This project has received funding from the European Union’s Seventh Framework Programme for research; technological development and demonstration under grant agreement n: 607480

D6.1 State-of-the-art analysis on tools for multimedia indexing and search

LASIE Project

FP7 - SEC-2013.1.6-1 - Framework and tools for (semi-) automated exploitation of massive amounts of digital data for forensic purposes – Integration Project

Grant Agreement n°: 607480

Start date of project: 1 May 2014

Duration: 42 months

Document. ref.: D6.1/SEN/ WP6/V0.1
Disclaimer

This document contains material, which is the copyright of certain LASIE contractors, and may not be reproduced or copied without permission. All LASIE consortium partners have agreed to the full publication of this document. The commercial use of any information contained in this document may require a license from the proprietor of that information. The reproduction of this document or of parts of it requires an agreement with the proprietor of that information. The document must be referenced if it is used in a publication.

The LASIE Consortium consists of the following partners:

<table>
<thead>
<tr>
<th>Partner Name</th>
<th>Short name</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Engineering Ingegneria Informatica S.p.A.</td>
<td>ENG</td>
<td>I</td>
</tr>
<tr>
<td>2 Centre for Research and Technology Hellas/Information Technologies Institute</td>
<td>CERTH</td>
<td>GR</td>
</tr>
<tr>
<td>3 Neuropublic A.E. Pliroforikis&amp;Epikoinionion</td>
<td>NP</td>
<td>GR</td>
</tr>
<tr>
<td>4 Queen Mary and Westfield College, University of London</td>
<td>QMUL</td>
<td>UK</td>
</tr>
<tr>
<td>5 Metropolitan Police Service</td>
<td>MET</td>
<td>UK</td>
</tr>
<tr>
<td>6 SenseGraph Limited</td>
<td>SEN</td>
<td>UK</td>
</tr>
<tr>
<td>7 Institutt for Fredsforskning Stiftelse</td>
<td>PRIO</td>
<td>N</td>
</tr>
<tr>
<td>8 Huawei Technologies Düsseldorf GmbH</td>
<td>HUA</td>
<td>D</td>
</tr>
<tr>
<td>9 Technische Universität Berlin</td>
<td>TUB</td>
<td>D</td>
</tr>
<tr>
<td>10 United Technologies Research Center Ireland</td>
<td>UTRC</td>
<td>IRL</td>
</tr>
<tr>
<td>11 Innovation Engineering</td>
<td>INNEN</td>
<td>I</td>
</tr>
<tr>
<td>12 Venaka Media Limited</td>
<td>VML</td>
<td>UK</td>
</tr>
<tr>
<td>13 ACIC Video Analytics</td>
<td>ACIC</td>
<td>B</td>
</tr>
<tr>
<td>14 Insitut Mines Telecom</td>
<td>IMT</td>
<td>F</td>
</tr>
<tr>
<td>15 Universidad Politecnica de Madrid</td>
<td>UPM</td>
<td>E</td>
</tr>
<tr>
<td>16 VisionWare – Sistemas de Informação S.A</td>
<td>VIS</td>
<td>P</td>
</tr>
<tr>
<td>17 Ayuntamiento de Madrid – POLICIA MUNICIPAL MADRID</td>
<td>ADM</td>
<td>E</td>
</tr>
<tr>
<td>18 UNIVERSITY OF GREENWICH</td>
<td>UoG</td>
<td>UK</td>
</tr>
<tr>
<td>PROGRAMME NAME:</td>
<td>SEC-2013.1.6-1</td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>PROJECT NUMBER:</td>
<td>607480</td>
<td></td>
</tr>
<tr>
<td>PROJECT TITLE:</td>
<td>LASIE</td>
<td></td>
</tr>
<tr>
<td>RESPONSIBLE UNIT:</td>
<td>SEN</td>
<td></td>
</tr>
<tr>
<td>INVOLVED UNITS:</td>
<td>CERTH</td>
<td></td>
</tr>
<tr>
<td>DOCUMENT NUMBER:</td>
<td>D6.1</td>
<td></td>
</tr>
<tr>
<td>DOCUMENT TITLE:</td>
<td>State-of-the-art analysis on tools for multimedia indexing and search</td>
<td></td>
</tr>
<tr>
<td>WORK-PACKAGE:</td>
<td>WP6</td>
<td></td>
</tr>
<tr>
<td>DELIVERABLE TYPE:</td>
<td>Report</td>
<td></td>
</tr>
<tr>
<td>CONTRACTUAL DATE OF DELIVERY:</td>
<td>30/04/2015</td>
<td></td>
</tr>
<tr>
<td>LAST UPDATE:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DISTRIBUTION LEVEL:</td>
<td>PU</td>
<td></td>
</tr>
</tbody>
</table>

**Distribution level:**

**PU** = Public,

**RE** = Restricted to a group of the specified Consortium,

**PP** = Restricted to other program participants (including Commission Services),

**CO** = Confidential, only for members of the LASIE Consortium (including the Commission Services)
# Document History

<table>
<thead>
<tr>
<th>VERSION</th>
<th>DATE</th>
<th>STATUS</th>
<th>AUTHORS, REVIEWER</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>v.0.1</td>
<td>27/02/2015</td>
<td>Draft</td>
<td>Sofia Segkouli (CERTH)</td>
<td>ToC preparation</td>
</tr>
<tr>
<td>v.1.0</td>
<td>04/03/2015</td>
<td>Draft</td>
<td>Maria Elena Alampi (SEN), Roberto Garigliano (SEN), Romain Serizel (IMT), Ernesto La Mattina (ENG)</td>
<td>Initial Input on text and audio indexing</td>
</tr>
<tr>
<td>v.2.0</td>
<td>20/04/2015</td>
<td>Draft</td>
<td>Theodoros Semertzidis (CERTH), Sofiane Yous (UTRC), Volker Eiselein (TUB)</td>
<td>Image/video indexing input added</td>
</tr>
<tr>
<td>v.3.0</td>
<td>21/04/2015</td>
<td>Draft</td>
<td>Sofia Segkouli (CERTH), Andrea Ciapetti (INNEN)</td>
<td>Text indexing input added</td>
</tr>
<tr>
<td>v.3.1</td>
<td>25/04/2015</td>
<td>Draft</td>
<td>Apostolos Axenopoulos (CERTH)</td>
<td>First version ready for review</td>
</tr>
<tr>
<td>v.4.0</td>
<td>30/04/2015</td>
<td>Final</td>
<td>Petros Daras (CERTH)</td>
<td>Final version</td>
</tr>
</tbody>
</table>
# Definitions, Acronyms and Abbreviations

<table>
<thead>
<tr>
<th>ACRONYMS / ABBREVIATIONS</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCR</td>
<td>Data Co-Reduction</td>
</tr>
<tr>
<td>DVC</td>
<td>Dimension Value Cardinalities</td>
</tr>
<tr>
<td>HMM</td>
<td>Hidden Markov Model</td>
</tr>
<tr>
<td>LSA</td>
<td>Latent Semantic Analysis</td>
</tr>
<tr>
<td>LSH</td>
<td>Locality-sensitive hashing</td>
</tr>
<tr>
<td>LSI</td>
<td>Latent Semantic Indexing</td>
</tr>
<tr>
<td>MSIDX</td>
<td>Multisort Indexing</td>
</tr>
<tr>
<td>NL</td>
<td>Natural-Language</td>
</tr>
<tr>
<td>NLP</td>
<td>Natural-Language Processing</td>
</tr>
<tr>
<td>ONLI</td>
<td>Ontology Natural Language Interaction</td>
</tr>
<tr>
<td>PCA</td>
<td>Principal Component Analysis</td>
</tr>
<tr>
<td>SPARQL</td>
<td>Simple Protocol and RDF Query Language</td>
</tr>
</tbody>
</table>
Table of Contents
Executive Summary ...........................................................................................................9
1 Introduction ..................................................................................................................10
  1.1 The LASIE Individual Search Modules .................................................................10
  1.2 Structure of the Deliverable ..................................................................................12
2 Technologies for Text Indexing and Search .............................................................13
  2.1 Latent Semantic Indexing (LSI) ...............................................................................13
  2.2 Other statistical analysis techniques .....................................................................15
  2.3 Natural-Language (NL) based queries ..................................................................16
  2.4 Applicability to LASIE framework .......................................................................21
3 Technologies for Audio Indexing and Search .........................................................24
  3.1 A survey of audio indexing techniques for large scale datasets .........................24
  3.2 Applicability to LASIE framework .......................................................................26
4 Technologies for Image Indexing and Search .........................................................29
  4.1 Tree-based Techniques .........................................................................................30
  4.2 Dimensionality Reduction ....................................................................................31
  4.3 Hashing Methods ...................................................................................................31
  4.4 The Multisort Indexing algorithm .........................................................................33
  4.5 The influence of image descriptors DVC to large-scale similarity search ..........37
  4.6 Applicability to LASIE framework .......................................................................40
5 Technologies for Video Indexing and Search ..........................................................42
  5.1 Shot Boundary Detection .......................................................................................42
  5.2 Key-frame extraction .............................................................................................43
  5.3 Scene segmentation ...............................................................................................43
  5.4 Video summarization .............................................................................................43
  5.5 Video hashing approaches ....................................................................................43
  5.6 Applicability to LASIE framework .......................................................................45
6 Conclusions ................................................................................................................46
7 References ...................................................................................................................47
List of Figures

