hackAIR

OPEN PLATFORM

D3.2: 2nd Environmental node discovery, indexing and data acquisition

WP3 – Collective sensing models and tools

Document Information

Grant Agreement Number	688363	Acronym	hackAIR	
Full Title	Collective awareness platform for outdoor air pollution			
Start Date	1st January 2016	Duration	36 months	
Project URL	www.hackAIR.eu			
Deliverable	D 3.2 – 2nd Environmental node discovery, indexing and data acquisition			
Work Package	WP 3 - Collective sensing models and tools			
Date of Delivery	Contractual	1st July 2017	Actual	3rd July 2017
Nature	Report	Dissemination Level	Public	
Lead Beneficiary	CERTH			
Responsible Author	Eleftherios Spyromitros-Xioufis (CERTH)			
Contributions from	Anastasia Moumtzidou (CERTH), Symeon Papadopoulos (CERTH), Stefanos Vrochidis (CERTH), Yiannis Kompatsiaris (CERTH), Dimitrios Messinis (DRAXIS)			

Document History

Version	Issue Date	Stage	Description	Contributor
0.1	10/04/2017	Draft	Document structure	E. Spyromitros-Xioufis (CERTH), S. Papadopoulos (CERTH), S. Vrochidis (CERTH)
0.4	19/06/2017	Draft	Integrated document	E. Spyromitros-Xioufis (CERTH)
0.7	22/06/2017	Draft	Internal review of the whole document	Anastasia Moumtzidou (CERTH), S. Papadopoulos (CERTH), S. Vrochidis (CERTH), Y. Kompatsiaris (CERTH), D. Messinis (DRAXIS)
0.9	29/06/2017	Pre-final	Final version	E. Spyromitros-Xioufis (CERTH), S. Papadopoulos (CERTH), S. Vrochidis (CERTH)
1.0	30/06/2017	Final	Approved by coordinator for submission	E. Spyromitros-Xioufis (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

