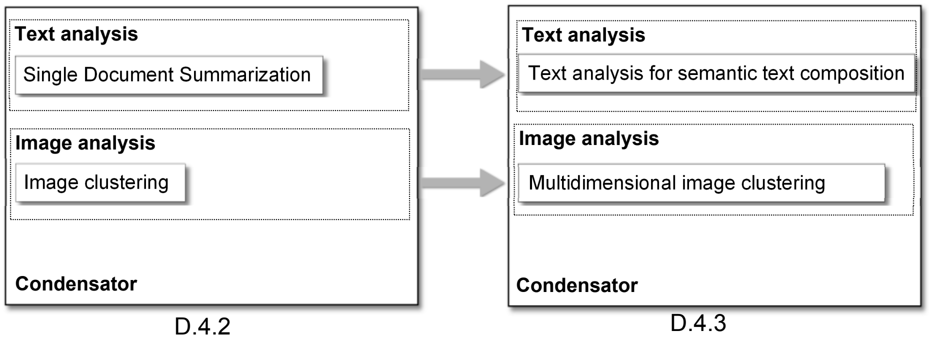

Text analysis
Single Document Summarization
Image analysis
Image clustering
Condensator
D.4.2
Text analysis
Text analysis for semantic text composition
Image analysis
Multidimensional image clustering
Condensator
D.4.3

Figure 3: The *Condensator's* subcomponents that were presented in D4.2 (left) and the new or updated subcomponents that are presented in the current deliverable (right)

3 Linguistically Motivated Text Simplification

3.1 Problem statement

There are many cases where full document summarization may not be appropriate but subtle simplifications would be acceptable. The approach described in this section simplifies sentences by removing words or phrases which are not required to convey the main point of the sentence. This can can be viewed as a first step in document summarization and also mirrors the way people remember conversations; the details but not the exact words used.

3.2 ForgetIT approach

The approach we have taken to this problem within ForgetIT is to use linguistic processing to determine potentially redundant words and phrases, which can be removed or replaced, without changing the meaning of the document, but which reduce its length.

The exact details of the language constructs that can currently be simplified were discussed in detail in deliverables D4.1 and D4.2 [D4.1, 2013, D4.2, 2014]. The main changes since D4.2 has been to improve the code, make a public release and include it within a web service for easy integration (see Section 3.4). No new approaches to simplification have been incorporated and as such D4.2 remains an accurate description of the technique.

Some examples of the condensation that can be applied include:

- “Any *particular type of* dessert is fine with me.” becomes “Any type of dessert is fine with me.” saving ten characters.
- “During that time period, many buyers preferred cars that were pink *in colour* and shiny *in appearance*.” becomes “During that time period, many buyers preferred cars that were pink and shiny.” saving twenty-three characters.
- “The function of this department is *the* collect*ion of* accounts.” becomes “The function of this department is to collect accounts.” saving seven characters.

3.3 Experimental Evaluation and Comparison

This approach to sentence simplification has been evaluated using data from the definition track of the TREC 2003 Question Answering evaluation [Voorhees, 2003]. The definition track consisted of 50 definition questions where answers had to be drawn from the AQUAINT collection of news text[1]. This document collection consists of approximately

[1]Linguistic Data Consortium (`http://www.ldc.upenn.edu`) catalogue number LDC2002T31.

Table 1: Evaluation results

Phrase Type	Number of Occurrences	% Correct	Average Character Saving
Gerund Clause	125	84.8%	49.32
Leading Adverbs	161	92.5%	4.27
Sentence Initial Expletives	6	16.7%	6.17
Redundant Category Labels	6	100%	6.67
Unnecessary Determiners and Modifiers	50	86.0%	7.94
Circumlocutions	1	100%	10.0
Unnecessary That and Which Clauses	20	95.0%	8.35
Noun Forms of Verbs	1	0.0%	6.0

1,033,000 documents in 3 gigabytes of text covering the Associated Press Worldstream News Service from 1998 to 2000, the New York Times news service from 1998 to 2000, and the English portion of the Xinhua News Service (Peoples Republic of China) from 1996 to 2000.

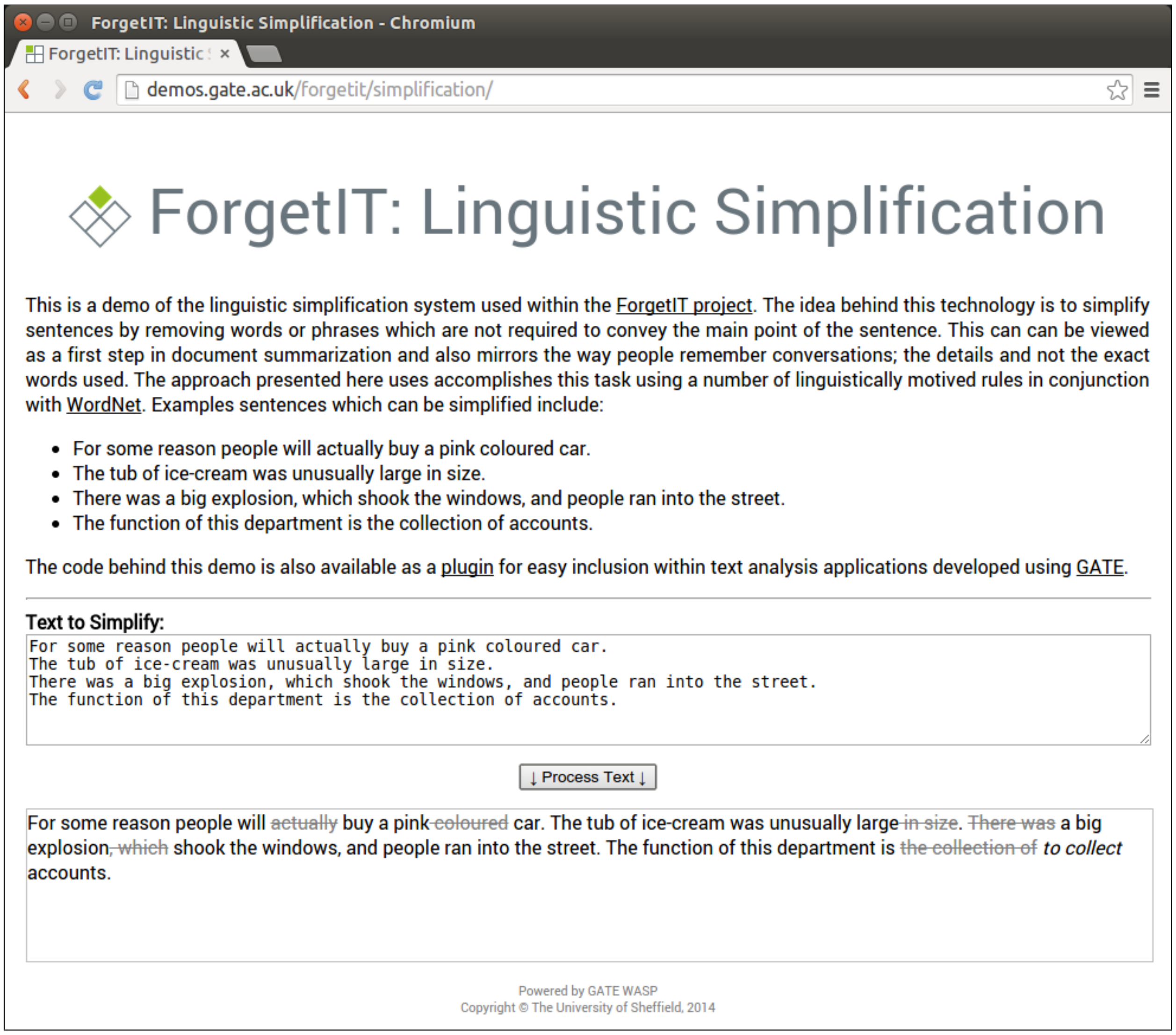
The evaluation of the sentence simplification was performed as part of a basic definitional question answering system. The full details of the question answering system are not important for this discussion but, the basic approach involved retrieving sentences from the AQUAINT collection which included the entity or event being defined. These sentences were then subject to simplification (the scoring metric for definitional questions penalises long sentences). Each of the sentences which were simplified were manually inspected to evaluate the simplification system in isolation.

The evaluation looked at the number of times each redundancy approach was applied and what percentage of these removals resulted in badly formed sentences. By badly formed sentences we mean that the sentences are no longer syntactically valid and not that important information relating to the target has been removed, as not a single instance of important information being removed was found. The results of this evaluation can be seen in Table 1.

It is clear that some of the methods of redundancy removal that were employed perform better than others and that some cover fairly rare sentence constructions.

3.4 Software implementation

This work has been implemented as a GATE [Cunningham et al., 2011] processing resource (PR) and uses WordNet [Fellbaum, 1998] as its main linguistic resource. As a GATE PR, this component can be used on its own or embedded within a larger application (named entity extraction, contextualization, etc.). The plugin (now called `Linguistic_Simplifier`) has been added to the main distribution of GATE as is available in the


ForgetIT: Linguistic Simplification - Chromium
ForgetIT: Linguistic
demos.gate.ac.uk/forgetit/simplification/

ForgetIT: Linguistic Simplification

This is a demo of the linguistic simplification system used within the ForgetIT project. The idea behind this technology is to simplify sentences by removing words or phrases which are not required to convey the main point of the sentence. This can can be viewed as a first step in document summarization and also mirrors the way people remember conversations; the details and not the exact words used. The approach presented here uses accomplishes this task using a number of linguistically motived rules in conjunction with WordNet. Examples sentences which can be simplified include:

- For some reason people will actually buy a pink coloured car.
- The tub of ice-cream was unusually large in size.
- There was a big explosion, which shook the windows, and people ran into the street.
- The function of this department is the collection of accounts.

The code behind this demo is also available as a plugin for easy inclusion within text analysis applications developed using GATE.

Text to Simplify:

```
For some reason people will actually buy a pink coloured car.
The tub of ice-cream was unusually large in size.
There was a big explosion, which shook the windows, and people ran into the street.
The function of this department is the collection of accounts.
```

↓ Process Text ↓

For some reason people will ~~actually~~ buy a pink ~~coloured~~ car. The tub of ice-cream was unusually large ~~in size~~. ~~There was~~ a big explosion~~, which~~ shook the windows, and people ran into the street. The function of this department is ~~the collection of~~ *to collect* accounts.

Powered by GATE WASP
Copyright © The University of Sheffield, 2014

Figure 4: Demo interface for easy experimentation

current nightly build[2]. The main characteristics of the software can be seen in Table 2.

The developed approach has also been made available as a web service, with an interactive demo, using GATE WASP (see Section 6) to allow for easy experimentation and integration within the use case tools. The service can be found at `http://services.gate.ac.uk/forgetit/simplification/`.

3.5 Discussion

It is clear from the results presented in the previous section that reducing sentences using a linguistically motivated approach results in shorter documents without any loss of

[2]`http://jenkins.gate.ac.uk/job/GATE-Nightly/lastSuccessfulBuild/`

information while retaining readability. Future work will involve producing a larger evaluation set that provides better coverage for some of the rarer constructs as well as the development of further rules.

Table 2: Linguistic simplifier plug-in: software technical details

Functional description	Linguistically Based Condensation
Input	Text Document
Output	Text Document
Language/technologies	Java, GATE, WordNet
Hardware Requirements	Any HW with Java support and sufficient RAM
OS Requirements	Any OS with Java support

4 Single Document Summarization

4.1 Problem statement

While text simplification, as described in Section 3, can be a useful tool in subtely reducing the amount of textual data to be preserved, without the loss of any information, there are many cases where a more radical reduction in document length would be beneficial. Summarization can be used to aid navigation of preserved documents by providing a brief overview of long documents as an aid to human memory, or it could be used to generate entirely new collections of information as with the diary generation idea being pursued in WP9.

4.2 ForgetIT approach

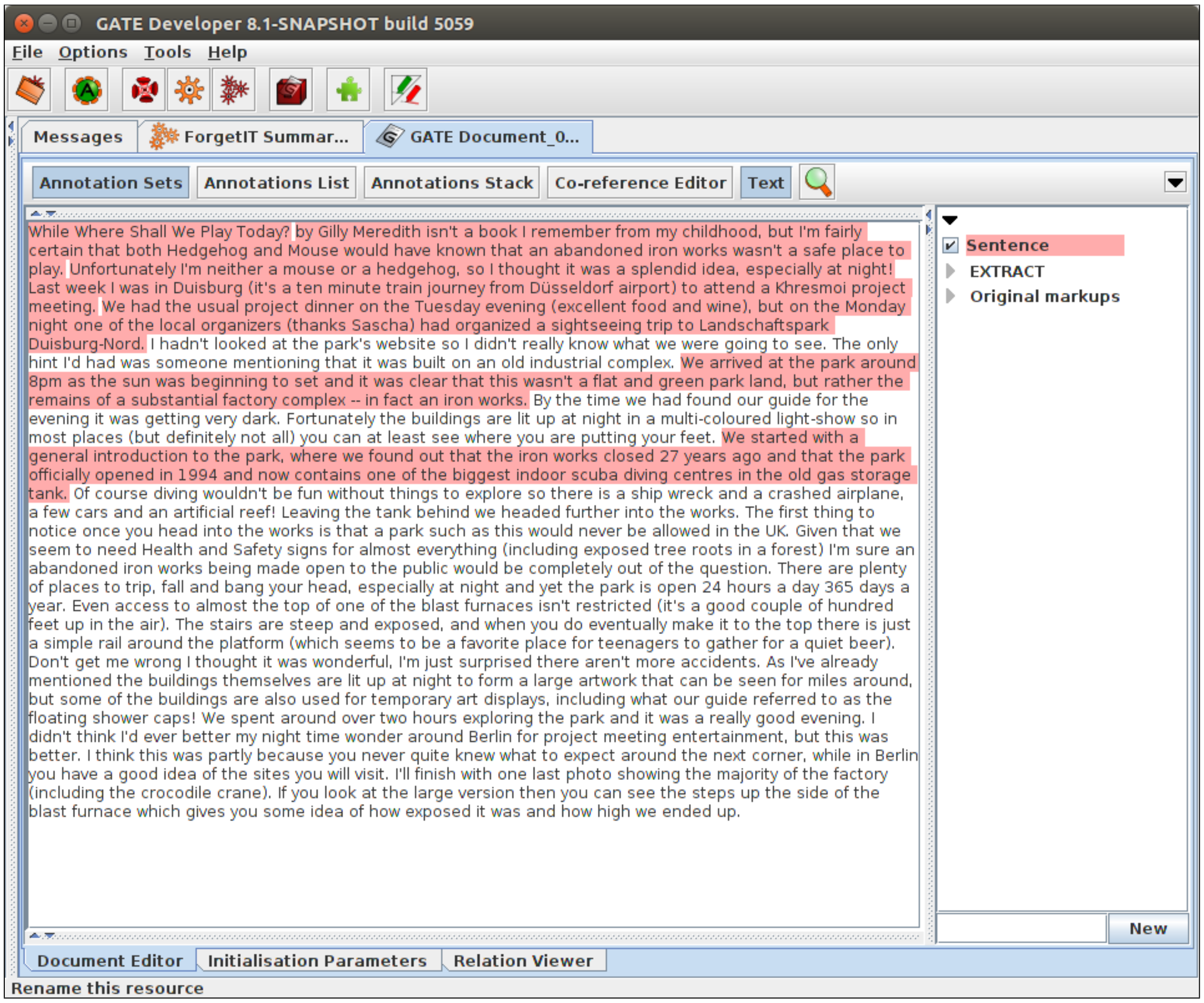
The approach to single document summarization we have adopted within the ForgetIT project uses the freely available[3] SUMMA toolkit [Saggion, 2008], which adds support for text summarization to GATE [Cunningham et al., 2011]. SUMMA provides numerous GATE processing resources which can be used to analyse text in order to produce a summary. These include both linguistically motivated resources as well as those that rely on document or corpus statistics to select relevant sentences as part of a summary.

Currently the single document summarization application we are using within ForgetIT performs topic-centered extractive summarization [Saggion et al., 2003, Barzilay and Mckeown, 2005] using a number of SUMMA components. The application uses the well known cosine similarity measure in conjunction with a term frequency-inverse document frequency, (TF.IDF), weighting function [Gerald et al., 1997] to extract the main information bearing sentences to form a summary. Different summary lengths can be generated using differing threshold values. As each document is currently considered in isolation we are using a pre-generated set of inverse document frequencies, computed over the AQUAINT collection of news text[4].

Specifically the current application:

- calculates the TF, IDF, and TF.IDF values for each Token annotation in the document
- builds a normalized TF.IDF token vector to represent the whole document
- builds a normalized TF.IDF token vectors for each sentence
- assignes a score to each sentence based on its position within the document
- uses the cosine similarity measure to score each sentence based on how similar it is to the whole document
- uses the cosine similarity measure to score each sentence based on how similarity it is to the first sentence (in general the first sentence in a document contains pertinent

[3]`http://www.taln.upf.edu/pages/summa.upf/`
[4]Linguistic Data Consortium (`http://www.ldc.upenn.edu`) catalogue number LDC2002T31.