Figure 1: The Evidence-based search engine of LASIE .................................................................11
Figure 2: WordNet ........................................................................................................................17
Figure 3: CIRI interface ...............................................................................................................17
Figure 4: KIM architecture .......................................................................................................18
Figure 5: The top of KIM Ontology (KIMO) class hierarchy with expanded Entity branch. (on the left); The Lexical Resources top class hierarchy (on the right). ............................................18
Figure 6: Example of AquaLog in action for basic generic-type queries .....................................19
Figure 7: Example of AquaLog additional information screen from Figure 8 example .................19
Figure 8: The Querix interface after executing the query "What is the biggest state in the US?" ....20
Figure 9: The ask box of Querix offers the possible meanings of "biggest" to the user for selection in order to resolve the ambiguity. ....................................................................................21
Figure 10: The LASIE architecture for Use Case 2 (Accident at Construction Site) ..................22
Figure 11: The graph of SenseGraph .........................................................................................23
Figure 12: Toy illustration of the ITerative Quantization method ................................................25
Figure 13: State-of-the-art method for t-vector indexing on training dataset. ............................26
Figure 14: The LASIE architecture for Use Case 3 (The Missing Person) ....................................27
Figure 15: t-vector based speaker identification on a database indexed with LSH .........................28
Figure 16: a typical high dimensional (3 dimensional) dataset represented in lower dimensions using different manifold learning dimensionality reduction methods. .............................................29
Figure 17: a kd-tree example (wikipedia kd-tree page) ..................................................................30
Figure 18: spherical hashing approach selects hyperspheres instead of hyperplanes to partition the space to binary codes. ........................................................................................33
Figure 19: MSIDX Pre-processing algorithm ................................................................................34
Figure 20: MSIDX insertion algorithm .......................................................................................34
Figure 21: The compare function for multiple sort ......................................................................34
Figure 22: The query processing algorithm ..................................................................................34
Figure 23: the influence of W in 240K images .............................................................................35
Figure 24: influence of W in 1M images .....................................................................................35
Figure 25: (a-e) Retrieval accuracy versus search time for the 5 datasets, (f) number of bits compared with the W parameter that gave similar search times ....................................................................36
Figure 26: Canonical Correlation Analysis ..................................................................................37
Figure 27: Experimental results of CCA for evaluating the performance of LSH, SKLSH, PCA-iTQ, SPH and MSIDX on the six evaluation datasets. ..........................................................37
Figure 28: Empirical cumulative distribution function of F(x) with x = DVC, for different sizes (%N) of the evaluation datasets ...........................................................................................................38
Figure 29: Performance of sequential search of descriptors by eliminating a percentage of the low-DVC dimensions ......................................................................................................................................38
Figure 30: Performance of sequential search of descriptors, after eliminating a percentage of low, high or randomly selected DVC dimensions per dataset. ........................................................................39
Figure 31: Energy preservation (as a percentage of the total energy) after eliminating a percentage of low, high or randomly selected DVC dimensions ......................................................................39
Figure 32: The LASIE architecture for Use Case 1 (Video Analysis for Offender identification) ....41
Figure 33: An illustration of the hash method of [125] within the event category “feeding an animal”. Discriminative local commonality is automatically discovered, e.g., the eyes of animals, the edges of tubs, and the parts of human hands. Temporal consistency is preserved, and successive frames are grouped and put into the same hash bucket. (figure taken from [125]). ...........................................44
Figure 34: Temporal pooling for video hashing (figure taken from [126]).
Executive Summary

The purpose of this document is to report the work carried out in WP6 with respect to the identification of technologies for large-scale multimedia indexing and search. These technologies will be used as starting point for the development, during the second year of the LASIE project, of the Individual Search modules that will constitute the main components of the LASIE Evidence-based Search Engine.

After a short introduction of the LASIE Evidence-based Search Engine and a description of its main components, the analysis of existing technologies for large-scale multimedia indexing follows. State-of-the-art analysis is performed separately for each content type, i.e. text, audio, image, video, since each of the above types has different requirements in terms of processing and indexing. More specifically, for text indexing and search, technologies for Latent Semantic Analysis (LSA), Latent Semantic Indexing (LSI), Natural-Language (NL) based querying as well as other statistical analysis techniques are analysed. Regarding audio indexing, emphasis is put on hashing techniques since they appear to be more efficient. Among the technologies for image indexing, tree-based techniques, dimensionality reduction and hashing methods are presented, explaining also the superiority of hashing methods. Finally, video indexing can be covered either by image indexing techniques or even technologies that consider the temporal dimension of the video content and generate index points for video segments or other structures.

For each content type, the applicability of related indexing methods to LASIE framework is explained and the role of the specific module within the LASIE architecture is presented. Then, an initial decision for selection of the most appropriate technology as well as suggestions for potential improvements of existing works takes place.

This document should be considered as a guide for the development of the Individual Search modules, in the context of Task 6.1 of LASIE, which will take place during the second year of the project.
1 Introduction

This deliverable belongs to WP6 of the LASIE project, which has as primary objective the development of an Intelligent Evidence-based Search Engine. The search engine will enable search and retrieval in both the existing forensic databases and the various captured forensic data. The searchable data types comprise text (in the form of documents, email, internet history, calligraphic, etc.), images (faces, tattoo, plates, etc.), video (stored in local disks, surveillance, etc.), audio (speech, emotions, etc.), social and biometric data.

The individual search modules will exploit technologies for distributed processing and large-scale indexing to enable fast and efficient search even to vast repositories of forensic content. Indexing will be performed on the metadata extracted with the processing tools that will be implemented within WP4 of the LASIE project. Due to the high heterogeneity of these data types, several search modules will be implemented, each one adapted to the specific technical constraints of each type.

The Task under which this work has been accomplished is Task 6.1 – Individual Search modules, led by SenseGraph (SEN) with ENG, CERTH, TUB, UTRC and IMT contributing to it. The purpose of this document is to present a thorough state-of-the-art analysis on existing technologies that have been proposed so far to tackle the problem of large-scale multimedia indexing and search. The analysis is performed on each content type separately (i.e. text, audio, image, video) taking into account their different requirements and technical constraints. Moreover, a brief description of the role of each indexing module within the LASIE framework is provided, followed by requirements that will drive the research of Task 6.1.

1.1 The LASIE Individual Search Modules

The proposed LASIE Evidence-based Search Engine is presented in Figure 1. It consists of the individual search modules, the complex query formulation module and the iterative refinement module. The individual search modules perform indexing and search capabilities for each type of forensic content that is available within LASIE and are described in the sequel.

Text Indexing Module: This module supports large-scale generic text-based search, as well as advanced text queries using Natural Language (NL) questions. Text that has been acquired from various sources and processed using the text processing modules (WP4) will be indexed for fast search and retrieval. The Text Indexing module will address problems such as term synonymy and polysemy using advanced techniques such as Latent semantic indexing (LSI). LASIE will also allow the user to query the system using NL questions. This method of query formulation will be applicable not only to text content but also to multimedia content (e.g. video, sound), where high-level information has been extracted.

Audio Indexing Module: The Audio Indexing Module evaluates the available sources as phone, video, and microphone arrays at sensible locations with respect to background signals and human voice signals. Due to the classifiers, a labelled background (vehicle or railway traffic, pedestrians, etc.) could be extracted. On the other side, the number of speakers, the speaker’s gender, emotion or obviously deviations from a native dialect should be indexed. Next challenge is the speaker-independent text recognition, which is to notate into NLP notation and then to proceed as text.

Image Indexing Module: The Image Indexing Module will enable search for image content in forensic data using both textual and visual queries. In the former case, a text-based search on the extracted metadata (EXIF, semantic descriptions after object detection and recognition) will be performed. In the latter case, a query by example process will be
realised, i.e. an image will be added as query and the indexing module will return results with visual similarity to the query.

**Video Indexing Module:** The task of the Video Indexing Module is to convert video streams into structured and indexed textual information entities. This significantly simplifies video querying as video object searching can be analogous to word searching (the NL-based query module can be used for querying). A great research challenge for LASIE is the event-based indexing. Event metadata that have been calculated for the video summarisation, will complement the simple ones and will offer a more efficient context-based indexing schema. Following the extraction of the corresponding event metadata, the indexing process will involve appropriate representation of the events in a semantic metadata description language, which will provide links between events and video items.