Table of Contents

1 Executive summary.....7

2 Environmental and Social Node Indexing and Data Collection.....9

2.1 Social Image Collection Improvements and Extensions.....9

2.1.1 Flickr Collector Improvements.....9

2.1.2 Collecting Flickr Images with Textual Queries.....10

2.2 Webcam Image Collection.....15

2.2.1 Collecting Images from AMOS Webcams.....15

2.2.2 Collecting Images from webcams.travel API Webcams.....17

2.3 Image Collection Statistics.....21

2.4 Collecting Measurements from Ground Stations.....24

3 Image Analysis Experiments and Statistics.....29

3.1 Evaluation of Sky Localization Methods.....29

3.2 Robustness of Image Analysis against Image Transformations.....34

3.3 Image Analysis Service.....44

3.3.1 Service Architecture.....44

3.3.2 Image Analysis Service Efficiency.....46

3.4 Image Analysis Statistics.....47

3.5 Visualization of Image Collection & Analysis.....50

4 Twitter-based air quality estimation.....52

4.1 Methodology.....52

4.2 Data Collection.....52

4.2.1 Twitter Data Collection.....53

4.2.2 Collection of Ground Station Measurements.....55

4.3 Predictive Feature Generation.....57

4.4 Air Quality Estimation Experiments.....61

4.4.1 One-to-one vs Many-to-one Models.....61

4.4.2 Feature Selection Experiments.....62

4.4.3 Contribution of Twitter-based Features.....64

5 Conclusions.....67

6 References.....69

7 Appendix.....71

7.1 Additional Results on Robustness against Image Transformations.....71

7.2 Additional Results on Twitter-based Air Quality Estimation.....77

Table of Figures

Figure 1 – Illustration of example term-cell probabilities calculated for the grid containing the city of New York. 11
Figure 2 – Location estimation performance with different cut-off thresholds using only terms in title. 13
Figure 3 – Location estimation performance with different cut-off thresholds using only terms in tags. 13
Figure 4 - Location estimation performance with different cut-off thresholds using only terms in description. 14
Figure 5 - Location estimation performance with different cut-off thresholds using the union of terms in title, tags and description. 14
Figure 6 - The MongoDB record of a webcam image from the AMOS dataset. 16
Figure 7 – Geographical distribution of AMOS webcams. 17
Figure 8 - Example response from the webcams.travel API. 19
Figure 9 - The MongoDB record of a webcam image from webcams.travel. 19
Figure 10 – Geographical distribution of webcams.travel webcams. 20
Figure 11 – Number of images collected daily from each source 21
Figure 12 – Percentage of the total number of collected images coming from each European country (top 20 are shown) 22
Figure 13 – Total number of Flickr and webcam images collected from each European country (top 20 are shown)..22
Figure 14 –Total number of images collected daily from Germany and Norway 23
Figure 15 – Example response from the latest endpoint of the OpenAQ API. 26
Figure 16 – Example MongoDB record of an environmental measurement from the OpenAQ API. 27
Figure 17 - Screenshots of the ground station data collection visualization web application. 28
Figure 18 – Reponses to Q1-a and Q1-b for Flickr images (left) and webcam images (right). 30
Figure 19 – Examples of sky-depicting images that are considered unusable for air quality estimation due to (left to right): a) cirrus clouds, b) clouds, c) hour of the day (too late or too early), d) humidity 31
Figure 20 - Percentages of correctly/incorrectly detected sky regions using each sky localization approach for Flickr images (left) and webcam images (right). 31
Figure 21 - Inappropriate sky masks extracted using the FCN (middle) and the heuristic (right) approach for a Flickr image (left). 32
Figure 22 – Comparison of the masks generated by the FCN approach (second column) with the masks generated by the heuristic approach (third column) for the images of the first column. The fourth column shows the masks generated by combining the FCN approach with the heuristic approach. 33
Figure 23 – Comparison of FCN, heuristic and FCN+heuristic sky localization approaches when all images are considered. 33
Figure 24 - Image enhancements (left) and filters (right) offered by Flickr's Android/IOS app. 34
Figure 25 - Scatterplots of R/G (left) and G/B (right) scores calculated from original (x axis) and transformed images (y axis) for the w:200, brightness:20 and improve transformations. 43
Figure 26 – Architecture of the image analysis service 45
Figure 27 – The body of an example (POST) request to the IA service (left) and the corresponding JSON response (right)r. 46
Figure 28 – Daily number of usable sky images in Europe (the increasing trend is due to the addition of webcam images). 48
Figure 29 – Daily number of usable sky images in Germany and Norway. 48
Figure 30 – Daily R/G vs G/B averages. 49
Figure 31 – Scatterplot of average daily R/G and G/B ratios. 49
Figure 32 – Screenshots of the image collection and analysis visualization web application. 50

Figure 33 – Example response of the REST web service that supports the image collection and visualization application. 51
Figure 35– Top 10 place fields from tweets assigned to London....................54
Figure 36 – Number of tweets retrieved daily for each city. 54
Figure 37 – Daily PM10 and PM2.5 means in London. 56
Figure 38 - Daily PM10 means in London, Manchester and Liverpool....................56
Figure 40 – Scatterplot of distances and PM10 correlations between cities in the UK. 57
Figure 41 – Daily numbers of AQ_D, AQ_S, High, Low and Irrelevant tweets in London. Irrelevant refers to tweets classified as irrelevant by the AQ_Discussion classifier. 59
Figure 42 - Daily and hourly tweet distributions....................60
Figure 43 - Feature correlations 61
Figure 44 - Regression with and without Twitter-based features in Manchester....................66
Figure 45 - Scatterplots of R/G (left) and G/B (right) scores calculated from original (x axis) and transformed images (y axis)....................77