While Where Shall We Play Today? by Gilly Meredith isn't a book I remember from my childhood, but I'm fairly certain that both Hedgehog and Mouse would have known that an abandoned iron works wasn't a safe place to play. Unfortunately I'm neither a mouse or a hedgehog, so I thought it was a splendid idea, especially at night! Last week I was in Duisburg (it's a ten minute train journey from Düsseldorf airport) to attend a Khresmoi project meeting. We had the usual project dinner on the Tuesday evening (excellent food and wine), but on the Monday night one of the local organizers (thanks Sascha) had organized a sightseeing trip to Landschaftspark Duisburg-Nord. I hadn't looked at the park's website so I didn't really know what we were going to see. The only hint I'd had was someone mentioning that it was built on an old industrial complex. We arrived at the park around 8pm as the sun was beginning to set and it was clear that this wasn't a flat and green park land, but rather the remains of a substantial factory complex -- in fact an iron works. By the time we had found our guide for the evening it was getting very dark. Fortunately the buildings are lit up at night in a multi-coloured light-show so in most places (but definitely not all) you can at least see where you are putting your feet. We started with a general introduction to the park, where we found out that the iron works closed 27 years ago and that the park officially opened in 1994 and now contains one of the biggest indoor scuba diving centres in the old gas storage tank. Of course diving wouldn't be fun without things to explore so there is a ship wreck and a crashed airplane, a few cars and an artificial reef! Leaving the tank behind we headed further into the works. The first thing to notice once you head into the works is that a park such as this would never be allowed in the UK. Given that we seem to need Health and Safety signs for almost everything (including exposed tree roots in a forest) I'm sure an abandoned iron works being made open to the public would be completely out of the question. There are plenty of places to trip, fall and bang your head, especially at night and yet the park is open 24 hours a day 365 days a year. Even access to almost the top of one of the blast furnaces isn't restricted (it's a good couple of hundred feet up in the air). The stairs are steep and exposed, and when you do eventually make it to the top there is just a simple rail around the platform (which seems to be a favorite place for teenagers to gather for a quiet beer). Don't get me wrong I thought it was wonderful, I'm just surprised there aren't more accidents. As I've already mentioned the buildings themselves are lit up at night to form a large artwork that can be seen for miles around, but some of the buildings are also used for temporary art displays, including what our guide referred to as the floating shower caps! We spent around over two hours exploring the park and it was a really good evening. I didn't think I'd ever better my night time wonder around Berlin for project meeting entertainment, but this was better. I think this was partly because you never quite knew what to expect around the next corner, while in Berlin you have a good idea of the sites you will visit. I'll finish with one last photo showing the majority of the factory (including the crocodile crane). If you look at the large version then you can see the steps up the side of the blast furnace which gives you some idea of how exposed it was and how high we ended up.

Figure 5: Example of Summarizing a Diary/Blog Entry

information and is highly likely to be included in a summary)

- assigns a final score to each sentence using a weighted combination of the different similarity scores

Differing length summaries can then be produced by selecting sentences with ever decreasing scores until the appropriate summary size has been reached. Currently the summarization length can be defined in terms of sentences as either a fixed number or as a percentage of the sentences in the full document.

4.3 Experimental evaluation and comparison

While it would be nice to be able to report a rigorous evaluation of the approach described above, we have been unable to find an appropriate pre-existing corpus which would be suitable for evaluation. We envisage using this system with web pages, diary entries etc.

yet the majority of available corpora are aimed at summarizing news documents which are clearly very different in style and structure. As such we hope to be able to collect relevant material as the use case tools further develop to allow for a rigorous evaluation (using the widely accepted ROGUE-L metric [Lin, 2004]) to be reported in D4.4 using data from the use cases. Having said that, the underlying components, taken from the SUMMA toolkit, have been used in numerous summarization systems before, albeit in differing configurations to that used here, and evaluations of these systems suggest that performance is suitable for our use.

Although there has, as yet, been no rigorous evaluation of the approach, the summaries produced during development have been examined to ensure that in general the results are reasonable and that the sentences selected produce summaries that make sense. An example summary generated during development can be seen in Fig. 5.

4.4 Software implementation

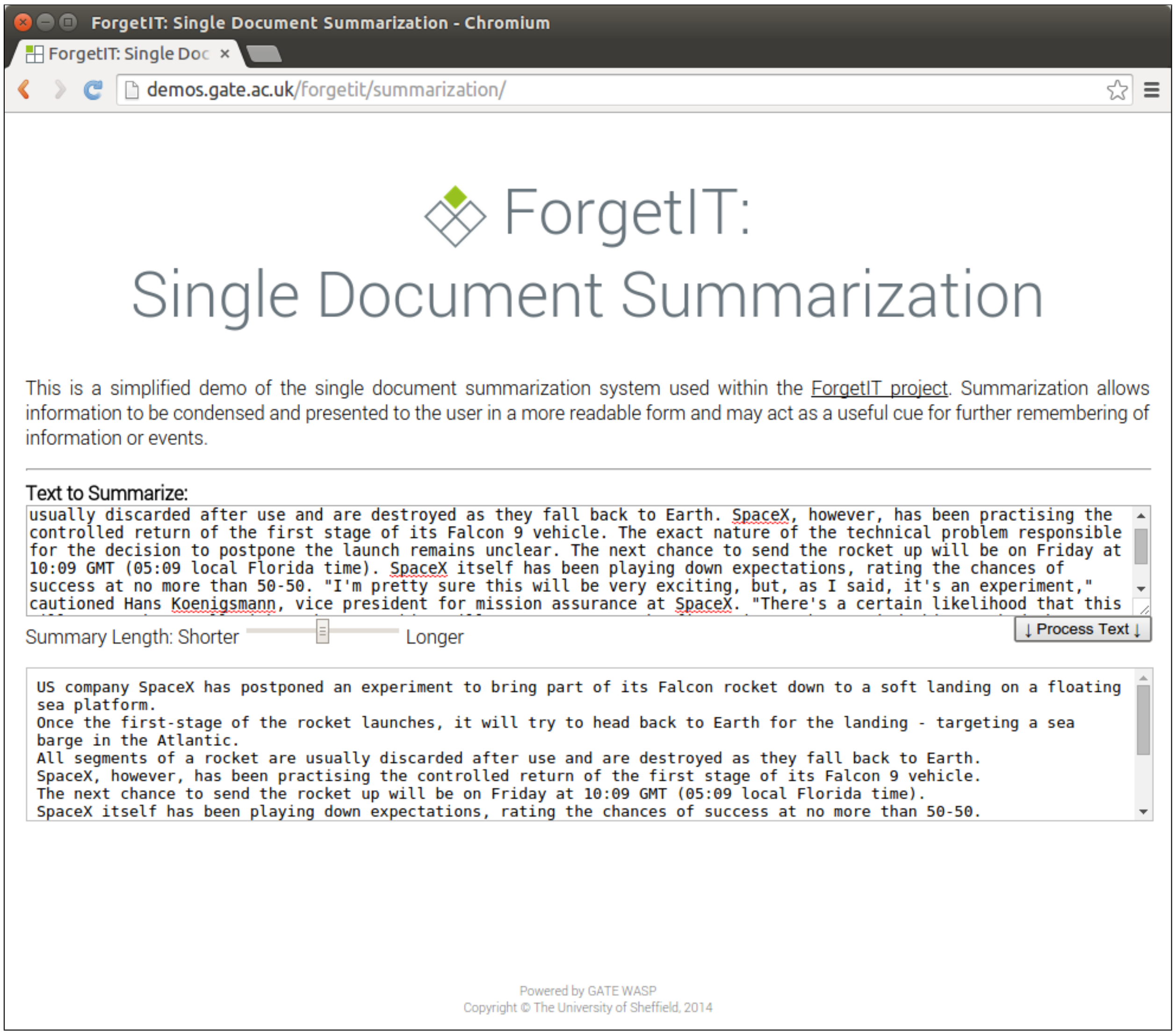
The approach has also been made available as a web service, with an interactive demo (Fig. 6), using GATE WASP (see Section 6) to allow for easy experimentation and integration within the use case tools. The service can be found at `http://services.gate.ac.uk/forgetit/summarization/`. The main characteristics of the software can be seen in Table 3.

As well as the standard configuration parameters provided as part of the GATE WASP service wrapper, the summarization size can also be configured using the `compression` parameter. The size of the summary can be set in one of two ways:

- if the value is between 0 and 1 then this is interpreted as a percentage (i.e. 0.1 is 10%) and the summary is produced by selecting that percentage of sentences from the original document.
- if the value is greater than 1 then it (after rounding to the nearest whole number) is interpreted as the number of sentences from the original document to use as the summary.

Table 3: Document summarization plug-in: software technical details

Functional description	Single document summarization
Input	text
Output	text
Limitations/Scalability	Currently only one document at a time
Language/technologies	Java, GATE, and SUMMA
Hardware Requirements	Any HW with Java support and sufficient RAM
OS Requirements	Any OS with Java support


ForgetIT: Single Document Summarization - Chromium
ForgetIT: Single Doc
demos.gate.ac.uk/forgetit/summarization/
ForgetIT:
Single Document Summarization

This is a simplified demo of the single document summarization system used within the ForgetIT project. Summarization allows information to be condensed and presented to the user in a more readable form and may act as a useful cue for further remembering of information or events.

Text to Summarize:

```
usually discarded after use and are destroyed as they fall back to Earth. SpaceX, however, has been practising the
controlled return of the first stage of its Falcon 9 vehicle. The exact nature of the technical problem responsible
for the decision to postpone the launch remains unclear. The next chance to send the rocket up will be on Friday at
10:09 GMT (05:09 local Florida time). SpaceX itself has been playing down expectations, rating the chances of
success at no more than 50-50. "I'm pretty sure this will be very exciting, but, as I said, it's an experiment,"
cautioned Hans Koenigsmann, vice president for mission assurance at SpaceX. "There's a certain likelihood that this
```

```
US company SpaceX has postponed an experiment to bring part of its Falcon rocket down to a soft landing on a floating
sea platform.
Once the first-stage of the rocket launches, it will try to head back to Earth for the landing - targeting a sea
barge in the Atlantic.
All segments of a rocket are usually discarded after use and are destroyed as they fall back to Earth.
SpaceX, however, has been practising the controlled return of the first stage of its Falcon 9 vehicle.
The next chance to send the rocket up will be on Friday at 10:09 GMT (05:09 local Florida time).
SpaceX itself has been playing down expectations, rating the chances of success at no more than 50-50.
```

Powered by GATE WASP
Copyright © The University of Sheffield, 2014

Figure 6: Demo interface for easy experimentation

4.5 Discussion

Summarization can be compared with the way in which human memories preserve pertinant details but allow extraneous information to fade over time. The level of compression applied during summarization alows us to mimic this process and we expect it to be a useful technique that will benefit the use case tools. Further development, tuning, and evaluation will take place over the next 12 months as the service is integrated within the user tools and the exact use of summarization within those tools becomes clearer.

5 Text analysis for semantic text composition

5.1 Problem statement

Semantic text composition aims at improving the process of text composition by building on the semantics of the text being composed rather than its syntactic features. From a ForgetIT point of view, this contributes to preservation by means of early contextualization.

As pointed out in D4.2, we have developed Seed, a semantic text editor capable - among other things - of recognizing entities (e.g. persons, countries, organizations, locations) in text. It is also capable of annotating the text with the discovered entities with little user intervention. Annotations add information about the entities from personal knowledge (i.e. PIMO) or from public knowledge sources (e.g. DBPedia [Lehmann et al., 2015] and Freebase [Bollacker et al., 2008]).

Reducing the mental load on the user during annotation of the text being composed is essential yet challenging. Therefore, in D4.3 we focus on discovering relations between candidate entities in the text. Those relations we extract are important for the following main reasons:

- Relations between entities in the text provide clues for better entity disambiguation.
- Relations between entities in the text on one side and others in the personal knowledge model of the user allow for further contextualization.

Addressing those two points is expected significantly to reduce the need for user intervention required to disambiguate entities or explicitly annotate the text.

5.2 ForgetIT approach

In our effort to help users naturally author texts while utilizing the knowledge extracted from content to enrich it, Seed attempts to automatically annotate entities (i.e. important things like persons, locations, institutions ... etc) mentioned in the text. When it is not possible to disambiguate a recognized entity with certainty, Seed allows users to quickly review and confirm/reject the suggestions for disambiguation by themselves.

Despite the value of automatic suggestions, user intervention still incurs a certain amount of mental load, which may make the text composition experience feel less natural.

In order to further decrease the required user intervention we focus, in D4.3, on utilizing knowledge about relations between entities mentioned in the text.

In comparison to the initial Seed version reported in D4.2, the users perceive the improvement through observing one or more of the following:

- getting more automatically disambiguated entity annotations

- getting less suggestions per annotation in case of an ambiguous annotation
- getting the same number of suggestions in the case of an ambiguous annotation but better reordered

5.2.1 Knowledge based relation extraction

In D4.2, Seed analyzed the text and recognized references to named entities. Using the Linked Open Data (LOD) and PIMO components, it then tried to annotate the mentions with entities from various knowledge repositories. Depending on the results of search in a knowledge base such as Freebase, Seed may fail to automatically disambiguate an entity reference. Therefore, it offered the user a list of suggestions from which the user can confirm or reject entity disambiguation offer.

In D4.3 we attempt to improve disambiguation by searching the source knowledge base for relations between an individual entity reference's suggestions and other non-ambiguous entities mentioned in its context in the text. Considering the relations between the involved entities, Seed assigns the results scores, reorders them, and selects candidates among the top ranking entity suggestions. This may result in an completely disambiguated entity or may improve the quality of the disambiguation suggestions.

5.2.2 Coreference resolution for improved relation extraction

In some texts, the number of entities mentioned is small due to short length or because the focus of the text is on a few entities even if the text is lengthy. In such cases, the benefit of applying relation extraction techniques such as those previously mentioned, is limited. We tackle this problem by using coreference resolution as a pre-processing step. This consequently introduces more mentions of entities in the text which can be used for relation extraction.

5.3 Experimental evaluation and comparison

Due to the difficulty in arranging for a user study with test subjects producing texts over a considerable amount of time, we have performed a proof of concept evaluation as follows:

1. Using a set of Wikipedia article excerpts of average length 106.8 words and average number of unique entity mentions per text of 8.
2. We manually ambiguated the mentions of at least one entity in the text.
3. We chuncked the text into sentences and incrementally fed it into Seed's back-end (i.e. the NLP and LOD components).

4. We saved the list of suggestions after adding every additional sentence.

5. We compared the contents and order of entity suggestions after adding every additional sentence to see how addition of more contextual entity mentions in the text improves the quality of the suggestions.

In 83.33% of the set of texts, the order and contents of the suggestions list changes. 52% of the changes resulted in a better ordering of the list of suggestions, meaning that the actually meant entity suggestion has moved up in the list. 35% of the changes resulted in a disambiguated entity using relative ranking. In the rest of the cases, the added contextual entity mentions shared a similar if not higher relation count with the entity mention to be disambiguated. For those cases, another additional step of disambiguation would be required to further improve the order and contents of the suggestions list.

5.3.1 Comparison to D4.2

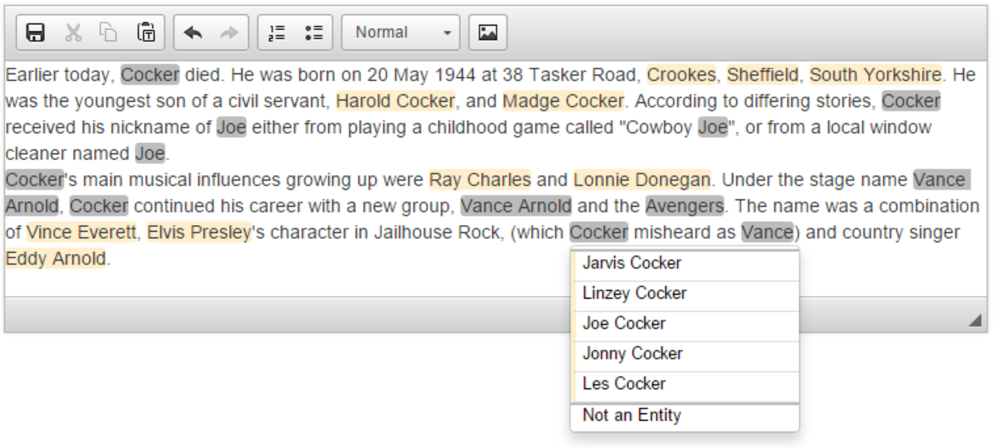
To illustrate the new changes in D4.3 in contrast to D4.2, we will use the following sample text passage adapted from Wikipedia about Joe Cocker, a musician and singer:

“Earlier today, Cocker died. He was born on 20 May 1944 at 38 Tasker Road, Crookes, Sheffield, South Yorkshire. He was the youngest son of a civil servant, Harold Cocker, and Madge Cocker. According to differing stories, Cocker received his nickname of Joe either from playing a childhood game called “Cowboy Joe”, or from a local window cleaner named Joe. Cocker’s main musical influences growing up were Ray Charles and Lonnie Donegan. Under the stage name Vance Arnold, Cocker continued his career with a new group, Vance Arnold and the Avengers.”