The Complex Query Formulator and Iterative Process Refinement are also parts of the Evidence-based search engine. Since work of these modules is part of other WP6 tasks, only a brief description is provided here. More specifically, the Complex Query Formulator will be implemented as a workflow that will orchestrate the entire search process. It will receive the input from the LASIE user interface, which will be multimodal, invoke the search services of the appropriate indexing modules, collect the results from the multiple search modules and, finally, fuse the results in order to present a unified result list to the investigator. The Iterative Process Refinement module will take as input the result list from the query formulation module and the user's feedback, who marks which of the retrieved results are relevant or not. Then, an appropriate relevance feedback mechanism will be responsible for formulating a new refined query and trigger a new search request to the corresponding search modules.
1.2 Structure of the Deliverable

The rest of the document is organised as follows:

Section 2 – Technologies for Text Indexing and Search – analyses technologies for Latent Semantic Analysis (LSA) and Latent Semantic Indexing (LSI), technologies for Natural-Language (NL) based querying as well as other statistical analysis techniques. Then it describes the requirements for text search within the LASIE framework, more specifically to Use Cases 2 and 3, and briefly outlines the work proposed towards this direction.

Section 3 – Technologies for Audio Indexing and Search – provides a survey on audio indexing techniques for large-scale datasets, followed by a solution proposed in order to fulfil the requirements of audio search in LASIE.

In Section 4 – Technologies for Image Indexing and Search, a summary of the main categories of image indexing techniques is given, namely tree-based techniques, dimensionality reduction and hashing methods. A more extensive description is provided for hashing methods, since they have proven more appropriate for such tasks, explaining also how techniques for hashing could be exploited to fulfil Use Case 1 of LASIE and what improvements are needed to address the large-scale requirements.

Section 5 – Technologies for Video Indexing and Search – complements the previous section, by covering those technologies that consider the temporal dimension of the video content and generate index points for video segments or other structures.

Finally, conclusions are drawn in Section 6.
2 Technologies for Text Indexing and Search

2.1 Latent Semantic Indexing (LSI)

Latent Semantic Analysis (LSA) is a theory and method for extracting and representing the contextual-usage meaning of words by statistical computations applied to a large corpus of text [128]. The underlying idea is that the aggregate of all the word contexts in which a given word does and does not appear provides a set of mutual constraints that largely determines the similarity of meaning of words and sets of words to each other. The adequacy of LSA’s reflection of human knowledge has been established in a variety of ways. For example, its scores overlap those of humans on standard vocabulary and subject matter tests; it mimics human word sorting and category judgments; it simulates word–word and passage–word lexical priming data and it accurately estimates passage coherence, and the quality and quantity of knowledge contained in an essay. Research reported in example by [129], [130], [131] exploits a new theory of knowledge induction and representation [128] that provides a method for determining the similarity of meaning of words and passages by analysis of large text corpora. After processing a large sample of machine-readable language, Latent Semantic Analysis (LSA) represents the words used in it, and any set of these words—such as a sentence, paragraph, or essay—either taken from the original corpus or new, as points in a very high (e.g. 50-1,500) dimensional “semantic space”.

LSA is closely related to neural net models, but is based on singular value decomposition, a mathematical matrix decomposition technique closely akin to factor analysis that is applicable to text corpora approaching the volume of relevant language experienced by people.

Word and passage meaning representations derived by LSA have been found capable of simulating a variety of human cognitive phenomena, ranging from developmental acquisition of recognition vocabulary to word-categorization, sentence-word semantic priming, discourse comprehension, and judgments of essay quality.

LSA technique can be construed in two ways:

1. simply as a practical expedient for obtaining approximate estimates of the contextual usage substitutability of words in larger text segments, and of the kinds of meaning similarities among words and text segments that such relations may reflect, or

2. as a model of the computational processes and representations underlying substantial portions of the acquisition and utilization of knowledge.

As a practical method for the characterization of word meaning, we know that LSA produces measures of word-word, word-passage and passage-passage relations that are well correlated with several human cognitive phenomena involving association or semantic similarity. The correlations demonstrate close resemblance between what LSA extracts and the way peoples’ representations of meaning reflect what they have read and heard, as well as the way human representation of meaning is reflected in the word choice of writers.

As one practical consequence of this correspondence, LSA allows to closely approximate human judgments of meaning similarity between words and to objectively predict the consequences of overall word-based similarity between passages, estimates of which often figure prominently in research on discourse processing.

It is important to note that the similarity estimates derived by LSA are not simple contiguity frequencies, co-occurrence counts, or correlations in usage, but depend on a powerful mathematical analysis that is capable of correctly inferring much deeper relations (thus the definition “Latent Semantic”), and as a consequence are often much better predictors of
human meaning-based judgments and performance than are the surface level contingencies that have long been rejected [132] by linguists as the basis of language phenomena.

LSA, as currently practiced, induces its representations of the meaning of words and passages from analysis of raw text. None of its knowledge comes directly from perceptual information about the physical world, from instinct, or from experiential intercourse with bodily functions, feelings and intentions. Thus its representation of reality is bound to be somewhat sterile and bloodless. However, it does take in descriptions and verbal outcomes of all these juicy processes, and so far as writers have put such things into words, or that their words have reflected such matters unintentionally, LSA has at least potential access to knowledge about them. The representations of passages that LSA forms can be interpreted as abstractions of “episodes”, sometimes of episodes of purely verbal content such as philosophical arguments, and sometimes episodes from real or imagined life coded into verbal descriptions. Its representation of words, in turn, is intertwined with and mutually interdependent with its knowledge of episodes. Thus while LSA’s potential knowledge is surely imperfect, we believe it can offer a close enough approximation to people’s knowledge to underwrite theories and tests of theories of cognition.

LSA differs from some statistical approaches discussed below in two significant respects. First, the input data “associations” from which LSA induces representations are between unitary expressions of meaning (words and complete meaningful utterances in which they occur) rather than between successive words. That is, LSA uses as its initial data not just the summed contiguous pairwise (or tuple-wise) co-occurrences of words but the detailed patterns of occurrences of very many words over very large numbers of local meaning-bearing contexts, such as sentences or paragraphs, treated as unitary wholes. Thus it skips over how the order of words produces the meaning of a sentence to capture only how differences in word choice and differences in passage meanings are related.

Another way to think of this is that LSA represents the meaning of a word as a kind of average of the meaning of all the passages in which it appears, and the meaning of a passage as a kind of average of the meaning of all the words it contains. LSA’s ability to simultaneously derive representations of these two interrelated kinds of meaning depends on an aspect of its mathematical machinery that is its second important property.

LSA assumes that the choice of dimensionality in which all of the local word-context relations are simultaneously represented can be of great importance, and that reducing the dimensionality (the number parameters by which a word or passage is described) of the observed data from the number of initial contexts to a much smaller (but still large) number will often produce much better approximations to human cognitive relations. It is this dimensionality reduction step, the combining of surface information into a deeper abstraction that captures the mutual implications of words and passages. Thus, an important component of applying the technique is finding the optimal dimensionality for the final representation. A possible interpretation of this step, in terms more familiar to researchers in psycholinguistics, is that the resulting dimensions of description are analogous to the semantic features often postulated as the basis of word meaning, although establishing concrete relations to interpretable features poses daunting technical and conceptual problems and has not yet been much attempted.

Finally, LSA, unlike many other methods, employs a pre-processing step in which the overall distribution of a word over its usage contexts, independent of its correlations with other words, is first taken into account; pragmatically, this step improves LSA’s results considerably.

However, as mentioned previously, there is another, quite different way to think about LSA. Landauer and Dumais [128] have proposed that LSA constitutes a fundamental computational theory of the acquisition and representation of knowledge. They maintain that its underlying mechanism can account for a long-standing and important mystery, the
inductive property of learning by which people acquire much more knowledge than appears to be available in experience, the infamous problem of the "insufficiency of evidence" or "poverty of the stimulus." The LSA mechanism that solves the problem consists simply of accommodating a very large number of local co-occurrence relations (between the right kinds of observational units) simultaneously in a space of the right dimensionality. Hypothetically, the optimal space for the reconstruction has the same dimensionality as the source that generates discourse, that is, the human speaker or writer's semantic space. Naturally observed surface co-occurrences between words and contexts have as many defining dimensions as there are words or contexts. To approximate a source space with fewer dimensions, the analyst, either human or LSA, must extract information about how objects can be well defined by a smaller set of common dimensions. This can best be accomplished by an analysis that accommodates all of the pairwise observational data in a space of the same lower dimensionality as the source. LSA does this by a matrix decomposition performed by a computer algorithm, an analysis that captures much indirect information contained in the myriad constraints, structural relations and mutual entailments latent in the local observations available to experience.

The principal support for these claims has come from using LSA to derive measures of the similarity of meaning of words from text. The results have shown that:

1. the meaning similarities so derived closely match those of humans,
2. LSA's rate of acquisition of such knowledge from text approximates that of humans, and
3. these accomplishments depend strongly on the dimensionality of the representation.

In this and other ways, LSA performs a powerful and, by the human-comparison standard, correct induction of knowledge. Using representations so derived, it simulates a variety of other cognitive phenomena that depend on word and passage meaning.

**Latent semantic indexing (LSI)** is an indexing and retrieval method that uses the mentioned mathematical techniques (called singular value decomposition, SVD) to identify patterns in the relationships between the terms and concepts contained in an unstructured collection of text. LSI is based on the theory of LSA and, specifically, on the principle that words that are used in the same contexts tend to have similar meanings. A key feature of LSI is its ability to extract the conceptual content of a body of text by establishing associations between those terms that occur in similar contexts.