Table of Tables

Table 1 – Total, geotagged and geotagged in Europe images returned by the Flickr API for different upload date windows....................11
Table 2 – European countries for which data is available in the OpenAQ platform along with data source (second column), number of locations with PM 10 data (third column) and number of locations with PM2.5 data (fourth column). The URLs of the data sources are used in some cases because the source name is not provided by OpenAQ. 24
Table 3 - Mapping of absolute PM10/PM2.5 values to pollution classes. 28
Table 4 – Category (1st column), name (2nd column) and description (3rd column) of the tested image transformations, along with their effects on an example image (4th column) and their impact on the computed R/G and G/B ratios (5th and 6th column). A hyphen (-) in the 5th, 6th column means that no R/G, G/B ratios were computed because no usable sky was detected. 36
Table 5 – Pearson correlations between the R/G (2nd column) and G/B (3rd column) ratios of the original images and their transformed versions. The last column reports the percentage of images with no usable sky after each transformation....................41
Table 6 – Time efficiency characteristics of the IA service....................47
Table 7 – Numbers and percentages of sky-depicting and usable sky images collected from each source. 47
Table 8 – Daily average number of tweets retrieved per city. 55
Table 9 - Numbers of air pollution stations per city....................55
Table 10 - Predictive features. 57
Table 11 – AQ_General, AQ_Sense and AP_Severity classifier details....................58
Table 12 - One-to-one vs many-to-one (Accuracy)....................62
Table 13 - One-to-one vs many-to-one (AUC). 62
Table 14 – Comparison of feature selection metrics (Accuracy) 63
Table 15 - Comparison of feature selection metrics (AUC)....................63
Table 16 - Confusion matrix for London 64
Table 17 - Confusion matrix for Manchester 64
Table 18 - Accuracy obtained with Twitter-based vs mean_PM10_nearby vs all features....................64
Table 19 - AUC obtained with Twitter-based vs mean_PM10_nearby vs all features 65

Table 20 - MSE obtained with Twitter-based vs mean_PM10_nearby vs all features 65
Table 21 – Examples of London tweets classified as relevant by the AQ_Discussion classifier 77
Table 22 – Examples of London tweets classified as relevant by the AQ_Sense classifier........77
Table 23 - Examples of London tweets classified as High by the AP_Severity classifier 78
Table 24 - Examples of London tweets classified as Low by the AP_Severity classifier 78
Table 25 – Top-20 features selected using each feature selection metric. 78

1 Executive summary

This document reports on the second and final version of the research and development of techniques that facilitate the search and indexing[1] of environmental nodes, as well as the acquisition of air quality-relevant data from the Web and from social media. The work described here builds upon the work reported on D3.1 and includes: a) the description of several new data collection methods that were implemented and integrated into the hackAIR platform, b) an in-depth study of the performance of the image analysis techniques that were described in D3.1 that lead to an extension with significant improvements in terms of performance, c) the description and evaluation of an experimental framework for air quality estimation from Twitter data. With respect to data collection an important change is the extension of its coverage to the whole European continent instead of only big cities in the countries of the pilot studies (Germany and Norway). This change was motivated by the requirement of the data fusion module developed in WP4 for geographically scattered measurements that include both urban and rural areas and the fact that the throughput rate of the image processing module was considerably improved and can now handle the expansion.

Initially, we focused on improving data collection from Flickr which is the main source of social images after the shutdown of the Instagram API. Besides the geographical expansion, we also implemented several improvements (section 2.1.1) such as rejection of images with uncertain capture dates and better handling of the API's responses. In addition, we studied the possibility of extending Flickr image collection to include non-geotagged images of which the geolocation is estimated based on the textual metadata of the image. To this end, experiments with a state-of-the-art geolocation estimation method were carried out (section 2.1.2) and it was found that for a significant percentage of non-geotagged Flickr images (27.3%), it is possible to infer their location with high precision. This is a very encouraging finding as it shows that a significantly larger number of Flickr images can be exploited for air quality estimation, compensating for the smaller number of images uploaded on Flickr compared to Instagram.