Taking one entity reference as an example, Seed analyzed the text and figured out references to “Cocker” probably point to a an entity of the type person. Depending on the results of search in a knowledge base such as Freebase, Seed may fail to automatically disambiguate the entity reference. Therefore, it may offer the user the following unordered list of suggestions: (Jarvis Cocker, Linzey Cocker, Joe Cocker, Jonny Cocker and Les Cocker).

In D4.3, we attempt to further disambiguate by searching the source knowledge base for relations between the suggested entities for the entity reference “Cocker” and other non-ambiguous entities mentioned in the text. Considering the relations between the involved entities, seed assigns the results scores, reorders the results, and selects candidates among the top ranking entity suggestions. As a result it may offer the user the following ordered list of suggestions (Joe Cocker, Jarvis Cocker, Mac Cocker, John Cocker and Mark Cocker)

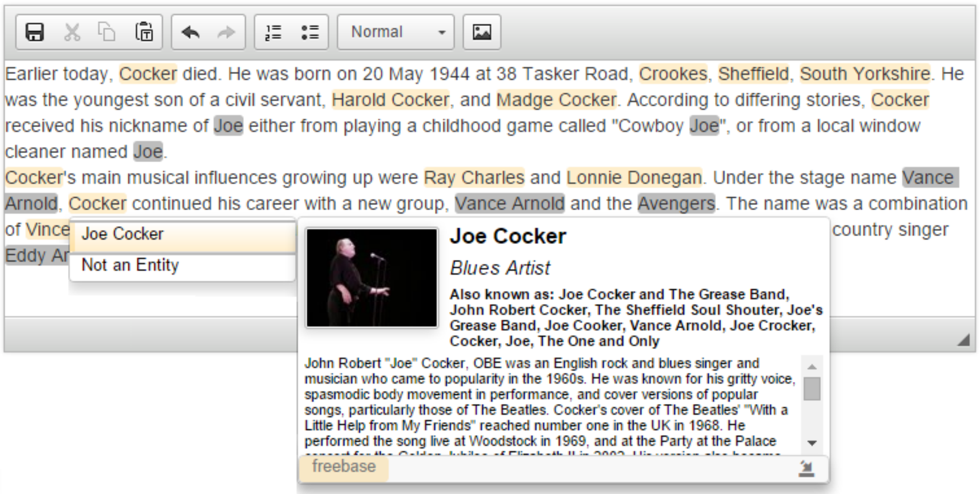
Fig. 7 and Fig. 8 depict sample user interaction before and after integrating relation extraction improvement into Seed.



Earlier today, Cocker died. He was born on 20 May 1944 at 38 Tasker Road, Crookes, Sheffield, South Yorkshire. He was the youngest son of a civil servant, Harold Cocker, and Madge Cocker. According to differing stories, Cocker received his nickname of Joe either from playing a childhood game called "Cowboy Joe", or from a local window cleaner named Joe.
Cocker's main musical influences growing up were Ray Charles and Lonnie Donegan. Under the stage name Vance Arnold, Cocker continued his career with a new group, Vance Arnold and the Avengers. The name was a combination of Vince Everett, Elvis Presley's character in Jailhouse Rock, (which Cocker misheard as Vance) and country singer Eddy Arnold.

Jarvis Cocker
Linzey Cocker
Joe Cocker
Jonny Cocker
Les Cocker
Not an Entity

Figure 7: Sample entity disambiguation as in D4.2



Earlier today, Cocker died. He was born on 20 May 1944 at 38 Tasker Road, Crookes, Sheffield, South Yorkshire. He was the youngest son of a civil servant, Harold Cocker, and Madge Cocker. According to differing stories, Cocker received his nickname of Joe either from playing a childhood game called "Cowboy Joe", or from a local window cleaner named Joe.
Cocker's main musical influences growing up were Ray Charles and Lonnie Donegan. Under the stage name Vance Arnold, Cocker continued his career with a new group, Vance Arnold and the Avengers. The name was a combination of Vince country singer Eddy Ar

Joe Cocker
Not an Entity

Joe Cocker

Blues Artist

Also known as: Joe Cocker and The Grease Band, John Robert Cocker, The Sheffield Soul Shouter, Joe's Grease Band, Joe Cooker, Vance Arnold, Joe Crocker, Cocker, Joe, The One and Only

John Robert "Joe" Cocker, OBE was an English rock and blues singer and musician who came to popularity in the 1960s. He was known for his gritty voice, spasmodic body movement in performance, and cover versions of popular songs, particularly those of The Beatles. Cocker's cover of The Beatles' "With a Little Help from My Friends" reached number one in the UK in 1968. He performed the song live at Woodstock in 1969, and at the Party at the Palace

Figure 8: Sample entity disambiguation as in D4.3

5.4 Software implementation

In D4.3, we focus on server components of Seed shown in Fig. 9, namely the NLP, LOD and PIMO components. The front-end editor component remains unchanged in that regard. The software implementation has the same requirements and limitations and can be used in the same way with the previous Seed version presented in Section 4 of [D4.2, 2014]. The main characteristics of the software can be seen in Table 4.

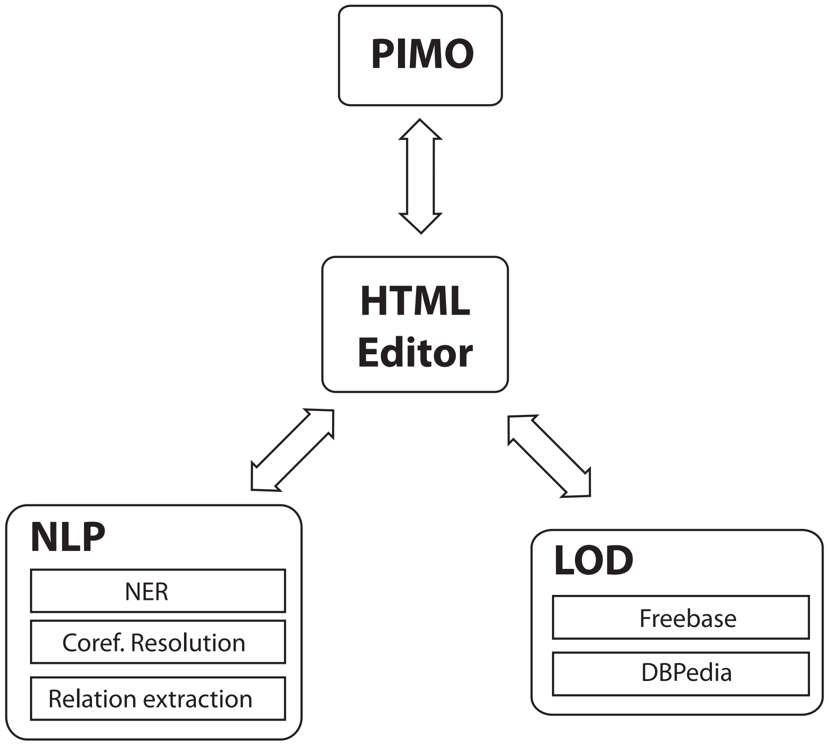


Figure 9: *seed* Components

Table 4: Semantic text composition tool: software technical details

Functional description	A semantic text composition tool
Input	Rich text
Output	Annotated rich text with information about its content
Language/technologies	HTML, JavaScript, Java
OS Requirements	Server: Any OS with Java support Client: Any OS with an internet browser

6 Easy Integration of Text Services using GATE WASP

6.1 Problem statement

There are many text processing applications which it would be useful to be able to include within the ForgetIT use case tools, and many of these can be quickly and easily developed using GATE [Cunningham et al., 2011]. This includes domain specific approaches (text from social media is very different to that found in a newspaper [Bontcheva et al., 2013]), language specific tools (currently text processing in ForgetIT has focused on English) as well as techniques for co-reference resolution, and date and measurement normalization which can be used to produce a richer semantic representation to aid in indexing and searching of large collections. The complexities of these tools could be hidden behind the middleware interfaces being developed within WP8, but sometimes it is is necessary or beneficial to be able to move the processing closer to the data and to analyse text within the use case tools themselves. While GATE can be embedded within other applications via the Java API, it has become clear that this is not always an ideal approach due to scaling and the web-based nature of some of the use case tools. To this end we have developed GATE WASP (Web Application Service Provider) to allow the rapid deployment and integration of GATE applications as web services. This same approach can also be used to integrate GATE based applications easily within the ForgetIT middleware, as a single integration component could support multiple different applications.

6.2 ForgetIT approach

Unlike the other tools and techniques reported in this deliverable, not only does GATE WASP not build upon previously reported work, it is also not a tool that warrants independent evaluation etc. (tools hosted through it will of course be properly evaluated) and so this discussion will focus on the approach taken to expose other tools reported in this and other ForgetIT deliverables.

The driving force behind the development of GATE WASP is to enable the fast development, deployment, and integration of text processing algorithms with the tools being developed for the use cases within both WP9 and WP10. In both work packages there is a desire to perform at least some of the text analysis close to the natural location of the data for reasons of speed and efficiency. While GATE can be embedded within other applications via the Java API, this can be quite complex, especially for developers who are new to GATE. Basic embedding of GATE does not cover issues of parallelism and scaling and is clearly not appropriate for use with tools developed in languages other than Java. This is particularly important in ForgetIT as the WP10 use case builds upon TYPO3, which is PHP-based application.

6.3 Software implementation

To maximize the situations in which GATE WASP can be used to integrate text analysis algorithms within other tools, we have choosen to take a web service approach. As GATE is written in Java, exposing the API as a web service can be achieved using the Apache Tomcat web server[5]. GATE on its own is not designed to handle concurrent requests, although a lot of work has gone into ensuring that it can be used in long running processes such as web services. To allow for concurrent requests (something that is highly likely in both use case scenarios) we utilise the pooling support provided by Spring[6] software framework.

Full usage instructions for deploying GATE WASP are outside the scope of this deliverable, but the basics are as follows. The standard service requires the text you wish to process to be passed using one of the following three request parameters:

Parameter	**Supported Request**	**Description**
text	GET or POST	Plain text to process
url	GET or POST	The URL of a document to process
file	POST	A file to process

Each deployed application may also offer further configuration parameters that can be used to control the processing or output of the application. For example, the single document summarization application (described in Section 4) can be configured to retrun summaries of different lengths using the `compression` parameter. The main characteristics and requirements of GATE WASP are summarized in Table 5.

Table 5: Overview of GATE WASP Requirements

Functional description	Easy integration of text analysis services
Input	textual content
Output	configurable based on required output
Limitations/Scalability	Currently only supports single documents
Language/technologies	Java, GATE, Tomcat
Hardware Requirements	Dependent upon application
OS Requirements	Java support

6.4 Discussion

GATE WASP is still under active development and has not yet been publicly released. The current version has been shared with project partners and their feedback has and will continue to contribute to it's future development. It is hoped that a full public release will

[5]http://tomcat.apache.org/
[6]http://spring.io/

be made during 2015. Even though it is still under development, it has been successfully used to provide easy access to a number of text applications both within ForgetIT and other projects[7].

GATE WASP has been used to allow the integration of both the simplification and summarization approaches documented in this deliverable and D6.3:

- The linguistic simplification algorithm, reported in Section 3, is available as a service (along with a demo) at `http://services.gate.ac.uk/forgetit/simplification/`
- The single document summarization approach , detailed in Section 4, is available as a service (along with a demo) at `http://services.gate.ac.uk/forgetit/summarization/`
- The real-world aspects of contextualization, see D6.3, are perfomed using YODIE which is joint work with the TrendMiner project[8]. A demo and online service version of YODIE can be found at `http://services.gate.ac.uk/yodie/`

Two services, which use GATE WASP, have also been made public through the Decarbonet project[9]:

- ClimaTerm, available at `http://services.gate.ac.uk/decarbonet/term-recognition/`, aims to annotate documents with terms related to climate change. Where appropriate, these terms are matched and linked to the instance of that term in relevant Linked Open Data ontologies.
- ClimateMeasure, available at `http://services.gate.ac.uk/decarbonet/indicators/`, aims to extract useful indicators of climate change such as “energy use”, “carbon pollution”, etc. for particular locations, together with measurable effects such as percentages, measurements etc. and the relevant dates.

The adoption of WASP outside of the ForgetIT project shows that it has wide appeal and has already proven to be useful in reducing the time required to make demo services available and as an aid to integration with other systems.

[7]There are more services using GATE WASP than document here but a number of the projects are of a commercial nature and are currently covered by non-disclosure agreements which prevent their inclusion in this discussion.

[8]`http://www.trendminer-project.eu/`

[9]`http://www.decarbonet.eu/`

7 Visual concept detection and online training

7.1 Problem statement

In D4.2 we introduced the preliminary ForgetIT visual feature extraction and concept detection approaches. The steps followed for feature representation of each image are: i) dense sampling for keypoint selection, ii) SURF and color variations of SURF (RGBSURF, OPPONENT SURF) descriptor are extracted on each keypoint, iii) $1x3$ spatial pyramid decomposition and iv) Bag-of-Words (BoW) encoding for the final image representation. In this deliverable, we experimented with Vector of Locally Aggregated Descriptors (VLAD) [Jégou et al., 2010a] encoding, that is a more elaborate feature encoding process. We compared VLAD encoding with BoW and the results shown that VLAD outperforms BoW. Thus, we adopted VLAD encoding for the descriptor-based representations of the images and we have included it in our updated service. Moreover, we replaced the SURF, RGBSURF and OPPONENT SURF with the corresponding ORB [Rublee et al., 2011] (Oriented FAST and Rotated BRIEF), RGBORB and OPPONENT ORB descriptors. ORB is a binary local descriptor, thus due to its compact size and its low storage needs it is a viable alternative to the non-binary descriptors previously currently used for this task.

Another issue of concept detection that has significant impact at both personal and organizational scenarios is the part of training concept classifiers. In D4.2, the training of the SVM-based classifiers was performed using the manually-generated training examples for 346 concepts of TRECVID 2012 [Over et al., 2012] Semantic Indexing Task. The human effort needed for annotating such a large amount of data is of course significant, and also limits the scalability of concept detection. To overcome these limitations we propose an automated approach of collecting positive training data from the Web. Web images are often annotated with tags, generated by the creators of the images or other Internet users. The use of such annotated images for automatically assembling training corpora for visual concept learning is a promising direction.

Several approaches have been introduced in the literature dealing with the issue of sampling positive and negative images from the web. Negative image sampling mostly follows a random sampling, as described in [Li and Snoek, 2009], or by selecting the most misclassified images of a concept as described in [Li et al., 2011]. Focusing on positive training data, works on automatic sampling of Web images include [Zhu et al., 2012],[Schroff et al., 2011],[Li and Fei-Fei, 2010],[Li et al., 2009].

7.2 ForgetIT approach

7.2.1 Method overview

Most of the related works introduced in the literature follow the approach illustrated in Fig. 10b, according to which: Given a target concept: i) one or a few queries of the target concept term are constructed, ii) a Web image search engine is queried with the constructed concept terms, iii) a large amount of images is downloaded, which contains both truly positive but also noisy images, iv) a ranking or filtering algorithm is applied on the returned images in order to discard those which are false positives and v) the final training set is select. On the contrary, our approach shifts the burden of selecting a high-quality set of positive samples to the Web image search engines, by exploiting the fact that when the number N of returned images by a Web image search engine receives a significantly low value, the N top results of a web query are almost always correct. We confirmed this fact by performing some tests; in Fig.11 the average precision (AP) versus the number of returned images is shown for two Web searches. The Google [10] image search engine is tested by querying for concept "bird" (Fig. 11a) and the Flickr [11] image search engine by querying for concept "animal" (Fig.11b). The top 1000 results returned by each search engine were manually evaluated. The shape of the curve shows that the precision decreases as the number of returned images increases, as expected. However, downloading a very small number of images per target concept and keeping the average precision high (above 90%), we significantly limit the number of positive samples that we would collect. To alleviate this we propose a multi-query formulation technique which formulates K queries, and downloads N top-ranked images for each one, Fig. 10b. The queries which derive from three different query formulation sets, the *Translation set*, *wordNET set* [Fellbaum, 1998] and *Language model set*, are used to query three different image search engines and collect the returned examples which serve as the training corpus. The advantages of the method derive from the low complexity of the method and the accuracy of the resulting concept detectors.