### 2.2 Other statistical analysis techniques

Statistical natural language processing is concerned with the creation of computer programs that can perform language-processing tasks by virtue of information gathered from (typically large) corpora. Usually this information is in the form of statistics, but it may be distilled into other forms, such as decision trees, dictionary entries, or various kinds of rules. The techniques have been applied to areas as diverse as lexicography and document retrieval [133].

Most statistical language programs sacrifice depth or accuracy to breadth of coverage. That is, statistical NLP programs typically work on most everything one throws at them, but they do not provide very deep analyses and/or may make occasional mistakes. This has the salutary effect of making it easier to compare competing techniques since they will work on the same body of data (i.e., "everything"). They work as well as they do because one can get a great deal of leverage from relatively weak (and easily collected) pieces of information --
combined with the fact that, at least at the shallower linguistic depths, computers seem better at collecting linguistic phenomena than people are at thinking them up.

Historically, the push behind statistical techniques came primarily from the speech community that discovered in the early 1970s that programs automatically trained from corpora worked better than their hand-tooled versions. These programs exploited the training properties of HIDDEN MARKOV MODELS. These models can be thought of as finite state machines in which the transitions between states have probabilities. There are well-understood mathematical techniques for training them -- adjusting the probabilities to better fit the observed data. The "hidden" in their name comes from the fact that in such models one cannot know the sequence of states that produced the output (or, equivalently, accepted the input), but there are linear-time techniques for finding the most probable of such state sequences. The trick is then to identify states with the properties one wishes to infer (e.g., the word uttered).

These speech programs also typically incorporate language models, assignments of probabilities to all sequences of words in the language. The most popular such model is the simple but remarkably accurate trigram model in which the probability of the next word is conditioned on just the two previous words. The language model enables the speech recognizer to pick the best word in context when the speech signal analysis by itself is insufficient.

An early successful application of HMM technology to language tasks was the HMM "taggers," programs that try to assign the correct part of speech to each word in a text. For example, in the sentence "can will rust," the program should identify can as a noun (not a modal verb) and will as a modal verb (not a noun). In these programs, the states of the HMM correspond to the parts of speech, so that finding the most probable sequence of states when the HMM is driven by the input gives us the desired tagging. Other schemes compile the statistics into rules. Moderately well-crafted programs achieve about 96 percent accuracy. For comparison, human taggers are consistent with one another at a 98 percent level.

2.3 Natural-Language (NL) based queries

There are many query expansion implementations used in keyword search that make use of ‘thesaurus ontology navigation’ as a step in query expansion such as [1], [2] and [3]. In these works, the large, well-known WordNet (Figure 2) ontology (http://wordnet.princeton.edu) is utilised.
WordNet Search - 3.1
- WordNet home page - Glossary - Help

Word to search for: police

Display Options: Show all

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: [frequency] [offset] <lexical filename > [lexical file number] (gloss) "an example sentence"

Display options for word: word#sense number (sense key)

Noun
- (34)(08226608) <noun.group>[14] S: (n) police#1 (police%1:14:00::), police
  force#1 (police_force%1:14:00::), constabulary#1 (constabulary%1:14:00::), law#7
  (law%1:14:01::) (the force of policemen and officers) "the law came looking for him"

Verb
- (4)(02460361) <verb.social>[41] S: (v) patrol#1 (patrol%2:41:00::), police#1
  (police%2:41:00::) (maintain the security of by carrying out a patrol)

Figure 2: WordNet

These kinds of systems function according to the same basic scheme:
- first, the keywords are located in the ontology, then,
- various other concepts are located through graph traversal, after which the terms related to those concepts are used to either broaden or constrain the search.

At the same time, systems such as the CIRI system [4] provide an ontological front-end text search (Figure 3). The search is performed through an ontology browser that visualises the ontologies as sub-assumption trees, from which concepts can be selected to constrain the search. The actual search is done through keywords annotated to these concepts and sub-concepts, using a traditional search engine.

Figure 3: CIRI interface

A multitude of tools for accessing data contained in ontologies and knowledge bases have been proposed so far. A popular idea is adding search and browsing support to ontology editing environments. For instance Protégé [5] provides the Query Interface, where the user can specify the query by selecting some options from a given list of concepts and relations. Alternatively, advanced users can type a query using a formal language such as SPARQL. Query languages have complex syntax, requiring a deep understanding of the representation schema. One step to simplify the approach it is creating a form-based graphical interface.
where the information request can be expressed by putting together a set of restrictions. Typically, these restrictions are provided as ready-made building blocks that the user can add to create a complex query. Such systems can be customised for a particular application domain or general purpose.

An example is provided by the KIM (Figure 4, Figure 5) knowledge management platform [6]. This type of interfaces is suited for domain specific uses; when customised to a particular application, they provide the path to the information.

Figure 4: KIM architecture

Figure 5: The top of KIM Ontology (KiMO) class hierarchy with expanded Entity branch. (on the left); The Lexical Resources top class hierarchy (on the right).

Their disadvantages, however, become apparent when moving to a general purpose search interface; in this case, they can be either too restrictive or too complex with either too many
input fields or too many alternative options [7]. SemSearch [8] is a concept-based system which aims to have a Google-like Query Interface. It requires a list of concepts (classes or instances) as an input query.

Another example is AquaLog [9]. It uses a controlled language for querying ontologies with the addition of a learning mechanism, so that its performance improves over time in response to the vocabulary used by the end users (Figure 6, Figure 7). The system works by converting the natural language query into a set of ontology-compatible triples that are then used to extract information from a knowledge store. It utilises shallow parsing and WordNet, and so requires syntactically correct input. It seems geared mainly towards queries containing up to two triples and expressed as questions (e.g., who, what).

![Figure 6: Example of AquaLog in action for basic generic-type queries](image)

![Figure 7: Example of AquaLog additional information screen from Figure 8 example](image)
Orakel [10] is a natural language interface (NLI) to knowledge bases. It supports compositional semantic construction, which in turn helps supporting questions involving quantification, conjunction and negation.

ONLI (Ontology Natural Language Interaction) [11] is a natural language question answering system used as front-end to the RACER reasoner and to nRQL, RACER’s query language. ONLI assumes that the user is familiar with the ontology domain and works by transforming the user’s natural language queries into nRQL.

Querix [12] is another ontology-based question answering system that translates generic natural language queries into SPARQL (Figure 8, Figure 9). In case of ambiguities, Querix relies on clarification dialogues with users.

Figure 8: The Querix interface after executing the query "What is the biggest state in the US?"
2.4 Applicability to LASIE framework

There are two Use Cases, within the LASIE framework, in which text indexing will be exploited:

- Use Case 2: Accident at Construction Site, and
- Use Case 3: The Missing Person

In Figure 10 and (Figure 14), the role of the Text Indexing module and its interaction with the other LASIE modules is depicted for Use Cases 2 and 3, respectively.

The module takes as input the outcome of text processing modules (WP4), such as the NLP module, and allows Natural-Language (NL) queries, i.e. the user can ask questions to the system using natural language, through the user interface (User Experience Framework).

The models proposed in the literature (those described in the previous subsections) are essentially based on SPARQL language and other existing language-based query interfaces that either support keyword-like search or require full-blown. It can be easily imagined the increase of the quality that could be achieved through the application of a dynamic model based on linguistic pragmatic inferences, not only in measurable terms of accuracy recognition of a word but also endowed with the ability to force an interpretation connecting linguistic fragments coming from the speech in a consistent manner with notions derived from ontology and from the current context. SenseGraph approach deals with free texts which often involve various interpretations and for this reason the complexity of the analysis is substantial.

Even a normal sentence, such as those that can be found in a Wikipedia article or on a manual for a product, can produce hundreds of parse trees legitimate, arising from combinatorial explosion of ambiguity. The approach used in LASIE, however, intends to give a much more accurate understanding of the text; thus, such ambiguities can be avoided. This is especially important in the field of investigation, where there is the aspiration to find in the text crimes, victims and distinguish between events planned and executed.

The SenseGraph system relies on a graph structure where information is linked freely (Figure 11). In fact, to answer a query, the system must have already analysed a large text (modules described in WP4 of LASIE). After the analysis, the system can proceed with the query and then a graph appears based on it. For example, if we need to find information
about an arrest involving a group of men, with a specific location and time, the query will be: *Where and when were the men arrested?* Therefore, due to the inference system, there will be a match between the two graphs (the graph based on the analysis of the larger text and the one based on the query) from which the results shown will be the information requested.

Figure 10: The LASIE architecture for Use Case 2 (Accident at Construction Site).
The analysis is the most important part. The SenseGraph approach is different from the other systems based on query to database that use the natural language as prefabricated linguistic structures corresponding to language-based query interfaces such as SQL, where information is already organised in a structured table. Often, they are accessing structured data such as that encoded in ontologies and knowledge bases that can be done using either syntactically complex formal query languages like SPARQL or complicated form interfaces that require expensive customisation to each particular application domain. SenseGraph, on the contrary, uses free text (the original natural language employed in text and communication between people) that can be written in every possible form: free text entries (unstructured texts) into the exploitation of forensic data give the user flexibility to note observations and concepts that are not supported or anticipated by the constrained choices associated with structured data.
3 Technologies for Audio Indexing and Search

3.1 A survey of audio indexing techniques for large scale datasets

Since the early 00’s a multitude of approaches have been proposed to tackle indexing of high-dimensional multimedia data on large scale datasets [13], [14], [15], [55], [91]. Up to recently however, research was mainly focused on visual information and only limited work was done on indexing audio data. More specifically, work on audio data mainly addressed extracting of signal signatures through the analysis of the human perception of audio scenes [16], [17], [18], the extraction of features inspired by this perceptual behaviour [16], [19], [20], [21], [22], [23] and the design of statistical models that would allow to extract relevant information from these features [20], [21], [22], [24], [25], [26].