In addition, we describe the implementation of methods for collection of images from public webcams (section 2.2). A characteristic of webcam images that makes them highly valuable for the data fusion model developed in WP4 is that their geolocation is fixed and known in advance and it is possible to collect images from them at regular intervals. Thus, we put significant focus on collecting images from a large number of webcams across the whole Europe. To this end, two very large public webcam image repositories were leveraged, AMOS and webcams.travel. Specialized image collectors were implemented for the two repositories, leading to a total of around 3.5K webcams in Europe.

Besides the collection of image data that are analyzed and used as input for the air quality estimation models, we also implemented a collector of official environmental data (particulate matter PM_{10} and $PM_{2.5}$) from ground stations (section 2.4) that builds upon the results of the empirical study of environmental web services that was conducted in D3.1. This data will be used as input to the visualization module developed in T5.2 and also serve as ground truth labels for the Twitter-based air quality estimation models that we describe in this report (section 4). The collector retrieves data from the OpenAQ platform which provides data from many air quality stations that cover most European countries, including the two countries of the pilots. In addition to the data collector, we also developed and present a web interface that visualizes the current air quality conditions in Europe as reflected by the data that we collect from OpenAQ.

Another large part of our work (section 3) has focused on the detailed evaluation and finalization of the image analysis methods that we developed in D3.1, in order to turn them into an effective and efficient image analysis service for supporting the process of providing air quality estimations from images within the hackAIR platform. First, we performed a very realistic, air-quality oriented evaluation (section 3.1) of the two alternative sky localization methods: a) the one based on deep learning techniques (*FCN* approach) and b) the one based on heuristic rules that were provided by air quality estimation experts (*heuristic* approach). To this end, we created a dataset that contain the sky

[1] Discovery of environmental notes was mainly addressed in D3.1.

masks extracted from a random set of Flickr and webcam images, and asked air quality experts to evaluate the performance of the two sky localization methods. Based on this evaluation, a number of important conclusions were drawn such as the complementarity of the two approaches and the difficulty in rejecting images where the sky is covered by cirrus clouds. Motivated by the complementarity of the two approaches, a new approach that combines them was proposed and evaluated, leading to significantly better results than either of the two approaches alone. Moreover, we performed (section 3.2) a comprehensive study of the impact of various commonly applied image transformations and filters on the ratios (R/G and G/B) that we computed from the sky regions of the images and sent as input to the image-based air quality estimation models. The results showed that the results of the image analysis were very robust against most transformations, except for the most intense ones.

We also provide a detailed description (section 3.3) of the architecture of the image analysis service as well as the effectiveness of its three main components: a) sky concept detection, b) *FCN*-based sky localization, c) refinement of the *FCN*-based mask with the *heuristic* approach and calculation of the R/G and G/B ratios. Importantly, statistics regarding the results of the image analysis service on the collected images are given (section 3.4), including the numbers of sky-depicting and usable sky images collected daily from each data source as well as from all available data sources and the number of usable sky images collected from the countries of the pilots. Moreover, we present a web interface that we developed for the visualization of the data collection and image analysis results (section 3.5).

Finally, we present an experimental line of work that investigates the feasibility of making air quality estimations for areas (cities) without official air quality stations based on Twitter activity. Such estimations can be potentially useful in cases where e.g. due to high cloud coverage, there is not enough data to make image-based air quality estimations. For this purpose, a Twitter data collection framework is implemented that focuses on collecting air quality-related Twitter posts that are posted in specific cities. Text analysis machine learning techniques are then utilized that try to learn accurate mappings between the current air quality conditions and statistical attributes of the Twitter posts. Five cities in the UK are used as a case study and a transfer learning framework is developed where using data from one or more nearby cities for training the estimation models, we try to make estimations for another city that is assumed to not have official ground station measurements. A series of experiments are conducted using state-of-the-art machine learning techniques and some promising results are obtained.

