



D3.1: 1st Environmental node discovery, indexing and data acquisition

WP3 – Collective sensing models and tools



D3.1: Environmental node discovery, indexing and data acquisition

Document Information

Grant Agreement Number	688363	Acronym	hackAIR
Full Title	Collective awareness platform for outdoor air pollution		
Start Date	1 st January 2016	Duration	36 months
Project URL	www.hackAIR.eu		
Deliverable	D 3.1 - Environmental node discovery, indexing and data acquisition		
Work Package	WP 3 - Collective sensing models and tools		
Date of Delivery	Contractual	1 st October 2016	Actual 7 th October 2016
Nature	Report	Dissemination Level	Public
Lead Beneficiary	CERTH		
Responsible Author	Anastasia Moumtzidou (CERTH)		
Contributions from	Symeon Papadopoulos (CERTH), Stefanos Vrochidis (CERTH), Yiannis Kompatsiaris (CERTH), Christodoulos Keratidis (DRAXIS)		

Document History

Version	Issue Date	Stage	Description	Contributor
0.1	02/09/2016	Draft	Document structure	A. Moumtzidou, S.Vrochidis, S. Papadopoulos (CERTH)
0.5	25/09/2016	Draft	Integrated document	A. Moumtzidou (CERTH)
0.6	26/09/2015	Draft	Review of CERTH contribution	S.Vrochidis, S. Papadopoulos, Y. Kompatsiaris (CERTH)
1.0	28/09/2016	Draft	Internal Review of the whole document	Christodoulos Keratidis (DRAXIS)
1.1	01/10/2016	Pre-Final	Final version	A. Moumtzidou (CERTH), S. Papadopoulos (CERTH), S. Vrochidis (CERTH)
1.2	07/10/2016	Final	Approved by coordinator for submission	A. Moumtzidou (CERTH), S. Papadopoulos (CERTH), S. Vrochidis (CERTH)

Disclaimer

Any dissemination of results reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

Copyright message

© hackAIR Consortium, 2016

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both. Reproduction is authorised provided the source is acknowledged.

Table of Contents

1 Executive summary.....	7
2 Introduction.....	9
3 Empirical Studies	11
3.1 Empirical Study Methodology.....	12
3.2 Empirical Study of Environmental Websites.....	13
3.2.1 Dataset.....	13
3.2.2 Methodology for the Empirical Study of Environmental Websites.....	14
3.2.2.1 First phase	14
3.2.2.2 Second phase.....	15
3.2.3 Analysis of Environmental Websites.....	15
3.2.3.1 Air Pollution in the World - aqicn.org Website	15
3.3 Empirical Study of Web services.....	16
3.3.1 Introduction to Web Services	17
3.3.2 Dataset.....	17
3.3.3 Analysis of Web services.....	18
3.3.4 Findings.....	19
3.4 Empirical Study of Social Media Platforms	20
3.4.1 Social Media Platforms	20
3.4.1.1 Flickr vs Instagram	22
3.4.2 Methodology for the Empirical Study of Flickr Social Media Platform	25
3.4.2.1 First phase	25
3.4.3 Collection of Flickr images for a specific city	25
3.5 Empirical Study of Webcams	26
3.5.1 Dataset.....	26
3.5.1.1 Webcam sites and portals dataset.....	26
3.5.1.2 AMOS dataset.....	27
3.5.2 Methodology for the Empirical Study of Webcams	27
3.5.2.1 First phase	28
3.5.2.2 Second phase.....	28
3.5.3 Analysis of Webcams	28
3.5.3.1 Webcam in Hannover	28
3.5.3.2 AMOS dataset.....	29
4 Data Gathering	31
4.1 Discovery of environmental web sites.....	31
4.1.1 State of the art for domain-specific web search.....	31

D3.1: Environmental node discovery, indexing and data acquisition

4.1.2 Framework.....	32
4.1.3 Evaluation	33
4.1.4 Results	35
4.2 Collecting Images from Social Media Platforms.....	36
4.2.1 Flickr collector.....	36
4.2.2 Image Downloader	38
4.3 Collecting Images from the AMOS dataset	39
5 Data Analysis.....	40
5.1 Text-based Sources.....	40
5.1.1 Web services.....	40
5.1.1.1 Web service example.....	40
5.1.2 Web data extraction	41
5.1.2.1 easIE: easy Information Extraction framework.....	41
5.2 Image-based Sources.....	46
5.2.1 First method: Visual concept detection and localization	46
5.2.1.1 Concept detection state of the art	46
5.2.1.2 Sky localization.....	48
5.2.2 Second Method: Image processing heuristics for sky localization	50
5.2.2.1 Framework.....	50
5.2.2.2 Evaluation	52
5.2.3 Comparison of sky localization methods	53
6 Data Storage and Indexing.....	55
6.1 Indexing Profile Records	55
6.2 Node Repository	55
6.2.1 SOS server.....	55
6.2.2 PostgreSQL/PostGIS database	56
6.2.3 MongoDB.....	56
6.2.4 Comparison of SOS, PostgreSQL/ PostGIS and Mongo	57
6.3 Indexing techniques at the Repository	57
6.4 Repository Population	58
7 Conclusions.....	60
8 References.....	62
9 Appendix.....	68
9.1 Empirical Study of Web sites	68
9.1.1 Brief Description of Dataset.....	68
9.1.2 Detailed Study.....	70
9.2 Empirical Study of Web services.....	87

D3.1: Environmental node discovery, indexing and data acquisition

9.2.1 Brief Description of Dataset.....	87
9.2.2 Detailed Study.....	88
9.3 Empirical Study of Webcams	90
9.3.1 Sites and portals dataset.....	90
9.3.2 AMOS dataset.....	93
9.4 Configuration file examples	95

Table of Figures

Figure 1- Overview of architecture for environmental information collection, analysis and indexing.	10
Figure 2- Empirical cycle (Wikipedia: Empirical Research, 2010).....	12
Figure 3- URL and screenshot of the web page.	16
Figure 4 - The Web services technology stack.....	17
Figure 5 – JSON response for the city of Amsterdam (broken in two columns for the sake of space).	19
Figure 6 - KPCB Internet Trends Report 2016.....	21
Figure 7 – Instagram Statistics	21
Figure 8 – Chart showing number of images uploaded to Flickr per month – Report 2004-2016.....	22
Figure 9 – Total number of images and images containing sky for Flickr during 1/1/2016-31/5/2016.....	24
Figure 10 – Total number of images and images containing sky for Instagram during 1/1/2016-31/5/2016	24
Figure 11- URL and screenshot of the web page.	29
Figure 12 – Web page from the AMOS dataset pointing to a specific webcam.....	30
Figure 13 - Focused crawling based on the combination of multimedia evidence.....	33
Figure 14 - Precision (top) and F1-measure (bottom) of the focused crawl for the $fS'h$ link selection method at depth $\in 1,2,3$ for threshold $t1 \in 0.1, \dots, 0.9$ when strict relevance assessments (left) and lenient relevance assessments (right) are employed.....	34
Figure 15 - Precision of the f_E' (h), f_E'' (h), and f_E''' (h) exploration strategies by a focused crawler that employs the f_s' (h) text-based link selection method at depth = 3 for thresholds $t_1, t_2 \in \{0.1, \dots, 0.9\}$, s.t. $t_2 < t_1$, when lenient relevance assessments are employed.....	35
Figure 16 - Web extraction tool architecture	42
Figure 17 – Configuration file for extracting information from the ‘www.umweltbundesamt.de’ web page.	44
Figure 18 - Screenshot of http://aqicn.org web site for the city of Brussels.	45
Figure 19 – Configuration file of easIE tool for http://aqicn.org web site	45
Figure 20 – Output of easIE tool for the http://aqicn.org web site as CSV file	45
Figure 21 – DCNN-based features	47
Figure 22 – Semantic Segmentation using FCN	49
Figure 23 – Sky localization visual examples using FCN on images retrieved by the Flickr API.....	49
Figure 24 - Sky localization visual examples using FCN on images from SUN Database	50
Figure 25 - 8-neighbors of pixel p	51
Figure 26 – Flowchart of sky detection algorithm using the method proposed by the experts.	52
Figure 27 - Sky localization visual examples using the heuristics method on images from the SUN Database.	53
Figure 28 – Population of MongoDB with data coming from Flickr.....	59

Table of Tables

Table 1 – Cities studies through hackAIR.....	11
---	----



D3.1: Environmental node discovery, indexing and data acquisition

Table 2- Distribution of services offered.....	13
Table 3- Distribution of coverage area	13
Table 4- Distribution of temporal type of measurements	14
Table 5- Distribution of coverage area	18
Table 6- Distribution of temporal type of measurements	18
Table 7 – Total number of images and images containing sky for Instagram and Flickr during 1/1/2016-31/5/2016. .	22
Table 8 – Average number of images per day for five example cities.	23
Table 9 –Number of images posted in Flickr per month for the months January, February, March.	25
Table 10 –Number of images posted in Flickr per month for the months April, May, June.	25
Table 11- Distribution of coverage area	26
Table 12- Distribution of format type	26
Table 13- Number of webcams per country	27
Table 14- Information for cities with cameras.....	27
Table 15 – Mean number of cameras.....	27
Table 16 – List of seed URLs	33
Table 17 – URL Seed list for retrieving sites with air quality measurements	35
Table 18 – Web pages discovered by the focused crawler and their classification scores.	36
Table 20 – Comparison of sky localization methods.....	53
Table 21 – Advantages and disadvantages of SOS server.	56
Table 22 – Advantages and disadvantages of PostgreSQL/ PostGIS server.	56
Table 23– Advantages and disadvantages of Mongo server.....	57
Table 24 – MongoDB fields, their type and description.....	59

1 Executive summary

This document reports on the discovery, acquisition, processing and indexing of environmental data coming from the following sources:

- text-based web official sources including lists of web services and web sites, and,
- image-based sources that comprise: a) user-generated images including publicly available images posted through social media platforms (e.g. Flickr), b) images captured by the users of the hackAIR mobile app, and c) images from webcams.

The aim of having several different sources is that they can function in a complementary way. Text-based sources are the most accurate but they are limited in number, on the other hand, images coming from social media platforms offer the advantage of abundance and high coverage, while the images captured by the users of the hackAIR mobile app are expected to be of much higher quality and consistency since their capturing parameters will be controlled by the hackAIR app. Finally, webcam images offer the advantage of a fixed location and regular updates, but they have usually very low resolution which might affect the air quality (i.e. PM) estimation.

The methods applied for discovering, retrieving and processing environmental data (i.e. PM measurements) differ depending on the source of content. However, storing and indexing are handled in a unified way. In order to discover and collect efficiently the data from the aforementioned sources, it is necessary to study and further understand how the information in these sources is structured and encoded. The results of the study will determine the techniques that should be used for discovering, collecting and retrieving the information of interest (i.e. the PM measurements, the location and date/time they refer to) from each source. Thus, separate studies for each source that is targeted are conducted in order to draw conclusions for each one. In particular, as far as environmental sites are concerned, we employ the well-established method of the empirical cycle (Groot, 1961), and we study in detail each website and report, the parts of which contain important information which should be retrieved. To perform the aforementioned empirical study, we select a representative and balanced (in terms of areas, information provision, etc.) dataset of environmental sites from the Web. The findings of the study show that the important information to be considered when dealing with an environmental node concerns the following: 1) type of environmental measurement, 2) location, and 3) temporal information. An in-depth analysis reveals that such information can be retrieved from the web content and the URL address of such websites. Regarding the study on environmental services, and given that the number of publicly available services providing environmental information is very limited, we focus on presenting the services provided by the air quality experts. Furthermore, as far as social media platforms are concerned, we study the possible platforms that could be used for retrieving images provided by users and after comparing their specifications and their APIs, we deduce that the most appropriate candidate is Flickr. Finally, we study a significant number of webcams that are located in the area of interest of hackAIR, and draw several conclusions regarding the sites, and how webcams are embedded into web pages. We should note that the study of webcams is further divided into two datasets, one consisting of several sites returned by general purpose search engines and a second stemming from the AMOS dataset, which contains webcams manually or automatically discovered that are dispersed around the world (but with more focus on USA).

Based on the results of the aforementioned studies and the sources studied, we analyse a set of techniques that are used for retrieving data from each source. Thus, for the case of environmental sites, we present a domain-specific search technique that targets such nodes using a focused crawler, and we use it for automatically discovering a set of URLs. As far as Flickr is concerned, we present the Flickr API, its capabilities and limitations, along with examples of queries that can be submitted to retrieve images that cover the region of interest and specific time period. Finally as far as webcams are concerned, a framework for automatically discovering sites with webcams will be developed and presented in the next deliverable (D3.2).

In the sequel, and depending on the type of source (i.e. images, web page, web service), we present data analysis techniques for extracting environmental information. Specifically, image processing techniques are foreseen for the recognition of the sky part of the images (either coming from Flickr, webcams or the mobile app), and the estimation

D3.1: Environmental node discovery, indexing and data acquisition

of its R/G (Red to Green) and G/B (Green to Blue) ratios. Specifically, as far as the sky localization is concerned, two methods are studied; the first is based on Fully Convolutional Networks (FCN) and the second on an algorithm proposed by experts that is based on simple image processing heuristics. After evaluating the two methods, we deduce that the one based on FCN is more accurate; however there is room for improvement for both methods. The extracted R/G and G/B ratios are meant to be provided as input to the AQ Estimation service and to produce air quality (i.e. PM) estimates for the area depicted in the image (yet this is a part to be discussed in D3.3). Regarding the web sites, web information extraction techniques are required to extract the target information. Thus, a versatile web information extraction framework called easIE was used after appropriate adaptations to handle sites that contain environmental information. Finally, as far as web services are concerned, specific JSON/XML parsers were developed for handling the data provided by each service separately.

Last but not least, we discuss several options for storing the retrieved data, along with the type of information that needs to be stored as well as the strengths and weaknesses of each option. The employed repository must offer indexes and query mechanisms to handle textual, numerical and geospatial information. After careful consideration of each option, we decided that MongoDB is the most appropriate solution as a repository for hackAIR given the variety and scale of data that need to be stored.

2 Introduction

One of the main objectives of hackAIR is to develop collective sensing approaches for measuring air quality. The considered approaches include the following: a) collecting **measurements from existing air quality stations and open data** on the Web, which includes environmental related web pages and services, b) collecting and analyzing **sky-depicting images** including publicly available geo-tagged and time-stamped images posted through social media platforms (e.g. Flickr), images captured by the users of the hackAIR mobile app, and webcams, and c) **crowdsourcing measurements via low-cost open hardware devices**. The collection of air quality data from the first two sources are discussed in the current deliverable.

To achieve this objective, it is necessary to study and further understand how the information in the aforementioned sources is structured and encoded. The results of our studies determine the techniques that should be used for discovering, collecting and retrieving the information of interest from each source. To this end, dedicated studies for each target source have been conducted in order to draw conclusions. Those include a study on environmental sites using the method of the empirical cycle (Groot, 1961), a study of environmental web services, a study of social media platforms, and a study of sites/ portals containing webcams.

In the sequel, we present the techniques used for retrieving data from each source, by considering both the results of the aforementioned studies and the type of information source. As far as environmental sites are concerned, we present a domain-specific search technique that targets such nodes, while for web service an automatic discovery technique is not required given that their number is limited. Finally, as far as social media platforms are concerned, their APIs, parameters and the queries that can be submitted for retrieving relevant images are studied.

Then, we discuss the techniques for data analysis which vary according to the type of source (i.e. images, web pages, web services). Specifically, we employ **image processing techniques for the recognition of the sky-depicting regions** of the collected images, **semi-automatic information extraction techniques** for the extraction of information from web sites, and finally **JSON/XML parsers** specifically developed for each service for the extraction of data from web services.

Next, we present the alternatives that may be used for storing the retrieved data from all the sources mentioned, the type of information that needs to be stored as well the advantages and disadvantages of each option. Based on those, we decide on the best solution for hackAIR. Eventually, we present an initial design of the hackAIR repository.

Figure 1 depicts a conceptual view of the data collection, analysis and indexing architecture. This deliverable discusses all the modules presented in the figure apart from the “Domain-specific discovery module” that focuses on the discovery of webcams, which is discussed in deliverable 3.2, the “Mobile APP” which is discussed in deliverable 5.2, the “Air Quality Estimation Service” that is discussed in deliverable 3.3 and finally the “Hardware-based sources” that is discussed in deliverable 3.5.

The diversity of sources results in multimodal input data that comprise images, unstructured and structured text and numeric values. Depending on the type of data, different analysis procedures are foreseen. Specifically, images (coming from social media, the mobile app, and webcams) will be processed using image analysis techniques and image-based air quality (i.e. PM) estimation that provides an air quality index (e.g. low, high). In the case of websites, the data is provided in unstructured format and thus information extraction techniques are required to extract the target information. In the case of web services, the data is provided in structured format and thus no complex text processing is required. Finally, in the case of hardware sensors, the sources provide numeric values. Eventually, all collected data are stored into a central repository. These data can either be served through a REST web service to the end user as they are or sent to the Fusion Service that will combine all the measurements and create air quality maps.

D3.1: Environmental node discovery, indexing and data acquisition

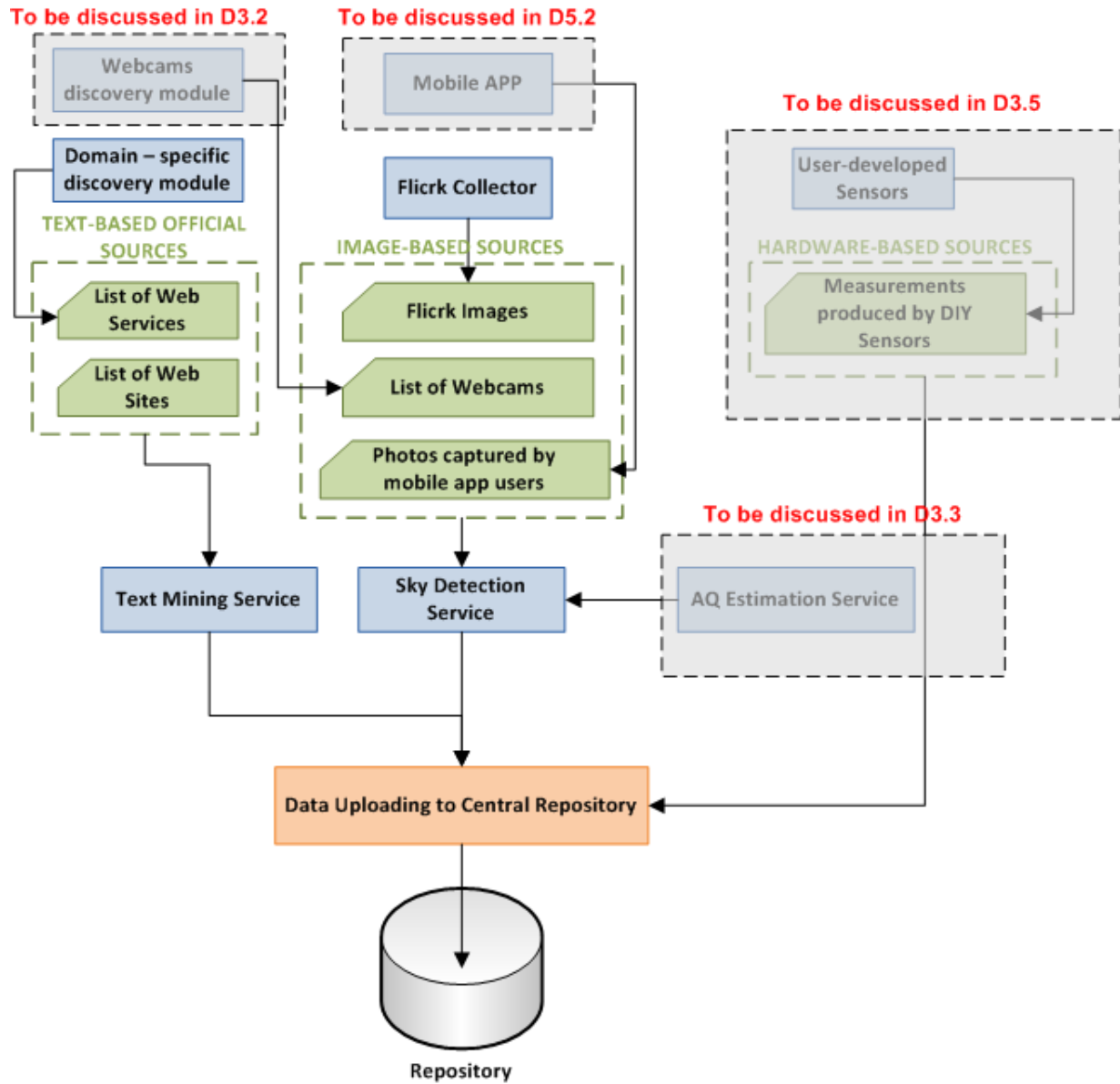


Figure 1- Overview of architecture for environmental information collection, analysis and indexing.

The remaining document is structured as follows: Section 3 presents the dataset comprising the type of sources and the regions that are targeted by hackAIR and a number of studies realized on those sources. More detailed information about the empirical study can be found in the Appendix. Section 4 discusses the data gathering techniques applied for each different source (i.e. Flickr, web sites, web services, webcams), and section 5 discusses the data analysis techniques that vary depending on the type of the source (i.e. image, text, JSON file) and provide some examples. Then section 6 deals with the selection of the suitable database for storing the data collected and the development of the environmental node repository. Finally, section 7 concludes this deliverable.

3 Empirical Studies

This section presents the general methodology applied during the empirical studies realized on text-based sources, i.e. environmental sites and services, and image-based sources, i.e. webcams and social media platforms, as well their purpose and their results. The ultimate goal of the studies is to facilitate the automatic discovery of similar sources on the web, to mine the target information and to build the necessary analysis tools.

Regarding the text-based sources, the aim of the empirical study is to determine the types of free and publicly available web nodes in English containing Particulate Matter (PM) measurements and their characteristic features, while as far as the webcams are concerned, the aim is to detect the common characteristics or the most usual ones found in sites containing webcam images and streaming. Finally as far as social media images are concerned the aim is to explore the available platforms that could provide a significant number of images on a daily basis.

At this point, we should note that as far as the web sites are concerned, a similar study targeting a larger set of environmental pollutants and weather related information was conducted in the context of the PESCaDO EU project in which CERTH was a partner. Thus, we will make use of the approach and the findings of the study presented in Deliverable D2.1 of PESCaDO (Moumtzidou, 2010). At the current study, we focus on sources provided by the partners of the hackAIR consortium that provide environmental measurements for Germany and Norway according to the project pilot cases, and several other European cities.

In particular, hackAIR targets the discovery of real-time or near real-time PM measurements (i.e. PM₁₀ and PM_{2.5}) that cover Germany and Norway in order to serve the hackAIR pilot studies. As near real-time measurements we considered measurements that are retrieved during the last 24 hours. Moreover, although not explicitly mentioned in the DoW, historical data are of interest, since they can be used for providing statistical information to end users. Regarding the geographical area of coverage, apart from Germany and Norway, some additional European cities are studied in order to cover the needs of all partners of the consortium, that is areas where the partners of the hackAIR consortium are based (e.g. Greece, Belgium and Netherlands) and places of general interest (e.g. London). Table 1 contains an overview of the cities that are targeted by hackAIR at the moment. Specifically, as far as the pilot cases are concerned, 20 cities were selected from Germany and 7 cities from Norway. Regarding Germany and in order to ensure as big a geographical coverage as possible, the selected cities are the capitals of the federal states and some additional big cities. Regarding Norway, the seven biggest cities of the country were selected. Finally, beyond Germany and Norway, six more cities are selected based on partners' location, and finally, London was included as it presents general interest given that several workshops and conferences related to air quality are held there. In total 34 cities are considered for the reported studies in this deliverable.

Table 1 – Cities studies through hackAIR

Cities related to Germany pilot case	Cities related to Norway pilot case
Stuttgart, Munich, Berlin, Potsdam, Bremen, Hamburg, Wiesbaden, Hanover, Schwerin, Dusseldorf, Mainz, Saarbrücken, Dresden, Magdeburg, Kiel, Erfurt, Rostock, Nuremberg, Leipzig, Cologne	Oslo, Bergen, Trondheim, Stavanger, Kristiansand, Drammen, Tromsø
	Cities related to partners
	Amsterdam (NL), Athens (GR), Brussels (BE), Piraeus (GR), Thessaloniki (GR), Xanthi (GR)
	Cities of general interest
	London (UK)

3.1 Empirical Study Methodology

Empirical research methods are typically divided into two main categories (Creswell, 2003): (i) quantitative methods that employ strategies of inquiry such as experiments and surveys to collect statistical, mathematical or numerical data that are analysed via computational techniques, and (ii) qualitative methods that collect evidence drawn from observations, interviews or/and documentary evidence and analyse them using appropriate qualitative data analysis techniques. Quantitative methods can be viewed as corresponding to confirmatory data analysis as they typically employ statistical testing of hypotheses drawn from existing theories, and thus tend to be more appropriate in cases where a theory is already well developed or for purposes of theory testing and refinement. Qualitative methods, on the other hand, can be viewed as corresponding to exploratory data analysis since they are not based on any pre-specified hypotheses, but seek to conceptualise and establish an understanding of a phenomenon and thus build a theory based on empirical evidence.

Our goal is to provide answers to the research questions outlined earlier regarding the characteristic features of free and publicly available Web nodes containing PM measurements. Given the exploratory nature of our research questions, the method that we apply is a qualitative research method.

A.D. de Groot proposed the empirical cycle as a technique of empirical research for developing theories within a dominant paradigm. This empirical cycle includes several phases, which are depicted in Figure 2 (Magrieta, 1994).

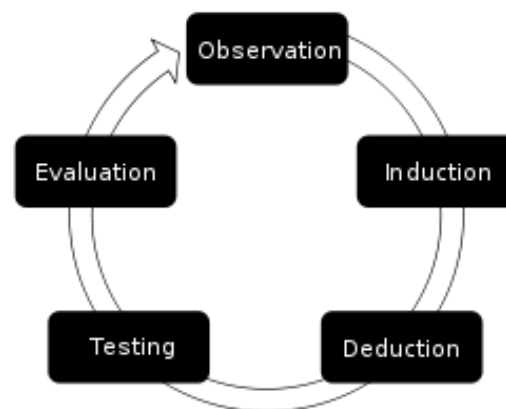


Figure 2- Empirical cycle (Wikipedia: Empirical Research, 2010).

In the Observation phase, the emphasis is on the collection and organisation of empirical facts. In the Induction phase, the aim is to specify explicitly the hypotheses on the basis of the observed facts. In the sequel, there is the Deduction phase, where hypotheses need to be defined in measurable variables, in order to derive concrete testable predictions. Then, the Testing phase follows where the aforementioned predictive statements are checked by collecting new empirical data in order to examine whether the relationships among variables can be found in the new data as predicted. The interpretation of the results within the framework of the specified hypotheses and theories is realized in the Evaluation phase, which connects back to the first phase of the empirical cycle.

In an ideal case, all phases should be carried out. However, this cannot be realized in all types of research, especially those that are of an explorative nature. For example, in descriptive studies, emphasis is put mostly on the first two phases, Observation and Induction according to Groot (1961).

Consequently, an empirical study is based on a set of experimental observations concerning a certain topic for which the researcher lacks sufficient theoretical knowledge and thus given that there is no official analysis of environment-related web pages, we will use this approach for identifying the characteristics of environment-related web pages. This study should reveal the characteristic features of the environmental service node pages and their encoding. The empirical study for sites with webcams follows a similar methodology. However, the studies for web services and social media platforms are somewhat different as these image sources have already been identified, and hence the study focuses on the Testing and Evaluation phase of the cycle.

3.2 Empirical Study of Environmental Websites

Based on the methodology presented in the previous section, we conduct an empirical study on environmental websites and discuss the results. We first present the selected dataset and the motivation behind its selection, and then the adopted study methodology and the environmental site analysis.

3.2.1 Dataset

A representative set of 18 environmental websites regarding air quality information was provided for this study by DRAXIS and NILU, since DRAXIS is an SME with strong experience in the domain and NILU is a research foundation focusing on air pollution. Given that the hackAIR pilot cases aim at providing air quality services and recommendations to citizens in Germany, and Norway, the dataset heavily emphasized on these regions. However, other cities that were mentioned earlier are included as well.

In order to make a first assessment of the dataset and describe it according to its content, we provide statistics regarding the services offered, the area covered, the language and the temporal information. In that way we are able to conclude whether this dataset is appropriate for the requirements of hackAIR. As far as the services offered are concerned, the studied websites address aspects (usually more than one) related to air quality and the distribution of the samples regarding the service offered for the selected dataset can be found in Table 2. It is quite evident that **the majority of studied nodes contain air quality, while only few sites provide general information.**

Table 2- Distribution of services offered.

Services Offered	Number of sites	Percentage (%)
Air Quality Measurements	16	88.9 %
General Environmental Information	2	11.1 %

Regarding the area covered (Table 3), the selected environmental sites provide information about several countries in Europe. It should be noted that the number of sites providing information exclusively for Norway and Germany is quite low but if we take into account that the sites providing information about Europe or the World contain data also about the pilot countries, the number is satisfactory. An interesting clue that arises from the study is that there is a significant number of sites providing air quality measurements for London.