Audio data was generally represented in a relatively low-dimensional space and indexing was then merely considered as a distance minimization [27], [28], [29], [30] sometimes weighted by an element importance function such as term frequency – inverse document frequency [29]. These approaches however are not scalable and perform poorly on high-dimensional spaces. With the increasing amount of audio data available and the surfaced of high-dimensional representations for audio data, researchers have started investigating methods that allow fast queries on large scale audio datasets with high-dimensional representations [31].

As a result from this relatively late focus on indexing for high-dimensional spaces and large scale datasets, methods applied on audio datasets are generally adapted from methods designed for textual or visual data. Among these methods two main approaches can be considered: tree-based methods and hashing methods.

Tree-based methods [15], [32], [33], [34] are efficient to retrieve exact solutions on low-dimensional problems but their applicability on high-dimensional data is limited due to the fact that they involve search on a large fraction of the dataset and only guarantee linear time query. Methods relying on features quantization [32] and decision trees [34] have been explored to overcome these limitations but tree-based methods’ use in audio still remains limited.

Hashing techniques [55], [14], [31], [35], [36], [37], [38], [45], [67] on the other hand, are designed to guarantee sub-linear time query even on high-dimensional data. In the nearest neighbour problem, we consider a set of points \( \{x_1, …, x_n\} \subseteq X \), a query point \( q \) and a distance measure:

\[
d : X \times X \rightarrow \mathbb{R}^+.
\]

The goal is then to find the point \( x_i \) that minimizes the distance \( d(x_i, q) \). Locality-sensitive hashing (LSH) relies on the premise that in many nearest neighbour problems, the exact solution is not needed and that an approximation is often “good enough”. Besides, an approximation is still useful as the measure \( d \) is often an approximation of the ground truth itself. Hashing techniques allow to trade-off between the estimation quality and the query time. Geometrically, LSH can be seen as approximating the nearest neighbour solution with projections on random hyperplanes.

The main idea is to hash points with several simple hashing functions that ensure, for a specific distance, points that are close to each other are likely to collide [55], [14]. A family of hashing functions \( \mathcal{H} = \{h : X \rightarrow [0, 1]\} \) is called \((r_1, r_2, p_1, p_2)\)-sensitive for \( d \) if:
∀ \( x, q \in X \) if \( d(x, q) \leq r_1 \) then \( p[h(q) = h(x)] \geq p_1 \)
if \( d(x, q) > r_2 \) then \( p[h(q) = h(x)] \leq p_2 \)

In order for the LSH family to be useful to assert similarity, the following inequalities have to be satisfied by \((r_1, r_2, p_1, p_2)\):

- \( p_1 > p_2 \)
- \( r_1 < r_2 \)

For a LSH family, the algorithm builds a set of hash tables, in which several hash functions are concatenated in order to avoid collisions and several tables can be maintained to increase the probability to find a close neighbour.

Across the years, LSH functions have been introduced to ensure compatibility with several distance metrics, among which: \( l_p\)-norms [56], Hamming distance [45], inner product [14], cosine distance [31] and general kernels [70]. To further improve the classification time and performance, Weis et al. [67] have proposed spectral hashing, a method based on spectral partitioning and Gong et al. [91] have introduced iteration quantization, an alternative way to solve binary code problems while relaxing the distribution assumption of spectral hashing.

In iterative quantization, the points are projected on hyperplanes, similarly as in hashing techniques, but the hyperplanes are now obtained after principal component analysis (PCA) (see Figure 12 - a). Jegou et al. [39] have suggested that applying random orthogonal transformation on the PCA-projected data (see Figure 12 - b) allows to balance the variance along different PCA directions and to outperform approaches as spectral hashing [67]. Gong et al. [91] proposed an iterative method to find optimal orthogonal projection that minimizes the quantization error (see Figure 12 – c).

Recently, LSH based methods and iterative quantization have been applied to index l-vectors [26] databases for speaker recognition with promising results [31], [40], [41]. The distance \( d \) is then the cosine distance defined as follows:

\[
d(x, y) = \frac{x \cdot y}{\|y\| \|y\|}
\]
The cosine distance gives competitive performance on speaker identification [26], not too far from the state-of-the-art approaches based on probabilistic discriminant linear analysis [25]. It also has the advantage to be well approximated by LSH functions [14] and in particular the following functions:

\[
    h_r(x) = \begin{cases} 
        1 & \text{if } r \cdot x \geq 0 \\
        0 & \text{if } r \cdot x < 0 
    \end{cases}
\]

where \( r \) is a random Gaussian vector. Geometrically, the hash function \( h_r \) is projecting on the hyperplane orthogonal to \( r \) and \( h_r(x) \) indicates on which side of the hyperplane \( x \) lies.

### 3.2 Applicability to LASIE framework

The Audio Indexing module will be exploited in Use Case 3 of the LASIE framework: The Missing Person. The role of the module within the LASIE framework is summarised in Figure 14. The module takes as input the features extracted from the audio processing modules, such as the Machine Listening module and the Speech Analysis module. Then, a hashing method is used for indexing these features. The indexing module interacts with the user interface (User Experience Framework) during the search and retrieval phase. A more detailed description of this process is provided in the sequel.
D6.1 State-of-the-art analysis on tools for multimedia indexing and search

Figure 14: The LASIE architecture for Use Case 3 (The Missing Person).
Regarding the problems of speaker recognition (speech analysis module) and acoustic scene recognition (machine listening module), within the context of the LASIE project, the first part is to index the training dataset (Figure 13). As an initial step, we extract low-level acoustic features from the audio recordings in the training datasets. Then, the input audio recordings in the training datasets have to be segmented into homogeneous parts. High level representations can be extracted, for example I-vectors for speaker identification. Finally, indexing methods described above such as LSH [55], [14], [31], [35], [36], [37], [38], [67], [45] or iterative quantization [91] can be applied on training datasets to compute hash tables and provide speaker-based indexing of the datasets (speaker recognition) and acoustic-scene-based indexing of the datasets (acoustic scene recognition).

The information is accessed with a query by example approach that is speaker or acoustic scenes are identified from an unknown incoming audio recording (Figure 15). Acoustic features are extracted from the incoming test audio recording. Then, high level representation can be extracted from acoustic features. The hash functions are evaluated from the high level representation of the query and a search through the hash tables (obtained during the training phase) allows finding the best candidate.

As far as event detection (machine listening module) and speech recognition (speech-to-text modules) are concerned, approaches relying on event-based indexing and semantic queries are to be considered [42], [43], [44].

---

**Figure 15:** I-vector based speaker identification on a database indexed with LSH.
4 Technologies for Image Indexing and Search

The widespread availability of all-connected devices and the wide adoption of digital cameras have resulted in vast amounts of digital images that are generated on a daily basis. To cope with the increasing amounts of visual information and easily access the petabytes of publicly available images, image search engines have recently become an invaluable tool (such as the text search engines in the first years of the Internet era). Most of the existing image search engines follow the content-based search approach, in order to bring visually similar content. The most naïve approach is exhaustive search; however, it is infeasible for large scale applications due to its extensive time requirements. Moreover, taking into account the constant creation of new images on a daily basis, image search systems should also address the issue of emerging storage requirements.

In an attempt to deal with the above issues, large-scale indexing techniques are able to provide efficient search time and retrieval accuracy and, at the same time, compress the information that is needed to be stored. However, multimedia objects like compressed images, video and audio streams are usually described by sequences of descriptor vectors with over than a thousand dimensions, and their similarity is examined by nearest neighbour search using typical distance metrics. In this high dimensional space, the performance of most indexing methods is challenged by the well-known problem of Curse of Dimensionality [45][46].

A typical approach to address the above issue is by dimensionality reduction methods [47],[48],[49],[50],[51],[52],[53],[54] and hashing techniques [55], [56], [57], [58], [59], [60], [61], [62], [63], [64],[65],[66],[67],[68],[69],[70],[71],[72],[73],[74]. These approaches have been proposed to overcome the problems of the tree-based methods and thus to provide efficient solutions for high-dimensional data. In the first case, dimensionality reduction methods try to reduce the number of dimensions of the high-dimensional data. In such methods the data is projected on lower-dimensional spaces by using dimensionality reduction methods and then are indexed in their new projected space.