In Fig. 12, the proposed method's steps are shown, and a detailed description of each query set is presented below.

i. Translation set of queries

The first set of queries is the "Translation set" which is a method that translates the concept term into fifteen languages[12]. Then each one of the translated terms is used to query the Google and Bing image search engines. In each case the Google or Bing [13] domain of the specific country is called, in order to avoid receiving duplicate results. Flickr image search does not provide such functionality, thus it is not used in this query set. The maximum number of images that we can collect by applying the *Translation set* of queries for a target concept is calculated by:

$$T_{transl} = E * N * L$$

[10] `http://www.google.com`

[11] `http://www.flickr.com`

[12] German, French, Greek, Italian, Spanish, Chinese, Indonesian, Russian. Romanian, Bulgarian, Danish, Dutch

[13] `http://www.bing.com`

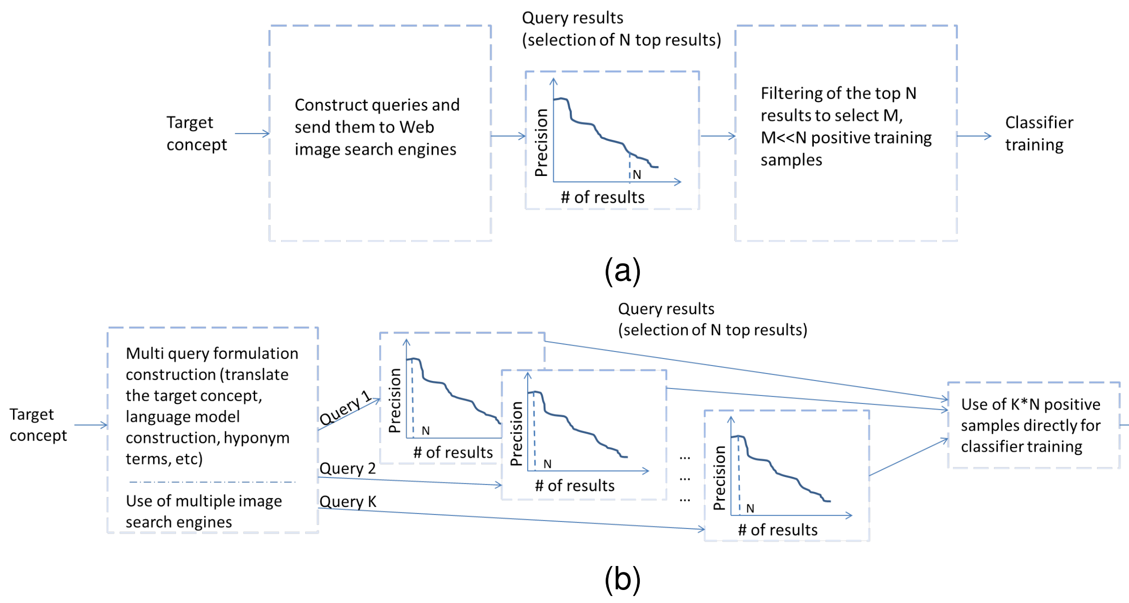

Target concept
Construct queries and send them to Web image search engines
Query results (selection of N top results)
Precision
N
# of results
Filtering of the top N results to select M, M<<N positive training samples
Classifier training
(a)
Multi query formulation construction (translate the target concept, language model construction, hyponym terms, etc)
Use of multiple image search engines
Query 1
Query 2
Query K
Use of K*N positive samples directly for classifier training
(b)

Figure 10: (a) Typical approach for collecting positive training samples from the Web, for visual concept detector training. Given the target concept, a web image search engine is queried so as to return the N top-ranked results (N being in the order of hundreds), which are then pruned in order to identify the M, $M << N$ images that will serve as positive training samples. (b) Proposed approach. Given the target concept, a multitude of queries are constructed and are submitted to several image search engines. The N top-ranked results (N receiving much lower value in this case, typically around 20) of each query are retained, and a set of $K * N$ (K being the number of query result-sets) directly serves as the positive training set of a visual classifier, without the need for further pruning

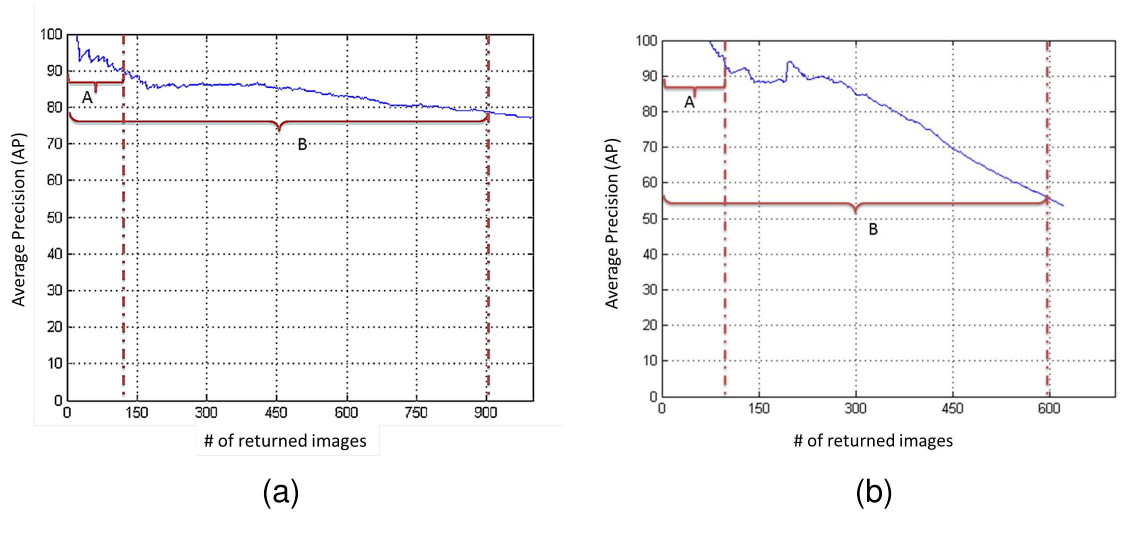


Figure 11: Average precision of returned images for Google and Flickr image search (a) concept animal queried Flickr and (b) concept bird queried Google

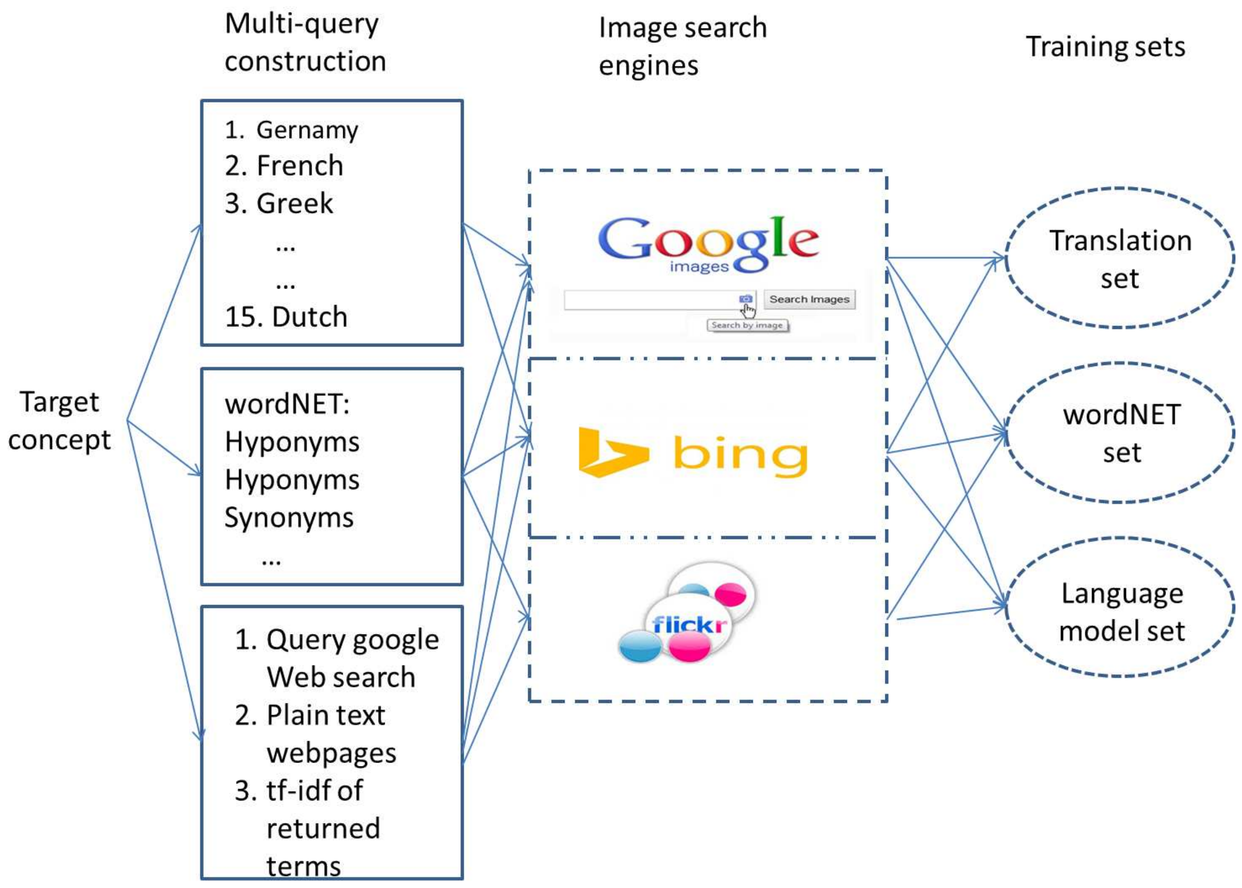


Figure 12: Proposed method sampling steps

where E is the number of different image search engines that we take into account ($E = 2$), N is the number of top-ranked images that we retain and L is the number of different languages ($L = 15$). Although we use different domains to avoid duplicate images we will still receive some of them. Specifically, after several experiments we conclude that the amount of duplicate images per target concept varies between 5% and 45% of T_{transl}.

ii. wordNET queries

WordNet [Fellbaum, 1998] is a lexical database for the English language. It groups English words into sets of synonyms, called *synsets*, which are connected to other *synsets* by means of semantic relations. Hyponyms and hypernyms, which are such categories, can be explained as: *Y is a hyponym of X if every Y is a (kind of) X (e.g. dog is a hyponym of canine)* and *Y is a hypernym of X if every X is a (kind of) Y (canine is a hypernym of dog))*. A text query is made by combining the selected terms with the target concept using AND (&). Although the extracted terms should be related to the target concept, there are cases in which the term is not semantically related to the concept and the images downloaded by querying this term do not indicate the requested concept. For example, for the concept "dog" a synset term returned by the wordNet database is "hot dog". Images downloaded by querying "dog AND hot dog" will be images that do not indicate the animal dog, see Fig. 13. These images are considered as false positive images and should be discarded from the training set. In order to filter and reject the extracted terms which will return false positive images we use use two similarity distances. The *umbc* one, proposed in [Han et al., 2013], which combines the use of thesaurus (e.g. WordNET) and statistics from a large corpus for computing word similarity, and the *easyesa* one, proposed in [Carvalho et al., 2014], which uses Wikipedia commonsense knowledge base on statistical analysis of the co-occurrence of words in the text. In order to accept an extracted wordNET-term, its *umbc* and *easyesa* distances with the concept in question should be above thresholds thr_{umbc} and $thr_{easyesa}$. The value of the thresholds is chosen heuristically by experimenting with five concepts and used for all concepts throughout all our experiments.

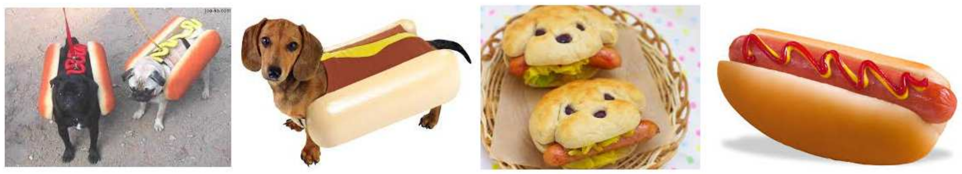


Figure 13: Images downloaded by querying Google image search with text query "dog & hot dog"

iii. Language model set of queries

The third set of queries is using Web text search results. Apart from the plentiful image items, the Web is flooded with text data and we use this information in order to retrieve more terms related to the concept in question. These terms are combined with the target concept and images are downloaded by querying the different image search engines. Specifically, given a target concept, i) we query Google Web text search with the target concept, ii) the W first Web pages are returned, iii) the text of the returned Web pages is extracted, iv) term frequency $-$ inverse document frequency (TF.IDF) [Ramos, 2003], which is intended to reflect how important a word is to a document in a collection or corpus, is applied on all documents and a vector of related or not-related words is returned. Terms with TF.IDF higher than a threshold t_{tfidf} are accepted. The value t_{tfidf} is again heuristically chosen by conducting similar experiments to those described above for thr_{umbc} and $thr_{easyesa}$. We also assume that the chosen threshold value is suitable for all concepts and is used in all our experiments. However, an additional filtering of the terms is required before querying with them the Web image search engines. This pruning is required in order to discard terms which are related to the target concept but do not visually describe it. For example for concept *airplane* the returned terms with high TF.IDF are:

airplane, plane, aircraft, jet, paper airplane, rc airplane, flight, cockpit, flying, **passenger**, **pilot**, **passenger cabin**, aviation, airport, off-airport, airline

As we can notice, the terms which are bold, are related terms but will return images that will typically not depict the target concepts. A further assessment of the terms is conducted using the similarity distances thr_{umbc} and $thr_{easyesa}$ together with their previously set thresholds.

7.2.2 Advances in comparison to the previous version

Our previously introduced version of concept detection method was based on SURF descriptors and BoW encoding. We replaced them with ORB descriptor and VLAD encoding. This combination allowed us to develop concept detectors exhibiting significantly increased accuracy and low computational requirements.

Furthermore, the previous concept detectors were built using manually-generated data as training examples. Specifically, 346 concept classifiers had been trained from the available labelled examples of TRECVID dataset. In order to expand the number of trained concepts and also avoid the human effort needed for labelling the training examples, we shifted to an automatic approach of collecting training examples from the Web.

7.3 Experimental evaluation and comparison

7.3.1 Feature extraction and representation

The experiments were performed on the TRECVID 2013 Semantic Indexing (SIN) dataset [Over et al., 2013]. We evaluate the 38 concepts that were evaluated as part of the TRECVID 2013 SIN Task in terms of Mean Extended Inferred Average Precision (MX-infAP) [Yilmaz et al., 2008], which is an approximation of the Mean Average Precision (MAP) suitable for the partial ground truth that accompanies the TRECVID dataset [Over et al., 2013].

From the experiments conducted in D4.2 [D4.2, 2014] the configurations that achieved the best performance and were implemented in the concept detection REST service are: the three SURF descriptors (SURF, RGBSURF and Opponent SURF) combined with dense sampling and soft assignment BoW encoding. The performance of concept detection for BoW and VLAD encoding is presented in Table 6 for the above three configurations and their combination using late fusion (averaging). The results of the same VLAD encoding when using the (faster to compute) ORB descriptors are also shown in the same table.

7.3.2 Collecting positive training samples from the Web

We experimented with a pool of forty concepts and an evaluation set of 17881 images from the ImageNET dataset [Russakovsky et al., 2014]. The positive training samples are collected from the Web using Google, Bing and Flickr image search engines. Negative

Table 6: Comparison of BoW and VLAD encoding for the SURF descriptor, and of VLAD encoding for the ORB descriptor

	MXinfAP (%)			MXinfAP (%)
descriptor	BoW	VLAD	descriptor	VLAD
SURF	6.97	14.68	ORB	11.43
RGBSURF	7.86	15.99	RGBORB	13.58
Opponent SURF	7.33	15.26	Opponent ORB	12.73
SURF combination	12.87	19.48	ORB combination	17.45

sample selection is out of the scope of our study; thus, the commonly-used solution of random selection is adopted and approximately 5000 images are used as the negative training samples. The same negative samples are used throughout the experiments. For evaluating a concept detector, this is applied to the entire evaluation set, the images of this set are ordered according to the output score of the detector (in descending order) and Average Precision (AP) [Smeaton et al., 2006] is calculated, Mean Average Precision (MAP) is calculated as the mean value of the AP across all concepts.