Table 3- Distribution of coverage area

Area	Number of sites	Percentage (%)
Germany	3	16.7 %
Norway	1	5.6 %
Other European countries/cities (Germany and Norway are not included) – one or two countries per site	7	38.8 %
Europe (several European countries)	4	22.2 %
World cities or countries	3	16.7 %

Finally we provide information for the sample dataset on the type of measurements with respect to time. More specifically, this information reveals the temporal period to which the site data refer. In this context, the temporal types have been identified as Real Time Data, Historic Data and Forecast. The statistics of the environmental nodes regarding the temporal information are presented in Table 4. Observing the distribution, we can deduce that more than half of the sites contain real time and historic data and fewer sites contain forecasts. It should be mentioned that most of the websites usually contain more than one temporal data type (e.g. real time data and historic data).

D3.1: Environmental node discovery, indexing and data acquisition

As hackAIR is aiming at providing real time services, we are mostly interested in sites containing real- or near real-time information, while forecast data are not of interest for hackAIR at the current stage.

Table 4- Distribution of temporal type of measurements

Temporal Type of Measurements	Number of sites	Percentage (%)
Real Time data/ Near Real time data	11	61.1 %
Historical data	10	55.6 %
Forecast data	8	44.4 %

3.2.2 Methodology for the Empirical Study of Environmental Websites

This section presents an application of the empirical cycle technique described in section 3.1 on environmental nodes. Given that this is an iterative procedure and that it is possible to apply part of the phases of the empirical cycle in a less strict order, the study was conducted into two iterations and the phases that we applied in iteration are observation, induction/deduction and testing. These two iterations will be presented in the sequel.

3.2.2.1 First phase

1. **Observation:** During the observation phase, part of the sample environmental nodes was observed in order to understand the type of information that is of importance. The number of sites that were studied at this phase was not defined in advance, but it was specified by the quality and quantity of the knowledge gained. More analytically, we stopped the procedure of observation when it was clear that further studying of environmental nodes was not leading to additional knowledge.

- **Induction and Deduction:** The aim of this step is to specify explicitly the hypotheses on the basis of the observed facts and to draw conclusions regarding the dataset. Thus, based on the analysis/study conducted at the observation step, the hypothesis/deduction formulated is that the key information that should be detected for each environmental website includes the following: a) **Type of measurement** (and data), b) **Location**, and c) **Time and date**.

It should be mentioned that although the exact measurements (i.e. numerical values) are not required for discovering and indexing purposes, we will need to be able to extract and constantly update them in the repository to support the data retrieval service.

2. **Testing:** During this step, the sites that were not studied during the observation phase were observed in order to identify the type of information that can be detected into the selected dataset based on the findings of the observation and induction phases. Thus during this step, the hypotheses that emerged from the study of the observation dataset were tested. The results showed that the information that was considered as meaningful (i.e. type of measurement, location, time and date) is indeed the most useful information.

After the first iteration was completed successfully, a second one was initiated that focused on identifying where the important information is located and how it is encoded.

3.2.2.2 Second phase

- **Observation:** In this phase, we used the same methodology (as in the 1st iteration) to detect all the possible parts of a webpage that contain important information about the site and the way the information is encoded.
- **Induction and Deduction:** In this phase and in the context of identifying key places where meaningful information resides, we deduce that the constitutive elements of web pages are their content, comprising structured text, and URL. The conclusions drawn from the assessment of the sample dataset were that the web pages did not share a common template or layout but on the contrary their structure resembled that of other non-environmental websites. Thus, the elements that are descriptive for the environmental web pages are their content/HTML structure and URL. Moreover, as far as the encoding of information is concerned, it was found that meaningful information was in the form of keywords, HTML tags, inside the URL and embedded in images.
- **Testing:** During this phase, the sites forming the testing dataset were studied in the light of the knowledge that has emerged from the study of the observation dataset in order to test the conclusions we extracted in the previous step regarding the location and the encoding of important information. The results showed that the hypotheses we made during the induction step were valid.

3.2.3 Analysis of Environmental Websites

Based on the aforementioned methodology we have observed all the websites included in the sample dataset. After having identified the different sources, in which important information is encoded, we present a detailed insight of each source followed by conclusions regarding the type of information, the encoding schemas as well as potential extraction techniques. In the sequel, we present an example of the analysis realized on one environmental node of the dataset. Detailed information about the study of all nodes can be found in the Appendix section 9.1. An example is presented below.

3.2.3.1 Air Pollution in the World - aqicn.org Website

The site <http://aqicn.org/city/> is developed and maintained by the World Air Quality Index team and the data provided come from several sources including the Beijing Environmental Protection Monitoring Center, the U.S. Embassy Beijing Air Quality Monitor, and the Open Weather Map. We study a specific page that provided information for the city of Berlin in Germany (Figure 3), which is available on the following URL:

<http://aqicn.org/city/germany/berlin/>

D3.1: Environmental node discovery, indexing and data acquisition

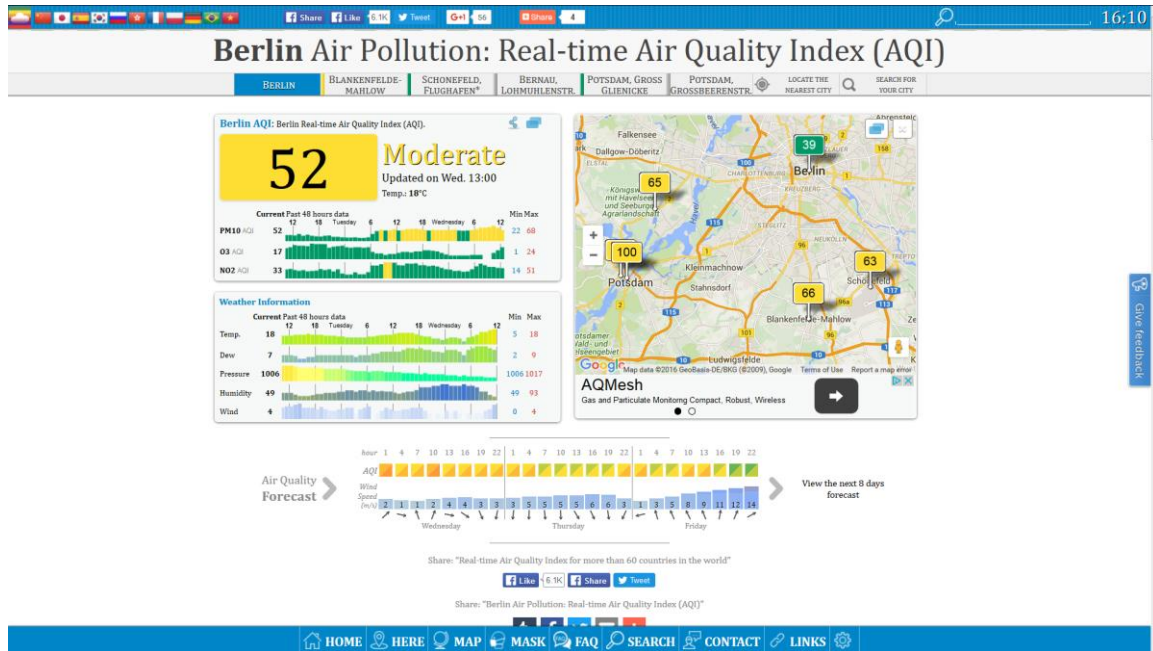


Figure 3- URL and screenshot of the web page.

After thorough investigation of the specific website, we identified the following meaningful information in the specific site:

- **Type of measurement:** air quality measurements and weather related information. Specifically, PM₁₀, O₃, NO₂, Air Quality Index and temperature, dew, pressure, wind and humidity
- **Location:** City of Berlin in Germany
- **Time and date:** Wednesday 13:00

These observations were made during the first phase of the cycle. At the second phase we detect the places in the web pages where this information was located. More specifically, the parts that were identified containing the aforementioned information are the content/text of the web page, and the URL.

As far as the content/text of the web page is concerned and after careful observation, we conclude that the measurements and the type of aspect (i.e. pollutant or weather phenomenon) is formatted in an HTML table where the first column indicates the type of aspect measured (e.g. temperature, pressure), the second the latest value of that aspect obtained by the site, the third a bar chart with historical values of past 48 hours data and the last two the minimum and maximum values observed during this 48 hour period. As far as the location is concerned, it can be found in bold format inside the text, but also in the title of the webpage (i.e. <title>Berlin, Germany Air Pollution: Real-time PM2.5 Air Quality Index (AQI)</title>). Date and time is not provided fully but it can be deduced given that the site is updated on a regular basis. We should also note that there is a Google map that contains the Air Quality Indexes of several locations close to Berlin and an air quality forecast for the upcoming days.

Finally, as far as the information that can be found in the URL of the web page is concerned, we can extract the location to which the measurements refer (i.e. Germany / Berlin). Only websites of the dataset containing PM measurements will be considered by this process.

3.3 Empirical Study of Web services

In this section, we present the study and analysis of air quality related web services. First, we analyse the concept of web services, then we describe the dataset of web services provided by the experts and present an example web service and finally we conclude with the findings of the study.

3.3.1 Introduction to Web Services

Web services, as defined by W3C, are software systems designed to support interoperable machine-to-machine interaction over networks and have an interface described in a machine-processable format. There are basically two parties involved in web services; the one providing a set of exposed APIs and the other, commonly known as web service consumers, uses the functionality provided by the web services. In general, web services provide an abstract reusable interface, they hide complexities from the web service consumers, they are easy to use and language independent. Other systems interact with the web services in a manner prescribed by its description. Web service technology has evolved around a stack of five technologies: Network, Transport, Packaging, Description, and Discovery. Each of these technologies sits on top of the ones before it (e.g. without a Network to build on, there could be no Transport). Figure 4 shows this stack of technologies on the left along with the most common choices for each of these layers on the right.

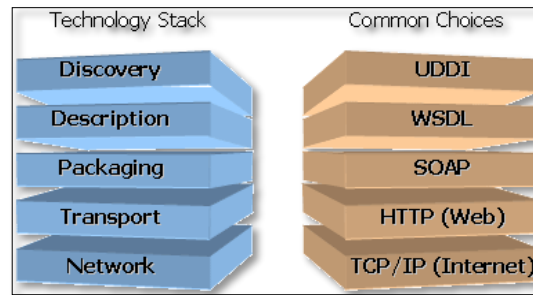


Figure 4 - The Web services technology stack

There are different methods for providing web services but the most common are SOAP, XML-RPC and REST (Ivak, 2014; Kumar, 2016). At this point we should mention that the formal definition of Web Services refers to SOAP Web Services which function as part of the stack presented in Figure 4. However, there are other types of web services as well, such as XML_RPC and REST (the most commonly used type of web service) that do not follow strictly this architecture. In the sequel, we present a concise description of the aforementioned types of web services.

- **XML-RPC:** This is simple to use, develop and consume and uses XML. It is simpler compared to SOAP and does not require/support WSDL. It also does not support i18n and allows only one mode of method serialization.
- **SOAP:** This is actually a modified and more powerful version compared to XML-RPS, and thus it is method for exchanging XML based message over the Internet for providing and consuming web services. It is based on WSDL and UDDI and its messages are transferred forming the SOAP-Envelope. However, SOAP is widely criticized for being too complex and verbose.
- **REST:** This actually is not a protocol but an architectural approach. RESTful is another term to define REST web services, and means that each unique URL is a representation of some object. It is not necessary to use XML as a data interchange format in REST. The format of information (representation) returned can be XML, JSON, plain text or even HTML, but usually developers favour JSON or XML. In general REST is very easy to develop and maintain. However, it depends on other security approaches like OAuth and it is confined to HTTP.

Given that XML-RPC is gradually becoming obsolete, the two main methods used are REST and SOAP. The main advantages of REST web services versus SOAP are that they are lightweight, the results are human readable and they are very easy to build. On the other hand, SOAP adheres to a contract which makes it more formal; it can leverage different transport protocols, including HTTP and SMTP and provide better support for attachments.

3.3.2 Dataset

Similar to the web sites, the environmental experts, i.e. DRAXIS and NILU, have provided a small dataset consisting of three web services. After an inspection of the web services, we deduce that all of them provide PM measurements. Regarding the area covered (Table 5) the web services provide information about countries in Europe; however Norway and Germany are not included.

D3.1: Environmental node discovery, indexing and data acquisition

Table 5- Distribution of coverage area

Area	Number of sites	Percentage (%)
Other European countries/cities (Germany and Norway are not included) – one or two countries per site	3	100 %

Finally as far as the temporal period to which the site data refer to and based on the information presented in Table 6, we can deduce that usually web services provide real time data, but based on the test queries, historical data are provided as well by the majority of them.

Table 6- Distribution of temporal type of measurements

Temporal Type of Measurements	Number of sites	Percentage (%)
Real Time data/ Near Real time data	3	100 %
Historical data	2	66.7 %

3.3.3 Analysis of Web services

In this section, we present an example of REST web service providing PM measurements in JSON format. The service is available for only a limited number of countries. Given the hackAIR cities of interest (Section 3.2.1), we focus specifically on the city of Amsterdam and city of London. It should be noted that neither Germany nor Norway is among the countries covered at least in the current version of the presented web service.

The website <https://docs.openaq.org/> offers documentation regarding the API, its parameters and the supported functions. Based on the guidelines provided, the HTTP GET query for retrieving PM10 measurements for the city of Amsterdam is the following:

<https://api.openaq.org/v1/measurements?city=Amsterdam¶meter=pm10>¹

Figure 5 depicts the JSON response, which provides some general information, e.g. the number of total results and the current page CSV file, and some information for each measurement including the location, the coordinates, the date and the pollutant value.

¹ See Appendix 9.2 for more detailed description of the web service. The Serial Number is S/N=1.

D3.1: Environmental node discovery, indexing and data acquisition

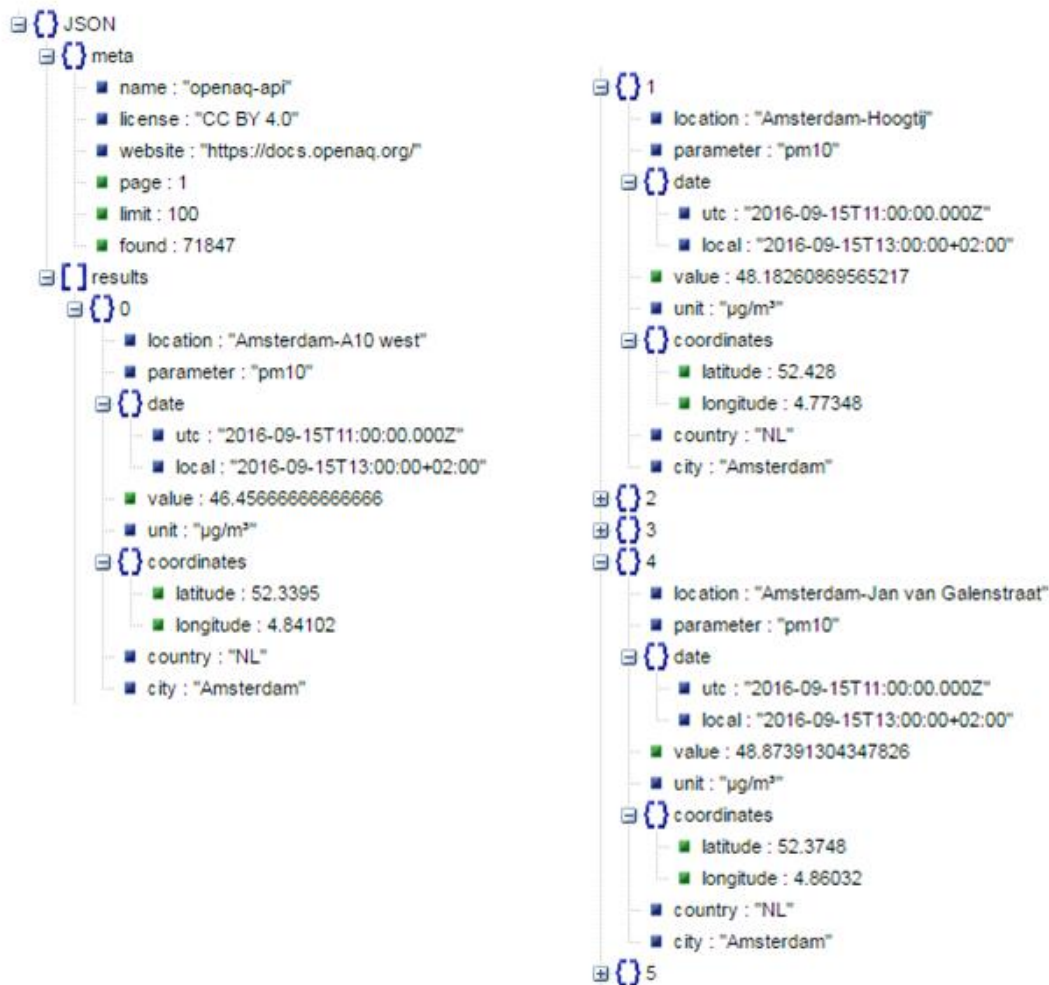


Figure 5 – JSON response for the city of Amsterdam (broken in two columns for the sake of space).

An example from the first page is the following:

1. country : "NL"
2. city : "Amsterdam"
3. location : "Amsterdam-A10 west"
4. coordinates: latitude : 52.3395, longitude : 4.84102
5. parameter : "pm10"
6. date UTC : "2016-09-15T11:00:00.000Z"
7. value : 46.456666666666666
8. unit : "µg/m³"

The information provided by the service cover our needs since time, location, pollutant type and value are all covered. Detailed information about the study of all nodes can be found in the Appendix section 9.2.

3.3.4 Findings

The aforementioned study showed that web services constitute a convenient means of obtaining information since they are structured in a well-defined way and thus the information extraction is straightforward. Regarding the functionality of the studied services, it can be recognised either indirectly from the name of the functions appearing in the WSDL file (usually the names indicative of the service provided) or if they are known in advance.

In order to detect web services existing in the Internet, it is possible to use UDDI, which is a protocol that defines the standard means for Web service discovery, and internet search engines. As far as UDDI is concerned, it is a platform-independent registry for businesses worldwide used for publishing service listings and for defining how these services interact over the Internet. Apart from UDDI, general search engines such as Google can be used for

D3.1: Environmental node discovery, indexing and data acquisition

discovering sites with web services. However, most of the found web services are converters (e.g. currency, temperature converters). In conclusion, it seems that there are not many publicly deployed web services offering air quality measurements, but given that is relatively simple to retrieve data from them, we will use data from the discovered web services.

3.4 Empirical Study of Social Media Platforms

In this section, we present the study and analysis of the social media platforms that users prefer for posting publicly their images. In hackAIR, images, after being processed, are used for producing an air quality (i.e. PM) estimation of the surrounding area. Thus, in this section, we introduce the social media platforms that can be used, then we perform a simplified empirical study on the source that is selected and eventually we present an example of its API and the returned results.

3.4.1 Social Media Platforms

Users upload billions of images per day in social media. However, not all social media are suitable or equally popular for posting images. The KPCB Internet Trends Report 2016² provides an overview of the trends related to image sharing/ posting for 2005-2015 (Figure 6). The conclusions that arise from Figure 6 are the following:

1. Users upload more than 3 billion images per day in social networks
2. First in that list is Snapchat³, with WhatsApp⁴ to follow and in the 4th place is Facebook Messenger⁵. However, neither of these networks distributes free, through an API, the user-contributed images.
3. Facebook is in the 3rd place; however the Facebook API allows access only to images from public pages and not from personal user profiles, while it also does not maintain the geotagging information of uploaded photos, thus rendering them useless for the purposes of hackAIR.
4. On the other hand, Instagram⁶ which is placed in 5th place, used to offer access (until June 2016) through its API to all public images of its users.

Based on the aforementioned analysis, one of our first choices was Instagram. The average number of uploaded photos/ videos per day is more than 95 million photos/videos according to the Instagram official page⁷.

² <http://www.kpcb.com/internet-trends>

³ <https://www.snapchat.com/>

⁴ <https://web.whatsapp.com/>

⁵ <https://www.messenger.com/>

⁶ <https://www.instagram.com/>

⁷ <https://www.instagram.com/press/>

Image Growth Remains Strong

Daily Number of Photos Shared on Select Platforms, Global, 2005 – 2015

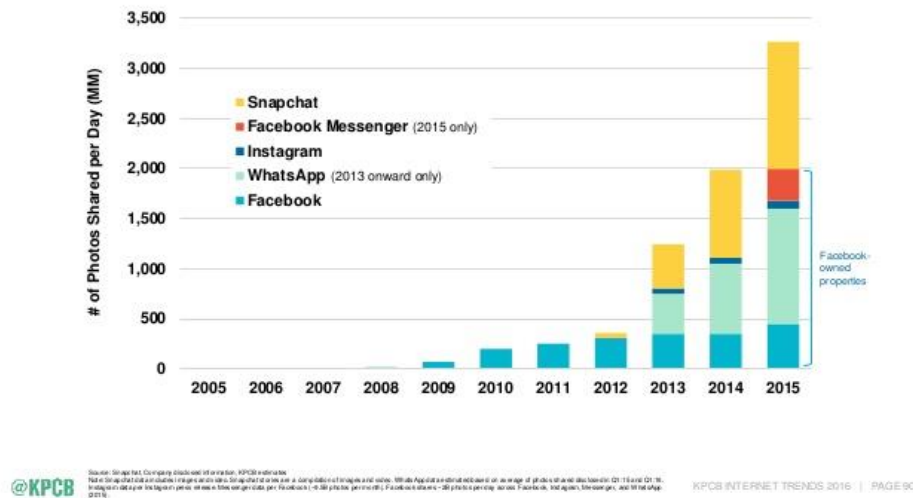


Figure 6 - KPCB Internet Trends Report 2016

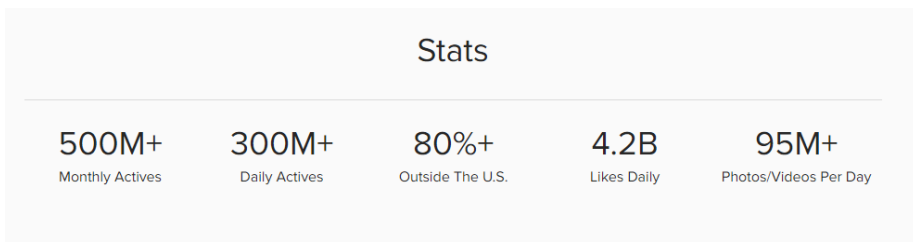


Figure 7 – Instagram Statistics

Unfortunately, Instagram announced in early 2016 a platform update⁸. This update was the result of the release of an app called beingtheapp⁹ which was developed by a third party and which gave users the ability to simulate someone else’s profile. Therefore, Instagram decided to change its API access to protect its users, adding strict limitations on the apps that could access the data and the number of data they could retrieve. As of June 1st, 2016 the API entered sandbox mode, returning only restricted data with the following conditions:

- Restricted API rate limits, from 5000 requests per hour to 500 per hour
- Data is restricted to the 10 users and the 20 most recent media from each of those users

Due to the aforementioned limitations, hackAIR examined other social networks as alternatives for collecting publicly available images provided by user on daily basis. Since the other platforms mentioned above (i.e. Snapchat, WhatsApp, Facebook and Facebook Messenger) did not change their API policies, we looked at KPCB Internet Report 2014¹⁰ for an alternative solution. Thus, according to KPCB Internet Report 2014, Flickr¹¹ is the next social network in terms of image uploads and also provides an open API that enables gathering all public images users share through their profiles. Figure 8¹² depicts in more detail the number of images uploaded per month during the period 2004-

⁸ <http://developers.instagram.com/post/133424514006/instagram-platform-update>

⁹ <http://www.beingtheapp.com/>

¹⁰ <http://www.kpcb.com/blog/2014-internet-trends>

¹¹ <https://www.flickr.com/>

¹² <https://www.flickr.com/photos/franckmichel/6855169886>

D3.1: Environmental node discovery, indexing and data acquisition

2016. The chart was produced by data provided by Flickr. Finally, using again Flickr’s API, we can retrieve some more specific information regarding the images uploaded in 2015 in total, on a monthly or a daily basis:

- 728 million in total for 2015
- 60.7 million per month on average for 2015
- 2 million per day on average for 2015

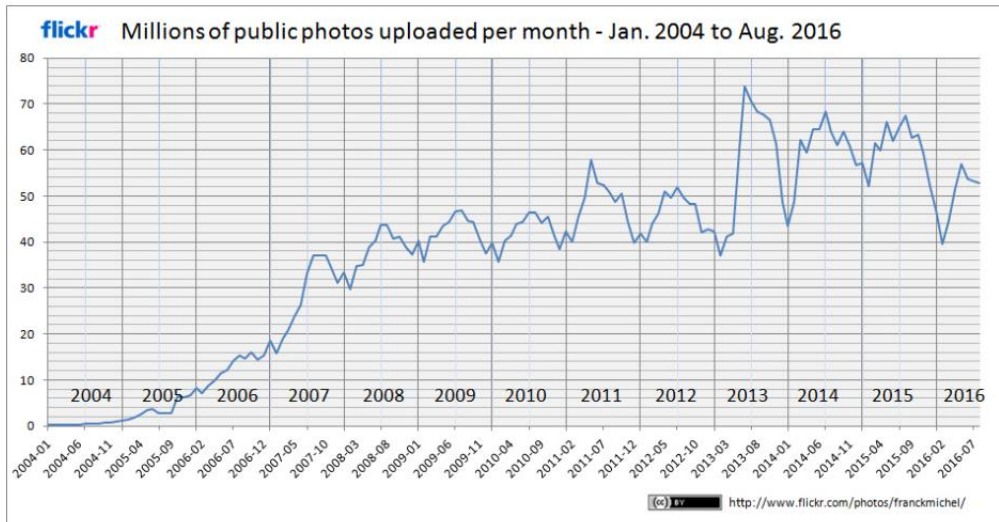


Figure 8 – Chart showing number of images uploaded to Flickr per month – Report 2004-2016

3.4.1.1 Flickr vs Instagram

In this section, we present a short comparison between Flickr and Instagram which were the two social platforms that were tested. The study was realized on 29 of the cities mentioned in Table 7, using the image collection modules that were implemented for collecting publicly accessible images that are located near the center of the selected cities. The analysis was made for the same period for the two networks (January 1st, 2016 – May 31st, 2016).

Table 7 contains the total number of images found per city both for Instagram and Flickr and the number of images that were recognized automatically to contain a part of the sky. The algorithm for sky detection was the same for both platforms and it is described in Section 5.2.1.1.3. It is quite obvious from simple observation of the values of Table 7 that the number of images uploaded on Instagram during the period 1/1/2016-31/5/2016 is considerably higher than that of Flickr. However, a rather interesting point that should be considered as well is that the ratio of images containing a part of the sky is higher in Flickr ($ratio = \frac{2112}{8282} = 0.26$) compared to Instagram ($ratio = \frac{7956}{50352} = 0.16$).

Table 7 – Total number of images and images containing sky for Instagram and Flickr during 1/1/2016-31/5/2016.

Cities	All Images		Images with Sky	
	Instagram	Flickr	Instagram	Flickr
Amsterdam	70,066	38,560	12,339	11,348
Attiki	101,222	8,749	13,280	3,107
Bergen	14,736	1,902	3,772	862
Berlin	97,038	37,972	20,104	10,597
Bremen	14,078	2,026	1,925	515
Brussels	143,027	29,604	20,568	3,615

D3.1: Environmental node discovery, indexing and data acquisition

Cologne	65,928	9,298	8,010	1,837
Drammen	14,122	617	2,548	199
Dresden	47,088	4,811	10,463	1,888
Dusseldorf	87,111	9,687	12,486	2,723
Erfurt	15,809	1,058	2,307	332
Hamburg	89,443	9,505	14,519	2,727
Hanover	35,943	3,547	4,515	1,272
Kiel	18,136	3,264	3,093	1,350
Kristiansand	22,697	495	4,672	122
Leipzig	36,261	7,183	5,935	1,157
London	259,459	29,970	34,949	7,399
Mainz	22,662	5,773	3,004	2,030
Oslo	90,380	4,475	18,097	965
Potsdam	27,198	1,344	7,945	517
Rostock	19,022	839	4813	568
Saarbrucken	19,148	2,603	1,844	815
Schwerin	7,735	514	1,982	306
Stavanger	29,685	4,196	7,108	1,244
Stuttgart	43,624	8,735	5,099	1,699
Tromso	18,343	1,562	7,081	848
Trondheim	25,077	1,962	4,734	1,033
Wiesbaden	13,813	434	1,898	161
Xanthi	11,360	28	1,099	16
Average	50,352	7,956	8,282	2,112

Table 8 – Average number of images per day for five example cities.