Figure 16: a typical high dimensional (3 dimensional) dataset represented in lower dimensions using different manifold learning dimensionality reduction methods.
In the second case, the data is encoded into binary codes using appropriate hash functions. Using hash functions, higher scalability is achieved due to the compactness of the data and the fast distance computations using hamming distance. Similar high-dimensional objects are mapped to similar binary codes. Therefore, approximate nearest neighbour search is performed by only examining similar binary codes. However, the hashing methods often fail to achieve high accuracy due to approximations, especially when the hashing functions are drawn independently from the data, or when a short binary code length is selected. On the other hand, long binary code lengths require significant pre-processing/training time that is prohibitive.

4.1 Tree-based Techniques

Several tree-based indexing methods have been proposed for the problem of nearest neighbour search, such as: KD-trees [75][76][77], R-trees [78], M-Trees [79], Quad-Trees [80], Vantage Point Trees (VPT) [81], Voronoi Trees (VT) [82], etc. Additionally, several tree-based indexing methods for approximate nearest neighbour search have also been proposed, such as: Spatial Approximation Tree (SAT) [83], Approximating Eliminating Search Algorithm (AESA)[84]. Extensive surveys for the most tree-based methods can be found in [85], [86], [87]. The main strategy for all tree-based indexing methods is to prune tree branches on the established bounding distances in order to reduce the node accesses.
However, in high-dimensional spaces, where the multimedia objects lie, the tree-based indexing methods are inefficient, performing worse than exhaustive search [88].

4.2 Dimensionality Reduction

Instead of indexing the data into the original high-dimensional space, dimensionality reduction methods aim at mapping the data into a lower-dimensional subspace. The main idea is to make such a transformation without losing much information and build an index on the subspace. Several local and global dimensionality reduction methods have been proposed so far [47], [49], [50], [51], [52], [53], [54]. Global dimensionality reduction methods map the whole dataset into a much-lower dimensional subspace. For example, the Isometric Feature Mapping method estimates geodesic distances and uses them to project the data into the embedded space. Local dimensionality reduction methods divide the dataset into correlated clusters and then each cluster is reduced in subspaces independently. For example, the Locally Linear Embedding method projects the data to a low-dimensional space, while preserving local geometric properties.

The pre-processing cost of such transformations is often high, due to dense matrix operations (especially products, eigenvector and eigenvalue calculations). Several optimization techniques have been proposed to reduce the pre-processing cost and keep the accuracy of the estimated distances [50], [51]. For example, the Laplacian Eigenmaps method [54] uses an extra weighted distance between the data points as a loss function in order to evaluate and optimize the dimensionality reduction results.

Dimensionality reduction methods can be used either for approximate or exact similarity search. In the first case, the similarity search is performed only into the transformed subspace. In the second case, firstly the similarity search is performed into the transformed space, where lower bounds on the distances are used for filtering, then a resulting set of candidates is returned, and finally the candidates are refined in the original space with exact search (e.g. [50]).

In the case of exact similarity search, the recently proposed Data Co-Reduction (DCR) method of [50] significantly outperforms the existing methods for lossless retrieval, especially in the presence of extremely high dimensionality. The DCR method reduces simultaneously both the size and the dimensionality of the original data into a compact subspace, where lower bounds of the actual distances in the original space can be efficiently established to achieve fast and lossless similarity search in a filter-and-refine approach. In the case of approximate similarity search, it has been demonstrated that the dimensionality reduction methods return accurate results in relative low dimensional spaces [51].

4.3 Hashing Methods

The hashing methods have proven to be suitable for approximate similarity search, since they support efficient indexing and data compression. The basic idea of the hashing methods is (a) to encode the distances between the data into the form of compressed sequences of bits by using hash functions, and (b) to store the encoding distances into buckets, in order to ensure that the probability of collision is much higher for data that are close to each other than those that are far apart. Then, they approximate exact similarity measures by comparing hash codes, using a hamming distance on binary codes or other measures. Different strategies are followed during the pre-processing for the generation of the binary codes. The existing hashing methods can be broadly categorized as data-independent and data-dependent.
In data-independent hashing methods, the hashing functions are defined independently from the data. One of the most popular methods is Locality Sensitive Hashing (LSH) [55], which is based on projection onto random vectors drawn from a specific distribution. Many hashing methods, where also a randomized process is followed in various metric spaces, are: p-stable LSH [56], min-hash [57], Shift-Invariant Kernels Hashing [58], Entropy based LSH [59], Multi-Probe LSH [60], posteriori Multi-Probe LSH [61], etc. Such methods are based on random projections and according to the Johnson-Lindenstrauss Theorem [89], at least \( O(\ln n/\epsilon^2) \) projection vectors are required (where \( n \) is the dataset size), so as to preserve the pairwise distances with a relative error \( \epsilon \). Therefore, in order to decrease the relative error and increase the probability that similar objects have similar hash codes, the random projection based methods require many random vectors to generate the hash tables (each table corresponds to one random vector), leading to a large storage space and a high computational cost. Alternatively, several methods instead of following randomization strategies use deterministic structuring based methods like grids [62], space filling curves [63],[64], or lattices [65]. Since the efficiency of the deterministic structuring methods depends on the data distributions, the randomized processes are more adaptive and usually more efficient. However, the data-independent hashing methods are often inefficient, especially for short lengths of binary codes, due to the fact that their hashing functions are drawn independently from the data.

In data-dependent hashing methods, the hashing functions are defined only for a preselected training dataset, which is usually a subset of the data, and involve similarity calculations for the training dataset. They try to fit the data distribution to the feature space in order to group the similar items and preserve locality. Notable examples of data-dependent hashing methods are: Spectral Hashing [67], which is based on spectral graph partitioning; K-means based hashing [68], which uses K-means clustering in the generation process of the binary codes; Subspaces Product Quantization [69], which decomposes the feature space into a Cartesian product of low-dimensional subspaces, each subspace is quantized separately, and the asymmetric distances are computed between the query and the quantized codes with the help of lookup tables; Kernelized Locality-Sensitive Hashing [70], which generalizes hashing to any Mercer kernel; Semi-Supervised Hashing [71], which exploits label information of the training set; Multiple Feature Hashing [90], which combines multiple features (i.e., global feature HSV color histogram and local visual features LBP) of videos, in order to learn the hash codes of the training keyframes and a series of hash functions in a joint framework; Iterative Quantization [91], which minimizes quantization error by rotating zero-centred PCA projected data; Joint Optimization [72], which jointly optimizes both search accuracy and search time using compact binary codes; Random Maximum Margin Hashing [73], which constructs hash functions by using large margin classifiers with arbitrarily sampled data points that are randomly separated into two sets.

Efficiency improvements of data-dependent methods over independent ones have been shown in several studies [69], [67], on condition that limited hash code sizes are considered. This happens because by increasing the number of hash functions there is a lack of independence between them. For example, Spectral Hashing [67][58] outperforms many data-independent methods for small code sizes, but it is outperformed by the data-independent method of Shift-Invariant Kernel Hashing [58] for sizes over 64 bits. Moreover, in all data-dependent hashing methods there is often a significant pre-processing cost for learning the selected training dataset and for generating the binary codes.

In all aforementioned hashing methods, the most common technique for assigning the binary codes is to partition the metric space of the projected data points with appropriate hyperplanes and set two different codes for each side. In a very recent approach, Spherical Hashing [74], hyperspheres (instead of hyperplanes) are used, so as to partition the data
points and to compute the binary codes. In the experimental evaluation of [74], authors showed that Spherical Hashing outperforms other state-of-the-art hashing methods.

![Figure 18: spherical hashing approach selects hyperspheres instead of hyperplanes to partition the space to binary codes.](image)

4.4 The Multisort Indexing algorithm

The Multisort Indexing method (MSIDX) is a novel approximate indexing scheme for efficient content-based image search and retrieval [92]. The proposed scheme analyses high dimensional image descriptor vectors, by employing the value cardinalities of their dimensions (DVC). Since dimensions with high value cardinalities have more discriminative power, a multiple sort algorithm is used to reorder the descriptors’ dimensions according to their value cardinalities, in order to increase the probability of two similar images to lie within a close constant range. The expected bounds of the constant range are defined in detail, following deterministic and probabilistic analyses in [92].

MSIDX’s approach requires minimum pre-processing of the datasets and thus minimum computational costs. The pre-processing algorithm (in Figure 19) computes a priority list for sorting the dimensions according to their value cardinalities. Then the insertion algorithm, presented in Figure 20 is used to insert any new vectors in the dataset, using the comparison function of Figure 21.
State-of-the-art analysis on tools for multimedia indexing and search

### Algorithm Preprocessing-MSIDX(S)

Input: $S$ = the set of $N$ $D$-dimensional descriptor vectors
Output: $S'$ = the multiple-sorted set

1. a. **compute** value cardinalities $c_i$ of the dimensions
2. b. $i = 1, \ldots, D$ of dataset $S$.
3. a. **sort** the value cardinalities $c_i$ descending.
4. b. create a priority index $p_i$ on dimensions, $i = 1, \ldots, D$.
5. c. based on the sorted value cardinalities.
6. a. [optional] **compute** $c_x$, for each image $o_j \in S$.
7. b. $j = 1, \ldots, N$, store it in an extra dimension $D + 1$.
8. a. [optional] **define** a specific priority index $p_{i_{x}}$.
9. b. with $k = D + 1$ for the $c_x$ values.
10. a. **multiple sort** the $N$ images, using the $p_i$ order.
11. b. in all dimensions.
12. a. **update** the positions in $L$ and.
13. b. **return** the multiple-sorted dataset $S'$.