Three sets of experiments are conducted in order to evaluate the proposed approach:

- Use of the different training sets: The proposed multi-query approach collects positive training samples using three sets of queries. Thus, we evaluate the proposed approach in two ways: i) Early fusion, where all collected positive training samples are considered as a training set and ii) Late fusion: a separate training set is formed with the positive training samples returned by each of the query sets. A final result is extracted by applying late fusion (averaging) on the results of the individual sets. In Table 7, the results for $N = 8$, $N = 16$, $N = 24$ and $N = 32$ for the above cases are presented. The Late fusion cases outperforms all others.
- Optimal number of top-ranked images to download: We experimented with different values of N. As discussed, the value of N should be low ($<$ 50) in order to exploit the precision of the Web image search engines. Experiments for $N = 8$ (*Top8*), $N = 16$ (*Top16*), $N = 24$ (*Top24*) and $N = 32$ (*Top32*) are presented in Table 7 showing that the optimal value of N is for $N = 24$ with MAP 0.41.
- Comparison with baselines and a state-of-the-art approach: We compared our proposed approach with:
  - Baseline: a query of the target concept term is sent to the Flickr image search engine (Flickr is used here as the image search engine because it is widely used in many related works) and a fixed number X of examples is returned. The value of X is empirically set to 600, as is often the case in the literature. All returned images are considered positive training examples, and no further pruning is applied on them.
  - ImageNet examples: ImageNet has more than 10000 concepts, each of them with a manually annotated set of examples. We downloaded the manually annotated images of the target concepts and trained the classifiers.

– Zhu et al. [Zhu et al., 2012]: We implemented the sampling approach introduced in [Zhu et al., 2012] and trained SVM-based classifiers (following the feature extraction and representation discussed above) with the selected examples.

The results of all evaluated approaches are shown in Table 8. As expected, the Baseline approach performed worst with MAP equal to 0.252 and the ImageNet examples approach best with MAP of 0.46. The state-of-the-art approach of Zhu et al. [Zhu et al., 2012] performed better than the Baseline with a MAP of 0.3 and worse that our proposed approach which outperformed both Zhu et al. [Zhu et al., 2012] and Baseline with a MAP of 0.41. The proposed approach fall behind the ImageNet trained classifiers and this can be explained by the different domains of training and testing data and the noisy positive training examples which remained in the training sets of the concept detectors.

Table 7: MAP for different training sets and top-ranked images $N = 8, 16, 24$ and 32. Optimal value is marked in bold.

	MAP			
Training set	Top8	Top16	Top24	Top32
Translation set	0.347	0.363	0.37	0.316
wordNET set	0.308	0.331	0.35	0.34
Language model set	0.281	0.305	0.33	0.32
ForgetIT approach - Early fusion	0.35	0.36	0.37	0.35
ForgetIT approach - Late fusion	0.3872	0.399	**0.41**	0.4

Table 8: Comparison of baseline, manual, state-of-the-art and proposed method approaches in terms of MAP

Approach	MAP
Baseline	0.252
ImageNet examples	0.46
Zhu et al. [Zhu et al., 2012]	0.3
ForgetIT approach	0.41

7.4 Software implementation and integration

The ORB descriptor and VLAD encoding are C++ implementations and updated the already existing REST service for concept detection.

The service call for both GET and POST methods are as follows:

- GET method

  `http://multimedia.iti.gr:8080/ForgetITImageAnalysis/EXTRACTOR/methodGET?imagePaths=’’&method=’’`

URL paths of the images are separated with special character ∼.

- POST method

  `http://multimedia.iti.gr:8080/ForgetITImageAnalysis/EXTRACTOR/methodPOST`

  Data should be encoded as: application/x-www-form-urlencoded and URL paths of the images are separated with special character newline (\n).

The input arguments are:

- *imagePaths*: url paths of the images of the collection (separated with special character depending on GET or POST call) or URL path of a compressed file (currently only .zip format is supported)
- *method*: `conceptF` or `conceptS`, where `conceptF` is executing one configuration thus it is faster but loses in accuracy and contrary, `conceptS` improves accuracy at expense of larger execution time.

A summary of the concept detection software's technical details is presented in Table 9.

Table 9: Software technical details for the ForgetIT concept detection method

Functional description	Concept detection
Input	An image or an image collection
Output	An XML file containing a vector of confidence scores for each image
Language/technologies	C++, JAVA Rest Service
Hardware Requirements	NVIDIA graphic card-CUDA drivers
OS Requirements	Windows

The sampling approach of positive training samples is implemented using java scripts. A zip file containing all required scripts to perform the sampling procedure and detailed instructions on how to call them are included in `http://www.forgetit-project.eu/en/downloads/workpackage-4/`

A summary of the software's technical details for Web data collection is presented in Table 10.

7.5 Discussion

It this Section the updated version of feature extraction and representation is presented as well as a new approach for collecting positive training examples from the Web for training concept detectors.

Table 10: Software technical details for the collection of positive training examples from the Web approach

Functional description	Sampling of positive training examples from the Web
Input	A concept term
Output	Concept classifiers
Language/technologies	Java
OS Requirements	Windows with Java support

8 Face / person clustering in image collections

8.1 Problem statement

Person clustering is the procedure that, given an image database, groups all the appearances of each person in different clusters. It can be used in order to find the dominant or main persons of the collection. This information can be employed in image collection summarization (e.g. to keep the images that contain a specific person/s or dominant persons that appear in large clusters) mainly in the personal preservation scenario.

Person clustering of photo collections is a challenging task. Several methods have been presented in the literature both for video and image media. The basic skeleton of most of the approaches is the following: initially faces are detected from the media, then features are extracted from the facial regions and finally, a clustering method is used in order to create the clusters. The fact that there are no constrains on the environmental parameters such as lighting, face position, orientation, image quality etc. makes this approach very weak. For this reason, side information is employed in many works in order to increase the clustering accuracy. For example, in video data face clustering, facial regions of a video shot are grouped in facial tracks by employing face tracking after face detection. Thus, instead of face similarity, face track similarity is employed. Since pose variation within a face track varies, it is more likely to find similar faces between face tracks than by considering only single faces. Besides facial info, other characteristics have been employed such as hair, clothing etc. Some face clustering methods for video are [Castrillón et al., 2011, Wu et al., 2013, Jaffré et al., 2004, El Khoury et al., 2010] and for photo collections [Sivic et al., 2006, Zhang et al., 2010, Zhang et al., 2014].

8.2 ForgetIT approach

8.2.1 Method overview

Since face clustering will be mainly applied in consumer photo collections, the face orientation, pose and illumination of each person will vary across the photos. Hence, face clustering using only facial characteristics is not adequate. Thus, other information such as hair and clothing is used in order to improve the method's accuracy. As reported in [Gallagher and Chen, 2008], average recognition rate in humans is increased significantly if clothing and face is available instead of just face.

However, if the image collection time span is long (larger than 4-5 hours), clothing might not remain the same. For this reason, the collection is initially clustered in time and then, within each time cluster, the algorithm searches for similar people using clothing and hair information. The procedure above results in small initial person clusters. After this, face cluster merging follows using clusters from the entire collection (without any temporal constrains). At this stage, merging is performed by employing facial information similarity

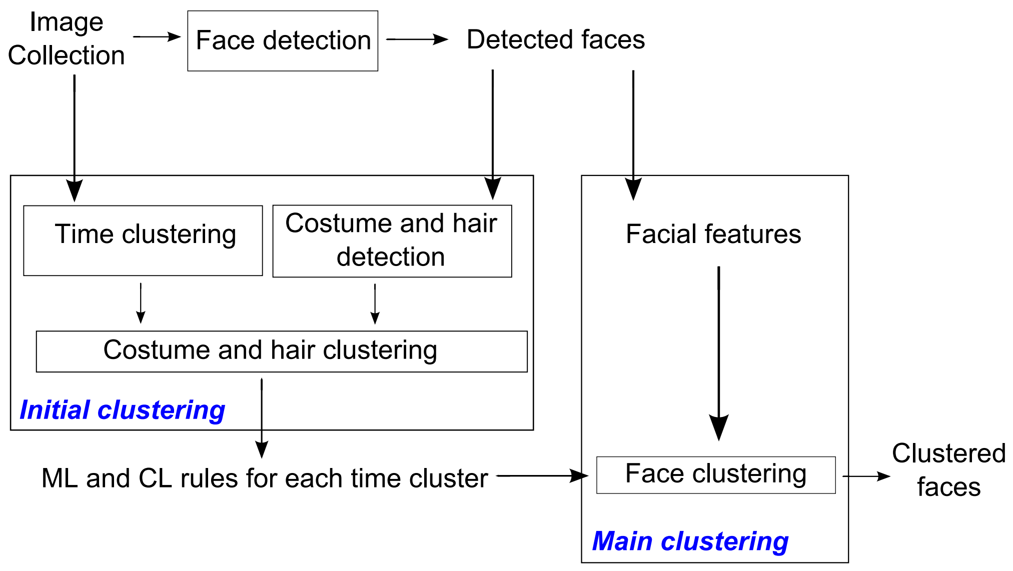


Figure 14: Face clustering method overview

between face clusters. The method overview is illustrated in Fig.14.

Initial clustering

The initial clustering is performed using on costume and hair similarity. The main idea of using costume similarity is that different costumes can be easier discriminated than faces, especially when the latter show significant pose, illumination or expression variations.

Clothing and hair detection. Clothing has been widely used for person authentication. In [Jaffré et al., 2004] clothing is used for automatic video content indexing where face detection is followed by clothing localization and person is recognized only from its clothing (compared to the one in the database). Based on the detected face locations, the body and the hair of each person is estimated. The persons' clothing region is located below face region, having width equal to 1.2 the face width and height equal to 1.4 the face height. Using a similar approach we estimate the hair region. The hair bounding box has the same width and half the height compared to the face one and its center is located in the top of the face bounding box. For the modelling of both hair and clothing we have used the hue and saturation channels of the HSV color space.

Since skin is almost always included in the clothing bounding box (usually parts of neck), the clothing bounding box is masked before modelling in order not to include skin colored pixels using the procedure described below. Similarly, hair bounding box is also masked in order to include hair colored pixels only. For both tasks we have used the method proposed in [Chen and Lin, 2007].

Time clustering. Costume similarity is useful only between photos taken at similar time. For this reason, the images are initially clustered according their capture time. Time has also been employed in [Choi et al., 2010, Zhang et al., 2014]. In [Choi et al., 2010] the notion of ”situation cluster” is proposed which is is defined as a group of photos having similar capture times and visual characteristics.

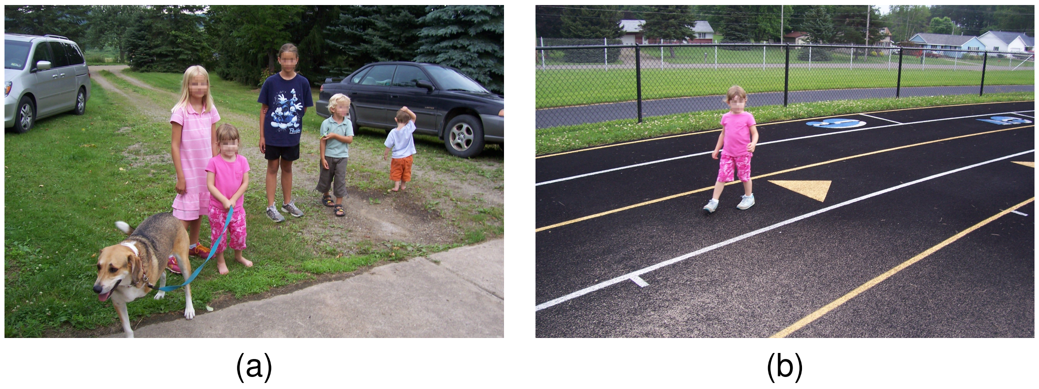


Figure 15: Consequtive images that are clustered in the same time-cluster (temporal difference: 35 min) - children faces have been pixelized due to privacy issues.

We have adopted a simple time clustering approach in which the images are ordered temporarily and time differences between consecutive images are calculated. If the temporal difference is above a threshold, then these images are grouped in different clusters. In this approach only time is employed, since we don't consider visual characteristics important, as also suggested in previous works (e.g. [Choi et al., 2010]). For example, in Fig. 15, two consecutive images of the Gallagher set [Gallagher and Chen, 2008] are illustrated, having different background but depicting the same person wearing the same clothing.

Let us assume that $I_C = \{I_1, I_2, ..., I_{N_I}\}$ is the input image collection, where N_I is the number of images in the collection. Let also $F_C = \{f_1, f_2, ..., f_{N_F}\}$ be the set of detected faces, where N_F is the number of detected faces. A face f_i that is the kth face of the jth image can be written as $f^{I(j,k)}$ and the set of faces belonging to image I_j can be written as F^{I_j}. After time clustering the images are grouped in clusters $I_{T_I(i)}$ and the faces are grouped in face sets $F^{I_{T_I(i)}}$.

Clothing and hair similarity. We consider that in each face set $F^{I_{T_I(i)}}$ similar persons can be matched using the clothing and hair information. The χ^2 distance between HS histograms is employed for distance calculation, for both clothing and hair regions. Before distance calculation, each histogram is filtered in order not to have large spikes and get more robust and reliable results. More specifically, the histogram indices are converted to a 2D disk (where H corresponds to angle and S to radius) in order to be perceptually uniform and then a gaussian filter is applied. Finally, gabor features for both costume and hair regions are extracted using five scales and eight orientations. Using the χ^2 distance of the gabor representations, costume and hair similarity is estimated.

Can not link rules. Taking into account that faces belonging in the same image can not correspond to the same person, a *can not link* (CL) rule can be created. CL will be used as input in the clustering method that will prevent a CL pair to be included in the same cluster.

Must link rules. The first step is to use the clothing and hair similarities as well as the CL information in order to make an initial clustering. It should be noted that the clustering will be performed separately on each set $F^{I_{T_I(i)}}$. Thus, for each $F^{I_{T_I(i)}}$ the four distance matrices (for hair and costume using color and gabor features) are clustered separately (employing also the CL info) using a modified version of Rank-Order distance based clustering method ([Zhu et al., 2011]), which will be presented afterwards. A pair of faces will be considered to belong in the same cluster if they are grouped together in all the four aforementioned clusterings. We consider that this initial clustering does not manage to create all the groups, since the number of ungrouped faces is large, but ensures that its accuracy is quite high. We name the extracted pairs as *must-link* (ML) pairs, which are assumed as pairs that should be grouped in the same cluster. ML rules can extend CL rules since if $CL(f_a, f_b) = TRUE$ and $ML(f_a, f_c) = TRUE$ then $CL(f_b, f_c) = TRUE$.

Main clustering

Face features. Local binary pattern (LBP) histograms have been utilized as the facial features. Initially, the method crops the facial region in order to make it square and then scales it to 128×128. Then, the facial region is divided in 256 blocks of size 8×8 and the LBP histogram of each region is calculated. Finally, all histograms are concatenated, forming the feature vector.

Clustering. Main clustering is performed using the face features but also taking into account the restrictions given by the ML and CL pairs. The face distance matrix (using χ^2 or *EMD*-(Earth mover's distance) of LBP face features) is given as input in the clustering method. In order to incorporate the ML and CL info, the distance values for the ML pairs are changed to zeros while the ones for the CL pairs are changed to a large number M, which has to be larger than the largest value of the distance matrix. We perform a two step clustering. Initially, clustering is performed within the faces that belong to the same time clusters and then, we perform a clustering of the entire collection using as input the output of the first one.

Clustering method We have used a modified version of the Rank Order Distance based clustering method presented in [Zhu et al., 2011]. The method was modified in order to import the CL constraint in the algorithm. Thus, the distance of two clusters C_i, C_j is given by

$$d(C_i, C_j) = \begin{cases} \min\limits_{\forall f_a \in C_i, f_b \in C_j} d(f_a, f_b) & \text{, if } \max\limits_{\forall f_a \in C_i, f_b \in C_j} d(f_a, f_b) < M \\ M & \text{, if } \max\limits_{\forall f_a \in C_i, f_b \in C_j} d(f_a, f_b) = M \end{cases}$$

namely their distance equals the closest distance between the two clusters if they do not contain any CL pair.

As described in [Zhu et al., 2011], the method is an agglomerative one where each sample is considered as a cluster and successively at the next levels clusters are merged in a bottom-up way. It is initialized by assigning all $N = |F|$ faces to N single element clusters, where $|S|$ denotes the cardinality of set S. Two clusters are merged if their cluster level Rank-order distance D^R is below a threshold t and the cluster level locally normalized distance D^N is below 1. Thus, it is clear that the distance definition above, prevents the cluster merging of a cluster pair C_i, C_j if they have faces $f_a, f_b : f_a \in C_i, f_b \in C_j$ and $CL(f_a, f_b) =$ TRUE.