Cities	All Images		Images with Sky	
	Instagram	Flickr	Instagram	Flickr
Amsterdam	217	125	69	60
Berlin	361	120	131	52
Dusseldorf	314	27	83	13
Kristiansand	69	1	25	1
Wiesbaden	137	1	15	1

D3.1: Environmental node discovery, indexing and data acquisition

Figure 9 and Figure 10 depict the same data in the form of bar charts. Moreover, Table 8 contains the average number of images posted per day both in Flickr and Instagram for five example cities (three big and two small ones). It is evident that for larger cities, this number is sufficient for providing an estimation of the air quality considering that 50 images per day constitute an adequate sample per region. However, for small cities, social media platforms and especially Flickr (given that Instagram cannot be used anymore) cannot always be considered as a sufficient source for air quality information. Overall, at the current stage, **user-generated images from Flickr can be used as a valuable source for testing the air quality estimation methodology from user-generated images (described in D3.3) through a first round of pilot experiments in several of the regions targeted by the project.**

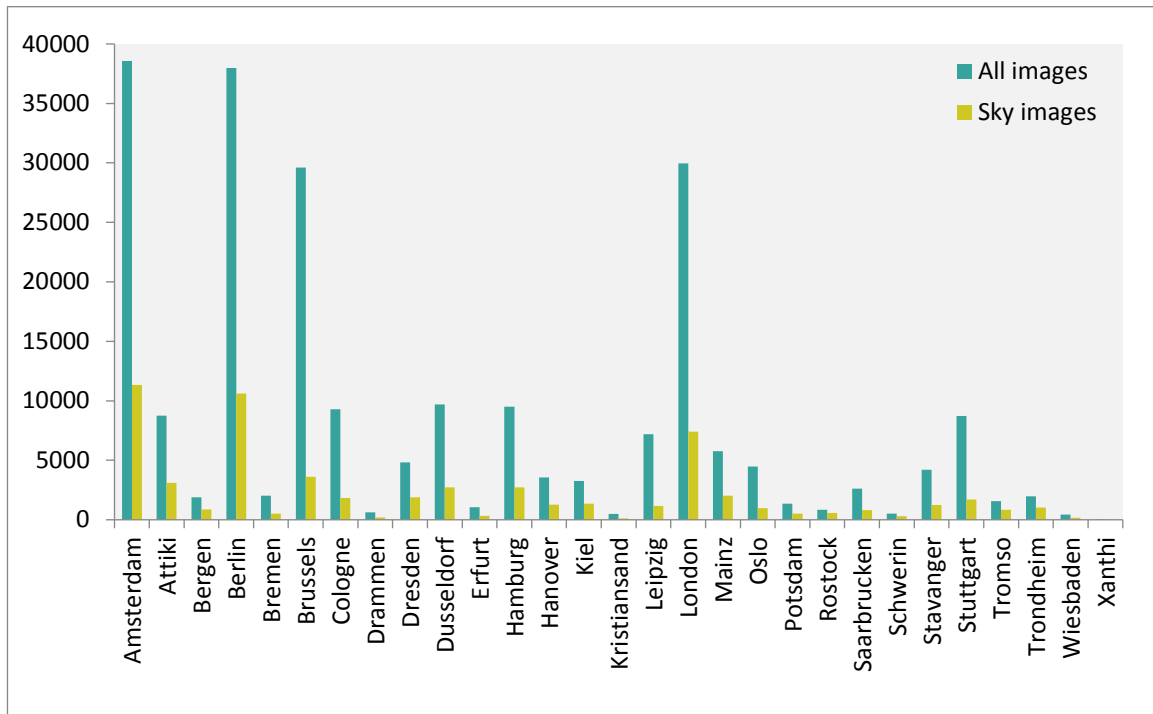


Figure 9 – Total number of images and images containing sky for Flickr during 1/1/2016-31/5/2016

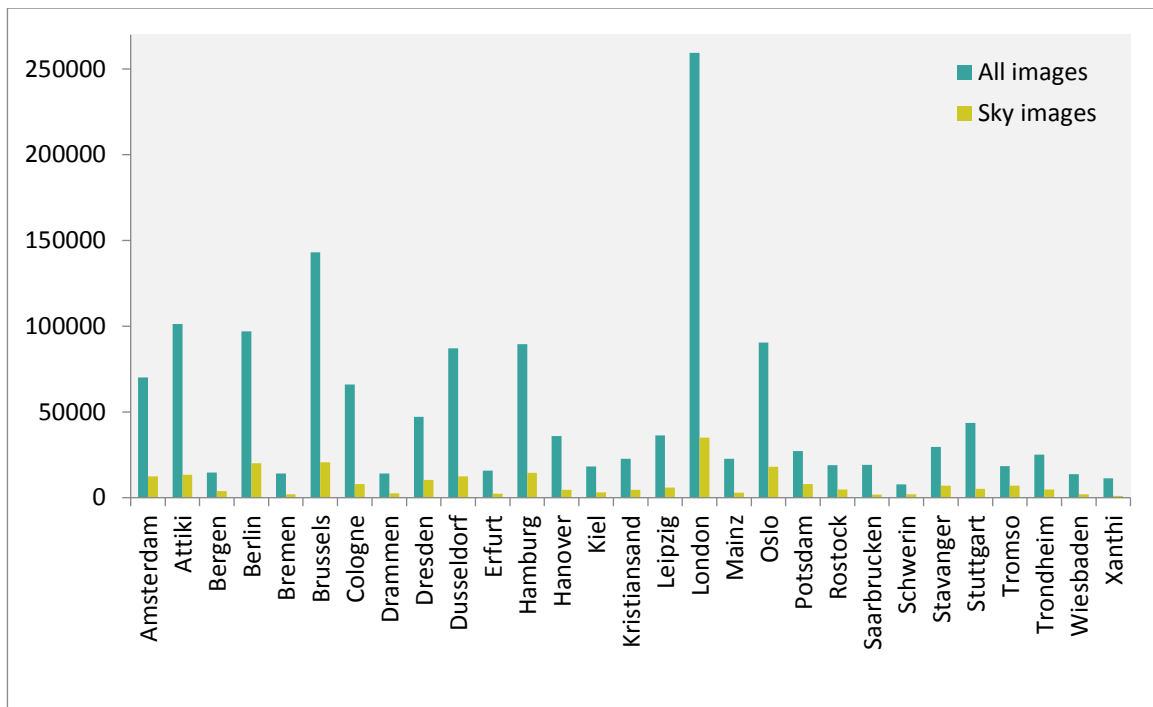


Figure 10 – Total number of images and images containing sky for Instagram during 1/1/2016-31/5/2016

3.4.2 Methodology for the Empirical Study of Flickr Social Media Platform

This section presents a modified application of the Empirical cycle technique described in section 3.1.1 on the Flickr social media platform. The aim of the study is to check whether the number of images for a set of indicative cities remains more or less stable through time. Given the fact that this study focuses only to the time aspect, only a single iteration is required. The phases that are applied are observation, induction/deduction and testing.

3.4.2.1 First phase

1. **Observation:** During the observation phase, we observe the number of images posted in Flickr during the three first months of the year (2016), January, February and March. Table 9 contains the available data and it is obvious that in general the numbers are relatively similar during the different months. A slight increase is observed during March, but this applies to all cities. However, the number of images per city remains similar and no dramatic increase or decrease is observed.

Table 9 –Number of images posted in Flickr per month for the months January, February, March.

Month	All images				Images with Sky			
	Amsterdam	Berlin	Dusseldorf	Kristiansand	Amsterdam	Berlin	Dusseldorf	Kristiansand
January	4,597	4,384	1,075	43	1,500	1,056	303	12
February	4,230	4,262	1,149	105	1,386	1,049	349	28
March	5,801	6,448	1,346	31	1,769	1,685	425	8

2. **Induction and Deduction:** The aim of this step is to specify explicitly the hypotheses on the basis of the observed facts and to draw conclusions regarding the dataset. Thus, based on the analysis/study conducted at the observation step, the hypothesis/deduction formulated is that the number of images retrieved per city and per month should be similar or at least have the same order of magnitude.
3. **Testing:** During this step, we check the number of images posted per city for the upcoming three months April, May, June, that were not studied during the observation phase. Thus, during this step, the hypotheses that emerged from the study of the observation dataset were tested. The results showed that the number of images per city remains similar for all the studied months and no significant changes should be expected. Another conclusion that emerges is that we should always expect bigger cities to have significantly more data throughout the year.

Table 10 –Number of images posted in Flickr per month for the months April, May, June.

Month	All images				Images with Sky			
	Amsterdam	Berlin	Dusseldorf	Kristiansand	Amsterdam	Berlin	Dusseldorf	Kristiansand
April	8,971	7,928	1,441	40	2,800	2,463	507	13
May	8,297	8,209	3,197	83	2,061	2,372	774	18
June	6,664	6,741	1,479	193	1,832	1,972	365	43

3.4.3 Collection of Flickr images for a specific city

In this section, we present an example of a call realized to Flickr API for retrieving data for the city of Berlin, which has center coordinates (52.520007, 13.404954) for August 2016, in order to provide more evidence regarding the validity of the outcome of the empirical study.

```
https://api.flickr.com/services/rest/?method=flickr.photos.search&media=photos&extras=geo,description,date_taken,owner_name,views,url_l&api_key=a560*****c0e&format=json&nojsoncallback=1&page=1&per_page=250&lat=52.520007&lon=13.404954&radius=16&radius_units=km&min_taken_date=1470009600&max_taken_date=1472601600
```

D3.1: Environmental node discovery, indexing and data acquisition

The initial number of photos returned is around 5,900, but after considering several restrictions imposed the number drops significantly to 2,171, which is close to the data posted from January to June. Detailed analysis of the call, the parameters and the response of the Flickr API is presented in Section 4.2.1.

3.5 Empirical Study of Webcams

Based on the methodology presented in the previous section, we conducted an empirical study on webcams and discussed the results. However, whether we will actually use the images of webcams for the hackAIR purposes is not finalized yet given their low quality and thus this issue requires further exploration in cooperation with the air quality expert partners, which will be realized in deliverable 3.2. At first, we present the selected dataset, then we present the applied study methodology and finally we study an example webcams site.

3.5.1 Dataset

A representative set of webcams that contain sky and cover the areas mentioned in Section 3.2.1 was provided for this study by NILU and BUND which are based in Norway and Germany respectively, since these are the main areas of interest. The majority of the webcams are linked to portals that cover several cities among which is the AMOS dataset¹³ that retrieves images from 29,838 webcams located around the world. Compared to other portals, which simply host webcams, AMOS stores the images retrieved by them locally and makes them available through a Python script and REST API. Thus, it can be considered as a web service for webcams and will be studied separately in a later section.

3.5.1.1 Webcam sites and portals dataset

To make a first assessment of the dataset and describe it according to its content, we provide numerical statistics regarding the area covered, and how the webcam is embedded in the site (i.e. single image, website, flash). Regarding the area covered (Table 11) the selected sites contain webcams that are located in Germany or Norway. However, a significant number of sites are portals and thus they link to webcams hosted in several countries around the world. Moreover, regarding the format of the sites (

Table 12) hosting the webcams, the majority (~62 %) are simple websites and thus images can be extracted relatively easy since they are stored in form of images. Almost a 30% of the sites contain webcams displayed using Flash, which are rather difficult to obtain (if not impossible, given that the link of the flash video can be encrypted, URL-escaped, wrapped in JSON, etc.). And finally a 10% only provides simple images that are updated regularly. A detailed description of the dataset can be found in Appendix 9.3.1.

Table 11- Distribution of coverage area

Area	Number of sites	Percentage (%)
Single German cities	15	48.4 %
Single Norwegian cities	7	22.5 %
Several cities of Germany	2	6.5 %
Several cities of Norway	1	3.2 %
Countries around the world	6	19.4 %

Table 12- Distribution of format type

Temporal Type of Measurements	Number of sites	Percentage (%)
Image	3	9.7 %

¹³ <http://amos.cse.wustl.edu/dataset>

D3.1: Environmental node discovery, indexing and data acquisition

Website/ image	19	61.3 %
Website/ flash	9	29.0 %

3.5.1.2 AMOS dataset

The AMOS dataset (Jacobs, 2007) contains 983,688,543 images taken from 29,838 webcams located around the world, the vast majority in the United States. The construction of AMOS began in March, 2006 and continues to this day. To make a first assessment of the AMOS dataset and describe it according to its content, we provide numerical statistics regarding the number of the cameras identified for the areas of interest described in Section 3.2.1. Specifically, we identify the total number of webcams per country, the number of cameras that work until now and the number of cameras that contain part of the sky (Table 13).

Table 14 contains the number of cities from the dataset in Section 3.2.1 that have either no cameras or no working cameras or no cameras containing a part of the sky, and finally

Table 15 shows the mean number of cameras that are expected to be found in every city. A detailed description of the dataset can be found in Appendix 9.3.2.

Table 13- Number of webcams per country

Country	Cities Number	Matches	Working Cameras	Cameras containing sky
Germany	20	17	12	12
Norway	7	78	38	19
Greece	2	10	2	1
Brussels	1	0	0	0
UK	1	9	3	2
Netherlands	1	8	1	0
Sum	32	122	56	34

Table 14- Information for cities with cameras

Query	Number of sites
Total number of cities with no cameras	12
Total number of cities with no working cameras	16
Total number of cities with no working cameras with sky (i.e. Thessaloniki, Brussels, Amsterdam, Stuttgart, Munich, Dresden, Dusseldorf, Potsdam, Wiesbaden, Schwerin, Mainz, Saarbrucken, Magdeburg, Kiel, Erfurt, Nuremberg, Leipzig, Bergen)	18

Table 15 – Mean number of cameras

Query	Number of sites
Mean number of cameras per cities (by considering only cities with working sky cameras)	2.5
Mean number of cameras per cities (by considering all cities)	1

3.5.2 Methodology for the Empirical Study of Webcams

This section presents an application of the Empirical cycle technique described in section 3.1 on webcams sites and portals. The AMOS dataset is not included since it resembles mostly a web service that provides images. Given the fact that the Empirical cycle is an iterative procedure and that it is possible to apply part of the phases of the empirical cycle in a less strict order, the study, similar to the one presented for websites, was conducted in two iterations and the phases that we have applied in iteration are observation, induction/deduction and testing. These two iterations are presented in the sequel.

3.5.2.1 First phase

1. **Observation:** During the observation phase, part of the samples sites/ portals containing webcams was observed in order to understand which information should be retrieved. The number of sites/portals that were studied was specified by the quality and quantity of the knowledge gained.
2. **Induction and Deduction:** The hypothesis/deduction that is formulated during this step is that the important information that should be detected from each source is:
 - Webcam image/ flash
 - Webcam location
 - Time and date
3. **Testing:** During this step, we test whether the hypotheses that emerged from the study of the observation dataset is valid. The results showed that the information that were considered as meaningful (i.e. webcam image/flash, location , time and date) is indeed the most useful information.

After the first iteration was completed successfully, a second one was initiated that focused on identifying where the important information is located and how it is encoded.

3.5.2.2 Second phase

1. **Observation:** In this phase, we follow the same methodology described in the 1st iteration in order to detect the way the information is encoded.
2. **Induction and Deduction:** The constitutive elements of web pages are their content, and URL. The assessment of the sample dataset showed that in the case that webcams are hosted in sites/ portals, they do not share a common layout. However, it also revealed that the interesting information is found inside images, in html tags close to the image, inside META keywords and in the URL.
3. **Testing:** During this phase, the sites forming the testing dataset were studied based on the results produced during the study of the observation dataset. The study focused on testing the location and the encoding of important information. The results showed that the hypotheses we made during the induction step were valid.

3.5.3 Analysis of Webcams

Based on the aforementioned methodology we have studied carefully all the websites/ portals as well as the AMOS service. After having identified the different sources, in which important information is encoded, we present a detailed insight of each source used. In the sequel, we present an example of the analysis realized an indicative website containing a webcam and a page holding the information of an indicative camera from the AMOS dataset. Detailed information about the study of all nodes can be found in the Appendix section 9.3.

3.5.3.1 Webcam in Hannover

The page studied is part of the site of a regional television channel targeting northern Germany, specifically the states of Schleswig-Holstein, Lower Saxony, Mecklenburg-Vorpommern, Hamburg and Bremen. It is broadcast by both Norddeutscher Rundfunk (NDR) and Radio Bremen. In the following example, we study a specific page that hosts a webcam from Hannover (Figure 11). The webpage URL is the following:

D3.1: Environmental node discovery, indexing and data acquisition

http://www.ndr.de/fernsehen/sendungen/mein_nachmittag/wettermelder/Wetter-Webcam-in-Hannover,wettermelder144.html¹⁴

After thorough investigation of the specific website, we identified the following information of interest:

- **Type:** Webcam image
- **Location:** City of Hannover
- **Time and date:** updated daily

These observations were made during the first phase of the cycle. At the second phase we detect the places in the web pages where this information was located. More specifically, the parts that were identified containing the aforementioned information are the image itself, the URL of both the web page and the image¹⁵ and the page title.



Figure 11- URL and screenshot of the web page.

3.5.3.2 AMOS dataset

In this section, we present an example of an indicative page from the AMOS dataset. The page holds information of an example camera. The URL that holds information of each camera is generally of the following form:

<http://amos.cse.wustl.edu/camera?id=<Camera Id>>

In order to study the content of such page, we select randomly one camera id. Thus, for the camera with id=65, the URL is following: <http://amos.cse.wustl.edu/camera?id=65> and the page's screenshot is depicted in Figure 12.

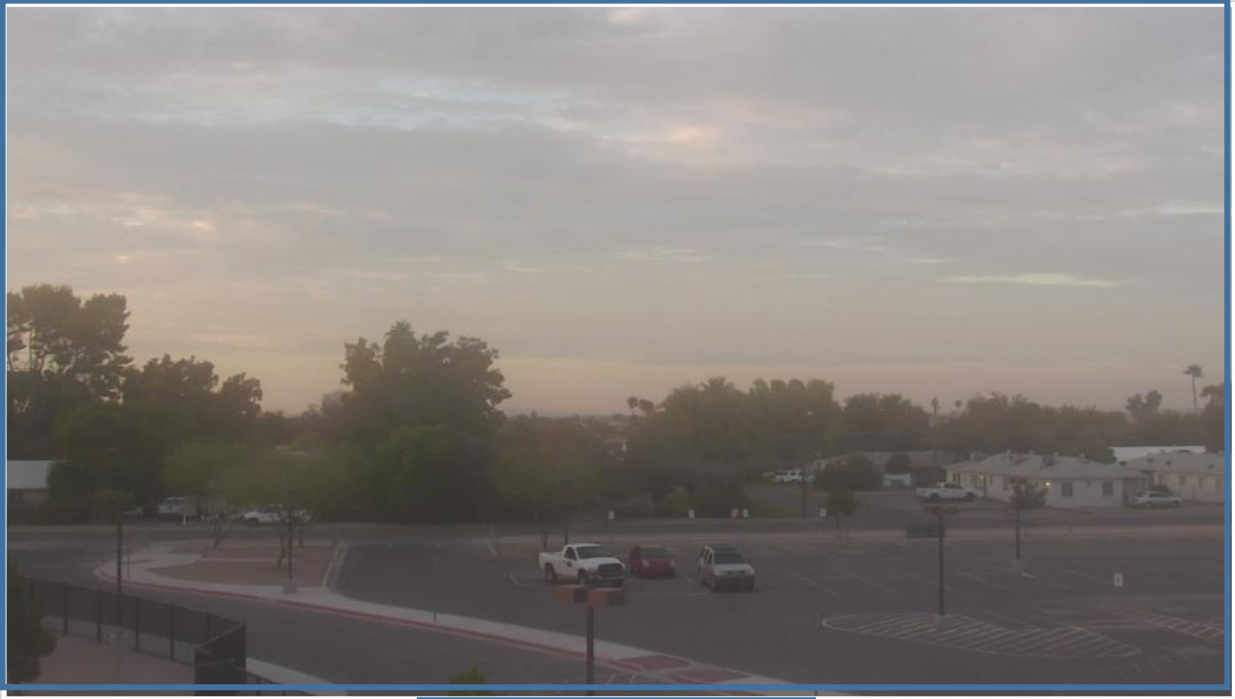
¹⁴ See Appendix 9.3.1 for more detailed description of the site containing the webcam. The Serial Number of the site is S/N=12.

¹⁵ <http://www.ndr.de/public/webcams/wetter/hannover.jpg>

D3.1: Environmental node discovery, indexing and data acquisition

The information that is of interest for the AMOS dataset is the camera id, the image captured, the date/time of the image and the geolocation of the camera. The image can be found in a central place of the webpage and the date and time information is directly below it. Moreover, the camera id is retrieved from the page URL and finally the geolocation of the camera can be found at the bottom left corner of the page (highlighted bounding box).


« Previous **Coolidge High School** Next »



Wed Sep 21 2016 16:33:22 GMT+0300 (GTB Daylight Time)


[Compare this image to an image from a previous day.](#)
[Analyze GCC score!](#)

« Back 2016 Forward » RGB PCA Error



Day of Year → Time of Day ↓

Geolocation Map



Camera Information

Name: Coolidge High School
Date Added: Jul 31, 2008 at 19:37:43 UTC
Last Captured: Sep 21, 2016 at 13:33:22 UTC
Next Scheduled Capture Time: 23 minutes 48 seconds.
Active: ✓
Tags: buildings crosswalk geoValid google highway
google highway google morning google morning
google vehicle google vehicle google _filhaze
grass neverShifts nightVision personal
possibly stable rockSolid sky trafficlight trees
Dimensions: 1280 x 720 pixels
tags, to, add
★★★★★ (Current rating: 5.0)

Geolocation Information

United States
● Known Location: (44.726403726, 16.6475062663)
● IP Location: (45.4116, -75.6982)

Image Information

Tags:
tags, to, add

Figure 12 – Web page from the AMOS dataset pointing to a specific webcam.

4 Data Gathering

In this section, we present the methods used for gathering data from the different sources. Specifically, as far as the web sites are concerned we apply domain specific web search and crawling techniques, i.e. we develop a focused crawler that is oriented towards finding sites with PM measurements. Regarding social media platforms, we present the collector that is developed for collecting data from Flickr, and regarding web services no discovery techniques are applied since not many public web services are available (based on the analysis realized in Section 3.3).

Finally, as far as webcams are concerned, we plan on developing an automatic discovery method for discovering sites with webcams. However, this will be developed at a later stage and the results will be presented on the upcoming deliverable D3.2 which will be submitted on month 18 of the project. In this deliverable, we present only the method for retrieving data from the AMOS dataset, which constitutes a good starting point for the planned experiments.

4.1 Discovery of environmental web sites

Building on the outcomes of the empirical studies of section 3 and additionally using domain specific web search and crawling techniques, we expand the list of sources provided by the experts. Domain specific search refers to the automatic discovery of Web resources related to a given domain. The resulting domain-specific search engines, also known as *vertical search engines*, attempt to index only web pages that correspond to a pre-defined area, publisher, topicality, media type or purpose. In general, the area of domain specific search can be considered as a well-established research area since a considerable number of techniques have been developed to tackle this issue. The main methodologies used for the implementation of a domain specific search engine contain several techniques such as web searching, crawling and query expansion techniques. In the sequel, we present an overview of the techniques proposed for domain specific search, the proposed framework for discovering environment related web sites, the evaluation results and a first set of links that are discovered with the help of the framework.

4.1.1 State of the art for domain-specific web search

Domain-specific search is mainly addressed by two categories of techniques: (i) domain-specific query submission to a general-purpose search engine followed by post-retrieval filtering, and (ii) focused crawling. Regarding the environmental domain, existing efforts are based on techniques from the first category (Moumtzidou, 2013; Moumtzidou, 2012), while techniques from the second category are rarely proposed.

The approach employed for the environmental domain that is relevant for hackAIR involves the generation of domain specific queries which are generated using empirical information, including the incorporation of geographical terms (Moumtzidou, 2013), and expanded using ‘keyword spices’ (Oyama, 2004). Then a post-processing step is performed using Support Vector Machines (SVMs) (Moumtzidou, 2012). Such approaches are complementary to the discovery of web resources using focused crawlers.

Focused crawling is a technique that has been researched since the early days of the Web (De Bra, 1994). Based on the ‘topical locality’ observation that most Web pages link to other pages that are similar in content (Davison, 2000), focused crawlers try to estimate the benefit of following a hyperlink extracted from an already downloaded page by mainly exploiting the web graph structure. The majority of such crawlers (Chakrabarti, 1999; De Bra, 1994; Olston, 2010; Pant, 2005; Pant, 2006) use the text around a given hyperlink in one form or another. The results of the study conducted by Pant (2006) showed that crawling techniques that exploit terms both in the immediate vicinity of a hyperlink, as well as in its entire parent page, perform significantly better than those depending on just one of them.

Early focused crawlers, e.g. (Cho, 1998), estimated the relevance of the hyperlinks pointing to unvisited pages by computing the textual similarity of the hyperlinks’ local context to a query. Recent focused crawlers though, use supervised machine learning methods to decide whether a hyperlink is likely to lead to a web page on the topic or not (Olston, 2010). Finally, the studies realized regarding the efficiency of different classification algorithms showed that SVMs are a better choice than Neural Network-based, while Naive Bayes perform rather poorly.

4.1.2 Framework

Focused crawlers use the web graph structure and follow only the hyperlinks that lead them to resources relevant to the topic of interest, by considering two sources of evidence: (i) the local context of hyperlinks, typically the textual content appearing in their vicinity within their parent page, and (ii) the global context of hyperlinks, typically associated with the entire parent page. Although existing focused crawling approaches have employed textual evidence for estimating the relevance of the global context of hyperlinks (Olston, 2010; Pant, 2006), the presented method exploits the frequent occurrence of images on the web during the link selection process, by combining the textual evidence (local context) with visual evidence (global context). This novelty is achieved by the late fusion of textual and visual classification confidence scores obtained by supervised machine learning methods. The application domain includes environmental web resources about air quality measurements, commonly encoded as heatmaps, i.e. graphical illustration of pollutant concentrations over geographic regions.

An overview of the proposed focused crawling method (Tsirikika, 2016) is depicted in Figure 13. Given an initial set of seed URLs, pages are picked in an iterative manner and are parsed to retrieve their hyperlinks and their embedded images. The relevance of each hyperlink h is estimated as $f_s(h) = a_1 \times score_T(h) + a_2 \times score_V(parent(h))$, where $score_T(h)$ is the relevance of the textual feature vector of h to the topic, $score_V(parent(h))$ is considering the most relevant image within the parent page of h , and a_1, a_2 are parameters that can be adjusted to express different strategies over the link selection:

- $a_1 = 1$ and $a_2 = 0$: $f'_s(h) = score_T(h)$; a hyperlink h is selected if its text-based classification confidence score is over a threshold t_1 , irrespectively of multimedia,
- $a_1 = 0$ and $a_2 = 1$: $f''_s(h) = score_V(parent(h))$; a hyperlink h is selected if its parent page contains at least one relevant image (score is 1 or 0, due to binary image classifiers), and
- $a_1 = score_V(parent(h))$ and $a_2 = 0$: $f'''_s(h) = f'_s(h) * f''_s(h)$; this approach selects all hyperlinks h with $score_T(h) \geq t_1$, within pages containing at least one relevant image.

Moreover, link exploration strategies can be applied to address the issue where a hyperlink is ignored as irrelevant, despite leading to highly relevant hyperlinks. When the focused crawler encounters a hyperlink pointing to a page with estimated relevance $f_s(h) < t_1$, an additional threshold $t_2 < t_1$ is set to select the links to be explored. Equivalently to the link selection approaches, the following exploration strategies are proposed: (i) $f'_E(h) = score_T(h)$, (ii) $f''_E(h) = score_V(parent(h))$, and (iii) $f'''_E(h) = f'_E(h) * f''_E(h)$.

Regarding $score_T(h)$, the local context of a hyperlink is defined as its anchor text a and a text window s of 50 characters, and is evaluated using a supervised machine learning approach based on SVMs. In the training phase, 711 samples (100 positive, 611 negative) were collected from 10 pages about air quality measurements, selected from the empirical study of Section 3.2, so as to build a vocabulary for representing the samples in the textual feature space and also for training the model. The vocabulary is built upon the $a + s$ context of the positive samples, generating a lexicon of 207 terms (10 most frequent: days, ozone, air, data, quality, today, forecast, yesterday, raw, and current), used then in the *tf-idf* weighting scheme to represent all samples. The SVM classifier is built using an RBF kernel and 5-fold cross-validation is performed on the training set to select the class weight parameters. On the other hand, $score_V(parent(h))$ is evaluated by training an SVM classifier with an RBF kernel for MPEG-7 features that capture color and texture, while both text and visual classifiers are implemented based on the LIBSVM library (Chang, 2011). The implementation of the focused crawler is based on Apache Nutch¹⁶, a highly extensible and scalable open source web crawler software project, by modifying its parser to function as a focused crawler.