---

### Algorithm Insert-MSIDX(image $o_{new})$

Input: $o_{new}$ the new image in the form of $D$-dimensional descriptor vector
Output: the updated dataset $S''$

1. a. [optional] **compute** the value of $c_x$ of the optional extra dimension for image $o_{new}$.
2. b. **compare** function in Figure 2.
3. a. allocate the storage position $pos_{new}$ using the same.
4. b. **compare** function in Figure 2.
5. a. insert the image $o_{new}$ into the allocated position $pos_{new}$.
6. b. and **update** the positions in $L$.
7. a. **return** the updated dataset $S''$.

---

**Figure 19:** MSIDX Pre-processing algorithm.  **Figure 20:** MSIDX insertion algorithm.

**Figure 21:** The compare function for multiple sort.  **Figure 22:** The query processing algorithm.

In the query time, MSIDX follows the algorithm of **Figure 22** to position the query inside the dataset using the same sorting strategy. Next, descriptors in a constant range above and below the positioned query are compared to prepare the ranked list. This constant range is determined by the parameter $W$ which represents the actual number of descriptor vectors that should be checked to compute their distance from the query vector.
Figure 23 presents the influence of parameter $W$ in the performance of the indexer compared with the exhaustive search for three different datasets of size 240K images, while Figure 24 presents the same experiment with 2 datasets of 1 million images.

Finally, Figure 25 presents a set of experiments comparing the performance of MSIDX with state of the art hashing methods. The red vertical lines in the plots represent the exhaustive search time with mAP 100%. As it is shown in the figure, MSIDX outperforms all other hashing methods of the experiments by giving lower search times with equal or better mAP performance.
Figure 25: (a-e) Retrieval accuracy versus search time for the 5 datasets, (f) number of bits compared with the W parameter that gave similar search times.

The promising results of this work required further investigation of the DVC characteristics to verify that this approach can be widely used. Towards this goal, a new study for the impact of DVC on multimedia similarity search was performed.
4.5 The influence of image descriptors DVC to large-scale similarity search

In this work we evaluated the impact of DVC on the performance of large scale similarity search [93]. To do so we performed a Canonical Correlation Analysis to verify that the DVC characteristics of a dataset correlate with the performance of indexing methods in terms of mAP and Speedup factor. For our analysis we used 6 publicly available datasets of image descriptors with different dimensionality and sizes to support the generality of our approach. Figure 26 depicts the setup of the analysis we performed and Figure 27 the results of the CCA analysis which clearly demonstrate the correlation of DVC and the performance of the methods in terms of mAP.

![Canonical Correlation Analysis](image1)

Figure 26: Canonical Correlation Analysis.

<table>
<thead>
<tr>
<th></th>
<th>LSH</th>
<th>SKLSH</th>
<th>PCA-ITQ</th>
<th>SPH</th>
<th>MSIDX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wilk’s Λ</td>
<td>0.121 (p &lt; 0.001)</td>
<td>0.018 (p &lt; 0.001)</td>
<td>0.187 (p &lt; 0.001)</td>
<td>0.822 (p &lt; 0.004)</td>
<td>0.525 (p &lt; 0.001)</td>
</tr>
<tr>
<td>Can. Load. rDVC,</td>
<td>0.475</td>
<td>0.431</td>
<td>0.558</td>
<td>0.656</td>
<td>0.440</td>
</tr>
<tr>
<td>rDVC,</td>
<td>0.306</td>
<td>0.502</td>
<td>0.196</td>
<td>0.867</td>
<td>0.595</td>
</tr>
<tr>
<td>rDVC,</td>
<td>0.061</td>
<td>0.036</td>
<td>0.167</td>
<td>0.965</td>
<td>0.732</td>
</tr>
<tr>
<td>Can. Load. rPerf,</td>
<td>0.445</td>
<td>0.427</td>
<td>0.503</td>
<td>0.239</td>
<td>0.303</td>
</tr>
<tr>
<td>rPerf,</td>
<td>0.286</td>
<td>0.497</td>
<td>0.177</td>
<td>0.316</td>
<td>0.410</td>
</tr>
<tr>
<td>rPerf,</td>
<td>0.057</td>
<td>0.036</td>
<td>0.150</td>
<td>0.351</td>
<td>0.504</td>
</tr>
<tr>
<td>Can. Cross-Load. rDVC,</td>
<td>-0.074</td>
<td>-0.014</td>
<td>-0.072</td>
<td>-0.182</td>
<td>0.018</td>
</tr>
<tr>
<td>rDVC,</td>
<td>0.973</td>
<td>0.996</td>
<td>0.946</td>
<td>0.894</td>
<td>-0.078</td>
</tr>
<tr>
<td>rPerf,</td>
<td>-0.074</td>
<td>-0.014</td>
<td>-0.072</td>
<td>-0.182</td>
<td>0.018</td>
</tr>
<tr>
<td>Can. Cross-Load. rPerf,</td>
<td>0.911</td>
<td>0.987</td>
<td>0.852</td>
<td>0.325</td>
<td>-0.476</td>
</tr>
<tr>
<td>rDVC,</td>
<td>-0.069</td>
<td>-0.014</td>
<td>-0.065</td>
<td>-0.066</td>
<td>0.0120</td>
</tr>
<tr>
<td>Can. Correlation Coefficient</td>
<td>0.936</td>
<td>0.991</td>
<td>0.900</td>
<td>0.364</td>
<td>0.688</td>
</tr>
</tbody>
</table>

Figure 27: Experimental results of CCA for evaluating the performance of LSH, SKLSH, PCA-ITQ, SPH and MSIDX on the six evaluation datasets.

Moreover, in order to ensure that our analysis and in general the usage of DVC is robust in terms of scaling we computed the empirical cumulative distribution function of DVC for every dataset to ensure that the produced DVC distributions do not change with different dataset sizes (Figure 28).

As it is shown in Figure 29 and Figure 30, the study confirmed that the high DVC dimensions have high discriminative power and thus extra effort in encoding such information is required to ensure better performance of indexing methods.
Figure 28: Empirical cumulative distribution function of $F(x)$ with $x = \text{DVC}$, for different sizes ($\%N$) of the evaluation datasets.

The descriptor extraction techniques tend to produce similar DVC distributions from the same distribution family, and thus similarity search strategies can exploit image descriptors’ DVC, irrespective of the N datasets’ sizes.

Figure 29: Performance of sequential search of descriptors by eliminating a percentage of the low-DVC dimensions.

The results are grouped based on the datasets’ dimensionality in (a) low and (b) high dimensional.
Aiming to formulate an analytical study of the DVC elimination, we followed the concept of Orthogonal Centroid Feature Selection algorithm [94][95], used on text categorization applications. A score function for each dimension (i.e. feature) is computed according to:

\[ S(w_l) = \lambda \sum_{j=1}^{k} \frac{n_j}{n} (w_j^T (m_j - m))^2 + (1 - \lambda) \sum_{i=1}^{n} (w_i^T (x_i - m))^2 \]

which combines objective functions of supervised, semi-supervised and unsupervised methods in a unified framework, with \( w \) the element of the projection matrix that selects dimension \( l \), \( n \) the number of records, \( m \) the mean of each dimension and \( n_j \) and \( m_j \) the number of records and the mean of each class in the supervised methods. Parameter \( \lambda \) takes values from \{0; 1; 2\} for the unsupervised, supervised or semi-supervised approach respectively. In our analysis we followed an unsupervised approach and thus for \( \lambda = 0 \), the function is transformed to:

\[ S(w_l) = \sum_{i=1}^{n} (w_i^T (x_i - m))^2 \]

which gives the (scaled) variance of each variable \( l \), i.e. each dimension, of the dataset \( X \in \mathbb{R}^n \). Recall here that for a positive semidefinite matrix \( A \in \mathbb{R}^{d} \), the energy of the matrix may be defined by the summation of its eigenvalues which equals also to the trace of the matrix \( A \) as in:

\[ E_{total} = \sum_{i=1}^{d} \lambda_i = \text{tr}(A) \]

According to the above equation, the energy of the covariance of the full dataset \( X \) is equal to summation of the scores of \( S(w_l) \) for all dimensions. Finally, based on [94][95], the energy function after the selection of \( p \) dimensions (features), sorted by their scores in descending order, is defined as:
Given the above formulations, to evaluate our statement that low-DVC dimensions hold much less information than high-DVC or randomly selected ones, we computed the energy function of each dataset by eliminating the dimensions from 5% to 50% with a step of 5% as in our previous experiments. However, the sorting of scores followed the DVC sorting of each experiment. The results, presented in Figure 31, make clear that, by eliminating only low-DVC dimensions, the preserved energy is much higher compared to eliminating randomly selected or high-DVC dimensions.