8.2.2 Advances in comparison to the previous version

As mentioned in D4.2 [D4.2, 2014], the first process that should be performed before face clustering, is face detection. Face detection in consumer photos is not always accurate due to the extreme pose and illumination variety of faces. For this reason, in order to increase the number of detected faces, we extended our previous approach to face detection by multiple face detectors. Also, in the second version of the face detector, the detected potential facial regions are considered as faces if they contain at least one facial feature (eyes, nose or mouth) and the percentage of the skin-color pixels of it is above a threshold, or if they have been detected from all the five detectors (regardless of the existence of facial features or the ratio of skin-color pixels). The last case is effective in the case of dark images, where facial characteristics can not be detected and face color is too dark to be considered as skin-like.

Besides the advances to face detection, face clustering, which is performed on top of the face detection results, is a new functionality that was not available in the previous version of the ForgetIT content analysis techniques.

8.3 Experimental evaluation and comparison

8.3.1 Face clustering

We have tested our method in a collection gathered from partners of the ForgetIT project. It consists of 484 images, and contains images from the vacation of a group of five people. Initially face detection is performed. Then, the list of detected faces is clustered, taking into account the capture time, the costume and hair similarity and the fact that faces of the same person can not appear twice in the same image. The clustering output is a set of L face clusters $\mathbf{C} = \{C_1, C_2, ..., C_L\}$ and a cluster of ungrouped faces C_{un}. Ideally, C_{un} should contain faces that appear only once (usually faces detected in the image bachground) and false facial detections. Then, the *Precision*, *Recall* and *Compression Ratio* measures have been used for evaluating the clustering result as defined in [Zhu et al., 2011]. *Precision* is related with the number of mis-grouped faces in each cluster, *Recall* measures the non-noise faces that are grouped together and *Compression Ratio* gives the average cluster size. They are defined as:

$$Precision = 1 - \frac{\sum_{i=1}^{L}|M_i|}{\sum_{i=1}^{L}|C_i|}, \quad Recall = \frac{\sum_{i=1}^{L}(|C_i|-|n_i|)}{N_F-|nf|}, \quad Comp.Ratio = \frac{1}{L}\sum_{i=1}^{L}|C_i|,$$

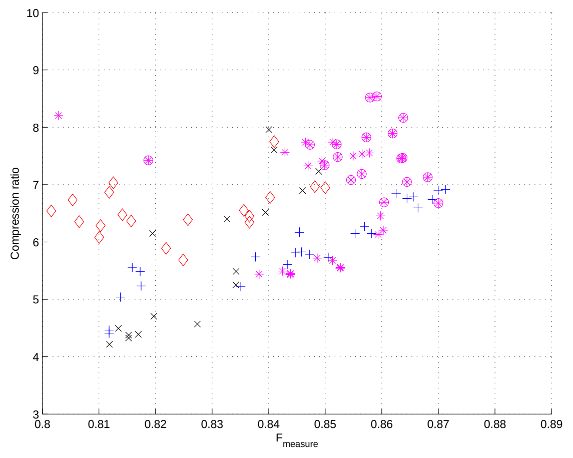
where $|M_i|$ and $|n_i|$ are the number of misgrouped faces and the number of non facial detections (erroneous facial detections) in C_i respectively. $|nf|$ is the number of erroneous facial detections of the entire collection. Also, $F_{measure}$ is calculated, combining $Precision$ and $Recall$ as $F_{measure} = 2\frac{Precision \cdot Recall}{Precision + Recall}$.

For facial feature vector calculation, three LBP parameter sets have been used. The first LBP histogram has been obtained using the $(8, 1)$ neighborhood, the second using the $(16, 2)$ neighborhood and third the concatenation of the histograms above. Facial feature vector distance was calculated utilizing the χ^2 and *emd* distances. The experiments were performed using the modified Rank Order Distance based clustering method for several values of parameters t and K of the algorithm [Zhu et al., 2011] ($t : 8$ to 20 with step 3, $K : 12$ to 60 with step 4). For comparison reasons, the algorithm has also been executed without taking into account the ML and CL constrains.

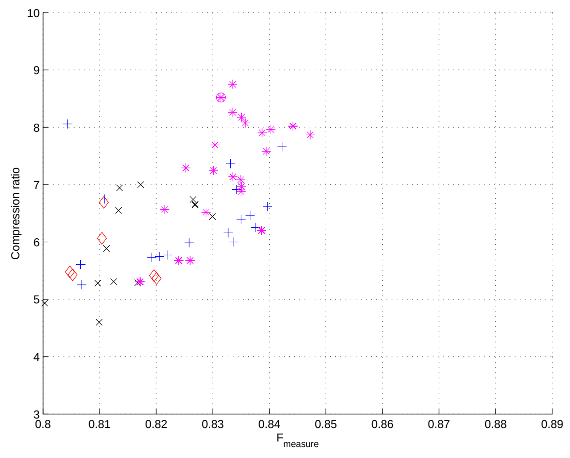
The experimental results are illustrated in Fig. 16. The figure presents only the results for which both precision and recall are above 0.7. If both precision and recall are above 0.8 an extra circle is printed around the result marker. From Fig. 16 it is clear that the ML and CL constrains improve the method's performance in terms of both accuracy and compression ratio. Also, it is worth mentioning that almost all results attaining precision and recall above 0.8 come from the full constraint (ML and CL) case. Based on these results, in our ForgetIT software, we adopted the case of using LBP with $(8, 1)$ neighborhood, *emd* as vector distance and $t = 11, K = 14$ parameters for Rank Order Clustering. For this case, *Precision*=0.93, *Recall*=0.746, $F_{measure} = 0.8554$ and *Compression ratio* = 7.23.

8.3.2 Face detection

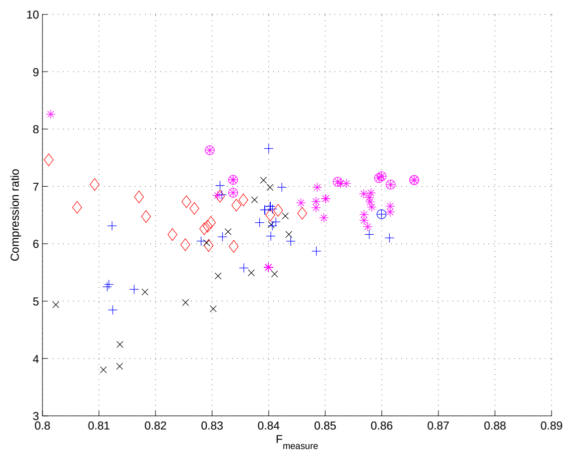
Face detection has been tested in the same dataset. As stated in D4.2, the use of many face detectors along with facial element validation increases the face detection performance. In this study, we also utilized face color validation and multiple detector validation. The study results are depicted in Table 11. Face detectors 1 and 2 have precision above 80% but the number of correct detections is small (< 470). On the other hand, detectors 3 and 4 detected more faces but also more non-facial elements. As expected, if we use the output of all detectors (row 5), the number of correct detections increases but, on the other hand, precision becomes very low. In order to decrease false detections we employed various constrains. We have accepted the face detector output if the number of skin-color pixels are above a threshold (row 6), if at least one facial element (mouth, eyes or nose) has been detected (row 7) or if the face has been detected by all detectors (row 9). Skin color thresholding and all-detectors cases returned more precise results but with low number of correct detections. Combinations of the above filtering have been examined, such as intersection of skin thresholding and facial elements existence results (row 8), union of skin thresholding and 'all detectors' results (row 10) and union of row 8 and row 9 results (row 11). The optimal results are achieved in case 10 and 11, where the number of correct detections is high enough and the precision values around 85%.



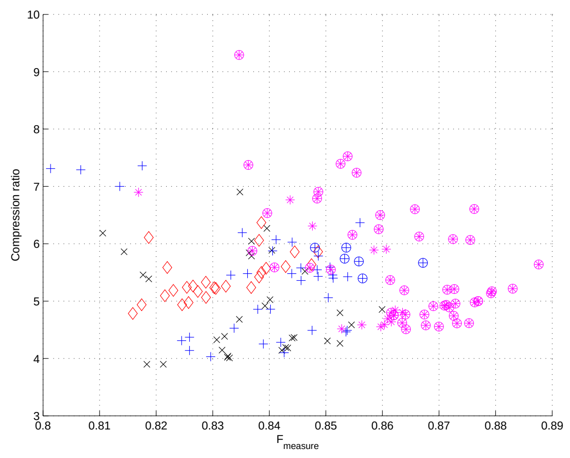
(a)



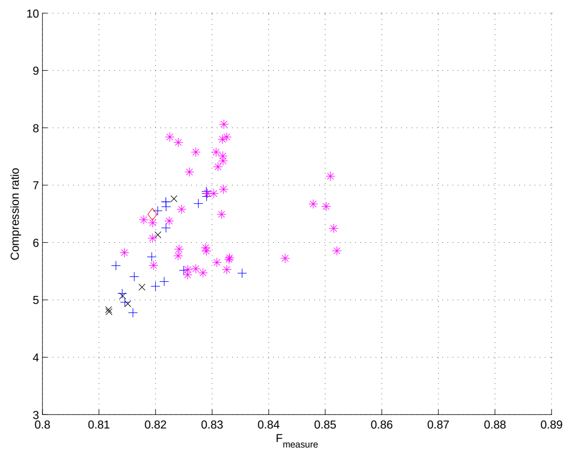
(b)



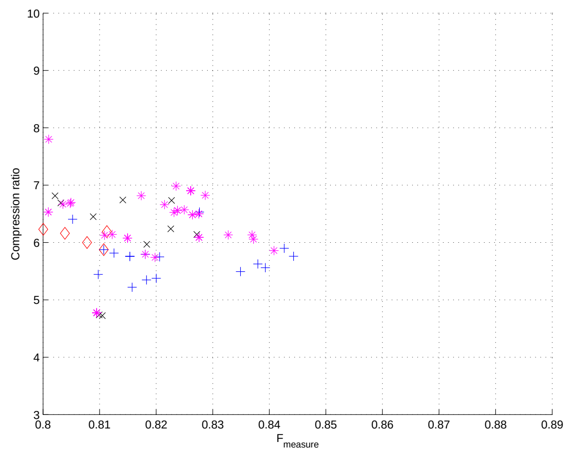
(c)



(d)



(e)



(f)

Figure 16: $F_{measure}$ **VS** $Compression\_Ratio$ **diagrams for three feature vector types and vector distances. Left column (a,c,e)** χ^2 **distance, right column (b,d,f)** ***emd*** **distance. Diamond (**$\diamond$**) symbol: without ML and CL constraints, times (**$\times$**) symbol: with CL constraint only, plus (**$+$**) symbol: with ML constraint only, asterisk (**$*$**) symbol : with both ML and CL constraints. Symbol in a circle: Precision and recall greater than** 0.8

Table 11: Face detection results using many face detectors and extra validation rules

		Number of correct detections	Number of total detections	Precision (%)
1	Detector 1	467	576	81.08
2	Detector 2	449	544	82.54
3	Detector 3	482	627	76.87
4	Detector 4	486	733	66.3
5	Merged	495	763	65.74
6	Skin color thres.	442	486	90.95
7	Facial elements	471	653	72.13
8	Skin thr. AND facial elem	424	457	92.78
9	All detectors	439	490	89.59
10	Skin thr. OR All detectors	487	576	84.55
11	(Skin thr. AND facial elem) OR All detectors	484	564	85.82

8.4 Software implementation

The updated face detection and face clustering has been developed in Matlab [MATLAB, 2013]. The method returns two files which contain the face detection and face clustering results. Software technical details are listed in Table 12. The prototype component described in this section can be downloaded from `http://www.forgetit-project.eu/en/downloads/workpackage-4/`. Please contact the project coordinator for accessing to this protected section of the website.

Table 12: Face detection and face clustering software technical details

Functional description	Face detection and clustering of Image collection
Input	A directory that contains image files
Output	Two txt files, one with face coordinates and another with clusters of face ROIs indices
Language/technologies	Matlab R13a, Computer Vision toolbox
OS Requirements	Windows

Each line of the face detection output file contains information about each detected face (Fig. 17a). The first number is the face index (which is used in the face clustering output), the second the image index (image list is saved in another file), the third is the face index in the image, the next two are the (x, y) coordinates of the upper left corner of the face region bounding box and the last two the height and width of it.

Face clustering output (Fig. 17b) contains the face bounding box indexes of each cluster in a separate line.

```
Face detection output
1 1 1 1987 1089 534 534
2 2 1 1910 1802 161 162
3 4 1 2636 1882 427 427
4 5 1 3755 1126 119 120
5 7 1 1519 1569 117 117
6 8 1 935 1573 147 145
7 8 2 1219 1622 136 136
8 9 1 1924 1972 203 203
9 13 1 113 1453 195 195
...
```

(a)

```
Face clustering output
1 3 30 31 37 41 46 48 61 ...
2 57 72 102 207 224 396 561
4 29 185
5 8 11 13 16 26 27 333
10 12 17 18 21 24
14 76 353 475
19 23 324 399
20 22
32 33 35 81 103 105 107  ...
...
```

(b)

Figure 17: (a) Face detection output sample, (b) Face clustering output sample

8.5 Discussion

In this section a face clustering method and an updated version of face detection was presented. In order to improve the face clustering performance, side information (hair, costume, time) is employed. Similarly, face detection uses many detectors, face color, and facial features to get more accurate results.

Face clustering can be further improved after employing more side information, such as social relationship, relative pose etc.

9 Near-duplicate detection and time synchronization for multi-user collection summarization

9.1 Problem statement

In D4.2 [D4.2, 2014] user collection summarization has been performed using visual-feature-based image clustering. In this document, user collection summarization is extended by i) evaluating more approaches than the ones in D4.2 ii) employing more data dimensions, such as time and GPS coordinates. Furthermore, near duplicate images, namely images portraying the same scene are detected. Near duplicate detection (NDD) can enrich the user collection summarization algorithm, since the existence of near duplicate images may indicate the importance of the specific scene. Also, the updated user collection summarization algorithm uses the NDD results and forces all the near duplicate images to be clustered in the same cluster. Finally, single-user collection summarization is extended to a multi-user approach. Since time is employed in our clustering approach, user galleries should be temporally aligned before clustering, because the photo capturing devices of different users are not necessarily synchronized. This is achieved with the multi-user collection time synchronization method which is also proposed in this section.

9.2 ForgetIT approach

9.2.1 Image clustering using visual information

In D4.2 [D4.2, 2014] an image clustering method for summarization using visual information was proposed. Six methods were studied and evaluated (K-means, Hierarchical clustering using simple/complete linkage, Partitioning Around Medoids, Affinity Propagation and Farthest First Traversal Algorithm).

For further improvement of the clustering results, “Dimension/Feature Reduction” algorithms are also implemented and tested. “Dimension Reduction” (DR) is the mapping of data to a lower dimensional space such that uninformative variance in the data is discarded, or such that a subspace in which the data lives is detected. DR can lead to better models for inference. The motivation for implementing these methods was the dimension of concept detection results, which is currently 346. For this, various DR algorithms are implemented and tested.

Some of the tested DR algorithms are Principal Component Analysis (PCA), Classical multidimensional scaling (MDS), Linear Discriminant Analysis (LDA), Laplacian Eigenmaps, Diffusion Maps and Kernel PCA.

Along DR algorithms, the clustering algorithms that are implemented and tested are: K-Means, hierarchical clustering, *Fuzzy C-Means Clustering* and *Gaussian Mixture Models (GMM) Clustering*.

In K-means, data is divided into crisp clusters, where each data point belongs to exactly one cluster. In Fuzzy C-Means, the data points can belong to more than one cluster, and associated with each of the points are membership grades which indicate the degree to which the data points belong to the different clusters. Fuzzy C-Means was developed by Dunn in 1973 and improved by Bezdek et al. [Bezdek et al., 1984].

The Gaussian Mixture Models clustering method implements the expectation-maximization (EM) algorithm for fitting mixture-of-Gaussian models. In GMM, clusters are assigned by selecting the component that maximizes the posterior probability. Clustering using Gaussian mixture models is considered a soft clustering method, just like Fuzzy C-Means.

9.2.2 Image clustering using time and geolocation metadata

Three approaches of image clustering using additional data dimensions were developed and evaluated in ForgetIT. In all of them, given an image collection, the capture time and geolocation information (if any) of each image are extracted from the EXIF tags of the image files, and visual information is extracted using the concept detection method presented in D4.2. All three types of information (time, geolocation and visual) are passed as input to the clustering method as follows:

First approach

- Temporal event clustering: the time information extracted from the images is converted into UNIX timestamps (i.e. the number of seconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), 1 January 1970) and the images are sorted according to their UNIX timestamps. The similarity matrix S of the temporally ordered images is constructed using the similarity measure presented in [Cooper et al., 2005] for different values of parameter K. In order to find the boundaries of the sub-events, novelty scores [Cooper et al., 2005] are calculated for each image. The first differences of the novelty scores are computed and the peaks that correspond to differences which are greater than a threshold based on the maximum peak are selected. This procedure is repeated for all K values. The result of the above procedure is the formation of a set of boundary lists. The appropriate one is selected using a confidence measure from the average within-class similarity and the between class dissimilarity of the data as in [Cooper et al., 2005].