¹⁶ <http://nutch.apache.org/>

D3.1: Environmental node discovery, indexing and data acquisition

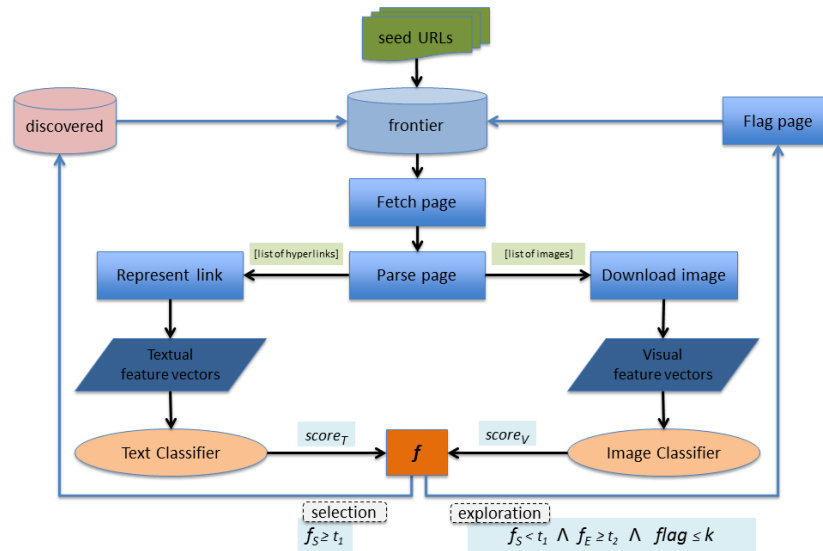


Figure 13 - Focused crawling based on the combination of multimedia evidence

4.1.3 Evaluation

The goal of the following series of experiments is to evaluate the effectiveness of the proposed focused crawling approach, involving the outcome of the empirical study conducted by domain experts in the context of the project hackAIR. Table 16 lists the seed URLs used in the experiments; they are different to the ones used for training the classifiers, while half of them contain at least one heatmap. A relatively small seed set is employed so as to keep the evaluation tractable while investigating larger crawling depths. Starting from these six seeds, crawls at depth $\in \{1, 2, 3\}$ and for t_1 values ranging from 0.1 to 0.9 at step 0.1 were performed in June 2016, fetching a total of 444, 3,912 and 19,564 unique pages, respectively to depth. Annotation of the Web pages pointed to be the extracted hyperlinks was manually performed using the following three-point relevance scale:

- **(highly) relevant:** web resources that provide air quality measurements or/and forecasts;
- **partially relevant:** web resources about air quality measurements and forecasts, but with no actual data;
- **non-relevant:** web resources that are not relevant to air quality measurements and forecasts, including resources that discuss air quality and pollution in general, e.g., its causes and effects.

Given that most performance metrics require binary relevance judgements, these multiple grade relevance assessments are also mapped onto the two dimensional space: (i) strict, when considering only highly relevant Web resources as relevant, while the rest (partially relevant and non-relevant) as non-relevant, and (ii) lenient, when considering both highly relevant and partially relevant Web resources as relevant.

The standard retrieval evaluation metrics of precision and recall, as well as their harmonic mean, the F1-measure, were applied for assessing the effectiveness of the proposed focused crawler. Precision corresponds to the proportion of fetched pages that are relevant and recall to the proportion of all relevant pages that are fetched. Since the latter requires the impossible knowledge of all relevant web pages on a given topic, the recall base of the experiments is defined as the relevant pages crawled by the most comprehensive link selection approach, i.e., $f_S(h)$.

Table 16 – List of seed URLs

	URL	Heatmap present
1.	http://aircarecolorado.com/	
2.	http://db.eurad.uni-koeln.de/en/	✓
3.	http://gems.ecmwf.int/	✓
4.	http://uk-air.defra.gov.uk/	

D3.1: Environmental node discovery, indexing and data acquisition

The first part of the experiments evaluates the effectiveness of the text-based focused crawler (i.e., the one that employs the $f'_S(\mathbf{h})$ link selection method) at depth $\in \{1, 2, 3\}$, when applying strict or lenient relevance assessments, and the results are depicted in Figure 14. The precision achieved is comparable to that of state-of-the-art text-based classifier-guided focused crawling approaches that start from a similar (or larger) number of seeds and crawl a similar number of pages, while employing SVMs as their classification scheme (Pant, 2005). The F1-measure also achieves similar results. As expected, the absolute values for both effectiveness metrics are much higher in the lenient case, compared to the strict. Furthermore, an indication of the accuracy of the underlying text-based link classifier can be obtained by considering the hyperlinks selected at depth = 1 (67 hyperlinks: 29 positive and 38 negative) and is close to 0.8. The highest overall precision for each depth value is achieved for $t_1 = 0.2$ for the case of lenient relevance assessments, while the results are mixed for the strict case, where the best results are observed for $t_1 = 0.4$, $t_1 = 0.7$, and $t_1 = 0.2$ for depth $\in \{1, 2, 3\}$, respectively. For the F1-measure, the best results are observed for $t_1 = 0.2$ for both lenient and strict relevance assessments. The usage of visual evidence (in the form of heatmaps) as global context shows significant improvement in combination with the textual evidence (i.e., the multimedia-based $f_S(\mathbf{h})$ link selection method), but the strictly visual-based $f''_S(\mathbf{h})$ method does not achieve better performance (results not shown).

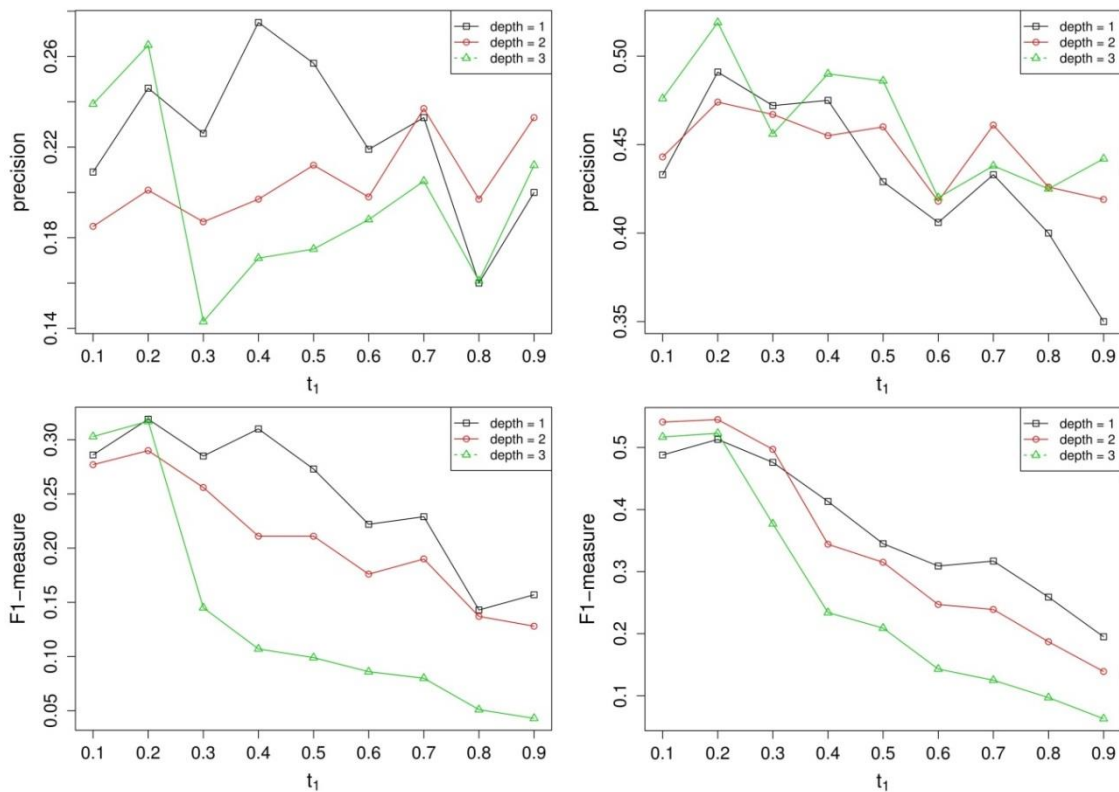


Figure 14 - Precision (top) and F1-measure (bottom) of the focused crawl for the $f'_S(\mathbf{h})$ link selection method at depth $\in \{1,2,3\}$ for threshold $t_1 \in \{0.1, \dots, 0.9\}$ when strict relevance assessments (left) and lenient relevance assessments (right) are employed

The second part of the evaluation refers to the link exploration strategies, where experiments are performed for depth = 3, since the application of such strategies is only meaningful for larger crawling depths. Figure 15 presents the precision of the focused crawler that employs the text-based $f'_S(\mathbf{h})$ link selection method in conjunction with the $f'_E(\mathbf{h})$, $f''_E(\mathbf{h})$, and $f'''_E(\mathbf{h})$ exploration strategies, when employing lenient relevance assessments; t_1 and t_2 values range from 0.1 to 0.9 at step 0.1, while maintaining $t_2 < t_1$. These are compared against a baseline corresponding to the best performing text-based focused crawling approach over all t_1 values.

Improvements in the effectiveness are observed only for the text-based exploration strategy $f'_E(\mathbf{h})$ and for low values of t_2 ($t_2 \leq 0.2$) across all t_1 values (apart from $t_1 = 0.2$). Moreover, in all these cases, the precision improves, as t_1 increases, leading to significant improvements over the baseline for some configurations of t_1, t_2 ; e.g., for $t_1 = 0.9$ and $t_2 = 0.2$, the improvement is 17% over the best performing text-based focused crawler (0.609 vs. 0.519) and reaches 38 % when compared against the text-based focused crawler for the same $t_1 = 0.9$ (0.609 vs. 0.442). Results for the case of strict relevance and the F1-measure metric are both skipped, since similar trends are

D3.1: Environmental node discovery, indexing and data acquisition

observed. The evaluation indicates that the text-based link exploration strategy in conjunction with the text-based link selection method is beneficial, as it ensures that web pages classified with a high degree of confidence are fetched (by using high t_1 values), while providing the freedom to remove potential “blocks” in the trails towards such pages (through appropriate t_2 values). Likewise, the application of the multimedia focused crawler $f'_S(h)$ in conjunction with the $f'_E(h)$, $f''_E(h)$, and $f'''_E(h)$ exploration strategies appears to be beneficial.

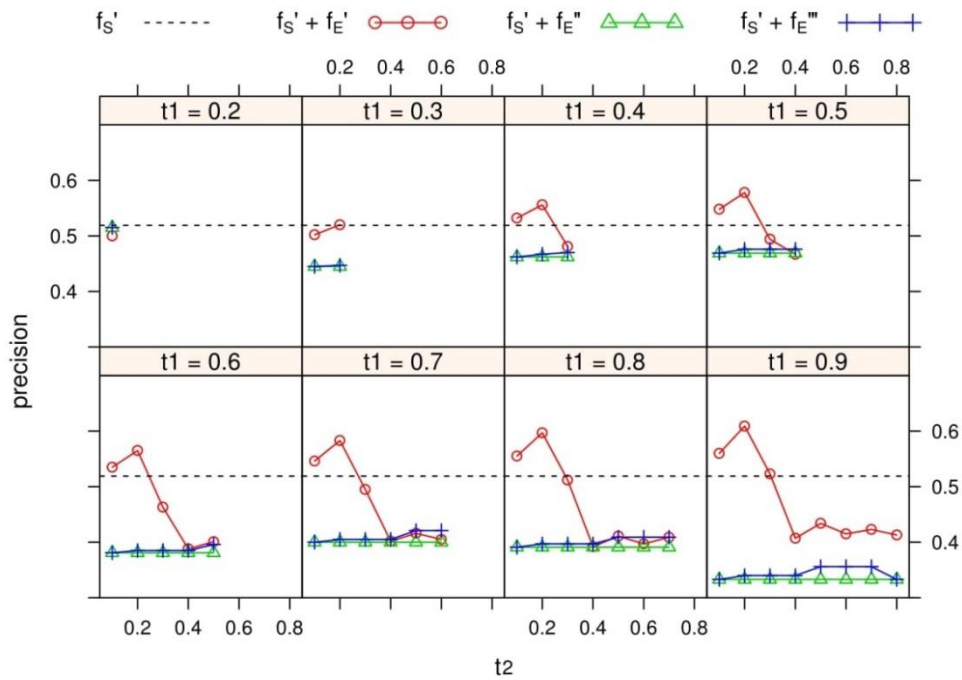


Figure 15 - Precision of the $f'_E(h)$, $f''_E(h)$, and $f'''_E(h)$ exploration strategies by a focused crawler that employs the $f'_S(h)$ text-based link selection method at depth = 3 for thresholds $t_1, t_2 \in \{0.1, \dots, 0.9\}$, s.t. $t_2 < t_1$, when lenient relevance assessments are employed.

4.1.4 Results

In this section, we present a short list of URLs retrieved after running the developed focused crawler. In order to limit the crawler to the sites of our interest and also to obtain a reasonable set of results quickly we used the following crawl parameters:

- *heatmap = false*: this parameter receives a Boolean value and it was set to false given that heatmaps are usually found in web pages containing forecast measurements, which are out of the scope for hackAIR.
- *t1 = 0.6*: this parameter defines the threshold of the value of the link-based classifier
- *depth = 2*: this parameter defines how deep the Nutch crawler should crawl.

Table 17 contains the set of URLs used as a seed list by the crawler. The command that was used is the following:

```
./bin/nutch crawl seed_urls -depth 2 -modelFile air_quality_svm_default_2.model -t1 0.6 -heatmap false
```

Table 17 – URL Seed list for retrieving sites with air quality measurements

URL	
1.	http://www.airtext.info/
3.	http://aqicn.org/city/
4.	http://www.irceline.be/en/
5.	http://www.airqualitynow.eu/index.php
6.	http://macc-raq-op.meteo.fr/index.php

D3.1: Environmental node discovery, indexing and data acquisition

7. <http://www.ypeka.gr/Default.aspx?tabid=708&locale=en-US&language=el-GR>

The results are depicted in Table 18, which contains the URLs of the discovered pages and their classification scores.

Table 18 – Web pages discovered by the focused crawler and their classification scores.

URL	Classification score
http://aqicn.org/forecast/europe	0.9999999999999699
http://aqicn.org/forecast	0.9999999999978254
http://aqicn.org/forecast/world	0.999999999990522
http://www.airqualitynow.eu/pollution_home.php	0.9999987811036768
http://www.airtext.info/alerts	0.9936403118901396
http://aqicn.org/city/beijing/m	0.9647315660546448
http://www.cerc.co.uk/environmental-software/ADMS-Forecast.html	0.9608465146411661
http://www.airqualitynow.eu	0.9577701380381032
http://www.airqualitynow.eu/comparing_home.php	0.9459896758647054
http://www.cerc.co.uk/environmental-software/prices.php	0.9285865716130299
http://www.airtext.info/health	0.9159369954244506
http://macc-raq-op.meteo.fr/?category=data_access&subensemble=dataserver_services	0.9002924267882652
http://aqicn.org/nearest	0.8682327323658214
http://www.airqualitynow.eu/about_home.php	0.8520431363729027
http://aqicn.org/map/world	0.7898312980281005
http://www.londonair.org.uk	0.6004213861392532

4.2 Collecting Images from Social Media Platforms

In this section, we present the Flickr collector for collecting public images users upload to their profiles that fulfill certain geographical and time criteria, and also the image downloader which is used for downloading all images provided by the Flickr collector in an efficient manner. All the desired information (e.g. image, upload time, coordinates, text, etc.) can be extracted from the API response, which is serialized in JSON.

4.2.1 Flickr collector

The Flickr API¹⁷ is used for collecting public images users upload to their Flickr accounts that are located near a selected point. For each city of the dataset cited in Section 3.2.1, we define its center and a radius around it. The employed call to the Flickr's API is the following:

https://api.flickr.com/services/rest/?method=<API endpoint>&media=<Media type>&extras=<Extra information>&api_key=<Key>&format=<Result format>&nojsoncallback=<Function wrapper>&page=<Page Number>&per_page=<Number of results per page>&lat=<Latitude>&lon=<Longitude>&radius=<Radius>&radius_units=<Radius unit measure>km&min_taken_date=<Minimum timestamp>&max_taken_date=<Maximum timestamp>

and the main parameters that should be filled are the following:

¹⁷ <https://www.flickr.com/services/api/>

D3.1: Environmental node discovery, indexing and data acquisition

- **method:** This points to the API's endpoint.
- **media:** This filters results by media type. Possible values are: all (default), photos or videos.
- **extras:** A comma-delimited list of information to fetch for each photo, including: description, license, date_upload, date_taken, owner_name, icon_server, original_format, last_update, geo, tags, machine_tags, o_dims, views, media, path_alias, url_sq, url_t, url_s, url_q, url_m, url_n, url_z, url_c, url_l, url_o
- **api_key:** In order to use the API, a key that is tied to a user name is required. This key is used for authorizing the API calls and thus tracking its usage. The purpose of having a key is to limit the access to the API and thus prevent abuse.
- **format:** This is the format of response. Possible values are: xml (default) or json.
- **nojsoncallback:** To return raw JSON with no function wrapper. It is a Boolean value with values either 0 or 1.
- **page:** The page of results that will be returned. It is an incremental number.
- **per_page:** It identifies the number of results returned per page. The maximum value is 250.
- **lat:** Latitude of the center search coordinate.
- **lon:** Longitude of the center search coordinate.
- **radius:** The search radius around the center point set in *lon*, *lat* parameters. At this point, it should be noted that the radius parameter can take values up to 32 kilometers.
- **radius_units:** The unit of measure for radius.
- **min_taken_date:** This is to collect images with upload date (expressed in timestamp) later than this value.
- **max_taken_date:** This is to collect images with upload date (expressed in timestamp) earlier than this value.

The standard values that we use in the aforementioned parameters are the following:

1. `method=flickr.photos.search`
2. `media=image` (used for retrieving only photos)
3. `extras= geo,description,date_taken,owner_name,views,url_l` (used for adding to the response information regarding location, text that comes with the image, date taken, username of owner, views in Flickr and URL of the image respectively)
4. `api_key=XXX`
5. `format=json`
6. `nojsoncallback=1`
7. `per_page=250` (use the maximum available number)
8. `radius= 16` (usually even the bigger cities do not have radius over 16km)
9. `radius_units= km`

The aforementioned values remain static throughout all the calls to the API. From the following ones, certain change when a new call is made for the same city while others change for each city.

Specifically, the parameters changing per call for the same city are:

1. `page= Desired page`
2. `max_taken_date= Current timestamp`
3. `min_taken_date= max_taken_date-86400` (24 hours in seconds for retrieving the last days photos)

While, the parameters that change when a new city is target are the following

- `lat= Latitude of the city center.`
- `lon= Longitude of the city center.`

By considering all the aforementioned parameters, we formulate a sample call of the Flickr API:

```
https://api.flickr.com/services/rest/?method=flickr.photos.search&media=photos&extras=geo,description,date\_taken,owner\_name,views,url\_l&api\_key=a5609f6d69e\*\*\*\*\*9d4d37dc0e&format=json&nojsoncallback=1&page=1&per\_page=250&lat=53.480759&lon=-2.242631&radius=16&radius\_units=km&min\_taken\_date=1473033600&max\_taken\_date=1473120000
```

A sample response from Flickr API is the following:

```

{
  "photos": {
    "page": 1,
    "pages": 1,
    "perpage": 250,
    "total": "20",
    "photo": [
      {
        "id": "28849659653",
        "owner": "23621946@N00",
        "secret": "256c07b991",
        "server": "8161",
        "farm": 9,
        "title": "\u266b Big wheels keep on turning \u266b",
        "ispublic": 1,
        "isfriend": 0,
        "isfamily": 0,
        "description": {
          "_content": "New Cathedral street, Manchester."
        },
        "datetaken": "2016-09-05 10:39:07",
        "datetakengravity": "0",
        "datetakenunknown": "0",
        "ownername": "tootdood",
        "views": "337",
        "latitude": "53.483723",
        "longitude": "-2.244712",
        "accuracy": "15",
        "context": 0,
        "geo_is_family": 0,
        "geo_is_friend": 0,
        "geo_is_contact": 0,
        "geo_is_public": 1,
        "url_l":
"https://farm9.staticflickr.com/8161/28849659653_256c07b991_b.jpg",
        "height_l": "1024",
        "width_l": "683"
      }
    ]
  },
  "stat": "ok"
}

```

All the desired information (e.g., image, upload time, coordinates, text, etc.) can be extracted from the above JSON using a JSON parser. Eventually, all collected information is unique (check for duplicate images is implemented) and is stored in a repository. The repository that is selected for storing the data is a MongoDB. The selection of MongoDB is justified in Section 6.2. This process is followed for every city on a daily basis (24 hours). It should be mentioned that there is a limit of 3,600 queries per hour that can be submitted to the API. In case of abusing the service, Flickr either expires the used key or turns the service temporarily off for the particular user.

4.2.2 Image Downloader

After the information of the API is stored in a repository, we download the images locally. Since the process of image downloading takes longer (compared to downloading metadata), especially for high resolution images, this is not done serially. To this end, we use multithreading, with each thread downloading an image independently. The number of initial threads varies and is based on the machine specifications. Eventually, we end up with using 50 threads and we achieve a rate of approximately 1,000 images per minute, which is considerably faster compared to the average 17 images per minute that was achieved when images were downloaded in a serial manner.

D3.1: Environmental node discovery, indexing and data acquisition

Similar to the Flickr collector, the Image Downloader runs periodically every 24 hours. Specifically, when the collector finishes with the requests to the suitable API, and stores all data to the repository, the downloader checks the repository for all the images that have not been downloaded yet and creates a list with their URLs. Each thread then tries to download an image from the given URL. As soon as the image is downloaded it takes the next available URL (if any) from the list.

4.3 Collecting Images from the AMOS dataset

The creators of the AMOS dataset have developed a REST service, which allows open access to all images retrieved from the indexed webcams. In this section, we present an example of how to call the AMOS REST web service for downloading images. The query is focused on retrieving cameras found in the city of Berlin and is realized through a python script, called *download_amos.py*, that is provided by the people gathering and preserving the AMOS dataset.

In order to gather data from a specific set of cameras, there is a set of variables that need to be edited inside the python script, i.e. `CAMERAS_TO_DOWNLOAD`, `YEARS_TO_DOWNLOAD`, and `MONTHS_TO_DOWNLOAD`. While the variables `YEARS_TO_DOWNLOAD`, and `MONTHS_TO_DOWNLOAD` receives pretty straightforward values, in order to fill the variable `CAMERAS_TO_DOWNLOAD`, it is necessary to find and check all the cameras located in Berlin. In order to provide such information, the site hosting the AMOS dataset provides the possibility of browsing through its cameras by allowing advanced search, where the user can specify the latitude and longitude range of the cameras. The bounding box around Berlin can be defined by the following values:

- Latitude: [52.323774, 52.690990]
- Longitude: [13.085535, 13.747448]

The corresponding query that is submitted for retrieving the webcams in Berlin is the following:

http://amos.cse.wustl.edu/browse_with_filters?search=&id_0=&id_1=&id_list=&latitude_0=52.323774&latitude_1=52.690990+&longitude_0=13.085535&longitude_1=13.747448&images_captured_0=&images_captured_1=&years_captured_0=&years_captured_1=&land_use=&o=-rating_rating_average

The search returns a set of six cameras, two of which are not functioning for some years. The ids of the cameras are shown below:

- 5783
- 7239
- 7980 (not working since 2010)
- 9478 (not working since 2011)
- 19396
- 24112

After filling in all the variables into the python script, it starts downloading the images that fulfill the camera ids, year and month criteria. We should also note that each image is named after the date and time that it was captured. In order to extract the exact location of each camera, as we have already mentioned in Section 3.5.1.2, there is a page holding the information of each one that holds additional information regarding each camera as well its geolocation.

5 Data Analysis

This section includes the data analysis techniques applied on the different types of data. The text-based sources include web sites and web services that provide PM measurements. As far as the web sites are concerned, the data is provided in unstructured format and thus web information extraction techniques are required to extract the target information (i.e. value of pollutant, pollutant name, time/date and location). In the case of web services, the data is provided in structured format and thus simple JSON/XML parsing is required. For the image-based sources that include images either coming from social media, the mobile app or the webcams, image analysis techniques are foreseen that recognize images containing a part of the sky and the part of the image with the sky that fulfils certain criteria. The results of the analysis are sent to the AQ Estimation service that is part of deliverable D3.3 to provide an air quality estimate (i.e. PM) according to an air quality index (e.g. low, medium, high).

5.1 Text-based Sources

This module involves the extraction of environmental content from web sites and services. In case of web services the format of the data is well defined and data can be retrieved by simple JSON/XML parsing. However, in the case of web sites, we apply Web Data Extraction techniques that tackle the problem of extracting data from Web sources and allow collection of large amounts of data from the web.

5.1.1 Web services

Web services as already mentioned in Section 3.3.1, provide a standard means for interoperating between different software applications which may run on a variety of platforms. The most commonly used methods for providing web services are SOAP, XML-RPC and REST. However, regardless of the method used, the most common formats of representation (data serialization) are XML and JSON. Thus, the extraction of data from web services is realized using either JSON or XML parsers.

5.1.1.1 Web service example

The web service used as example here, i.e. <https://openaq.org/#/>, was presented earlier in the empirical study section. In the documentation section regarding the API, its parameters and the supported functions are explained. The service is available for only a limited number of countries dispersed around the world. Based on the hackAIR cities of interest, our interest focuses specifically on the cities of Amsterdam and London. Neither Germany nor Norway are among the countries covered by the current version of the system. Based on the guidelines provided, the HTTP GET query for retrieving PM₁₀ measurements for Amsterdam is the following:

<https://api.openaq.org/v1/measurements?city=Amsterdam¶meter=pm10>

Figure 5 depicts the JSON response, which provides some general information, e.g. the number of total results and the current page CSV file, and some information for each measurement including the location, the coordinates, the date and the pollutant value. Thus, an example from the first page is the following:

- country : "NL"
- city : "Amsterdam"
- location : "Amsterdam-A10 west"
- parameter : "pm10"
- date UTC : "2016-09-15T11:00:00.000Z"
- value : 46.456666666666666
- unit : "µg/m³"
- coordinates: latitude : 52.3395, longitude : 4.84102

5.1.2 Web data extraction

The extraction of data from web sources is realized by Web Data Extraction (WDE) techniques. WDE techniques allow collecting large amounts of data from the Web and are used in a wide range of research fields. Web Data Extraction tools find use in a wide range of research fields, from business intelligence (Baumgartner, 2005), and social media (Catanese, 2011), to bio-informatics (Plake, 2006). To extract data from a particular web source, a web wrapper needs to be created, which is a procedure that seeks and finds the “interesting” data as defined by the human users, extracts it and transforms it to structured data in a semi-automatic or fully automatic way (Ferrara, 2014). Web wrapper building attracted the interest of many researchers and as a consequence many studies were conducted in the field of WDE. In general, building manually Web wrappers is impractical and time consuming. Thus, researchers have focused in the task of wrapper induction, which refers to the automatic wrapper generation by learning extraction rules from a set of manually labeled HTML documents. The first algorithms that were designed for attacking the problem of wrapper induction were supervised.

The first algorithm was introduced by Kushmerick (1997), which required a set of labeled instances based on which a hypothesis (wrapper) is constructed. Moreover in order to ensure the quality of the produced wrapper, PAC analysis is used. Using as a starting point the work of Kushmerick, several researchers studied the problem of supervised wrapper induction (Muslea, 1998; Hsu, 1998, Wong, 2010). However, gradually and in order to eliminate the main downfall of supervised wrapper induction, which is the human effort required during the labeling phase, several unsupervised were proposed. Such works include the system RoadRunner proposed by Crescenzi (2001) and the DeLa system proposed by Wang (2003).

Although many of the aforementioned methodologies achieve high performance, there are cases that they fail or are unable to extract important pieces of information. In order to eliminate any problems of misunderstanding or information overlooking, we opted for a semi-automatic data extraction framework that achieves 100% of precision and recall. The employed system is called easIE (easy Information Extraction). It was initially developed for the WikiRate EU Project (Papadopoulos, 2015) and was inspired by the Ducky system (Kanaoka, 2014) where data extraction is implemented by means of a set of rules that are serialized in a configuration file. Compared to Ducky, easIE also supports the information extraction from dynamic HTML pages. For the needs of hackAIR, we applied easIE by generating appropriate configuration files as will be explained in the following.

5.1.2.1 easIE: easy Information Extraction framework

Implementing a WDE tool involves overcoming a number of issues. Specifically, the tool should be general enough in order to be easily applied to different domains, it should reduce the user effort as much as possible by providing a high degree of automation, and finally it should be easy to use and modify by users with little programming skills. In general, easIE supports the extraction of data both from static and dynamic HTML pages. HTML (HyperText Markup Language) is the standard markup language for describing and creating Web pages by annotating content using tags. Eventually, the structure of an HTML page corresponds to a tree. Therefore, when browsers load a Web page, they create the corresponding Document Object Model¹⁸ (DOM), consisting of a tree of Objects as specified in the HTML document. The tree structure of HTML pages is exploited by many WDE methods (Dalvi, 2009). The resulting DOM can be used to extract specific elements in an HTML document by exploiting XPath Selectors and CSS Selectors¹⁹. Selectors are patterns that address elements in a tree. XPath²⁰ is a language based on a tree representation of the XML or HTML document, and provides the ability to navigate around the tree, selecting nodes by a variety of criteria. Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. CSS uses Selectors for assigning style properties in the elements of a document.