The results of Figure 31 explain the outcomes of our previous experiments, since the higher the energy is the more information is preserved. An interesting observation is also that the curves for the randomly eliminated dimensions fall always in-between the low-DVC and high-DVC curves, with the high-DVC elimination curves to preserve the less possible energy. One may interpret this as follows: by removing the high-DVC dimensions from the dataset, the most informative features (dimensions) are lost.

Figure 31: Energy preservation (as a percentage of the total energy) after eliminating a percentage of low, high or randomly selected DVC dimensions.

4.6 Applicability to LASIE framework

In Figure 32, a diagram of LASIE architecture is depicted, which has been specifically designed for Use Case 1: Video Analysis for Offender identification. This use case involves mainly processing of visual data acquired from surveillance cameras, aiming at the identification of a suspect in one or more videos. Several processing modules (WP4) are used to extract useful information for automatic (or semi-automatic) analysis of the visual content, namely EXIF extraction, plate detection, tattoo recognition, face detection, DROP tracking, people tracking, event detection, logo detection, biometrics processing, abandoned baggage detection, vehicle recognition and tracking, video summarisation. Applying these modules on the hundreds or thousands of hours of video content acquired from surveillance cameras results in a vast amount of extracted features, which is impossible to exploit for search and retrieval without an appropriate indexing scheme.
The role of image and video indexing modules is to organise and store the extracted features in a more efficient way. Search and retrieval will be then realised by querying directly the indexing scheme using query by example or keyword-based queries.

Within LASIE, our expertise in image indexing (MSIDX and DVC influence studies) will be used as a starting point for further investigations. More specifically, we will focus on extending our work with hashing methods and bitmap indexing methods to be able to work with even larger datasets. By introducing a hybrid approach on hashing methods and MSIDX we aim to exploit the efficiency of the hashing methods in terms of storage space with the efficiency of MSIDX in terms of mean average precision. Finally, we will give effort in the extension of our components in large computing clusters with efficient query rooting and filtering approaches.

Figure 32: The LASIE architecture for Use Case 1 (Video Analysis for Offender identification).
5 Technologies for Video Indexing and Search

Video indexing and search are very popular fields of research for a variety of researchers from different fields, due to the multimodal nature of the content. Video combines visual and audio information but goes beyond these two modalities due to the existence of time dimension. As such, there are different techniques and approaches to index and search for videos and this has to do with the level of detail and the depth of search one needs to perform. Finally, the indexing approach that is followed largely depends on the scenarios and use cases one should support within each system.

In this state of the art report, we discuss the basic approaches of video indexing mainly in terms of the visual characteristics since these are the dominant ones. Of course the combination of such methods with the already described audio ones, may support extra scenarios or improve the performance of the algorithms.

There are two basic categories of methods for indexing videos. The one is by indexing lower level features for each video, in order to support query by example functionalities. The second one is by higher level indexing methods (i.e. event based indexing, concept based indexing etc.) that index higher level concepts, meanings or structures for each video. For the low level features indexing, there is also another level of discrimination. That is, a) each video is considered as a set of frames. Thus, techniques that apply in image indexing may be also used in video indexing in almost the same way. b) Techniques that consider the temporal dimension of the video content and generate index points for video segments or other structures. In this section we will focus on the techniques for higher level event and concept indexing, since the frame based techniques are covered by the previous section [96].

Moreover, these higher level methods generate typically, more compact metadata than the lower level features. As such, after the initial processing phase, these may be indexed in common structures such as SQL or NoSQL databases along with their timestamps and other meta information to construct typical queries.

5.1 Shot Boundary Detection

A shot is a consecutive sequence of frames captured by one camera from start to stop and typically there is a large correlation between frames in one shot. Shot boundary detection aims to find these video segments [96]. Several survey papers exist on shot boundary detection such as [98][99]. The boundaries are classified as either “cut” for abrupt change, or gradual transitions such as fade in/out, dissolve etc. Cut detection is easier to detect than gradual transitions. The overall approach of shot boundary detection algorithms is to extract features for each frame, find similarities and identify the change in distances to detect a boundary.

The threshold based approaches compare such pairwise similarities to define or learn a threshold for deciding whether there is a shot boundary or not [100][101]. Thresholds may be global or adaptive (or combined). Then, there are also supervised statistical learning techniques that are based on SVM [102][103][104], adaboost [105][106] or other classifiers (kNN or HMM classifiers etc.) [107]. Finally, unsupervised techniques also exist. The most popular of them, are using clustering approaches to group frames and create shot boundaries [108][109][110].
5.2 **Key-frame extraction**

As it is obvious there is great redundancy among the frames of the same shot. As such, it is useful to select only a small set of frames that best represent the video. In the key frame extraction process there is a wide set of features used such as colour histograms, edge detectors, shape, optical flow and other motion vectors as well as features derived from the image variations due to camera motion.

5.3 **Scene segmentation**

The scene segmentation is creating story units from the continuous shots that are coherent to a theme. Scenes contain higher level semantics and follow different approaches to select scenes. These use either background information (e.g. the background is not changing) or key frame and audio-visual information to check if the theme/subject/concept is the same.

5.4 **Video summarization**

Video summary has been defined in as "a sequence of still or moving images presenting the content of a video in a way that the respective target group is rapidly provided with concise information about the content while the essential message of the original is preserved”. A number of methods and techniques have been proposed for the automatic extraction of video summaries; however, no robust answer has been given to overcome all challenges. Two are the main aspects of video summarization: the selection of frames that can represent the content of the video and the visualization of those frames in a user intuitive manner. Representative frames, also known as key-frames, are extracted from the video source. A common approach for key-frame selection is clustering. Color histograms or color features are extracted from the video frames; then Delaunay triangulation, modified hierarchical clustering, or line Gaussian Mixture Model clustering are applied to produce clusters. The cluster centroids are selected as keyframes. The authors in present a summarization technique based on robust low-rank subspace segmentation. A series of video frames subspaces are segmented based on the Normalized Cuts algorithm and the key frames are chosen from the significant subspaces.

5.5 **Video hashing approaches**

An interesting set of research papers work on hashing methods for video sequences. As in the sections of audio and image that we already discussed on hashing methods, video hashing methods became popular due to their low storage and low computational cost requirements.

In the authors propose a hashing method that respects the temporal consistency while taking into account the discriminative commonalities between video frames to build robust binary codes. On the other hand, the authors in combine hash codes to build a semantically rich hash of each video. Their proposal is to combine different video components and generate temporal hashes from them.
Figure 33: An illustration of the hash method of [125] within the event category “feeding an animal”. Discriminative local commonality is automatically discovered, e.g., the eyes of animals, the edges of tubs, and the parts of human hands. Temporal consistency is preserved, and successive frames are grouped and put into the same hash bucket. (figure taken from [125]).

Figure 34: Temporal pooling for video hashing (figure taken from [126]).
5.6 Applicability to LASIE framework

The interaction of the video indexing module with the other modules of the LASIE architecture is depicted in Figure 32. As it was explained at the beginning of Section 5, video indexing will focus on higher-level event and concept indexing, as opposed to low-level (image) indexing of individual frames.

Since Use Case 1 of LASIE deals with very large volumes of video content, we need to take into account the temporal information to boost performance, while keeping computational complexity low. Towards this direction, a potential approach would be to combine semi-supervised hashing methods to support both low-level and higher level indexing and querying functionalities. This is a challenging research task and it is planned for the second year of the LASIE project.
6 Conclusions

In this document, a survey on existing technologies for large-scale multimedia indexing and search is presented. State-of-the-art analysis has been performed separately for each content type, i.e. text, audio, image, video, since each of the above types has different requirements in terms of processing and indexing. Instead of analysing each method in detail and producing a lengthy document, the main categories of indexing techniques for each content type (text, audio, image, video) have been presented and the most representative works for each category have been included in the study.

The main objective of the work in Task 6.1, for the first year of the LASIE project, is to identify the most appropriate technologies in multimedia indexing that fulfil the requirements of the LASIE framework. Thus, each survey is followed by a subsection that investigates the applicability of methods to LASIE framework. Four modules will be the outcome of Task 6.1:

- Text Indexing module
- Audio Indexing module
- Image Indexing module
- Video Indexing module

A brief description of the related LASIE Use Case is given for each module. Then, the role of the module in LASIE architecture and its interaction with the other LASIE modules is described. The functionality of the module is explained and initial suggestions for improvement of existing tools are made. The latter will provide the research directions and will drive the work within the second year of the LASIE project. It is worth mentioning that significant research towards this direction has been already done by the LASIE partners, which will be used as starting point and improved within the context of LASIE.
7 References


D.1 State-of-the-art analysis on tools for multimedia indexing and search


[81] T. Chiuheh, “Content-based image indexing”, in Proc. of Int. Conf. on Very Large Databases (VLDB), pp. 582-593, 1994.


D6.1 State-of-the-art analysis on tools for multimedia indexing and search


[124] Zhengzheng Tu, Dengdi Sun, and Bin Luo. Video summarization by robust low-rank subspace segmentation. In Zhixiang Yin, Linqiang Pan, and Xianwen Fang, editors, Proceedings of The ighth International Conference on Bio-Inspired Computing:
Theories and Applications (BIC-TA), 2013, volume 212 of Advances in Intelligent Systems and Computing, pages 929–