- Geolocation information: the above formed clusters are split using geolocation information. Geolocation information of images is checked and if the location of an image is close to the location of another image then images are assigned to the same cluster, or otherwise a new cluster is created. The threshold that determines if an image is close to another one is calculated heuristically. Specifically, the geolocation threshold is calculated by fitting two Gaussian distributions to the histogram of image distances, the first representing the images that were taken in close-by locations, and the second the distant ones. The threshold is defined by the mean of the first Gaussian curve plus its standard deviation.

- Visual information: if geolocation information is missing, visual information is used in order to assign the images to the clusters which are more similar according to the visual content.

Second approach

In the second approach, we detect time gaps between the events of each image collection. An overview of this approach is the following: we find the minimum time difference of dissimilar photos which is greater than the maximum time difference of the near-duplicate photos (based on the similarity matrix extracted from the method presented in subsection 9.2.3). The clusters that are formed by these time gaps are merged according to time and geolocation similarity. For the clusters that do not have geolocation information, the merging is continued by considering the time and low-level feature similarity (HSV histogram) or the time and the concept detector (CD) confidence similarity scores [Mezaris et al., 2010].

Third approach

In this approach, not only concept results, GPS (if available) and time but also *quality assessment* results are utilized. The images are clustered multiple times in a cascaded fashion and in all steps automatic hierarchical clustering with complete linkage is used, and mean distance between the links is used as a cut-off. Fig. 18 shows the cascaded structure. All the steps are applied in the given order. The GPS, time and quality clustering steps are applied to the images which have these information available. In each step, if in the collection some of the images have the info and the others do not, the non ones lacking a specific type of information are treated as a single cluster, and auto-clustering is applied to the rest. This summarization method is being integrated to the demo application that is described in *D9.3 - Personal Preservation Pilot I*.

9.2.3 Near duplicate detection (NDD)

Considering NDD, the pipeline of our method proceeds as follows: we detect a fixed number of SIFT keypoints per image and compute the corresponding descriptors. We perform PCA to reduce the dimensionality of SIFT vectors to 96. We construct a visual vocabulary using a fast approximate version of kMeans, enabling us to encode the descriptor-based representation of each image using VLAD encoding [Jégou et al., 2010b]. A componentwise mass normalization and square-rooting is performed on VLAD vectors [Delhumeau et al., 2013]. We then construct an index on VLAD vectors using a Randomized KD forest, to perform kNN search of a query. We match the descriptors of the query image to those of its nearest neighbours. Once we have the matches, we check the following ratios:

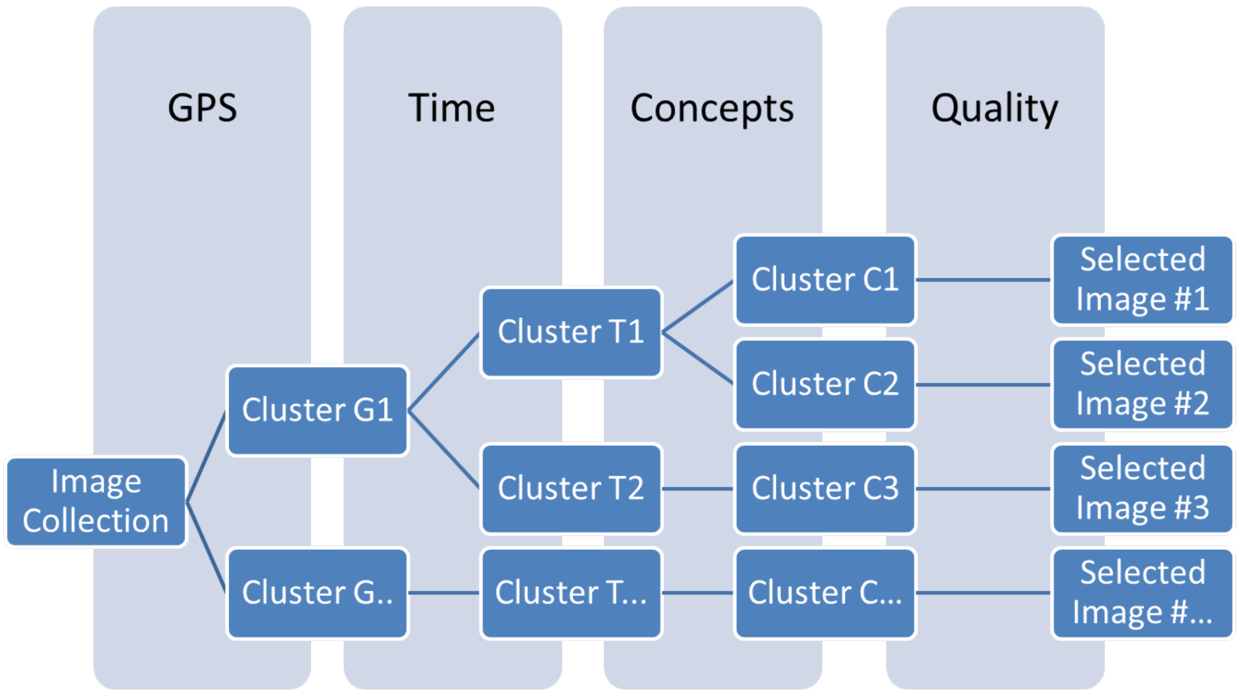


Figure 18: Image Collection Summarization #2

1. Total image area to matched area ratio, to optionally exclude partial duplicates.
2. Query matched area to neighbour matched area ratio, to optionally exclude scaled duplicates.

Finally, remaining neighbours are further refined by checking the geometric consistency of SIFT keypoints using Geomtric Coding (GC) [Zhou et al., 2013].

Alternatively, instead of querying a collection of images, we can directly find groups of near duplicate images in a collection of images using the following procedure: we form a matrix W, of $n \times n$ size (where n the total number of images). Each position (i, j) of W holds the visual similarity score of image i to image j. We consider W as a weight matrix of graph G and find the community structure of G using the modularity measure [Blondel et al., 2008]. Each community of G is a group of near duplicate images.

9.2.4 Muti-user collection time synchronization

Time synchronization makes use of the NDD software, as well as color information (HSV histograms) and scene similarity information (GIST) [Oliva and Torralba, 2001]. We perform a late fusion of this similarity measures assessed on the union of all galleries. Consequently, we filter out identified pairs of near duplicates according to the following rules:

1. Reject pairs when geolocation information is available and the location distance of the two photos is greater than a distance threshold.

2. Reject pairs when the time difference between the photos is above an extreme time threshold (which indicates that this time difference is unlikely to be due to a time synchronization error alone).

The remaining near duplicate photos belonging to different galleries are considered as links between those galleries. We then construct a graph whose nodes represent the galleries, and the edges represent these links between galleries. Each edge has a weight which is equal to the number of links between the two galleries. Having constructed the graph, we compute the time offset of each gallery by traversing it, as follows: Starting from the node corresponding to the reference gallery, we select the edge with the highest weight. We compute the time offset of the node on the other end of this edge as the median of the time differences of the pairs of near duplicate photos that this edge represents, and add this node to the set of visited nodes. The selection of the edge with the highest weight is repeated, considering as possible starting point any member of the set of visited nodes, and the corresponding time offset is computed, until all nodes are visited.

9.2.5 Advances in comparison to the previous version

Our previously introduced version of image clustering used only the visual information extracted by the images of the collection. In the current version we added more data dimensions as input to the clustering algorithm, such as the time taken for each image of the collection and its geolocation information, if available. Combining these data, we cluster images of a collection, which describes an event, into sub-events. Moreover, the near duplicate images of a collection are detected and this information is used in order to force the near duplicates to be assigned always into same clusters. Finally, since time is used as input in our method, we developed a time synchronization approach which addresses the existence of time measurement differences when multiple photo collections (captured with different devices, possibly by different users) of the same event are jointly considered for organizing the photos in sub-events.

9.3 Experimental evaluation and comparison

9.3.1 Image collection clustering

Feature reduction methods

In our experiments we used the same dataset used in D4.2 for image collection clustering experiments, which consists of 4 image collections whose size varies between 107 and 254 images and 5 collections of video keyframes whose number varies between 157 and 345. The evaluation of the clustering results is performed by calculating the Normalized Mutual Information (NMI) [Strehl and Ghosh, 2003] for all image collections for: i) a fixed number of clusters (equal to $K = 10$) is defined for all image collections and ii) different number of clusters is defined for each collection ($K = Var$). First the different dimensionality reduction algorithms are evaluated in combination with different clustering algorithms. In Table 13 the results (mean of the NMIs of all collections) for each DR algorithm are given for the optimal target dimensionality (TD) for each image collection (which is found by exhaustive search). However, since for the final software application the TD of DR should be fixed, another set of experiments with a fixed DR size for all image collections is conducted and the results are reported in Table 14.

Table 13: Clustering results for different DR methods with the TD of DR being optimized for each photo collection separately

	K=10				**K=Var**			
DR Method	**K-Means**	**Hier.**	**Fuzzy C-Means**	**GMM**	**K-Means**	**Hier.**	**Fuzzy C-Means**	**GMM**
PCA	0.404	0.388	0.400	0.395	0.366	0.343	0.350	0.355
LDA	0.388	0.373	0.386	0.402	0.351	0.338	0.346	0.368
MDS	0.405	0.388	0.396	0.403	0.367	0.343	0.347	0.351
Kernel PCA (Gaussian)	0.402	0.389	0.389	0.382	0.368	0.348	0.339	0.335
Kernel PCA (Linear)	**0.408**	0.388	0.397	0.405	0.362	0.343	0.348	0.358
Laplacian Eigenmaps	0.377	0.362	0.380	0.386	0.327	0.286	0.338	0.341
Diffusion Maps	0.347	0.300	0.368	0.360	0.316	0.202	0.336	0.325

Regarding the results, the PCA and K-means combination seems to perform slightly better than the rest of the DR-clustering combinations. However, comparing the results with those of the D4.2 clustering algorithms, (see Table 15), using an DR algorithm only improves the performance from 36% to 37.7%. Thus, for the corresponding component of the ForgetIT Condensator, we decided to continue to use K-means clustering without DR since it is much simpler and faster, and its results are only slightly worse than those of the other clustering combinations we evaluated.

Using time and geolocation information

The developed approaches for image collection summarization using multiple data dimensions (time, GPS, visual) are evaluated on the datasets of MediaEval SEM task [Conci et al., 2014]. The *Vancouver* dataset consists of 1351 photos and captures various sub-events of the Vancouver 2010 Winter Olympic Games, while the *London* dataset consists of 1358 photos and captures various sub-events of the London 2012 Olympic Games. In Table 16, the results for the three approaches of section 9.2.2 are presented in terms of NMI. We can see from these results that the first two of the approaches proposed in section 9.2.2 perform better than the third one.

Table 14: Clustering results for different DR methods with fixed reduced sizes

		K=10				K=Var			
DR Method	**DR TD**	**K-Means**	**Hier.**	**Fuzzy C-Means**	**GMM**	**K-Means**	**Hier.**	**Fuzzy C-Means**	**GMM**
PCA	150	0.376	0.310	0.358	0.083	0.304	0.263	0.313	0.062
LDA	94	0.335	0.318	0.324	0.337	0.300	0.271	0.295	0.297
MDS	90	0.366	0.337	0.361	0.167	0.321	0.294	0.317	0.117
Kernel PCA (Gaussian)	75	0.364	0.343	0.354	0.190	0.317	0.294	0.311	0.115
Kernel PCA (linear)	30	**0.368**	0.340	0.362	0.32	0.310	0.292	0.309	0.231
Laplacian Eigenmaps	10	0.330	0.302	0.338	0.320	0.248	0.201	0.252	0.271
Diffusion Maps	16	0.304	0.179	0.315	0.275	0.273	0.130	0.274	0.26

Table 15: Clustering results using the approaches presented in D4.2

Clustering algorithm	NMI	
	$k = 10$	$k = Var$
k-means	$\mathbf{0.36}$	0.31
hier-comp	0.35	0.29
hier-single	0.21	0.16
PAM	0.35	0.28
AP	0.35	0.28
Far. First	0.30	0.26

9.3.2 Near duplicate detection

For near duplicate detection evaluation we used the California-ND (CND) [Jinda-Apiraksa et al., 2013], UKBench (UKB) [Nister and Stewenius, 2006] INRIA Copydays and INRIA Holidays datasets. In successive experiments, we used each image in these datasets as a query image, and for each query, we compared the retrieved near duplicate images to the available ground truth, evaluating precision, recall and mean average precision (MAP). The evaluation results are shown in Table 17, where we can see that high precision and recall are consistently achieved in different datasets.

9.3.3 Muti-user collection time synchronization

The datasets used in section 9.3.1 are also used for evaluating the Muti-user collection time synchronization. We defined precision as the ratio between the number of synchronized galleries and the total number of galleries, and accuracy as the average time offset

Table 16: Clustering results (NMI) using multiple data dimensions (time, GPS, visual) for two datasets and the three approaches presented in Section 9.2.2

London dataset			Vancouver dataset		
First approach	Second approach	Third approach	First approach	Second approach	Third approach
0.8559	0.8842	0.6907	0.8667	0.7848	0.6539

Table 17: Evaluation Results for Near Duplicate Detection

	Precision	Recall	mAP
California ND	85.98%	81.55%	68.41%
UK Bench	90.34%	72.19%	65.22%
INRIA Copydays	97.51%	80.76%	79.87%
INRIA Holidays	97.46%	63.46%	62.09%

calculated over the synchronized galleries, normalized with respect to the maximum accepted time offset (which is 1800 seconds). The evaluation results are shown in Table 18. We can see that the proposed approach can accurately synchronize the majority of photo galleries in the collection. Our participation to the MediaEval SEM task [Conci et al., 2014] also showed that the ForgetIT approach compares favorably to the other literature approaches on this topic.

Table 18: Evaluation Results for Time Synchronization

	Precision	Accuracy
Vancouver Winter Olympics	91.18%	73.75%
London Olympics	61.11%	71.27%

9.4 Software implementation and integration

Near duplicate detection

The near duplicate detection method is implemented as a REST service, running in a CERTH server.

The service calls for both GET and POST methods are as follows:

- GET method

  `http://multimedia.iti.gr:8080/ForgetITImageAnalysis/EXTRACTOR/methodGET?imagePaths=''&method=''`

  URL paths of the images are separated with special character $\sim$.

- POST method

`http://multimedia.iti.gr:8080/ForgetITImageAnalysis/EXTRACTOR/methodPOST`

Data should be encoded as: application/x-www-form-urlencoded and URL paths of the images are separated with special character newline (\n).

The input arguments are:

- *imagePaths*: url paths of the images of the collection (separated with special character depending on GET or POST call) or URL path of a compressed file (currently only .zip format is supported)
- *method*: `duplicate`.

A summary of the near duplicate detection software's technical details is presented in Table 19.

Table 19: Software technical details for the ForgetIT near duplicate detection method

Functional description	Near duplicate detection
Input	An image collection
Output	An XML file containing classes of near duplicate images
Language/technologies	C++, Java Rest service
OS Requirements	Windows

Image clustering for summarization

Using visual information

The service call for both GET and POST methods are as follows:

- GET method

  `http://multimedia.iti.gr:8080/ForgetITImageAnalysis/CONDENSATOR/methodGET?userID=''&NumberOfClusters=''`

- POST method

  `http://multimedia.iti.gr:8080/ForgetITImageAnalysis/CONDENSATOR/methodPOST`

  Data should be encoded as: application/x-www-form-urlencoded.

Requirements: in order to call the image clustering for summarization service, the user must have already called at least the concept detection service (see subsection 7.4). Visual information is required for executing the initial clustering using model vectors and K-means clustering algorithm. Near duplicate detection is not required if the user wants to apply the initial visual clustering on his/her image collection. However, the duplicate detection method can enrich the clustering since duplicate images will be forced to be assigned to the same clusters. Note that if the user wants to use both concept detection and near duplicate detection results for image clustering, he/she should call the service once for both methods with the `method` argument set to `conceptF,duplicate` or `conceptS,duplicate` in order to receive a unique *userID*.

The input arguments are:

- *userID*: This is a numeric ID of the image collection, used for photo caching and session identification purposes.
- *NumberOfClusters*: The number of clusters is user-defined. Note that the NumberOfClusters must in any case be lower than the number of images.

A summary of the image clustering for summarization software's technical details is presented in Table 20.

Table 20: Software technical details for the ForgetIT image clustering for summarization method

Functional description	Image clustering
Input	An XML file containing the output of image analysis methods (concept detection, near duplicate detection)
Output	An XML file containing the formed clusters
Language/technologies	Matlab, Java Rest service
OS Requirements	Windows
Other Requirements	Matlab

Using visual, time and GPS information

The first two approaches of image clustering using time and geolocation metadata (subsection 9.2.2) have been implemented and are available in `http://www.forgetit-project.eu/en/downloads/workpackage-4/` as a zip file containing usage instructions. In the third year of the project, one of them will be included in the Rest service of image clustering, together with the existing one that uses only visual information.