¹⁸ <http://www.w3.org/DOM/>

¹⁹ <http://www.w3.org/TR/css3-selectors/>

²⁰ <http://www.w3.org/TR/xpath20/>

D3.1: Environmental node discovery, indexing and data acquisition

The easIE framework supports easy generation of wrappers for a web source based on a set of data extraction rules that can be set in a user-defined configuration file. The extracted data are stored in mongoDB or in the hard drive in JSON or CSV format. Initially and in the context of the WikiRate project, easIE was developed to perform information extraction about companies; however, it was applied with the same success and without any adjustments to the domain of hackAIR that is web pages containing environmental measurements. This is due to the flexibility of the tool and the fact that is mainly based on the configuration files that are general enough to cover different domains. In the sequel, we describe the architecture of the developed tool and the configuration file for generating custom web wrappers for a source.

5.1.2.1.1 easIE architecture

easIE is responsible for creating automatically wrappers for the requested source based on user input. Figure 16 depicts the architecture of the tool. The input of the tool is a configuration file in JSON Format, and the tool consists of four components:

- **JSON Parser:** it parses the configuration file into Java Objects that are used by the Web wrapper generator.
- **Web wrapper generator:** is generates web wrappers, based on user preferences defined inside the configuration file. It consists of two sub-modules:
 - *Dynamic Page Web wrapper generator:* it generates a wrapper for a dynamic Web page. It first launches a browser emulator instance, then executes the specified events, and parses the HTML document. The procedure can be repeated multiple times.
 - *Static Page Web wrapper generator:* it generates a wrapper for a static Web page. If the document is divided in multiple pages, then a Pagination Iterator is created. Moreover, in case of a group of similarly structured pages, a Bunch URL Iterator is created, while the Static HTML Wrapper is responsible for extracting the desirable data from each page of the group.
- **Wrapper executor:** it executes the created web wrappers.
- **Data storage:** it stores the extracted data into mongoDB or to a file in the hard drive.

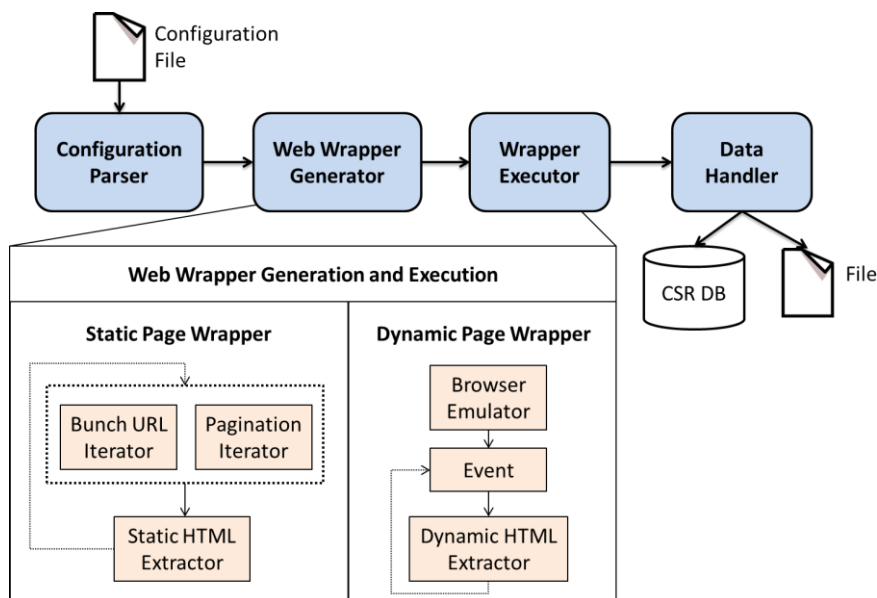


Figure 16 - Web extraction tool architecture

5.1.2.1.2 Configuration File Description

In this section, we briefly mention the main elements that can be found in the configuration file, while a detailed description of the elements and their description can be found in (Papadopoulos, 2015). The elements specify the type of data that is to be extracted, and the details of the steps that comprise the wrapper processing workflow (fetching, data extraction, post-processing, start-over). Therefore the main elements of the configuration file are: url,

D3.1: Environmental node discovery, indexing and data acquisition

source_name, unit_info, metrics, dynamic_page, events, list_selector, next_page_selector, group_of_urls, crawl, store. The workflow that is followed can be summarized into the following steps:

- The starting point for the wrapper is the url field that determines the seed Web page that the wrapper will first fetch in order to start the data collection process for the source under source name.
- An essential configuration element is the dynamic page that indicates whether the target page should be handled by the wrapper as static or dynamic. In case of static pages, the target data are available upon page load; instead, for dynamic pages the configuration file should also specify the user events that are required in order to load the necessary HTML snippets in the browser emulator. The types of event that can be handled are the Click and the Scroll Down.
- The data extraction process is initiated by pointing the wrapper to the DOM subtree specified by the list_selector_field.
- The wrapper, then, iteratively visits all the elements of the selected subtree and applies the data extraction rules specified by metrics and unit info that map parts of these elements to the respective structured fields.
- In case that the source of interest contains multiple pages with data, instead of having multiple similar configuration files with simply different URL field, the user needs to define an array of URLs to be collected by the same wrapper. Another multi-page data collection scenario pertains to paginated data. To enable paginated access to such data, one needs to specify the next page selector field by a CSS selector pointing to the element where the "next" button is located in the page.

Figure 17 depicts an example of the configuration file of a web page²¹ that contains PM measurements for German cities. Section 9.4, which is part of the Appendix, contains an example of more complicated configuration file that captures information from a site which contains PM measurements for Norwegian cities²².

²¹ <http://www.umweltbundesamt.de/en/data/current-concentrations-of-air-pollutants-in-germany>

²² <http://www.luftkvalitet.info/home/airquality.aspx?type=1&topic=1&id={03206c07-9d57-4af7-92eb-e66ee800dcac}>

D3.1: Environmental node discovery, indexing and data acquisition

```
{
  "url": {
    "base_url": "http://www.umweltbundesamt.de",
    "relative_url": "/en/data/current-concentrations-of-air-pollutants-in-germany"
  },
  "source_name": "Umweltbundesamt",
  "table_selector": "#transgression-data > tbody:nth-child(2) > tr",
  "unit_info": [
    {
      "label": "City",
      "value": {
        "selector": "td:nth-child(2)",
        "type": "text"
      }
    },
    {
      "label": "Date",
      "value": {
        "selector": "td:nth-child(6)",
        "type": "text"
      }
    }
  ],
  "metrics": [
    {
      "label": "PM10",
      "value": {
        "selector": "td:nth-child(3)",
        "type": "text"
      }
    }
  ],
  "dynamic_page": true,
  "events": {
    "selector": ".data-display > label:nth-child(6)",
    "times_to_repeat": 1,
    "extraction_type": "AFTER_ALL_EVENTS",
    "type": "CLICK"
  }
}
```

Figure 17 – Configuration file for extracting information from the www.umweltbundesamt.de web page.

5.1.2.1.3 easIE example

To make clear how easIE works, we apply it on an example web page from the empirical study of section 3.2. Figure 18 depicts a screenshot of the web page <http://aqicn.org/city/brussels/>, a zoom on the part of the page containing the information of interest, which is highlighted. Figure 19 depicts the configuration file that is created for extracting the target information of Figure 18, and finally, Figure 20 depicts the CSV file containing the extracted information. The first line of the file contains the title of each column and thus the information can be interpreted as follows:

D3.1: Environmental node discovery, indexing and data acquisition

- city =Brussels
- station = Brussels
- date =1473303600
- time = null
- Pollutant name = PM2.5
- Value = 59

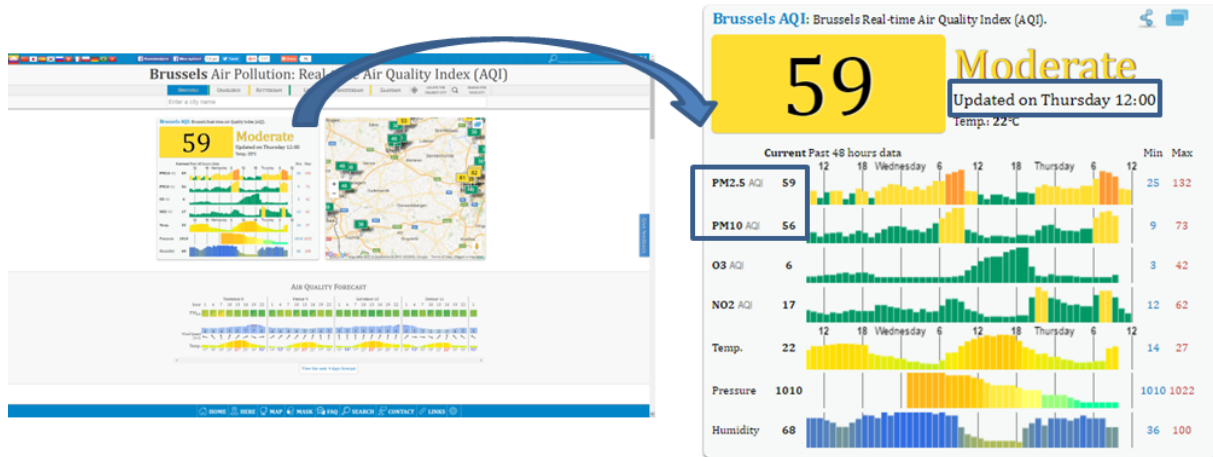


Figure 18 - Screenshot of <http://aqicn.org> web site for the city of Brussels.

```

{
  "url": {
    "base_url": "http://aqicn.org",
    "relative_url": "/city/brussels/"
  },
  "source_name": "aqicn_Belgium#Brussels",
  "company_info": [
    {
      "label": "City",
      "value": "Brussels"
    },
    {
      "label": "Station",
      "value": "Brussels"
    },
    {
      "label": "Date",
      "value": {
        "selector": "#citydivmain > div:nth-child(1) > div:nth-child(1) > div:nth-child(1) > table:nth-child(2) > tbody:nth-child(1) > tr:nth-child(1) > td:nth-child(2) > div:nth-child(2) > span:nth-child(1)",
        "type": "val"
      }
    }
  ],
  "metrics": [
    {
      "label": "PM2.5",
      "value": {
        "selector": "#cur_pm25",
        "type": "text"
      }
    },
    {
      "label": "PM10",
      "value": {
        "selector": "#cur_pm10",
        "type": "text"
      }
    }
  ],
  "dynamic_page": false
}

```

Figure 19 – Configuration file of easIE tool for <http://aqicn.org> web site

```

City, Station, Date, Time, Pollutant name, Value
Brussels, Brussels, 1473336000, null, PM2.5, 59
Brussels, Brussels, 1473336000, null, PM10, 56

```

Figure 20 – Output of easIE tool for the <http://aqicn.org> web site as CSV file

5.2 Image-based Sources

This section involves the processing of images in order to support air quality (i.e. PM) estimation based on their visual content. The techniques that are described are applied to images: 1) found in media sharing platforms such as Flickr, 2) captured from the mobile app and 3) found from webcams. The use of different sources aims at combining the advantages of each source. Specifically, the images found in media sharing platforms offer the advantage of high geographic coverage, while images captured from the mobile app have better quality and consistency since their capturing parameters will be controlled by the hackAIR mobile app. Finally, webcams are expected to provide constant information especially for city regions where they are usually placed. However, they typically have lower quality.

In order to process the image and retrieve the part of the image depicting a sky (which is of interest for the “Air Quality Estimation Service”), two approaches are studied. The first involves the use of visual concept detection and a sky localization algorithm that detects the position of sky in the image. The second approach is much simpler and it relies on heuristic rules that capture the characteristic features of images with a part of the sky. Eventually both approaches “translate” images and their sky parts to air quality. The results of both methods are the masks of the images that capture the sky part of the image. For this part, the R to G and G to B ratios are calculated. Both values are forwarded to the *AQ Estimation Service* (see Figure 1), that will provide along with other information the air quality estimation of the area captured by the image. Finally, it should be checked whether the images coming from webcams can provide “accurate” AQ estimations given that they usually have lower resolution compared to the images retrieved from the other sources. However, this will be presented in deliverable 3.2.

5.2.1 First method: Visual concept detection and localization

This method involves the use of visual concept detection algorithms based on low level features and classifiers for deciding whether an image contains substantial regions of sky, and a sky localization algorithm that detects the position of the sky in the image. In this section, we present an overview of state of the art methods for concept detection and localization, and then we present the framework proposed as well as some adjustments that could be realized in the next deliverable.

5.2.1.1 Concept detection state of the art

Concept detection in images aims at annotating them with one or more semantic concepts (e.g. hand, sky) that are chosen from a pre-defined concept list (Snoek, 2009). In general concept detection systems follow a process that first performs extraction of visual features, then training of classifiers for each concept using a ground-truth annotated training set, and finally, application of the trained classifiers to unlabeled images, after feature extraction is realized, that return a set of confidence scores for the appearance of the different concepts in the shot. Thus, the first step in concept detection is feature extraction and the second is the building of the classification model.

5.2.1.1.1 Feature Extraction

Feature extraction from images refers to methods that aim at the effective description of the visual content of images. Throughout the years, many descriptors have been introduced for representing various image features. These can be divided in two main groups: hand-crafted and DCNN-based descriptors. Given that DCNN-based features outperform the hand-crafted features in most applications, we will present the latter very briefly.

Hand-crafted features. These are further divided into global and local descriptors. Global descriptors capture global characteristics of the image. Some indicative descriptors are the MPEG-7 descriptors, the Grid Color Moments, and the Gabor Texture. Instead, local descriptors represent local salient points or regions and the most widely used are the SIFT descriptor (Lowe, 2004) and its extensions (e.g. HSV-SIFT, OpponentSIFT, RGB-SIFT), and the SURF descriptor (Bay et al., 2008) and its variations (e.g. dense-SURF). Usually, in the case of local descriptors a clustering algorithm is applied after the feature extraction in order to form a vocabulary of “visual words” that leads eventually to a global descriptor. The “bag-of-words” (BoW) representation (Qiu, 2002) is the most widely applied method for visual word

D3.1: Environmental node discovery, indexing and data acquisition

assignment. However, other improved approaches were proposed, including the Fisher vector (Perronnin et al., 2010) and the VLAD (Jegou et al., 2010).

DCNN-based features. The most recent trend in feature extraction and image representation is learning features directly from the raw image pixels using Deep Convolutional Neural Networks (DCNNs) (Figure 21). These consist of many layers of feature extractors and can be used both as standalone classifiers, i.e., unlabeled images are passed through a pre-trained DCNN that performs the final class label prediction directly, or as generators of image features, i.e., the output of a hidden layer of the pre-trained DCNN is used as a global image representation (Simonyan, 2014; Markatopoulou, 2015). The latter type of features is referred to as DCNN-based and these features will be used in hackAIR for concept detection due to their high performance both in terms of time and accuracy. Several DCNN software libraries are available, e.g., Caffe (Jia, 2014), MatConvNet (Vedaldi, 2015), and different DCNN architectures have been proposed, e.g., CaffeNet (Krizhevsky, 2012), GoogLeNet (Szegedy, 2014).

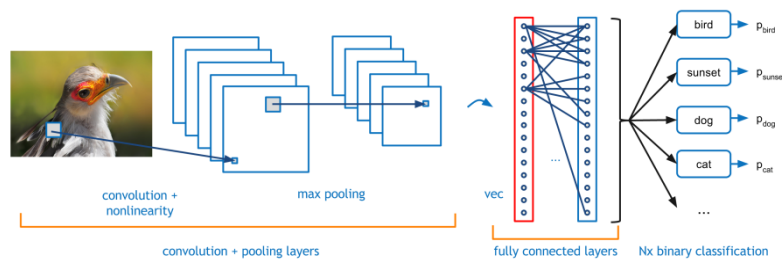


Figure 21 – DCNN-based features

5.2.1.1.2 Classification

The classification step is the last step of the multimedia concept detection process. It involves the construction of models for concept detection by using the low-level visual features, and then the application of these models for image labelling. For learning the associations between the image representations and concept labels, algorithms such as Support Vector Machines (SVM) and Logistic Regression are used (Markatopoulou, 2015). SVMs are trained separately for each concept, on ground-truth annotated corpora, and when a new unlabeled video shot arrives, the trained concept detectors will return confidence scores that show the belief of each detector that the corresponding concept appears in the shot.

5.2.1.1.3 Sky Detection Framework

In the employed framework, we train a 22-layer GoogLeNet (Szegedy, 2015) network on 5055 ImageNet concepts, which are a subset of the 12,988 ImageNet concepts. Then, we applied this network on the TRECVID SIN 2013 development dataset and we used as a feature (i.e., a global image representation) the output of the last fully-connected layer (5055 dimension) to train one Support Vector Machine (SVM) per concept. We evaluated the trained SVM concept detectors on the TRECVID SIN 2013 test. Subsequently, we applied this network on the test images to extract features, and served them as input to the trained SVM classifiers in order to gather scores for each of the 346 TRECVID SIN concepts. From this concept list, we select three, i.e. clouds, sky, and sun, in order to recognize images that are taken outdoors and contain significant part of sky.

At this point, we should mention that another method that we can apply instead of training SVM classifiers directly on one or more layers of an ImageNet network is to initially fine-tune its parameters on the TRECVID SIN dataset. Fine tuning (FT) compared to directly using SVMs trained on the ImageNet network provides in general a boost to the accuracy, however it is slower because the complete network has to be refined. FT is used generally for transferring knowledge which involves taking a network that has been trained on a large-scale source dataset and fine-tuning its parameters for the target dataset. Several approaches are proposed for knowledge transfer, but the one that we consider for the upcoming deliverable is the one that proposes the extension of a pre-trained DCNN by one or more fully-connected layers placed on the bottom of the classification layer (Oquab, 2014; Snoek, 2015). It has been observed that features learned on the top layers of a deep network are dataset specific. As a result learning

D3.1: Environmental node discovery, indexing and data acquisition

the weights of such layers from scratch and also extending the network with more layers seems to outperform the typical transfer learning approach that only replaces the classification layer.

5.2.1.1.4 Evaluation

In order to check the accuracy of the proposed concept detection framework, we annotated 23,000 images that were downloaded from social media platforms and were captured in the city of Berlin during the time period between 01/01/2016 to 15/04/2016. Then, we run each image from this annotated dataset through the concept detection framework and extracted the probability values of the concepts *clouds*, *sky*, and *sun*. We set the limit for considering a concept as positive (present) to 0.6 and calculate the precision, leading to precisions of 97.6% (clouds), 91.2% (sky) and 70.2% (sun) respectively, which are considered satisfactory for the needs of hackAIR.

5.2.1.2 Sky localization

Sky localization is an important problem in computer vision. Generally, the color and gradient of the pixel in sky border change greatly. An approach that was proposed by (Zhijie, 2015) suggests measuring the sky border points. The authors propose several modifications of the original sky border position function, namely the determination of multi-border points for detecting complex sky region identification in images. In (Irfanullah, 2013), the authors suggest using blue color for localizing and tracking RGB color in different applications of image processing. Specifically, they propose a pixel based solution utilizing the sky color information. The success of deep networks on several domains led to their application in semantic segmentation as well. Specifically, several recent works have applied Convolutional Networks (convnet) to dense prediction problems, including semantic segmentation by Ning et al. (2005), Farabet et al. (2013), and Pinheiro and Collobert (2014); boundary prediction for electron microscopy by Cirosan et al. (2012) and for natural images by a hybrid convnet/nearest neighbor model by Ganin and Lempitsky (2014). Moreover, Hariharan et al. (2014) and Gupta et al. (2014) adapt deep classification nets to semantic segmentation, but do so in hybrid proposal-classifier models. These approaches fine-tune an R-CNN system (Girshick, 2014) by sampling bounding boxes and/or region proposals for detection, semantic segmentation, and instance segmentation. Finally, fully convolutional training is rare, but used effectively by Tompson et al. (2014) to learn an end-to-end part detector and spatial model for pose estimation, although they do not analyze this method.

5.2.1.2.1 Framework

We employ the approach by Long et al. (2015), which draws on recent successes of deep nets for image classification (e.g. Krizhevsky, 2012) and transfer learning. Transfer learning was first demonstrated on various visual recognition tasks (e.g. Donahue, 2014), then on detection, and on both instance and semantic segmentation in hybrid proposal classifier models (Girshick, 2014; Gupta, 2014; Hariharan, 2014).

Unlike the existing methods, Long et al. (2015) adapt and extend deep classification architectures, using image classification as supervised pre-training, and fine-tune fully convolutionally to learn simply and efficiently from whole image inputs and whole image ground truths. It should be also noted that Long proposed the use of upsampling for two main reasons: (1) reduce training and prediction time, and (2) improve consistency of output. Figure 22 depicts semantic segmentation using Fully Convolutional Networks (FCN).

A schematic overview of the employed approach is illustrated in Figure 22

D3.1: Environmental node discovery, indexing and data acquisition

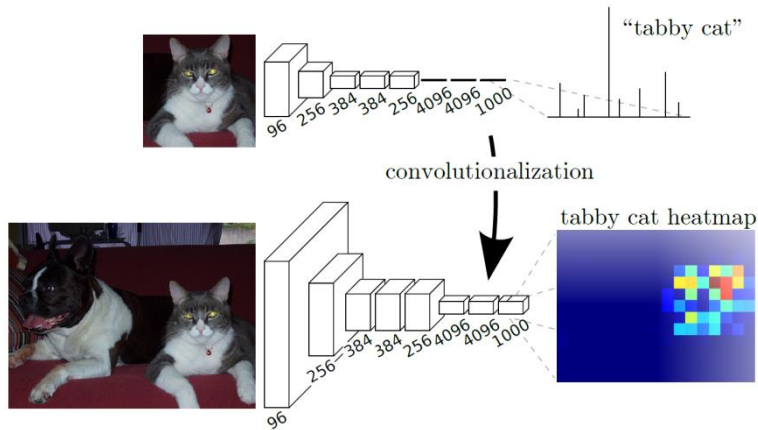


Figure 22 – Semantic Segmentation using FCN

In general the steps followed by the employed framework are the following:

1. A neural network is trained for image classification and on input images of a fixed size ($d \times w \times h$)
2. The network is interpreted as a single convolutional filter for each output neuron over the complete image area on which the original network was trained.
3. The network is run as a CNN over an image of any size with a stride $s \in \mathbb{N} \geq 1$
4. In case that $s > 1$, then an upsampling layer (deconvolutional layer) is required for converting the coarse output into a dense output.

Finally, it should be noted that according to the tests run by Long et al. (2015), the mean pixel accuracy for the semantic categories including sky is 85.2%.

5.2.1.2.2 Evaluation

The sky localization algorithm was evaluated both visually on a small set of images returned by the Flickr API and also on a set of images belonging to the SUN Database (<http://groups.csail.mit.edu/vision/SUN/>). The SUN database (Xiao, 2010) is a comprehensive collection of annotated images covering a large variety of environmental scenes, places and the objects within. Figure 23 depicts a sample set of visual examples of masks that capture the sky region.

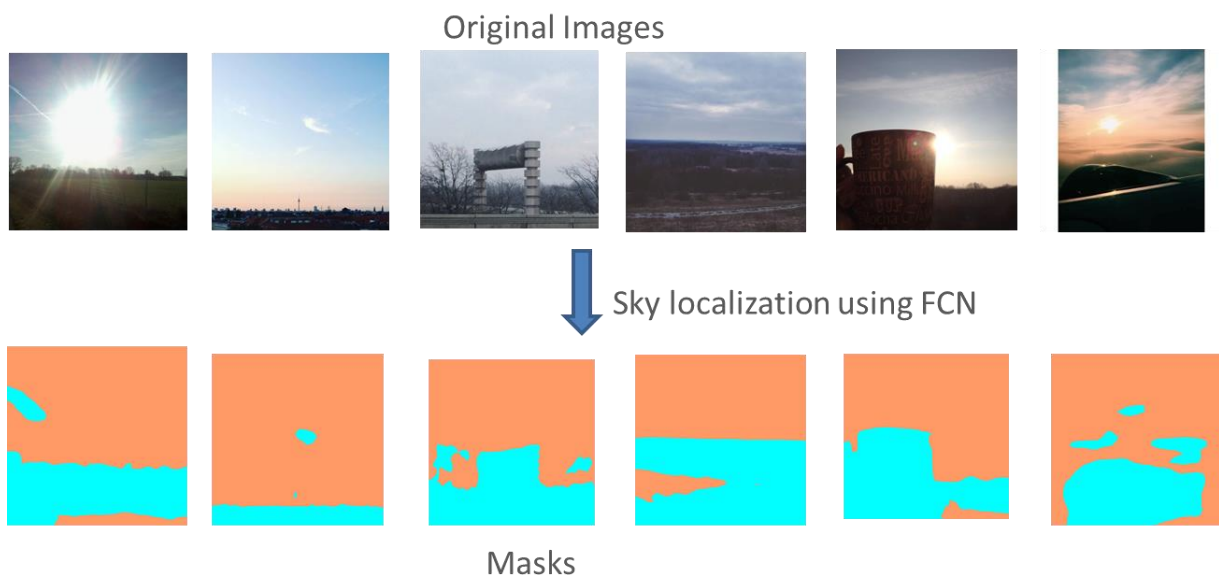


Figure 23 – Sky localization visual examples using FCN on images retrieved by the Flickr API

Regarding the evaluation of the sky localization algorithm on the SUN Database, 2,030 images were found in the database that are annotated with the concept sky and for which the polygons capturing the sky part is provided. Figure 24 depicts the results of sky localization on five randomly selected images from the SUN Database. It should

D3.1: Environmental node discovery, indexing and data acquisition

be noted that the upper part of the image shows the original images while the lower part their masks. Regarding the masks, the orange part of the image depicts the part of the image automatically recognized as sky.

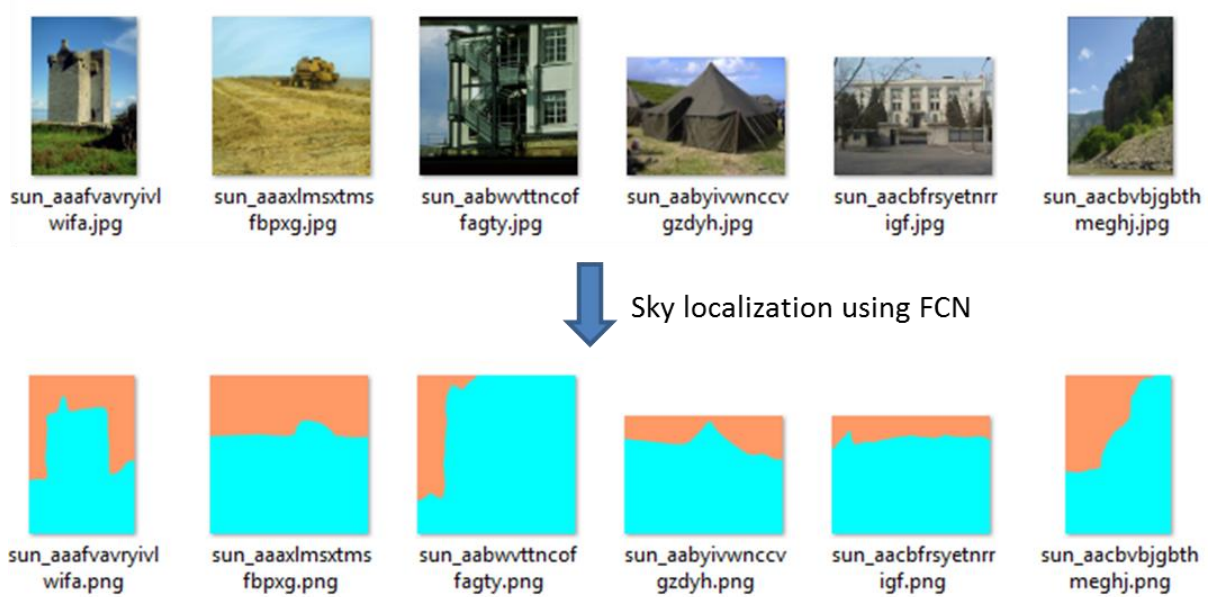


Figure 24 - Sky localization visual examples using FCN on images from SUN Database

Using the ground truth provided by the database, we calculated the precision and recall for every image. The mean precision is 0.9177, while the mean recall is 0.94245. It should be noted, that we are interested mainly on the precision of the algorithm given that what is required is recognizing accurately even a small part (it should be at least $(m \times n)/100$ pixels where m, n are the image dimensions) of the sky inside the image.

5.2.2 Second Method: Image processing heuristics for sky localization

5.2.2.1 Framework

The approach for sky detection presented in this section is more straightforward and simple and it is based on heuristic rules provided by experts (DUTH) that aim at recognizing the sky part of the images. In general the algorithm is based on identifying whether the pixels meet certain criteria involving their color values and the size of clusters (based on their color) they belong to. The output of the algorithm is a mask containing all pixels that capture the sky. The algorithm proposed can be summarized in the following steps and is illustrated in Figure 26.

- The upper 50% part of the image (size of image $M \times N$) is selected. This is based on the simple observation that usually images that contain sky, typically contain it in their upper part.
- The image pixels that satisfy the following conditions are recognized:

$$0.5 \leq \frac{R}{G} \leq 1 \text{ AND } 0.5 \leq \frac{G}{B} \leq 1 \text{ AND } \frac{B}{R} > 1.25$$

where R, G, B are respectively the Red, Green, and Blue value of each pixel, while $R/G, G/B$ and B/R are the ratios of these values.

- A loop through the pixels, identified in step 2, is realized. For each pixel, a control check with the 8 neighboring pixels (Figure 25) is realized that checks whether it satisfies step 2 criteria. If there are no neighboring pixels that satisfy step 2 criteria, the original pixel is discarded (i.e. pixel cannot be a part of output mask produced), otherwise it is kept and the corresponding R/G and G/B ratios are computed.

D3.1: Environmental node discovery, indexing and data acquisition

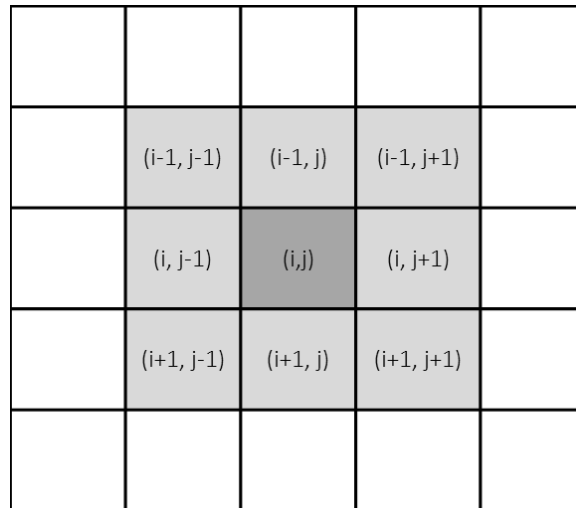


Figure 25 - 8-neighbors of pixel p

- The connected components of the pixels selected after step 3, are recognized, using the algorithm described in (Shapiro, 2001).
- The pixels belonging to the connected components with size over $(m \times n)/400$ are kept.
- The sum of the number of pixels (S_p) that passed successfully from step 5 is realized.
- A check is realized, regarding the value S_p . Specifically, if S_p is $\geq (m \times n)/100$, then we move to the next step, otherwise we consider that the image does not contain significant part of sky that can be used for air quality (i.e. PM) estimation, and thus it is discarded.
- The pixels with extreme upper or lower (outliers) R/G ratio values are discarded. As extreme values we consider values over $meanRG + 4 * stdRG$ and smaller than $meanRG - 4 * stdRG$. $meanRG$ is the average value of R to G ratio and $stdRG$ is the standard deviation of R to G ratio.
- The check described in step 7 is realized again. If the result of the check is true then we move to the next step, otherwise the image is discarded.
- A check regarding the monotonicity of G to B ratio is realized for random 20 vertical lines of the connected components. If the G/B increases in a monotonic way, we move to the next step. We should note that this check is realized for removing processed images that contain blue parts.
- The connected components that were recognized and passed successfully the aforementioned steps are used for drawing the image mask that captures the sky.

The current algorithm is far stricter than the other since sun and clouds are not considered part of the sky.

D3.1: Environmental node discovery, indexing and data acquisition

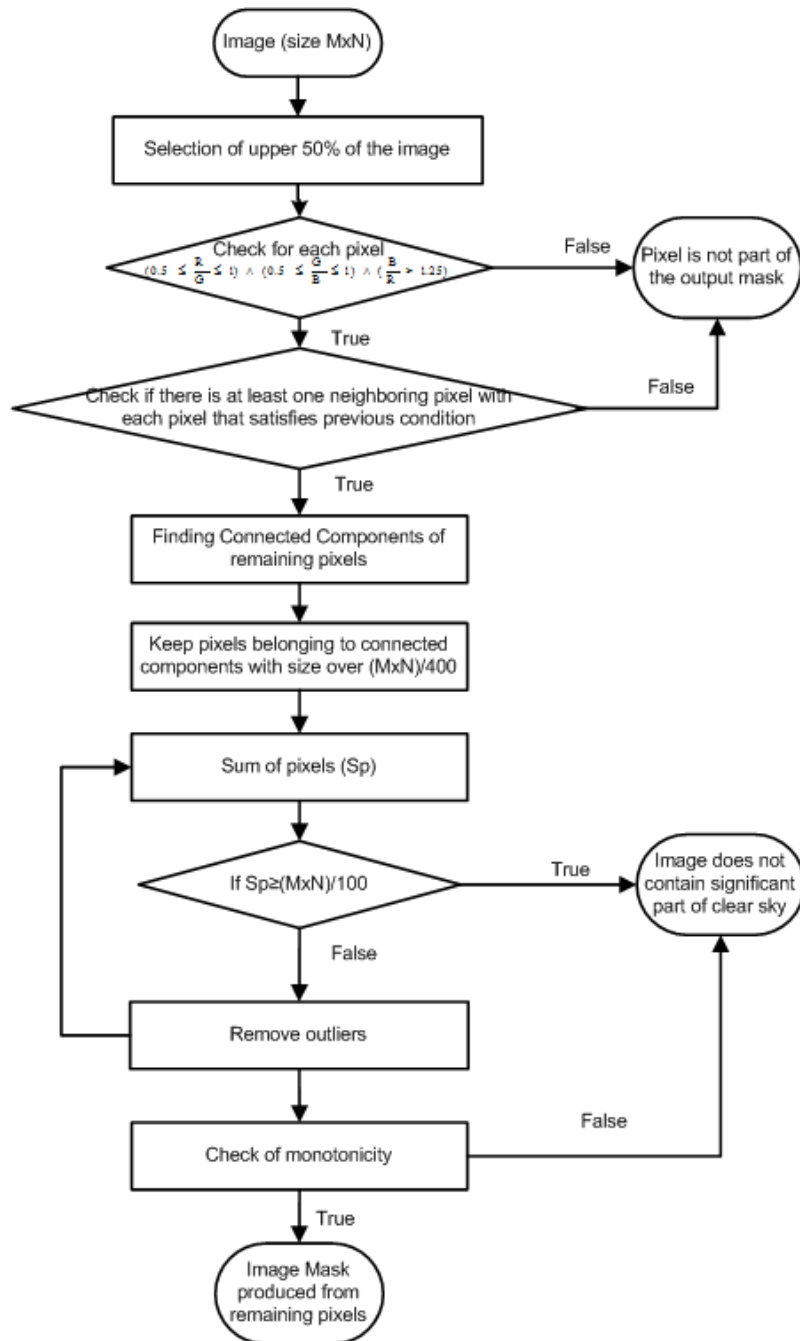


Figure 26 – Flowchart of sky detection algorithm using the method proposed by the experts.

5.2.2.2 Evaluation

The second method was evaluated solely on the SUN database (Xiao, 2010). The evaluation was realized on the same dataset, consisting of 2,030 images, as the previous method. Figure 27 depicts the results of sky localization on five random images belonging to the SUN Database (the same as the ones used with the previous method). Again, the upper part of the image shows the original images while the lower part their masks (painted in black). Using the ground truth provided by the database, we calculated the precision and recall for every image. The mean precision is 0.8245, while the mean recall is 0.5922. It should be noted, that we are interested mainly on the precision of the algorithm given that what is required is recognizing accurately even a small part (it should be at least $(m \times n)/100$ pixels where m, n are the image dimensions) of the sky inside the image.

D3.1: Environmental node discovery, indexing and data acquisition

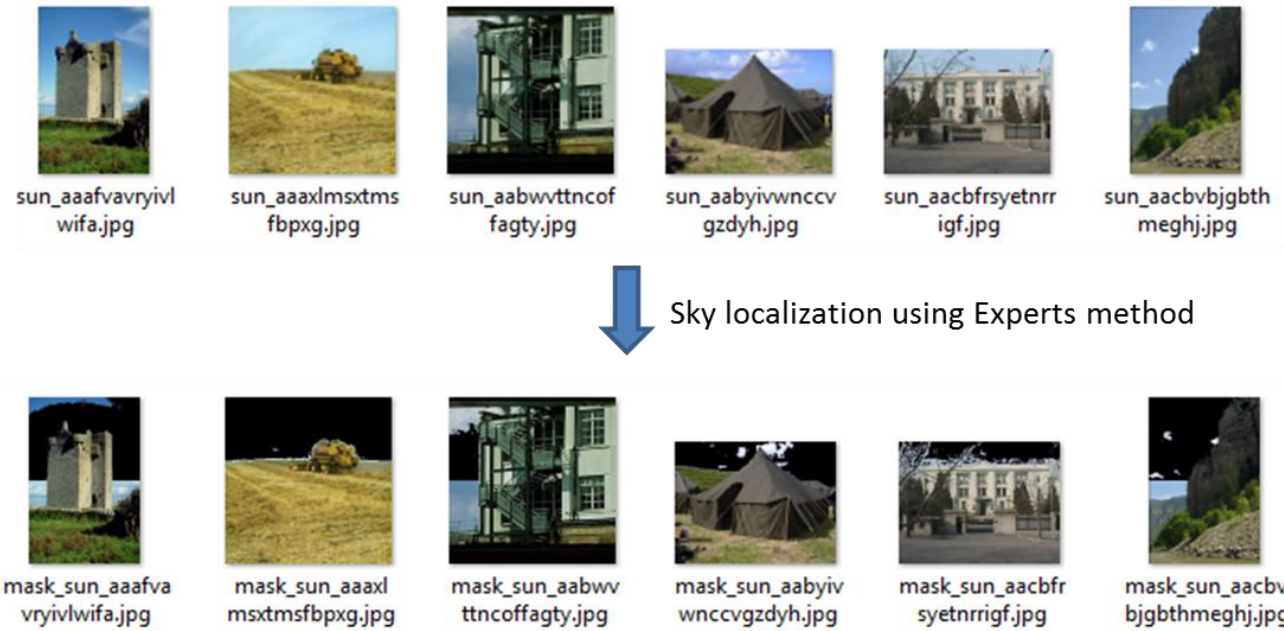


Figure 27 - Sky localization visual examples using the heuristics method on images from the SUN Database.

5.2.3 Comparison of sky localization methods

Based on the results reported in Sections 5.2.1.2.2 and 5.2.2.2, we can conclude that in general the first method that is based on FCN has much better performance given that both the mean precision and recall are better than the ones provided by the heuristics-based algorithm. Moreover, given that the main focus is on the precision and not recall we provide some visual examples where the precision in the second method is very bad, in order to see where it fails (Table 19), and see whether it can be improved. Based on Table 19, we can deduce that in certain cases both methods have low precision, but the precision of the second method is much lower. Therefore, based on the first results of sky localization, we can conclude that the method using FCN which is far more complex than the other has in general better results. However, both methods need to be improved in order to achieve better precision values and thus allow for more accurate Air Quality (AQ) estimations. This will be part of deliverable 3.2.

Table 19 – Comparison of sky localization methods

Image: sun_aacovuzcesaghkr.jpg			
Original Image	Ground Truth	Mask produced using <i>Method 1</i> (Precision = 0.8398)	Mask produced using <i>Method 2</i> (Precision = 0.005)
Image: sun_aefbyczapzrymptm.jpg			

D3.1: Environmental node discovery, indexing and data acquisition

			
Original Image	Ground Truth	Mask produced using <i>Method 1</i> (Precision = 0.4745)	Mask produced using <i>Method 2</i> (Precision = 0.063)
Image: sun_aarowrwdgbkiftpi.jpg			
			
Original Image	Ground Truth	Mask produced using <i>Method 1</i> (Precision = 0.526)	Mask produced using <i>Method 2</i> (Precision = 0.0275)
Image: sun_afnctlwkhlepeaf.jpg			
			
Original Image	Ground Truth	Mask produced using <i>Method 1</i> (Precision = 0.5137)	Mask produced using <i>Method 2</i> (Precision = 0.0111)

6 Data Storage and Indexing

This section describes the profile of the environmental nodes and the repository design and implementation. We first briefly mention the data that need to be stored by the repository, and based on the results we report on the options for node repository, their advantages and disadvantages. In the sequel, we present the indexing techniques of the database selected and eventually, we present the initial design of the repository that will store effectively all this retrieved information.

6.1 Indexing Profile Records

Based on the empirical study we have conducted in section 3, the information that we need to extract and store from a specific environmental node in order to achieve efficient indexing and orchestration is the following:

- Type of environmental measurement (e.g. Temperature)
- Value of measurement (e.g. 30°C)
- Area coverage (e.g. city of Berlin)
- Date (e.g. 05/08/2016)

The aforementioned information constitutes the basic fields that should be extracted from each node since they seem to cover efficiently the content (i.e. type of information) one can mine from them. In addition to them, the URL of the nodes is part of the node profile record since it describes in a unique way the specific node and can be considered as its fingerprint.

6.2 Node Repository

The options that were considered for storing the data (fields) coming from the aforementioned sources along with the data coming from the user-developed sensors are the following:

- SOS server
- PostgreSQL and PostGIS database
- MongoDB

In the sequel, we report on the advantages and disadvantages of each option and we will conclude with the option we consider as the best for covering hackAIR needs. Then, we will present a brief overview of the indexing techniques of the selected option and, finally, a first version of the database with some indicative data.

6.2.1 SOS server

A Sensor Observation Service (SOS) is a server used for handling data provided by sensors. The objective of the SOS specification is to organize and facilitate the retrieval of real time or archived data produced by all kinds of sensors (e.g. sites) (SOS2.0). At this point, it should be also noted that SOS is an approved standard of the Open Geospatial Consortium (SOS2.0). The wide application of SOS servers has led to the development of a considerable number of OGC compliant products, such as 52°North SOS. 52°North SOS implements the official OGC (Open Geospatial Consortium) specification 1.0 and depends heavily on PostgreSQL DBMS with the PostGIS extension for representing the data and the spatial information.

After a thorough investigation of the SOS server and also based on previous experience (52°North SOS was used in PESCaDO), we present in Table 20 its advantages and disadvantages.

D3.1: Environmental node discovery, indexing and data acquisition

Table 20 – Advantages and disadvantages of SOS server.

Advantages	Disadvantages
<ul style="list-style-type: none">• It is OGC compliant• The schema is defined• The services for storing and querying data are implemented• It is developed for handling sensor-base data• It supports queries on geographical objects (PostGIS)	<ul style="list-style-type: none">• Cannot handle efficiently the storing of random moving sensors which makes it inefficient for handling images captured from Social Media platforms and User-developed sensors.• The schema is not flexible, thus the different need of sources might create a rather sparse database

6.2.2 PostgreSQL/PostGIS database

PostgreSQL is an object-relational database management system (ORDBMS) with an emphasis on extensibility and standards-compliance. As a database server, its primary function is to store data securely, and to allow for retrieval at the request of other software applications. It can handle workloads ranging from small single-machine applications to large Internet-facing applications with many concurrent users. PostGIS is an open source software program that adds support for geographic objects to the PostgreSQL object-relational database. PostGIS follows the Simple Features for SQL specification from the Open Geospatial Consortium (OGC). To sum up, it adds spatial functions such as distance, area, union, intersection, and specialty geometry data types to the database.

Table 21 presents the advantages and disadvantages of this solution.

Table 21 – Advantages and disadvantages of PostgreSQL/ PostGIS server.

Advantages	Disadvantages
<ul style="list-style-type: none">• The schema developed in SOS can be used with alterations in order to be more simple• It supports queries on geographical objects (PostGIS)	<ul style="list-style-type: none">• The schema is not flexible, thus the different need of sources might create a rather sparse database• It is not very scalable• It is rather slow

6.2.3 MongoDB

MongoDB is a free and open-source cross-platform document-oriented database. Classified as a NoSQL database, MongoDB avoids the traditional table-based relational database structure in favor of JSON-like documents with dynamic schemas, making the integration of data in certain types of applications easier and faster. It is a rather new database, given that its first release was on 2009 and thus certain functionalities are not as mature yet compared to PostgreSQL/ PostGIS. However, Mongo already offers also a number of indexes and query mechanisms to handle geospatial information which is essential for the hackAIR project.

Table 22 presents the advantages and disadvantages of this solution.

D3.1: Environmental node discovery, indexing and data acquisition

Table 22– Advantages and disadvantages of Mongo server.

Advantages	Disadvantages
<ul style="list-style-type: none">• It is very scalable• It is rather fast• It does not have a standard schema and thus the documents stored can have different fields in order to cover the different need of different sources.	<ul style="list-style-type: none">• Although, it offers a number of indexes and query mechanisms to handle geospatial information, it cannot handle as complex or sophisticated queries as PostGIS.

6.2.4 Comparison of SOS, PostgreSQL/ PostGIS and Mongo

In general, hackAIR is expected to host large volume of data since data from different sources will be gathered. All this information should be stored in the same repository in order to allow efficient querying and retrieval by the Fusion Service. Moreover, geospatial queries should be handled given that the user will define hers/his area of interest in a form of a simple polygon.

Thus, based on the aforementioned needs, we consider as the best solution for storing all data produced during hackAIR, the MongoDB. The reasons for using MongoDB are its scalability, its ability to present fast results as well the fact that it allows storing of different documents in the same schema. Therefore, we could have slightly different documents, apart from the main basic fields, that would cover the different needs of the different sources. Finally, although it is not as mature in handling geospatial queries, the queries that are expected to be handled are quite simple and thus they can be easily handled by MongoDB.

6.3 Indexing techniques at the Repository

In this section, we present a brief overview of the indexes supported by MongoDB. In general, indexes support the efficient resolution of queries. Without indexes, MongoDB, or any other database, would have to scan every document of a collection to select those documents that match the query statement. This scan is highly inefficient and would require that MongoDB would process a large volume of data. However, if an appropriate index exists for a query, MongoDB can use the index to limit the number of documents it must inspect.

Indexes are special data structures that store a small portion of the data set in an easy to traverse form. The index stores the value of a specific field or set of fields, ordered by the value of the field as specified in index. The ordering of the index entries supports efficient equality matches and range-based query operations. In addition, MongoDB can return sorted results by using the ordering in the index. In general, MongoDB indexes use a B-tree data structure. In computer science, a B-tree is a self-balancing tree data structure that keeps data sorted and allows searches, sequential access, insertions, and deletions in logarithmic time. The B-tree is a generalization of a binary search tree in that a node can have more than two children (Comer 1979, p. 123). Unlike self-balancing binary search trees, the B-tree is optimized for systems that read and write large blocks of data. B-trees are a good example of a data structure for external memory. It is commonly used in databases and filesystems.

MongoDB provides a number of different index types to support specific types of data and queries²³:

²³ <https://docs.mongodb.com/v3.2/indexes/>

D3.1: Environmental node discovery, indexing and data acquisition

- **Single Field** MongoDB supports the creation of user-defined ascending/descending indexes on a single field of a document.
- **Compound Index** MongoDB also supports user-defined indexes on multiple fields, i.e. compound indexes. The order of fields listed in a compound index defines also the order of sorting.
- **Multikey Index** MongoDB uses multikey indexes to index the content stored in arrays. Thus, in case you index a field that holds an array value, MongoDB creates separate index entries for every element of the array.
- **Geospatial Index** To support efficient queries of geospatial coordinate data, MongoDB provides two special indexes: 2d indexes that uses planar geometry when returning results and 2dsphere indexes that use spherical geometry to return results.
- **Text Indexes** MongoDB provides a text index type that supports searching for string content in a collection. These text indexes do not store language-specific stop words and stem the words in a collection to only store root words.
- **Hashed Indexes** To support hash based sharding, MongoDB provides a hashed index type, which indexes the hash of the value of a field. These indexes have a more random distribution of values along their range, but only support equality matches and cannot support range-based queries.

Finally, a rather interesting functionality supported by MongoDB regarding indexing is the Index Intersection. This operation allows MongoDB to the intersection of indexes to fulfill queries.

6.4 Repository Population

In this section we present a first version of the MongoDB that is used for storing environmental data (i.e. PM measurements) from multiple sources. Currently, MongoDB is populated only with data coming from the Flickr social media platform, but we plan on inserting data from the other sources, i.e. environmental web sites, web services and webcams, and mobile app in the upcoming months.

Therefore, MongoDB now contains data coming exclusively from Flickr that cover the cities mentioned in Section 3.2.1 and the time period from 1/1/2016 until now. The total number of records retrieved is 539,323. The procedure that was followed for retrieving these images is described along the whole deliverable, but we briefly cite the main steps for the sake of completeness.

- **Step 1: Collection of Images using Flickr API.** It involves querying the Flickr API for specific regions in a daily basis. The number of images retrieved varies mainly according to the size of the city.
- **(optional) Step 2: Storage of data to MongoDB and local storage of images.** The images that were retrieved during the Step 1 are locally stored to a server, and also stored to MongoDB, along with several information that is returned inside the response including the geolocation of the image, the date/time the image was taken, the image views and its URL. Figure 28 is a screenshot of MongoDB, while Table 23 contains a list of the fields of a Flickr MongoDB document and their description. It is expected that the data coming from the other sources apart from the basic information (i.e. source, loc, city, date, timestamp, date_str) will require other fields for holding their properties. This step is only realized when the source is image-based (i.e. Flickr, webcams, mobile App)

The following steps are not yet part of the pipeline, but they will be added in the upcoming months.

- **Step 3: Running Image Data Analysis.** Application of data analysis techniques in order to find the part of the image containing the sky (if it exists) and estimate the R to G and G to B ratios. These ratios are stored into MongoDB. In case of other sources, the other methods for data analysis will be incorporated.
- **(optional) Step 4: Call of AQ Estimation Service.** This step is realized only when the source is image-based (i.e. Flickr, webcams, mobile App)
- **Step 4: Storing of air quality value to MongoDB and Indexing.** The air quality measurements that are retrieved from any source are inserted into MongoDB. In case of image-based sources, the existing records are updated.

The data stored in MongoDB are available to other services through a REST API service.

D3.1: Environmental node discovery, indexing and data acquisition

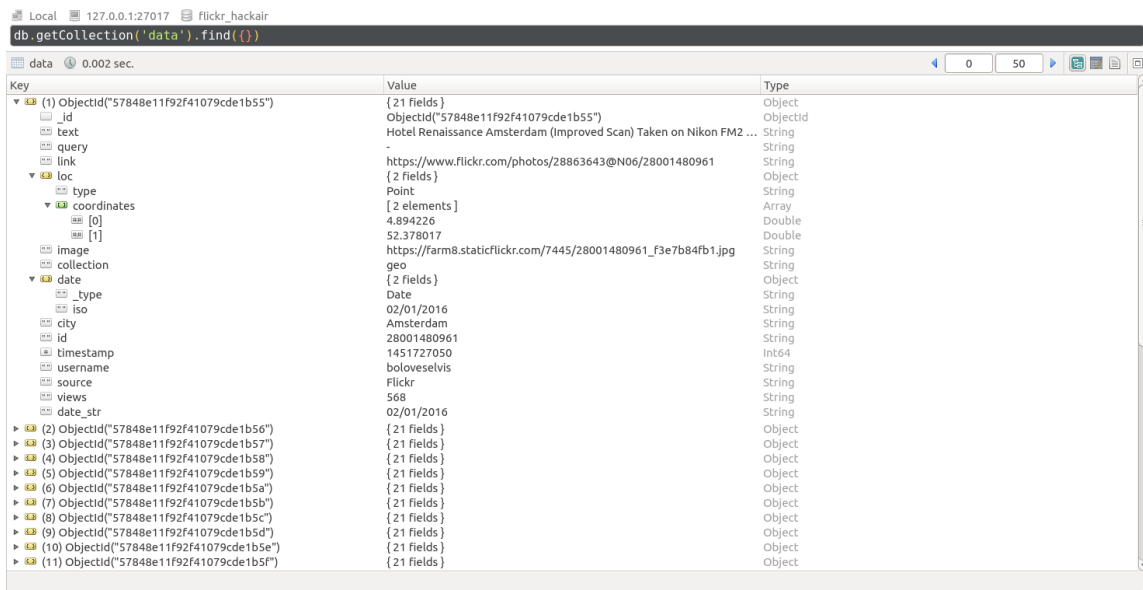


Figure 28 – Population of MongoDB with data coming from Flickr

Table 23 – MongoDB fields, their type and description.

Field	Type	Description
id	string	Flickr ID
source	string	Type of source (i.e. Flickr, site, service, webcam, sensor, mobileApp)
collection	string	In case the source is Flickr, there is the possibility to retrieve images not by using the geotagging information of images, but with their text description. This type of queries is performed when no data can be found using the geo information. Thus, when the query uses the geotagging information, the collection field takes the value 'geo', otherwise it takes the value 'text'.
query	string	In case the query to Flickr is realized using the text description of the image, this field holds the query send to the Flickr API.
text	string	It holds the image description/ text returned by the Flickr API.
views	int	It holds the number of views of the image as returned by the Flickr API.
username	string	It holds the name of the user who uploaded the image as provided by the Flickr API.
image	string	It holds the image URL.
link	string	It holds the page URL.
loc	GeoJSON object	It stores the city coordinates as a GeoJSON object <ul style="list-style-type: none"> • type: it holds the type of the object, e.g. Point, Polygon • coordinates: it holds the exact coordinates
city	string	It holds the name of the city
date	Date object	It holds the date taken of image <ul style="list-style-type: none"> • _type: It holds the type of the object, i.e. Date • iso: It holds the value of date in ISO format
timestamp	long	It holds the date that the image was taken in form of timestamp for easy sorting.
date_str	string	It holds the date that the image was taken in form of string for easy storing in the local server (i.e. images are stored into folders named after the city and the date).

7 Conclusions

In this deliverable, we conducted separate studies for the majority of the means of collective sensing that are handled by hackAIR including the environmental web sites and web services, images captured by users and uploaded to social media platforms that are publicly available, and webcams. As far as images captured by the users of the hackAIR mobile app are concerned, although they are processed in a common way with the images retrieved from social media platforms, the way they are captured is not part of this deliverable (but of deliverable 5.2). Similarly, the measurements provided by the user generated low cost devices are concerned, will be studied in another deliverable (3.5). However, it should be noted that eventually all measurements will be stored in a common repository in order to allow efficient querying, indexing and retrieval.

Thus, we conducted a thorough empirical study of the environmental web sites based on the empirical cycle methodology. To perform the study, we selected a balanced dataset that was provided by the environmental experts and that was representative of the overall data we address. After analysis of the environmental web nodes, we reached many important conclusions that are worth summarizing. First, it seems that many environmental websites are available for public usage covering a vast number of locations and environmental aspects. However, the sites do not have common formatting/presentation and thus in order to retrieve the information that is of interest, i.e. the type and data of environmental measurement, the area coverage, and the date, the information extraction is rather complicated. Apart from the limited set of sites provided by the experts, new sites are discovered by running a focused crawler that receives as input a set of URLs of environmental sites and which traverses the web for discovering web pages with similar content. After having collected a set of URLs containing environmental information, the next step involves the information extraction using web information extraction techniques using the easIE tool that can easily configured to handle pages. The configuration of the tool is realized in a manual way but the effort that is required is insignificant and can be realized by users with little programming skills. The output of the tool is a file containing all the information of interest that exists in the web page.

As far as web services are concerned, given that the number of publicly available services providing environmental information is very limited the study realized was limited to presenting the services provided by the experts and also the development of a technique for the automatic discovery of similar services it not practical. On the other hand the extraction of information from the web services is rather trivial with a use of a JSON or XML parser given that the formatting of the data is very structured.

Furthermore, regarding social media platforms, a study was realized that covered the possible platforms that allow the users to upload images and developers to download them easily without many restrictions. The results of the study showed that Flickr is the most appropriate candidate and thus we focused on presenting the Flickr API, by providing what queries are acceptable and how information is provided to the user. The retrieved images, along with the images retrieved from the other sources (i.e. webcams and mobile app) are processed in order to detect the sky part of the image and retrieve its R/G and G/B ratios. Two different methods were studied for the sky localization, the one based on Fully Convolutional Networks and the other based on an algorithm proposed by the air quality experts that were compared and the results showed the first method outperformed the second. The R/G and G/B ratios along with the geographical coordinates and the date/time information of the image are provided to the 'AQ Estimation service' which provided an estimation of the air quality of the area captured by the image.

The last study involved webcams that are located in the area of interest of hackAIR and it was realized into two steps. The first involved webcams found in web sites that were returned by Google search engine, while the second involved the AMOS dataset that contains already indexed a set of webcams located around the world. The extraction of the images from the first set of webcams requires information extraction techniques while AMOS provides a REST API for automatic downloading of the images captured by the cameras of interest. As we have already mentioned the image processing of images coming from webcams follows the same procedure with the images returned by Flickr.

D3.1: Environmental node discovery, indexing and data acquisition

Finally, we discussed the repository that will be used for storing and querying efficiently the data stored by all the aforementioned sources and based on the variety of the data as well on the characteristics of the databases studied, we concluded that the best option for hackAIR is MongoDB.