Also, an additional desktop demo application for ”Incremental Photo Preservation” that uses the time and geolocation-based clustering is being developed. The details for the application are provided in *D9.3 - Personal Preservation Pilot I*.

9.5 Discussion

The approaches that were presented in this section can be used for image collection summarization. Initially, a comparative study of several clustering methods using only visual information was presented. Near duplicate information was extracted and was exploited by placing constrains to the clustering algorithm. Moreover, time and geolocation information are used as inputs to the new clustering algorithm, which divides an initial collection, that describes an event into sub-events. Finally, due to time offsets of different image capturing devices (in the case of having many image collections for the same event, each collection captured with a different device and possibly also by a different user) a time synchronizing method is presented in order to temporary synchronize the images of the multiple collections and successfully organize all of them together in sub-events, so as to be able to produce a complete summary of the event.

10 Conclusions

10.1 Summary

In this deliverable the second release of the ForgetIT text and visual information analysis techniques for condensation and summarization were presented, based on the requirements and state-of-the-art approaches described in the previous work of D4.1 [D4.1, 2013] and the corresponding first release presented in D4.2 [D4.2, 2014]. Several software components performing textual and visual analysis are designed, implemented and evaluated.

10.2 Assessment of Performance Indicators

In this subsection, a short description is provided for explaining how and to what extent the methods described in this deliverable fulfil the success indicators of the five expected outcomes of WP4, as these are described in the DoW of the project. Each of the following objectives correspond to each one of the five WP4 tasks.

10.2.1 Textual Similarity and Redundancy

The success indicators of this objective are the *ability to achieve deep understanding* and the *number of features considered*. Deep understanding is addressed in Section 4 which introduces a document summarization method that condenses the documents preserving only the most useful information. Also, in Section 5 an improved semantic text editor is presented that is able to understand entities, identify new ones, and allow for manual annotation during document editing. This contributes to a better understanding of the text already with close interaction of the user with the result of more precise understanding of the text as opposed to with only an offline analysis w/o. user feedback. The improvements to the semantic text editor are achieved by utilizing relation extraction so as to make entity recognition more accurate.

10.2.2 Visual Quality, Similarity, Redundancy

The success indicators of this objective are the *ability to detect undesirable artifacts*, the *image/video similarity assessment* and the *number of information dimensions during clustering*. With respect to similarity assessment, we developed a new method for near duplicate image detection and we also examined a multitude of similarity distance functions as part of our experiments. Regarding the number of dimensions used for clustering, we extended our previous approaches that used only visual information by developing methods that combine appropriately visual information, time and geolocation metadata.

Concerning the detection of undesirable artifacts, this objective was addressed in [D4.2, 2014] with the development of a visual quality assessment method.

10.2.3 Semantic multimedia analysis

The success indicators of this objective are *the ability to detect concepts and complex events*. The ForgetIT concept detection method was extended in this deliverable, in comparison to the previous version, by adopting a more accurate image representation. Also, for the concept training, a method for acquiring training examples from the Web was presented. Face detection which is a specific case of concept detection, was also improved, and a face clustering method was developed. Complex event detection represents work in progress, which will be completed in the third year of the project and will be documented in D4.4.

10.2.4 Information condensation and consolidation

The success indicators of this objective are *the ability to summarize documents and multiple documents* and *the ability to combine the results of the first three objectives presented above (10.2.1-10.2.3) for selecting representative and diverse media*.

With respect to single and multiple document summarization, two methods were developed and presented in this deliverable. The first method, which simplifies text by removing words or phrases, was initially introduced in D4.2 and was extended in this deliverable (Section 3). The second method performs a more radical reduction of the text, as explained in Section 4.

Regarding results combination for representative media selection, we extended the clustering study initiated in D4.2 by i) evaluating more approaches to image clustering using visual information ii) employing more data dimensions such as time, GPS coordinates and faces. Furthermore, image collection summarization was extended in the direction of processing images from multi-user collections, and a time summarization method was also developed for supporting this task.

10.2.5 Evaluation of information condensation and consolidation

The success indicators of this objective are *the number of internal evaluations on ForgetIT datasets* and *the number of external benchmarking activities in which ForgetIT technologies participated in*. ForgetIT methods were widely tested in internal datasets as presented in the different sections of this deliverable. Furthermore, jointly with other EU projects, we participated in two benchmarking activities: the concept detection method was evaluated in the SIN (Semantic Indexing) task of TRECVID 2014, and our methods for near duplicate detection, multi-user collection time synchronization and clustering evaluated by participation to the MediaEval 2014 Event Synchronization Task (SEM) [Conci et al., 2014]. The results of the ForgetIT participation to these benchmarking activities were very encouraging.

10.3 Next steps

In the last year of the project, the aforementioned methods will be extended taking also into account the requirements that are being collected by WP2 (Foundations of forgetting and remembering) as well as the application workpackages which deal with personal (WP9) and organizational (WP10) preservation. Furthermore, these and all subsequent ForgetIT text and multimedia processing methods will continue to be evaluated, both on ForgetIT datasets and by participation to international benchmarking activities. Moreover, during the third project year, the presented methods and their software implementations will be also tested within the Preserve-or-Forget platform as part of the overall processing workflow and component communications in ForgetIT.

References

[Barzilay and Mckeown, 2005] Barzilay, R. and Mckeown, K. R. (2005). Sentence fusion for multi-document news summarization. *Computational Linguistics*, 31:297–328.

[Bezdek et al., 1984] Bezdek, J. C., Ehrlich, R., and Full, W. (1984). Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2):191–203.

[Blondel et al., 2008] Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

[Bollacker et al., 2008] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

[Bontcheva et al., 2013] Bontcheva, K., Derczynski, L., Funk, A., Greenwood, M. A., Maynard, D., and Aswani, N. (2013). TwitIE: An open-source information extraction pipeline for microblog text. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*. Association for Computational Linguistics.

[Carvalho et al., 2014] Carvalho, D., Callı, C., Freitas, A., and Curry, E. (2014). Easyesa: A low-effort infrastructure for explicit semantic analysis.

[Castrillón et al., 2011] Castrillón, M., Déniz, O., Hernández, D., and Lorenzo, J. (2011). A comparison of face and facial feature detectors based on the viola–jones general object detection framework. *Machine Vision and Applications*, 22(3):481–494.

[Chen and Lin, 2007] Chen, Y.-J. and Lin, Y.-C. (2007). Simple face-detection algorithm based on minimum facial features. In *Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE*, pages 455–460.

[Choi et al., 2010] Choi, J. Y., De Neve, W., Ro, Y. M., and Plataniotis, K. (2010). Automatic face annotation in personal photo collections using context-based unsupervised clustering and face information fusion. *Circuits and Systems for Video Technology, IEEE Transactions on*, 20(10):1292–1309.

[Conci et al., 2014] Conci, N., Natale, F. D., and Mezaris, V. (2014). Synchronization of multi-user event media (sem) at mediaeval 2014: Task description, datasets, and evaluation. In *MediaEval 2014 Workshop, Barcelona, Spain*.

[Cooper et al., 2005] Cooper, M., Foote, J., Girgensohn, A., and Wilcox, L. (2005). Temporal event clustering for digital photo collections. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, 1(3):269–288.

[Cunningham et al., 2011] Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Aswani, N., Roberts, I., Gorrell, G., Funk, A., Roberts, A., Damljanovic, D., Heitz, T., Greenwood, M. A., Saggion, H., Petrak, J., Li, Y., and Peters, W. (2011). *Text Processing with GATE (Version 6)*.

[D10.1, 2014] D10.1 (January 2014). Organizational Preservation: Application Mockups and Prototypes.

[D4.1, 2013] D4.1 (July 2013). Information Analysis, Consolidation and Concentration for Preservation - State of the Art and Approach. http://www.forgetit-project.eu/fileadmin/fm-dam/deliverables/ForgetIT_WP4_D4.1.pdf.

[D4.2, 2014] D4.2 (February 2014). Information analysis, consolidation and concentration techniques, and evaluation - First release. http://www.forgetit-project.eu/fileadmin/fm-dam/deliverables/ForgetIT_WP4_D4.2.pdf.

[D8.1, 2013] D8.1 (October 2013). Integration Plan and Architectural Approach. http://www.forgetit-project.eu/fileadmin/fm-dam/deliverables/ForgetIT_WP8_D8.1_Integration_Plan_Architecture.pdf.

[D8.2, 2014] D8.2 (April 2014). The Preserve-or-Forget Reference Model Initial Model. http://www.forgetit-project.eu/fileadmin/fm-dam/deliverables/ForgetIT_WP8_D8.2.pdf.

[D8.3, 2014] D8.3 (July 2014). The Preserve-or-Forget Framework First Release. http://www.forgetit-project.eu/fileadmin/fm-dam/deliverables/ForgetIT_WP8_D8.3.pdf.

[D9.1, 2013] D9.1 (July 2013). Application Use Cases & Requirements Document. http://www.forgetit-project.eu/fileadmin/fm-dam/deliverables/ForgetIT_WP9_WP10_D9.1.pdf.

[D9.2, 2014] D9.2 (January 2014). Personal Preservation Mockups. http://www.forgetit-project.eu/fileadmin/fm-dam/deliverables/ForgetIT_WP9_D9.2.pdf.

[Delhumeau et al., 2013] Delhumeau, J., Gosselin, P.-H., Jégou, H., and Pérez, P. (2013). Revisiting the vlad image representation. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 653–656. ACM.

[El Khoury et al., 2010] El Khoury, E., Senac, C., and Joly, P. (2010). Face-and-clothing based people clustering in video content. In *Proceedings of the International Conference on Multimedia Information Retrieval*, MIR '10, pages 295–304, New York, NY, USA. ACM.

[Fellbaum, 1998] Fellbaum, C., editor (1998). *WordNet - An Electronic Lexical Database*. MIT Press.

[Gallagher and Chen, 2008] Gallagher, A. and Chen, T. (2008). Clothing cosegmentation for recognizing people. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8.

[Gerald et al., 1997] Gerald, S., Amit, S., Mandar, M., and Chris, B. (1997). Automatic text structuring and summarization. *Information Processing & Management*, 33(2):193–207.

[Han et al., 2013] Han, L., Kashyap, A., Finin, T., Mayfield, J., and Weese, J. (2013). Umbc ebiquity-core: Semantic textual similarity systems. *Atlanta, Georgia, USA*, page 44.

[Jaffré et al., 2004] Jaffré, G., Joly, P., et al. (2004). Costume: A new feature for automatic video content indexing. In *Proceedings of RIAO*, pages 314–325. Citeseer.

[Jégou et al., 2010a] Jégou, H., Douze, M., Schmid, C., and Pérez, P. (2010a). Aggregating local descriptors into a compact image representation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3304–3311. IEEE.

[Jégou et al., 2010b] Jégou, H., Douze, M., Schmid, C., and Pérez, P. (2010b). Aggregating local descriptors into a compact image representation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3304–3311. IEEE.

[Jinda-Apiraksa et al., 2013] Jinda-Apiraksa, A., Vonikakis, V., and Winkler, S. (2013). California-nd: An annotated dataset for near-duplicate detection in personal photo collections. In *Quality of Multimedia Experience (QoMEX), 2013 Fifth International Workshop on*, pages 142–147. IEEE.

[Lehmann et al., 2015] Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., and Bizer, C. (2015). DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 6(2):167–195.

[Li and Fei-Fei, 2010] Li, L.-J. and Fei-Fei, L. (2010). Optimol: automatic online picture collection via incremental model learning. *International journal of computer vision*, 88(2):147–168.

[Li and Snoek, 2009] Li, X. and Snoek, C. G. (2009). Visual categorization with negative examples for free. In *Proceedings of the 17th ACM international conference on Multimedia*, pages 661–664. ACM.

[Li et al., 2009] Li, X., Snoek, C. G., and Worring, M. (2009). Learning social tag relevance by neighbor voting. *Multimedia, IEEE Transactions on*, 11(7):1310–1322.

[Li et al., 2011] Li, X., Snoek, C. G., Worring, M., and Smeulders, A. W. (2011). Social negative bootstrapping for visual categorization. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, page 12. ACM.

[Lin, 2004] Lin, C.-Y. (2004). ROUGE : A Package for Automatic Evaluation of Summaries. In *Proceedings of the Workshop on Text Summarization Branches Out*, Bacelona, Spain.

[MATLAB, 2013] MATLAB (2013). *version 8.1.0 (R2013a)*. The MathWorks Inc., Natick, Massachusetts.

[Mezaris et al., 2010] Mezaris, V., Sidiropoulos, P., Dimou, A., and Kompatsiaris, I. (2010). On the use of visual soft semantics for video temporal decomposition to scenes. In *Semantic Computing (ICSC), 2010 IEEE Fourth International Conference on*, pages 141–148. IEEE.

[Nister and Stewenius, 2006] Nister, D. and Stewenius, H. (2006). Scalable recognition with a vocabulary tree. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2161–2168. IEEE.

[Oliva and Torralba, 2001] Oliva, A. and Torralba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175.

[Over et al., 2013] Over, P., Awad, G., Michel, M., Fiscus, J., Sanders, G., Kraaij, W., Smeaton, A. F., and Quenot, G. (2013). Trecvid 2013 – an overview of the goals, tasks, data, evaluation mechanisms and metrics. In *Proceedings of TRECVID 2013*. NIST, USA.

[Over et al., 2012] Over, P., Awad, G., Michel, M., Fiscus, J., Shaw, B., Kraaij, W., Smeaton, A. F., and Quénot, G. (2012). TRECVID 2012 - An overview of the goals, tasks, data, evaluation mechanisms and metrics. In *TRECVID 2012*.

[Ramos, 2003] Ramos, J. (2003). Using tf-idf to determine word relevance in document queries. In *Proceedings of the First Instructional Conference on Machine Learning*.

[Rublee et al., 2011] Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). Orb: an efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE.

[Russakovsky et al., 2014] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2014). ImageNet Large Scale Visual Recognition Challenge.

[Saggion, 2008] Saggion, H. (2008). SUMMA. A Robust and Adaptable Summarization Tool. *TAL*, 49(2):103–125.

[Saggion et al., 2003] Saggion, H., Bontcheva, K., and Cunningham, H. (2003). Robust Generic and Query-based Summarisation. In *Proceedings of the European Chapter of Computational Linguistics (EACL), Research Notes and Demos*.

[Schroff et al., 2011] Schroff, F., Criminisi, A., and Zisserman, A. (2011). Harvesting image databases from the web. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(4):754–766.

[Sivic et al., 2006] Sivic, J., Zitnick, C. L., and Szeliski, R. (2006). Finding people in repeated shots of the same scene. In Chantler, M. J., Fisher, R. B., and Trucco, E., editors, *BMVC*, pages 909–918. British Machine Vision Association.

[Smeaton et al., 2006] Smeaton, A. F., Over, P., and Kraaij, W. (2006). Evaluation campaigns and trecvid. In *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*, pages 321–330. ACM.

[Strehl and Ghosh, 2003] Strehl, A. and Ghosh, J. (2003). Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3:583–617.

[Voorhees, 2003] Voorhees, E. M. (2003). Overview of the trec 2003 question answering track. pages 54–68.

[Wu et al., 2013] Wu, B., Zhang, Y., Hu, B.-G., and Ji, Q. (2013). Constrained clustering and its application to face clustering in videos. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3507–3514.

[Yilmaz et al., 2008] Yilmaz, E., Kanoulas, E., and Aslam, J. A. (2008). A simple and efficient sampling method for estimating ap and ndcg. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 603–610. ACM.

[Zhang et al., 2014] Zhang, L., Kalashnikov, D. V., and Mehrotra, S. (2014). Context-assisted face clustering framework with human-in-the-loop. *International Journal of Multimedia Information Retrieval*, 3(2):69–88.

[Zhang et al., 2010] Zhang, W., Zhang, T., and Tretter, D. (2010). Beyond face: Improving person clustering in consumer photos by exploring contextual information. In *Multimedia and Expo (ICME), 2010 IEEE International Conference on*, pages 1540–1545.

[Zhou et al., 2013] Zhou, W., Li, H., Lu, Y., and Tian, Q. (2013). Sift match verification by geometric coding for large-scale partial-duplicate web image search. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, 9(1):4.

[Zhu et al., 2011] Zhu, C., Wen, F., and Sun, J. (2011). A rank-order distance based clustering algorithm for face tagging. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 481–488. IEEE.

[Zhu et al., 2012] Zhu, S., Ngo, C.-W., and Jiang, Y.-G. (2012). Sampling and ontologically pooling web images for visual concept learning. *Multimedia, IEEE Transactions on*, 14(4):1068–1078.