Future work, which will be reported in D3.2, includes the further investigation of the webcams dataset in order to decide whether AMOS dataset can be used exclusively for retrieving data from webcams or external sites need to be considered and thus information extraction is required. Moreover, as far as webcams are concerned a framework for automatically discovering sites with webcams will be implemented by considering the characteristic features of the sites studied in the current study. In that case, methods for extracting frames from live video streaming should be considered given that significant part of the webcams found in web sites are provided in that form. And finally, as far as webcams are concerned, it is necessary to check whether the images coming from webcams can provide “accurate” AQ estimations given that they usually have lower resolution compared to the images retrieved from the other sources. Moreover, as far as the image processing techniques are concerned, it is necessary to further investigate the two methods presented in order to decide which one will be eventually used given that there is room for improvement in both of them. Finally, regarding the indexing and storing of the data in MongoDB, the current implementation is merely the first version which holds only data from one source and thus data from all sources should be considered as well as their features in order to decide on the fields to be used for each case.

8 References

- ♥ 52North (2009). *Installation Guide for Sensor Observation Service* version 3.1.1. [online] Available at: http://52north.org/communities/sensorweb/docs/how2install_SOS.pdf.
- ♥ 52north, (2014). *Sensor Observation Service (SOS)*. [online] Available at: https://wiki.52north.org/bin/view/Sensornet/SensorObservationService#SOS_tutorial.
- ♥ Battelle, J. (2005). *The Search: How Google and its Rivals Rewrote the Rules of Business and Transformed Our Culture*. New York: Portfolio.
- ♥ Baumgartner, R., Frolich, O., Gottlob, G., Harz, P., Herzog, M., & Lehmann, P. (2005). Web data extraction for business intelligence: the lixta approach. *na*, 30-47.
- ♥ Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-Up Robust Features (SURF). *Comput. Vis. Image Underst.*, vol. 110(3), pp. 346–359.
- ♥ Catanese, S. A., De Meo, P., Ferrera, E., Fiumara, G., & Provetti, A. (2011). Crawling Facebook for Social Network Analysis Purposes. *Proceedings of the international conference on web intelligence, mining and semantics*. (p. 52). ACM.
- ♥ Chakrabarti, S. van den Berg, M., and Dom, B.: Focused crawling: a new approach to topic-specific Web resource discovery. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, Vol. 31, Issue 11-16, pp. 1623 – 1640.
- ♥ Chakrabarti, S., Van den Berg, M., Dom, B. (1999). Focused crawling: A new approach to topic-specific web resource discovery. In: *Proceedings of the 8th International Conference on World Wide Web, (WWW 1999)*, pp 1623–1640
- ♥ Chang, C. C., & Lin, C. J. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3), 27.
- ♥ Chen, H., Fan, H., Chau, M., and Zeng, D. (2001). MetaSpider: Meta-Searching and Categorization on the Web. *Journal of the American Society for Information Science and Technology*, Vol. 52, No 13, pp.1134-1147.
- ♥ Cho, J., Garcia-Molina, H., Page, L. (1998). Efficient crawling through URL ordering. *Comput Netw* 30(1-7):161–172
- ♥ Cirezan, D. C., Giusti, A., Gambardella, L. M., and Schmidhuber, J. (2012). Deep neural networks segment neuronal membranes in electron microscopy images. In *NIPS*, pages 2852–2860.
- ♥ Crescenzi, V., Mecca, G., Merialdo, P., et al. (2001). Roadrunner. Towards automatic data extraction from large web sites. In *VLDB*, volume 1, pages 109-118.
- ♥ Creswell, J.W. (2013). *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage Publications, Inc.
- ♥ Dalvi, N., Bohanon, P., & Sha, F. (2009). Robust Web Extraction: An Approach Based on a Probabilistic Tree-Edit Model. *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data.*, (pp. 335-348).
- ♥ Davison, B.D. (2000). Topical locality in the web. In: *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, (SIGIR 2000)*, pp. 272–279
- ♥ De Bra. P, Post, R.D.J. (1994). Information retrieval in the world-wide web: Making client-based searching feasible. *Comput Netw ISDN Syst* 27(2):183–192

D3.1: Environmental node discovery, indexing and data acquisition

- 📍 de Groot, A.D. (1961). *Methodology, Foundations of Research and Thought in the Behavioral Sciences*. Mouton Publishers.
- 📍 Donahue, J., Jia, Y., Vinyals, O., et al. (2014). DeCAF: A deep convolutional activation feature for generic visual recognition. In *ICML, 2014*.
- 📍 Dreyfuss, Josh (2015). *The Ultimate JSON Library: JSON.simple vs GSON vs Jackson vs JSONP*, <http://blog.takipi.com/the-ultimate-json-library-json-simple-vs-gson-vs-jackson-vs-json/>
- 📍 Farabet, C., Couprie, C., Najman, L., and LeCun, Y. (2013). Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- 📍 Ferrara, E., Meo, P. De, Fiumara, G., & Baumgartner, R. (2014). Web data extraction, applications and techniques: A survey. *Knowledge-Based Systems*, 70:301-323.
- 📍 Ganin, Y., and Lempitsky, V. (2014). N4-fields: Neural network nearest neighbor fields for image transforms. In *ACCV*.
- 📍 Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*.
- 📍 Gupta, S., Girshick, R., Arbelaez, P., and Malik, J. (2014). Learning rich features from RGB-D images for object detection and segmentation. In *ECCV*. Springer.
- 📍 Hariharan, B., Arbelaez, P., Girshick, R., and Malik, J. (2014). Simultaneous detection and segmentation. In *European Conference on Computer Vision (ECCV)*.
- 📍 Hilbring, D., Moumtzidou, A. Mossgraber, J., and Vrochidis, S. (2013). Semantically Enriching an Open Source SOS Implementation for Accessing Heterogeneous Environmental Data Sources. *The Global Conference for Open Source Geospatial Software*, Nottingham, UK, September 17-21.
- 📍 Hsu, C.-N., Dung, M.-T. (1998). Generating finite-state transducers for semi-structured data extraction from the web. *Information systems*, 23(8):521-538.
- 📍 Huiskes, M. J., Thomee, B., & Lew, M. S. (2010). New trends and ideas in visual concept detection: the MIR flickr retrieval evaluation initiative. In *Proceedings of the 2010 International Conference on Multimedia Information Retrieval, MIR '10*, pp. 527–536, New York, NY, USA. ACM.
- 📍 Irfanullah, K. H., Sattar, Q., & Sadaqat-ur-Rehman, A. A. (2013). An Efficient Approach for Sky Detection. *IJCSI International Journal of Computer Science Issues*, ISSN (Print), 1694-0814.
- 📍 Ivak, Max (2016). REST vs XML-RPC vs SOAP – pros and cons, <http://maxivak.com/rest-vs-xml-rpc-vs-soap/>
- 📍 Jacobs, N., Roman, N., Pless, R. (2007). Consistent Temporal Variations in Many Outdoor Scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- 📍 Jegou, H., Douze, M., Schmid, C., Perez, P. (2010). Aggregating local descriptors into a compact image representation. In *IEEE on Computer Vision and Pattern Recognition (CVPR 2010)*. pp. 3304-3311. San Francisco, CA.
- 📍 Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*.
- 📍 Jurie, F., Triggs, B. (2005). Creating efficient codebooks for visual recognition. In *10th IEEE International Conference on Computer Vision, ICCV '05*, vol. 1, pp. 604–610.
- 📍 Kanaoka, K., Fujii, Y., Toyama, M. (2014). Ducky: a data extraction system for various structured web documents. In *Proceedings of the 18th International Database Engineering & Applications Symposium*, pages 342-347. ACM.

D3.1: Environmental node discovery, indexing and data acquisition

- ♥ Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.
- ♥ Kumar, Pradeep (2014). *Webservices Overview : XML RPC, SOAP and REST*. Presentation on Webservices expert speech on Techgig webinars
- ♥ Kushmerick, N. (1997). *Wrapper induction for information extraction*. PhD thesis, University of Washington.
- ♥ Long, J., Evan, S., and Trevor, D. (2015). Fully convolutional networks for semantic segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015.
- ♥ Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, vol. 60, pp. 91–110.
- ♥ Magrieta, M. D. (1994). *Understanding planning for effective decision support: a cognitive task analysis of nurse scheduling*. Thesis for Doctorate in Business, University of Groningen.
- ♥ Markatopoulou, F., Mezaris, V., & Patras, I. (2015). Cascade of classifiers based on binary, non-binary and deep convolutional network descriptors for video concept detection. In *Proc. of the IEEE Int. Conf. on Image Processing (ICIP 2015)*, pp. 1786–1790.
- ♥ McCallum, A., Nigam, K., Rennie, J. and Seymore, K. (1999). A Machine Learning Approach to Building Domain-Specific Search Engines. *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pp. 662 – 667.
- ♥ McCallum, A., Nigam, K., Rennie, J. and Seymore, K. (1999). *Building Domain-Specific Search Engines with Machine Learning Techniques*.
- ♥ Menemenis, F., Papadopoulos, S., Bratu, B., Waddington, S., and Kompatsiaris, Y. (2008). AQUAM: Automatic Query Formulation Architecture for Mobile Applications. In *Proceedings of the 7th international Conference on Mobile and Ubiquitous Multimedia (Umea, Sweden, December 3 - 5)*. ACM, New York, NY.
- ♥ Moutzidou, A., Vrochidis, S., Chatzilari, E., Kompatsiaris, I. (2013). Discovery of environmental nodes based on heatmap recognition. In: *Proceedings of the 20th IEEE International Conference on Image Processing (ICIP 2013)*
- ♥ Moutzidou, A., Vrochidis, S., Kompatsiaris, I., Tonelli, S., Pianta, E., Tarvainen, V., Karppinen, A., Myllynen, M., Koskentalo, T. (2010). Analysis of the codification of metadata and content in environmental service pages and design of the functional index repository. *Personalized Environmental Service Configuration and Delivery Orchestration (PESCaDO) deliverable*.
- ♥ Moutzidou, A., Vrochidis, S., Tonelli, S., Kompatsiaris, I., Pianta, E. (2012). Discovery of environmental nodes in the web. In: *Multidisciplinary Information Retrieval, Proceedings of the 5th International Retrieval Facility Conference (IRFC 2012)*, LNCS, vol 7356, pp 58–72
- ♥ Muslea, I., Minton, S., & Knoblock, C. (1998). Stalker: Learning extraction rules for semistructured, web-based information sources. In *Proceedings of AAAI-98 Workshop on AI and Information Integration*, pages 74-81. AAAI Press Menlo Park, CA.
- ♥ Ning, F., Delhomme, D., LeCun, Y., Piano, F., Bottou, L., and Barbano, P. E. (2005). Toward automatic phenotyping of developing embryos from videos. *Image Processing, IEEE Transactions on*, 14(9):1360–1371.
- ♥ OGC Network, (2011). *SOS 2.0 Tutorial*. [online] Available at: http://www.ogcnetwork.net/SOS_2_0/tutorial.
- ♥ OGC, (2016). *Sensor Observation Service*. [online] Available at: <http://www.opengeospatial.org/standards/sos>.
- ♥ Oliveira, B., Vasco S., and Belo, O. (2013). Processing XML with Java—a performance benchmark. *International Journal of New Computer Architectures and their Applications (IJNCAA) 3.1 (2013)*: 72-85.

D3.1: Environmental node discovery, indexing and data acquisition

- 📍 Olston, C., Najork, M. (2010). Web crawling. *Found Trends Inf Retr* 4(3):175–246
- 📍 Oquab, M., Bottou, L., Laptev, I., and Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. in *CVPR*, 2014.
- 📍 OSGeo Project. *What is PostGIS?* [online] Available at: <http://postgis.refrains.net/>
- 📍 Over, P., Awad, G., Kraaij, W., Smeaton, A.F. (2007). TRECVID 2007–overview. In: *TRECVID 2007 workshop participants notebook papers*. National Institute of Standards and Technology (NIST)
- 📍 Oyama, S., Kokubo, T., and Ishida, T. (2004). Domain-Specific Web Search with Keyword Spices. *IEEE Transactions on Knowledge and Data Engineering*, vol. 16 (1), pp. 17 – 27.
- 📍 Oyama, S., Kokubo, T., Ishida, T. (2004). Domain-specific web search with keyword spices. *IEEE Trans Knowl Data Eng* 16(1):17–27
- 📍 Oyama, S., Kokubo, T., Ishida, T., Yamada, T., and Kitamura, Y. (2001). Keyword Spices: A New Method for Building Domain-Specific Web Search Engines. *Proceedings of the 17th International Joint Conferences on Artificial Intelligence (IJCAI-01)*, pp. 1457-1463.
- 📍 Pal A., Tomar S. D., Shrivastava S.C. (2009). Effective Focused Crawling Based on Content and Link Structure Analysis. *International Journal of Computer Science and Information Security*, Vol. 2, No. 1.
- 📍 Pant, G., Srinivasan, P. (2005). Learning to crawl: Comparing classification schemes. *ACM Trans Inf Syst* 23(4):430–462
- 📍 Pant, G., Srinivasan, P. (2006). Link contexts in classifier-guided topical crawlers. *IEEE Trans Knowl Data Eng* 18(1):107–122
- 📍 Papadopoulos, S., Diplaris, S., Tsampoulatidis, Y., Kompatsiaris, Y., Gkatziaki, V., & Mills, R. (2015). Scalable Analytics Techniques for User Contributions v2. The Wikirate Project - Crowdsourcing and data mining, empowering all stakeholders of companies to play a role as more ethical economic citizens, deliverable.
- 📍 Percivall, G. (2011). Applying SWE to the Internet of Things (IoT). 77th OGC Technical Committee, Taichung, Taiwan, June 12.
- 📍 Perronnin, F., Sanchez, J., Mensink, T. (2010). Improving the Fisher kernel for large-scale image classification. In: *11th Eur. Conf. on Computer Vision: Part IV*. pp. 143-156. Springer-Verlag.
- 📍 Pinheiro, P. H., and Collobert, R. (2014). Recurrent convolutional neural networks for scene labeling. In *ICML*.
- 📍 Plake, C., Schiemann, T., Pankalla, M., Hakenberg, J., & Leser, U. (2006). ALIBABA: PubMed as a graph. *Bioinformatics* 22.19, 2444-2445.
- 📍 Qiu, G. (2002). Indexing chromatic and achromatic patterns for content-based colour image retrieval. *Pattern Recognition* 35, pp. 1675-1686.
- 📍 Russakovsky, O., Deng, J., Su, H. et al. (2015). ImageNet Large Scale Visual Recognition Challenge. *Int. Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252.
- 📍 Shakes, J., Langheinrich, M. and Etzioni, O. (1997). Dynamic reference sifting: a case study in the homepage domain. In *Proceedings of the 6th International World Wide Web Conference (WWW6)*, pp. 189– 200.
- 📍 Shapiro, L. G., & Stockman, G. C. (2001). *Computer Vision: Theory and Applications*.
- 📍 Simonyan, K., Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv technical report*, 2014.
- 📍 Smeaton, A.F., Over, P., and Kraaij, W. (2009). High-Level Feature Detection from Video in TRECVID: a 5-Year Retrospective of Achievements. In *Ajay Divakaran, editor, Multimedia Content Analysis, Theory and Applications*, pp. 151–174. Springer Verlag, Berlin.

D3.1: Environmental node discovery, indexing and data acquisition

- ♥ Snoek, C. G. M., Worring, M. (2009). Concept-Based Video Retrieval. *Foundations and Trends in Information Retrieval*, vol. 2, no. 4, pp. 215– 322.
- ♥ Snoek, C.G.M., Fontijne, D., van de Sande, K. E.A. , et al. (2015). Qualcomm research and university of amsterdam at trecvid 2015: Recognizing concepts, objects, and events in video. In *Proc. of TRECVID 2015*. NIST, USA, 2015.
- ♥ SOS 2.0 Tutorial, http://www.ogcnetwork.net/SOS_2_0/tutorial
- ♥ Szegedy, C. et al.(2015). Going deeper with convolutions. In *CVPR 2015*.
- ♥ Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014). Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*.
- ♥ Tompson, J., Jain, A., LeCun, Y., and Bregler, C. (2014). Joint training of a convolutional network and a graphical model for human pose estimation. *CoRR*, abs/1406.2984.
- ♥ Tsikrika, T., Latas, A., Moumtzidou, A., Chatzilari, E., Vrochidis, S. and Kompatsiaris, I. (2015). Discovery of Environmental Web Resources Based on the Combination of Multimedia Evidence. *2nd International Workshop on Environmental Multimedia Retrieval (EMR 2015)*, Shanghai, China, June 23.
- ♥ Tsikrika, T., Moumtzidou, A., Vrochidis, S. and Kompatsiaris, I. (2014). Focused Crawling of Environmental Web Resources: A Pilot Study on the Combination of Multimedia Evidence. *Proceedings of 1st International Workshop on Environmental Multimedia Retrieval (EMR 2014)*, Glasgow, UK, April 1.
- ♥ Tsikrika, T., Moumtzidou, A., Vrochidis, S., & Kompatsiaris, I. (2016). Focussed crawling of environmental Web resources based on the combination of multimedia evidence. *Multimedia Tools and Applications*, 75(3), 1563-1587.
- ♥ Tuytelaars, T., Mikolajczyk, K. (2008). *Local Invariant Feature Detectors: A Survey*. Now Publishers Inc., Hanover, MA, USA.
- ♥ Vedaldi, A. and Lenc, K. (2015). Matconvnet – convolutional neural networks for matlab. In *Proceeding of the ACM International Conference on Multimedia*.
- ♥ Wang, J., Lochovsky, F. H. (2003). Data extraction and label assignment for web databases. In *Proceedings of the 12th international conference on World Wide Web*, pages 187-196. ACM.
- ♥ Wei, A., Pei, Y., and Zha, H. (2012). Random-sampling-based spatial-temporal feature for consumer video concept classification. In *Proc. of the 2012 IEEE International Conference on Image Processing (ICIP)*, pp. 1861–1864.
- ♥ Wikipedia, (2010). *Empirical Research*. [online] Available at: http://en.wikipedia.org/wiki/Empirical_research.
- ♥ Wikipedia, (2010). *Empirical*. [online] Available at: <http://en.wikipedia.org/wiki/Empirical>.
- ♥ Wikipedia, (2010). *PostGIS*. [online] Available at: <http://en.wikipedia.org/wiki/PostGIS>.
- ♥ Wober, K. (2006). Domain Specific Search Engines. In *Travel Destination Recommendation Systems: Behavioral Foundations and Applications*, edited by D. R. Fesenmaier, H. Werthner, and K. Wober, Cambridge, MA: CAB International, pp. 205-26.
- ♥ Wong, T.-L., Lam, W. (2010). Learning to adapt web information extraction knowledge and discovering new attributes via a bayesian approach. *Knowledge and Data Engineering, IEEE Transactions on*, 22(4):523-536, 2010.
- ♥ Xiao, J., Hays, J., Ehinger, K., Oliva, A., and Torralba, A. (2010). SUN Database: Large-scale Scene Recognition from Abbey to Zoo. *IEEE Conference on Computer Vision and Pattern Recognition*.
- ♥ Yank, Kevin (2002). Web Services Demystified. <http://www.sitepoint.com/article/web-services-demystified>.

D3.1: Environmental node discovery, indexing and data acquisition

- 📍 Zheng, H.T., Kanga, B.Y., and Hong-Gee Kim (2008). An ontology-based approach to learnable focused crawling. *Information Sciences*, Vol. 178, Issue 23, pp. 4512-4522.
- 📍 Zhijie, Z., Qian, W., Huadong, S., Xuesong, J., Qin, T., & Xiaoying, S. (2015). A Novel Sky Region Detection Algorithm Based On Border Points. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 8(3), 281-290.

9 Appendix

The appendix contains the following:

9.1 Empirical Study of Web sites

In this section, we provide both a brief description of the dataset and a detailed study of the sites.

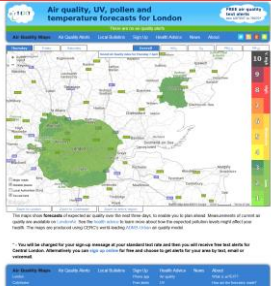
9.1.1 Brief Description of Dataset

S/N	URL	Content	Location	Language	Time	Extraction Method for Measurements
1	http://www.airtext.info/	Air Quality (PM _{2.5} , PM ₁₀ , O ₃ , NO ₂)	London, Colchester	English	Forecast data	Image processing
2	http://uk-air.defra.gov.uk/latest/currentlevels	Air Quality (PM _{2.5} , PM ₁₀ , O ₃ , SO ₂ , NO ₂)	United Kingdom	English	Real time & Forecast data	HTML parsing / Image processing
3	http://www.cleanerairforlondon.org.uk/londons-air/air-quality-data	Air Quality (PM _{2.5} , PM ₁₀ , O ₃ , SO ₂ , NO ₂ , CO)	London	English	Real time & Historical data	-
4	https://airvisual.com/	Air Quality (PM _{2.5} , PM ₁₀ , O ₃ , CO)	World	English	Real time & Historical & Forecast data	Parsing of dynamic pages
5	http://aqicn.org/city/	Air Quality (CO, SO ₂ , O ₃ , NO ₂ , PM ₁₀ , PM _{2.5})	Cities in the World	English	Real time & Historical & Forecast data	Image processing / HTML parsing
6	https://www.luchtmeetnet.nl/stations	Air Quality (SO ₂ , NO ₂ , O ₃ , NO, Smoke, Benzene, Toluene, CO, SO ₂ , H ₂ S, UFP, NH ₃ , PM _{2.5} , PM ₁₀)	Netherlands	English, Dutch	Real time & Historical data	HTML parsing / Image processing
7	http://www.irceline.be/en/	Air Quality (PM _{2.5} , PM ₁₀ , BC, O ₃ , NO ₂)	Belgium	English	Real time & Historical & Forecast data	HTML parsing / Image processing
8	http://www.luftkvalitet.info/home.aspx	Air Quality (PM _{2.5} , PM ₁₀ , O ₃ , SO ₂ , NO ₂ , CO)	Norway	Norwegian	Real time & Historical data	HTML parsing / Image processing
9	http://www.eea.europa.eu/data-and-maps/data/airbase-the-european-air-quality-database-7	Air Quality (SO ₂ , NO ₂ , O ₃ , NO, Benzene, CO, SO ₂ , PM _{2.5} , PM ₁₀ , Total Suspended Particulates)	Europe	English	Historical data	XLS/CSV parsing
10	https://luftdaten.brandenburg.de/home/-/bereich/datenexport	Air Quality (SO ₂ , O ₃ , NO ₂ , NO, CO, PM ₁₀ , PM _{2.5})	Germany	German	Historical + previous week data	XLS/CSV parsing
11	http://www.lanuv.nrw.de/luft/immissionen/aktluftqual/eu_luft_akt/	Air Quality (O ₃ , SO ₂ , NO ₂ , PM ₁₀)	Germany	German	Real time & Forecast data	HTML parsing / Image processing
12	http://www.airqualitynow.eu/index.php	Air Quality (NO ₂ , O ₃ , PM ₁₀ , PM _{2.5} , SO ₂ , CO)	Europe	English	Real time & Forecast data	HTML parsing
13	http://macc-raq-op.meteo.fr/index.php	Air Quality (SO ₂ , O ₃ , NO ₂ , CO, PM ₁₀ , PM _{2.5})	Europe	English	Forecast data	Image Processing / Parsing of data in netCDF or Grib format
14	http://opendata.thessaloniki.gr/group/pollution	Air Quality (SO ₂ , O ₃ , NO ₂ , NO, CO, PM ₁₀ , PM _{2.5})	Thessaloniki	Greek	Historical data	XLS/CSV parsing

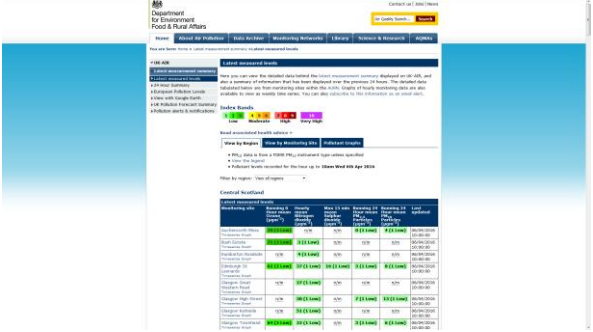
D3.1: Environmental node discovery, indexing and data acquisition

15	http://www.airqualitynews.com/	General Air Quality news and information	Europe	English	NA	-
16	http://cleanair.london/#	General Air Quality news and information	World	English	NA	-
17	http://www.ypeka.gr/Default.aspx?tabid=708&locale=en-US&language=el-GR	Air Quality (CO, SO ₂ , NO ₂ , O ₃ , PM ₁₀)	Attiki/ Greece	English/ Greek	Real time & Near Real Time data	HTML parsing
18	http://www.umweltbundesamt.de/daten/luftbelastung/aktuelle-luftdaten	Air Quality (CO, O ₃ , SO ₂ , NO, NO ₂ , PM _{2.5} , PM ₁₀)	Germany	German	Real time & Historical data	HTML parsing / Image processing


9.1.2 Detailed Study

Site S/N = 1		
General Information		
Content	Air Quality (Forecast data)	
Location	London, Colchester	
URL	http://www.airtext.info/	
Site Language	English	
Format	Dynamic pages with heatmaps	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	Air Quality (PM2.5, PM10, O3, NO2) Air pollutant name is returned as part of a javascript method	HTML parsing of dynamic content
Measurement	Measurement values are by default retrieved from the heatmap	Image processing
Time	Time/ date are returned as part of javascript method	HTML parsing of dynamic content
Exact Location	Location is by default retrieved from the heatmap	Image processing
Comment		


D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 2		
General Information		
Content	Air Quality (Real time & Forecast data)	
Location	United Kingdom	
URL	http://uk-air.defra.gov.uk/latest/currentlevels	
Site Language	English	
Format	Static pages that use tables for holding measurement values and other interesting clues (i.e. date, location)	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	<p>Air Quality (PM_{2.5}, PM₁₀, O₃, SO₂, NO₂)</p> <p>Air quality aspects for current data are embedded into the tables columns</p> <p>For forecast data only the Air quality index is provided.</p>	HTML table parsing
Measurement	<p>Real time measurements can be found inside tables. Each row usually contains data for specific date & location. Different columns are used for each aspect</p> <p>Forecast measurements can be found in heatmaps.</p>	<p>HTML parsing</p> <p>Image processing</p>
Time	Time is located inside specific column of the tables	HTML parsing
Exact Location	<p>Location is located inside specific column of the tables</p> <p>For forecast data, location can be found directly from maps.</p>	<p>HTML parsing</p> <p>Image processing</p>
Comment	-	-

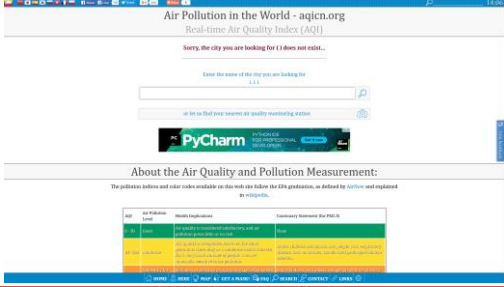
D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 3		
General Information		
Content	Air Quality (Real time & Historical data)	
Location	London	
URL	http://www.cleanerairforlondon.org.uk/londons-air/air-quality-data	
Site Language	English	
Format	Data are linked to "www.londonair.org.uk"	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	-	-
Measurement	-	-
Time	-	-
Exact Location	-	-
Comment	-	--

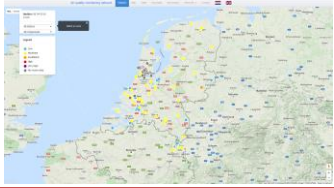
D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 4		
General Information		
Content	Air Quality (Real time & Historical & Forecast data)	
Location	World (hackAIR specific: Belgium, UK, Germany)	
URL	https://airvisual.com/	
Site Language	English	
Format	Dynamic pages using javascript	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	<p>Air Quality (PM_{2.5}, PM₁₀, O₃, CO, AQI)</p> <p>For the real time and the forecast data only the Air Quality Index is provided</p> <p>The pollutant names are provided as titles above the graphs</p>	Parsing of dynamically created pages
Measurement	<p>Real time and forecast data can be obtained by dynamic text</p> <p>Historical data can be obtained from dynamic graph</p>	Parsing of dynamically created pages
Time	<p>Time in real time and forecast data can be obtained by dynamic text</p> <p>Time in historical data can be obtained from dynamic graph</p>	Parsing of dynamically created pages
Exact Location	Exact location can be extracted from URL	URL parsing
Comment	Data are expected to be available through a REST API in the upcoming months.	

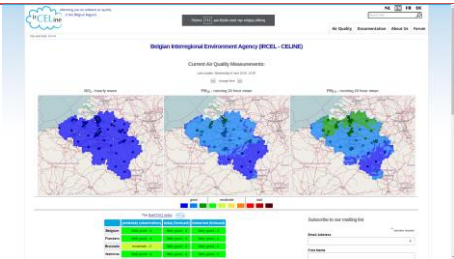
D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 5		
General Information		
Content	Air Quality (Real time, Historical & Forecast data)	
Location	Several cities in the world	
URL	http://aqicn.org/city/	
Site Language	English	
Format	Static pages (real time and forecast data), and graphs (Historical data)	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	<p>Air Quality (Air Quality Index, CO, SO₂, O₃, NO₂, PM₁₀, PM_{2.5})</p> <p>Air quality aspects for current data and historical data are embedded into specific elements ()</p> <p>Forecast data are provided only for Air Quality Index</p>	HTML parsing
Measurement	<p>Air quality aspects for historical data are located inside graphs.</p> <p>Current value and minimum and maximum values for each aspect for the last 48 hours are given in text format.</p> <p>Forecast data are given in text format.</p>	<p>Image processing</p> <p>HTML parsing</p> <p>HTML parsing</p>
Time	<p>Time is provided by found in central part of the page and is updated with a javascript</p> <p>or at the top of the html page</p>	HTML parsing for dynamic content
Exact Location	Location can be found in the URL of the page and in several places of the page	HTML parsing
Comment		

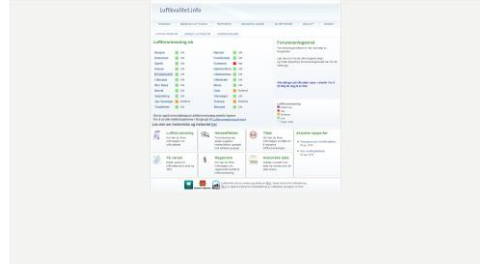
D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 6		
General Information		
Content	Air Quality (Real time & Historical data)	
Location	Netherlands	
URL	https://www.luchtmeetnet.nl/stations	
Site Language	English, Dutch	
Format	Both maps and static pages	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	<p>Air Quality (SO₂, NO₂, O₃, NO, Smoke, Benzene, Toluene, CO, SO₂, H₂S, UFP, NH₃, PM_{2.5}, PM₁₀)</p> <p>Air quality aspects for current + historical data are embedded either:</p> <ul style="list-style-type: none"> into the columns of tables in drop down lists 	HTML parsing
Measurement	<p>In tables, each cell contains a measurement and rows cover different regions and columns different pollutants.</p> <p>Heatmaps provide also historical and real time data.</p>	<p>HTML parsing</p> <p>Image processing</p>
Time	<p>For tables, time is found in the URL e.g. https://www.luchtmeetnet.nl/tabel?datetime=2016-04-06%2010:00:00</p> <p>For heatmaps, time is found in drop down lists</p>	HTML parsing
Exact Location	<p>In tables, each cell contains a measurement and rows cover different regions and columns different pollutants.</p> <p>In maps, location can be found directly from them.</p>	<p>HTML parsing to obtain data from text</p> <p>Image processing</p>
Comment		

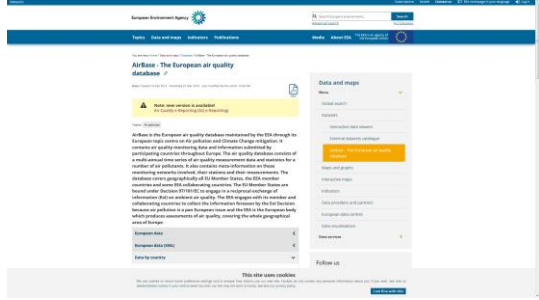
D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 7		
General Information		
Content	Air Quality (Real time & Historical & Forecast data)	
Location	Belgium	
URL	http://www.irceline.be/en/	
Site Language	English	
Format	Static pages (real time and historical data) and heatmaps (real time & forecast data)	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	<p>Air Quality (PM_{2.5}, PM₁₀, BC, O₃, NO₂)</p> <p>Air quality aspects for current data are embedded into the tables columns</p> <p>For forecast data only the Air quality index is provided.</p>	HTML table parsing
Measurement	<p>Real time measurements can be found inside tables. Depending on the table, rows either contain data for specific date & location and columns for different aspects or contain data for specific location and columns for different dates while the whole table refers to specific pollutant.</p> <p>Forecast measurements can be found in heatmaps.</p>	<p>HTML parsing</p> <p>Image processing</p>
Time	Time is located inside specific column of the tables	HTML parsing
Exact Location	<p>Location is located inside specific column of the tables</p> <p>For forecast data, location can be found directly from maps.</p>	<p>HTML parsing</p> <p>Image processing</p>
Comment	-	-

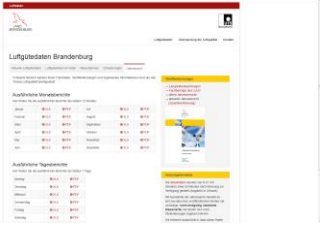
D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 8		
General Information		
Content	Air Quality (Real time & Historical data)	
Location	Norwegian	
URL	http://www.luftkvalitet.info/home.aspx	
Site Language	Norway	
Format	Static pages and maps	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	Air Quality (PM _{2.5} , PM ₁₀ , O ₃ , SO ₂ , NO ₂ , CO) Air pollutant names for current and historical data can be found in a drop down list.	HTML parsing
Measurement	Air pollutant values for current data can be found into tables. Air pollutant values for historical data into graphs.	HTML parsing Image processing
Time	Air pollutant values for current data can be found into tables. Air pollutant values for historical data into graphs.	HTML parsing Image processing
Exact Location	Location can be found either on the tables (specific column) or inside a specific html element (e.g. <div>)	HTML parsing
Comment		

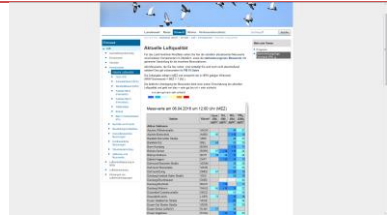
D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 9		
General Information		
Content	Air Quality (Historical data)	
Location	Europe	
URL	http://www.eea.europa.eu/data-and-maps/data/airbase-the-european-air-quality-database-7	
Site Language	English	
Format	Documents	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	Air Quality (SO ₂ , NO ₂ , O ₃ , NO, Benzene, CO, SO ₂ , PM _{2.5} , PM ₁₀ , Total Suspended Particulates) Air quality aspects are embedded into the tables.	XLS/CSV parsing
Measurement	Air quality aspects are embedded into the tables.	XLS/CSV parsing
Time	Time is located inside specific columns of the tables	XLS/CSV parsing
Exact Location	Air quality aspects are embedded into the tables.	XLS/CSV parsing
Comment		


D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 10		
General Information		
Content	Air Quality (Historical & previous week data)	
Location	Germany	
URL	https://luftdaten.brandenburg.de/home/-/bereich/datenexport	
Site Language	German	
Format	Documents (.xls and .pdf)	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	Air Quality (SO ₂ , O ₃ , NO ₂ , NO, CO, PM ₁₀ , PM _{2.5}) Each column in xls file is a different aspect.	XLS/CSV parsing
Measurement	Each row represents another area of Germany and each column another pollutant	XLS/CSV parsing
Time	Time is located on the top of the xls file	XLS/CSV parsing
Exact Location	Each row represents another area of Germany	XLS/CSV parsing
Comment		

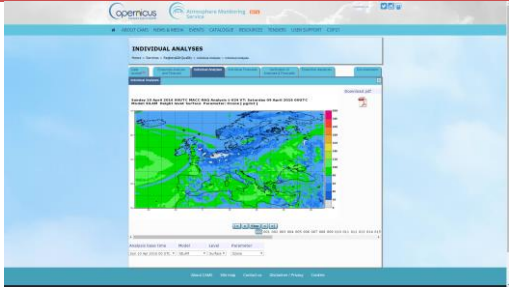
D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 11		
General Information		
Content	Air Quality (Real time & Forecast data)	
Location	Germany	
URL	http://www.lanuv.nrw.de/luft/immissionen/aktluftqual/eu_luft_akt/	
Site Language	German	
Format	Both static and dynamic pages depending on the type of data	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	<p>Air Quality (O₃, SO₂, NO₂, PM₁₀)</p> <p>Air quality aspects for real time data are embedded into the table columns.</p> <p>For forecast data, pollutant names can be found in several places of the page including some drop down lists</p>	HTML parsing
Measurement	<p>Real time measurements can be found inside tables. Each row usually represents another area of Germany and each columns another pollutant.</p> <p>Forecast data are captured into heatmaps.</p>	<p>HTML parsing</p> <p>Image processing</p>
Time	Time is located on top of each table	HTML parsing
Exact Location	<p>For real time measurements location can be found in specific column.</p> <p>For forecast data, location can be found directly from maps.</p>	<p>HTML parsing</p> <p>Image processing</p>
Comment		

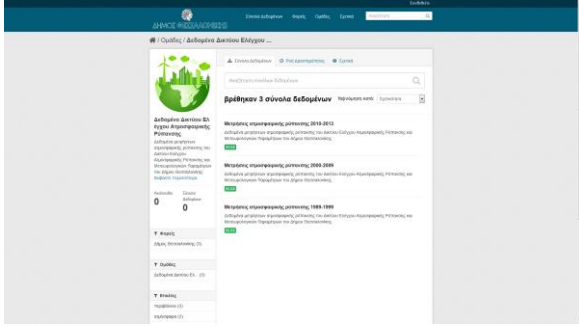
D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 12		
General Information		
Content	Air Quality (Real time & Forecast data)	
Location	Europe	
URL	http://www.airqualitynow.eu/index.php	
Site Language	English	
Format	Static pages	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	Air Quality (SO ₂ , NO ₂ , TRS, PM ₁₀) Air quality aspects can be found inside specific table columns.	HTML parsing
Measurement	Measurements can be found inside specific table columns.	HTML parsing
Time	Time can be found inside specific html elements (e.g.).	HTML parsing
Exact Location	Exact location can be found both in URL and in the tile of the web page.	HTML parsing
Comment		


D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 13		
General Information		
Content	Air Quality (Forecast data)	
Location	Europe	
URL	http://macc-raq-op.meteo.fr/index.php	
Site Language	English	
Format	Static pages and data retrieval in netCDF or Grib format	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	Air Quality (SO ₂ , O ₃ , NO ₂ , CO, PM ₁₀ , PM _{2.5})	HTML Parsing
Measurement	<p>Air pollutant can be found in drop down list</p> <p>Measurements are embedded in heatmaps.</p> <p>Data are also provided in numerical format using specific formatting.</p>	<p>Image processing of heatmaps</p> <p>Parsing of netCDF or Grib format</p>
Time	<p>Time can be found in drop down list</p> <p>In case of making of using directly the raw data, time can be found also in them.</p>	<p>HTML parsing</p> <p>Parsing of netCDF or Grib format</p>
Exact Location	<p>Location in case of heatmaps can be directly retrieved from the map.</p> <p>In case of making of using directly the raw data, location can be found also in them.</p>	<p>Image processing of heatmaps</p> <p>Parsing of netCDF or Grib format</p>
Comment		


D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 14		
General Information		
Content	Air Quality (Historical data)	
Location	Thessaloniki	
URL	http://opendata.thessaloniki.gr/group/pollution	
Site Language	Greek	
Format	Documents (.xls)	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	Air Quality (SO ₂ , O ₃ , NO ₂ , NO, CO, PM ₁₀ , PM _{2.5}) Each column in xls file is a different aspect.	XLS/CSV parsing
Measurement	Each row represents another date and each sheet represents another station.	XLS/CSV parsing
Time	Time is located on a specific column of file	XLS/CSV parsing
Exact Location	Each sheet is another station represents another station inside Thessaloniki.	XLS/CSV parsing
Comment		

D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 15		
General Information		
Content	General Air Quality news and information	
Location	Europe	
URL	http://www.airqualitynews.com/	
Site Language	English	
Format	Text articles/ HTML	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	-	-
Measurement	-	-
Time	-	-
Exact Location	-	-
Comment	The site contains Air Quality news/ articles and information. No measurements of any kind are provided.	

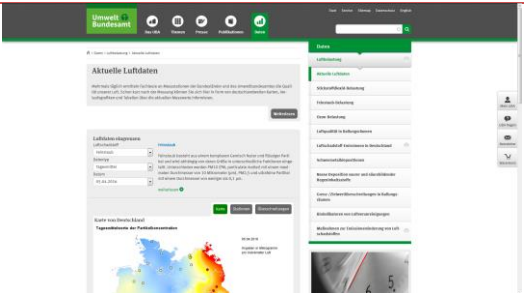
D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 16		
General Information		
Content	General Air Quality news and information	
Location	World	
URL	http://cleanair.london/#	
Site Language	English	
Format	Text articles/ HTML	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	-	-
Measurement	-	-
Time	-	-
Exact Location	-	-
Comment	The site contains Air Quality news/ articles and information. No measurements of any kind are provided.	

D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 17		
General Information		
Content	Air Quality (Real time data & near real time data)	
Location	Attiki	
URL	http://www.ypeka.gr/Default.aspx?tabid=708&locale=en-US&language=el-GR	
Site Language	Greek	
Format	Static pages	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	Air Quality (CO, SO ₂ , NO ₂ , O ₃ , PM ₁₀) Air quality aspects can be found in specific column of the tables.	HTML parsing
Measurement	All data are provided inside tables. Each row usually represents another area/ station and each column another time/date.	HTML parsing
Time	Time is located on the header of the table specific columns of the tables.	HTML parsing
Exact Location	Exact location of Belgium can be extracted from the rows of tables.	HTML parsing
Comment		

D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 18		
General Information		
Content	Air Quality (Real time & Historical data)	
Location	Germany	
URL	http://www.umweltbundesamt.de/daten/luftbelastung/aktuelle-luftdaten	
Site Language	German	
Format	Static pages and heatmaps	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	Air Quality (CO, O ₃ , SO ₂ , NO, NO ₂ , PM _{2.5} , PM ₁₀) Air quality aspects are elements of down lists.	HTML parsing
Measurement	Measurements can be found in tables and heatmaps. Each row refers to another location and there is a single measurement per day. Heatmaps contain the same information with tables.	HTML parsing Image processing
Time	There is a single measurement per day	HTML parsing
Exact Location	Exact location of Belgium can be extracted from the rows of tables or from the map itself.	HTML parsing
Comment		


9.2 Empirical Study of Web services

In this section, we provide both a brief description of the dataset and a detailed study of the services and their sites (if they exist).

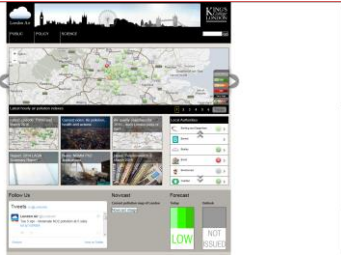
9.2.1 Brief Description of Dataset

S/N	URL	Content	Location	Language	Time	Extraction Method for Measurements
1	https://openaq.org/#/	Air Quality (PM _{2.5} , PM ₁₀ , O ₃ , SO ₂ , NO ₂ , CO)	Netherlands, United Kingdom	English	Real time & Historical data	JSON parsing
3	https://www.londonair.org.uk/LondonAir/Default.aspx	Air Quality (PM _{2.5} , PM ₁₀ , O ₃ , SO ₂ , NO ₂ , CO)	London	English	Real time & Historical data	Image processing / JSON parsing
16	http://www.envdimosthes.gr/deltioPop.php or http://www.envdimosthes.gr/getData.php	Air Quality (SO ₂ , NO ₂ , CO, O ₃ , PM ₁₀)	Thessaloniki	Greek	Near Real time data	HTML parsing / JSON Parsing


9.2.2 Detailed Study

Site S/N = 1		
General Information		
Content	Air Quality (Real time & Historical data)	
Location	Netherlands, United Kingdom	
URL	https://openaq.org/#/	
Site Language	English	
Format	JSON response from REST API + csv files	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	<p>Air Quality (PM_{2.5}, PM₁₀, O₃, SO₂, NO₂, CO)</p> <p>Aspect type can be provided as parameter to the REST query using as parameter: "parameter" (e.g.: "parameter=pm10") or it can be found inside the response of the REST API.</p>	<p>Calling REST API by sending GET query to https://api.openaq.org/v1/</p> <p>JSON parsing of the result returned by a query like: https://api.openaq.org/v1/measurements?city=London&parameter=pm10</p>
Measurement	<p>The measurement value is provided in double value inside the json response</p>	<p>JSON parsing of the result returned by a query like: https://api.openaq.org/v1/measurements?city=London&parameter=pm10</p>
Time	<p>Time can be provided as parameter to the REST query using as parameter: "date_from" and "date_to" (e.g.: "date_from =2016-04-03") or it can be found inside the response of the REST API.</p> <p>When time is not given, data for the current date are returned</p>	<p>JSON parsing of the result returned by a query like: https://api.openaq.org/v1/measurements?city=London&parameter=pm10</p>
Exact Location	<p>Location is provided as parameter to the REST query using as parameter either a country or a specific city. The parameters used are: "country" and "city" (e.g.: "country=GB", "city=London") or it can be found inside the response of the REST API.</p>	<p>JSON parsing of the result returned by a query like: https://api.openaq.org/v1/measurements?city=London&parameter=pm10</p>
Comment	<p>Queries can be created for:</p> <ul style="list-style-type: none"> retrieving the countries like: https://api.openaq.org/v1/countries retrieving cities of specific country like: https://api.openaq.org/v1/cities?country=GB retrieving measurements of specific city and aspect https://api.openaq.org/v1/measurements?city=London&parameter=pm10 	

D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 3		
General Information		
Content	Air Quality (Real time & Historical data)	
Location	London	
URL	https://www.londonair.org.uk/LondonAir/Default.aspx	
Site Language	English	
Format	Dynamic pages, heatmaps and REST API	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	<p>Air Quality (PM_{2.5}, PM₁₀, O₃, SO₂, NO₂, CO)</p> <p>Air pollutant names can be found from dynamic drop down lists.</p> <p>Pollutants can be found inside the REST response.</p>	<p>HTML parsing</p> <p>JSON parsing (Result of REST API: http://api.erg.kcl.ac.uk/AirQuality/Information/Species)</p>
Measurement	<p>In case of heatmaps, values are embedded into them using colours.</p> <p>In case of REST API, the values are part of the response.</p>	<p>Image processing</p> <p>JSON parsing</p>
Time	<p>In case of heatmaps, date/time can be found in several parts of the webpage.</p> <p>In case of REST API, time/date is part of the response.</p>	<p>Image processing</p> <p>JSON parsing</p>
Exact Location	<p>In case of heatmaps, location is found directly from the map.</p> <p>In case of REST API, location is part of the response.</p>	<p>Image processing</p> <p>JSON parsing</p>
Comment		

D3.1: Environmental node discovery, indexing and data acquisition

Site S/N = 16		
General Information		
Content	Air Quality (Near Real time data)	
Location	Thessaloniki	
URL	http://www.envdimosthes.gr/deltioPop.php or http://www.envdimosthes.gr/getData.php	
Site Language	Greek	
Format	Static page and API	
Site screenshot		
Clues	Information encoding	Methods for extracting information
Aspect Type	Air Quality (SO ₂ , NO ₂ , CO, O ₃ , PM ₁₀) Air quality aspects for current data are embedded into the tables or can be retrieved from JSON file.	HTML parsing or JSON Parsing (after calling REST API)
Measurement	In case of static pages, measurements are found inside tables. Each row represents another pollutant and each column another station in the city of Thessaloniki. In case of json, each pollutant's value can be found in the response.	HTML parsing or JSON Parsing (after calling REST API)
Time	In case of static pages, time/date is found inside specific html elements (e.g.) In case of json, there is a "datetime" attribute.	HTML parsing or JSON Parsing (after calling REST API)
Exact Location	In case of static pages, location is found in specific cells of the table (e.g.). In case of json, there are "lon" and "lng" attributes that point to the coordinates of the station.	HTML parsing or JSON Parsing (after calling REST API)
Comment		

9.3 Empirical Study of Webcams

9.3.1 Sites and portals dataset

In this section, we provide a description of the webcams dataset.

S/N	URL	Location	Percentage of sky visible in the webcam	Type	Surrounding	Where is information located		Comments
						Location	Time/Date	
1	http://www.met.fu-berlin.de/de/wetter/webcam/content/cms.html	Berlin (Germany)	60%	Website	City	Page URL & Inside html tags close to webcam image	Updated daily – embedded in image	-
2	http://download.hrz.tu-darmstadt.de/media/HRZ/mmAG/webcam/	Darmstadt (Germany)	20%	Image	City	Page URL	Updated daily – embedded in image	-

D3.1: Environmental node discovery, indexing and data acquisition

	bcam/s101-webcam02/current-popup.html							
3	http://www.st-michaelis.de/rundgang/turm-blick-live/	Hamburg (Germany)	30%	Flash (website)	Urban background	Page URL & Inside html tags close to webcam image	Live streaming & Embedded in image	-
4	http://www.rbb-online.de/wetter/webcam/rathaus.html	Berlin (Germany)	30%	Website	City	Inside html tags close to webcam image	Inside html tags close to webcam image	-
5	http://www.wetter.com/hd-live-webcams/deutschland/duisburg-sportschule-wedau/5295e2f857bad/	Countries around the World including Germany, Greece, Netherlands	Varies	Flash (website)	Varies	Inside html tags close to webcam image	Live streaming	-
6	http://www.wetterstation-bremen-aumund.de/	Bremen (Germany)	60%	Website	City	Page URL & Inside html tags close to webcam image	Updated daily – embedded in image	-
7	http://www.wdr.de/themen/global/webcams/domcam/domcam_960_live.jpg	Cologne (Germany)	30%	Image	City	-	-	-
8	http://www.mainhattan-webcam.de/	Frankfurt (Germany)	50%	Flash	City	-	Inside html tags close to webcam	Interactive map control
9	http://player.livespotting.tv/?alias=PS_7d900&ch=LS_053f2	Frankfurt (Germany)	40%	Flash (website)	Varies	Embedded in flash video	Live streaming	-
10	http://www.foto-webcam.eu/webcam/muenchen/	Several cities in Germany	Varies	Website	City	Page URL & Inside html tags close to webcam image & Embedded into image	Updated daily – embedded in image & html tags close to image	-
11	https://www2.duesseldorf.de/live-bilder-aus-duesseldorf/webcam-rheinufer.html	Düsseldorf (Germany)	40%	Website	City	Page URL	Updated daily – embedded in image	-
12	http://www.ndr.de/fernsehen/sendungen/mein_nachmittag/wettermelder/Wetter-Webcam-in-	Hannover (Germany)	20%	Website	City	Page URL & Inside html tags close to webcam image	Updated daily	-

D3.1: Environmental node discovery, indexing and data acquisition

	Hannover.wettermelder144.html							
13	http://www.frauenkirche.de/index.php/webcam	Dresden (Germany)	Varies	Website	Varies	META keywords	Updated daily – embedded in image	Updated every 25 sec.
14	http://www.etec.de/etec-webcam	Essen (Germany)	30%	Website	City	Embedded in image	Updated daily – embedded in image	-
15	https://www.dortmund.de/de/leben_in_dortmund/meldungen/dienportal/webcams/dortmunder_u/webcamdortmunderu.html	Dortmund (Germany)	Varies	Flash (website)	City	Page URL & Inside html tags close to webcam image	Updated daily	-
16	http://www.brem-en-tourism.de/webcams	Bremen (Germany)	Varies	Website	Varies	Varies depending on the site hosting the webcam	Varies depending on the site hosting the webcam	-
17	http://cityscope.panomax.com/nuremberg-fernsehturm	Nuremberg (Germany)	0%	Flash	City	Page URL	Live streaming	Interactive map control
18	https://www.skylinewebcams.com/en/webcam/norge/nordland/lofoten/reine.html	Lofoten Islands (Norway)	30%	Flash (website)	Harbour	Inside html tags close to webcam image	Live streaming	-
19	https://www.webcamgalore.com/webcam/Norway/Oslo/149.html	Oslo (Norway)	30%	Website	Harbour	Page URL & Inside html tags close to webcam image	Updated daily – & Inside html tags close to webcam image	-
20	http://www.ohv oslo.no/no/?template=webcam	Oslo (Norway)	30%	Website	Harbour	Page URL & Inside html tags close to webcam image	Updated daily – embedded in image	-
21	http://www.webcamsinnorway.com/webcams.php?viewcam=2534	Several cities in Norway	Varies	Website	Varies	Inside html tags close to webcam image	Updated daily – & Inside html tags close to webcam image	-
22	http://www.cruisinn.me/cruise-port-webcams/europe/trondheim-norway.php	Countries around the World including Norway	Varies	Website	City	Page URL & Inside html tags close to webcam image	Updated daily – embedded in image	Registration is required
23	http://www.aftenbladet.no/webkamera/livekamera/Jatta-retning-Jattavagen-ost-3623464.html	Stavanger (Norway)	20%	Flash (website)	City	Inside html tags close to webcam image	Live streaming	-

D3.1: Environmental node discovery, indexing and data acquisition

24	http://romeo.skybert.no/drammenh/k1/snapshot_pos1_1.jpg	Drammen (Norway)	20%	Image	Harbour	Image URL & embedded into image	Embedded into image	-
25	http://www.amihotel.no/ami/english/webcam_ami_hotel_tromso.html	Tromso (Norway)	30%	Website	City	Image URL & embedded into image & Inside html tags close to webcam image	Updated daily – embedded in image	-
26	http://weather.cs.uit.no/	Tromso (Norway)	60%	Website	City	META keywords & Inside html tags close to webcam image	Inside html tags close to webcam image & embedded into the image	-
27	http://www.123cam.com/	Countries around the World including Norway, Germany	Varies	Website	Varies	Varies depending on the site hosting the webcam	Varies depending on the site hosting the webcam	-
28	http://ip-24.net/main_index.php?lang=en	Countries around the World including Norway, Germany	Varies	Flash (website)	Varies	Varies depending on the site hosting the webcam	Varies depending on the site hosting the webcam	-
29	http://www.onthesnow.com/germany/webcams.html	Several mountains in Germany	Varies	Website	Mountain	Varies depending on the site hosting the webcam	Varies depending on the site hosting the webcam	-
30	http://www.webcamhopper.com/	Countries around the World including Germany	Varies	Website	Varies	Varies depending on the site hosting the webcam	Varies depending on the site hosting the webcam	-
31	http://www.webcamlocator.com/worldwide_welcome.htm	Countries around the World including Norway, Germany, Greece, etc.	Varies	Website	Varies	Varies depending on the site hosting the webcam	Varies depending on the site hosting the webcam	-

9.3.2 AMOS dataset

S/N	Country	City	Total number of cameras found for the area	Number of working cameras	Number of working cameras that contain sky
1	Germany	Berlin	6	4	4
2		Hamburg	1	1	1

D3.1: Environmental node discovery, indexing and data acquisition

3		Stuttgart	0	0	0
4		Munich	0	0	0
5		Dresden	0	0	0
6		Dusseldorf	1	0	0
7		Potsdam	0	0	0
8		Bremen	3	2	2
9		Wiesbaden	0	0	0
10		Hanover	1	1	1
11		Schwerin	0	0	0
12		Mainz	0	0	0
13		Saarbrücken	0	0	0
14		Magdeburg	0	0	0
15		Kiel	0	0	0
16		Erfurt	1	0	0
17		Rostock	1	1	1
18		Nuremberg	0	0	0
19		Leipzig	0	0	0
20		Cologne	3	3	3
21	Norway	Oslo	24	6	4
22		Bergen	5	0	0
23		Trondheim	16	15	6
24		Stavanger	7	2	1
25		Kristiansand	3	2	2
26		Drammen	12	9	2
27		Tromsø	11	4	4
28	Greece	Athens	8	1	1
29		Thessaloniki	2	1	0
30	Brussels	Brussels	0	0	0
31	UK	London	9	3	2
32	Netherlands	Amsterdam	8	1	0

9.4 Configuration file examples

```

{
  "url": {
    "base_url": "http://www.luftkvalitet.info",
    "relative_url": "/home/airquality.aspx?type=1&topic=1&id={03206c07-9d57-4af7-92eb-e66ee800dcac}"
  },
  "source_name": "Luftkvalitet_bergen_pm10",
  "table_selector": "#ctl100_cph_Map_nilu_usercontrols_met_met_ascx1_gwArea > tbody:nth-child(1) > tr",
  "unit_info": [
    {
      "label": "City",
      "value": {
        "selector": "td:nth-child(2)",
        "type": "text",
        "replace": {
          "regex": [
            "By"
          ],
          "with": [
            ""
          ]
        }
      }
    },
    {
      "label": "Station",
      "value": {
        "selector": "td:nth-child(1)",
        "type": "text",
        "replace": {
          "regex": [
            "Stasjon"
          ],
          "with": [
            ""
          ]
        }
      }
    },
    {
      "label": "Date",
      "value": {
        "selector": "td:nth-child(4)",
        "type": "text",
        "replace": {
          "regex": [
            ".+*"
          ],
          "with": [
            ""
          ]
        }
      }
    },
    {
      "label": "Time",
      "value": {
        "selector": "td:nth-child(4)",
        "type": "text",
        "replace": {
          "regex": [
            ".+*"
          ],
          "with": [
            ""
          ]
        }
      }
    }
  ],
  "metrics": [
    {
      "label": "PM10",
      "value": {
        "selector": "td:nth-child(3)",
        "type": "text"
      }
    }
  ],
  "dynamic_page": true,
  "events": {
    "selector": "#ctl100_cph_Map_nilu_usercontrols_met_met_ascx1_ddlComponentName > option:nth-child(3)",
    "times_to_repeat": 1,
    "extraction_type": "AFTER_ALL_EVENTS",
    "type": "CLICK"
  }
}

```