With the completion of D3.2, considerable progress has been achieved and we consider that the goals of WP3 have been fulfilled. The developed and tested components have been delivered and will be integrated in the hackAIR platform, and are expected to be continuously refined (within WP5 and WP7) throughout the second period of the project through the feedback acquired from the pilot studies, as well as by carrying out further lab experiments and implementing appropriate extensions to improve their quality and resilience.

2 Environmental and Social Node Indexing and Data Collection

2.1 Social Image Collection Improvements and Extensions

2.1.1 Flickr Collector Improvements

In this section, we report various updates and improvements that were implemented during the reporting period on the Flickr collector (of which the first version is described in D3.1). In short, the purpose of the Flickr collector is to periodically call the Flickr API in order to retrieve the URLs and necessary metadata (i.e. geolocation and timestamp) of images captured (and uploaded) recently (within the last 24 hours) around the locations of interest. The metadata of each image is stored in a MongoDB and the URLs are used to download the images and store them until image analysis for supporting air quality estimation is performed.

In the first version of the collector, the *flickr.photos.search* endpoint was used in order to collect images within a radius of 16 km around the center of 34 European cities located mainly in Germany, Norway and Greece. This was achieved by setting the *lat* (latitude) and *lon* (longitude) parameters of the endpoint and submitting one request per city every 24 hours using appropriate values for the *minimum* and *maximum date taken* parameters in order to retrieve only photos taken within the last 24 hours. Compared to that first version of the collector, the updated version includes the following updates and improvements:

Geographical coverage extension: We studied the feasibility of extending the geographical coverage of the Flickr collector to the whole Europe instead of specific European cities. This change was motivated by the fact that the data fusion module being developed in WP4 works better when the air quality measurements/estimations used as inputs are geographically scattered and include both urban and rural areas. To this end, an alternative way of performing geographical queries using the *flickr.photos.search* API method was employed, i.e. using the *woe_id* parameter. This parameter allows geographical queries based on a WOEID[2] (Where on Earth Identifier), a 32-bit identifier that uniquely identifies spatial entities and is assigned by Flickr to all geotagged images. Using this approach, extending the coverage to the whole Europe consists of replacing the multiple city-oriented requests with a single request where the *lat/lon* parameters have been replaced by the *woe_id* parameter set to the WOEID of Europe (24865675). Note that this approach was preferred over using a bounding box query (*bbox* parameter) because Europe's bounding box includes non-European countries (e.g. Turkey).

Taken date validation: In order to retrieve only photos taken within the last 24 hours, the *min/max_date_taken* parameters of the *flickr.photos.search* endpoint are used. These parameters operate on Flickr's 'taken' date field which is extracted, if available, from the image's Exif metadata. However, the value of this field is not always accurate as explained in Flickr API's documentation[3]:

- Flickr automatically sets the taken date to the time of upload when the taken date is not available in the Exif. Thus, we reject all images with a taken date equal to the upload date.
- When the taken date of the image is not specified with enough detail Flickr auto-completes the missing information with default values so that all taken dates are specified up to the second (e.g., a taken date of "2016-11-25" will be automatically transformed to "2016-11-25 00:00:00"). Fortunately, it is possible to identify such cases by taking taken date granularities into account (information provided in the *datetakengranularity* field of Flickr API's response). Flickr assigns a 'granularity' – the accuracy to which the date is known to be true – to taken dates. Currently, there are four taken date granularities (0, 4, 6, 8) on Flickr, only the finest of which (0) provides sufficient

[2] https://en.wikipedia.org/wiki/WOEID

[3] https://www.flickr.com/services/api/misc.dates.html

detail (up to the second) to be usable for our purposes. Thus, all images with a taken date granularity > 0 are rejected.

Improved API response management: After expanding the area of interest to the whole Europe, each API request now returns a much larger number of results compared to the city-oriented queries. An implication of this is that some queries return more than 4,000 results, bringing up an idiosyncrasy of the Flickr API, i.e. whenever the number of results for any given search query is larger than 4,000, only the pages (results are offered paginated) corresponding to the first 4,000 results will contain unique images and subsequent pages will contain duplicates of the first 4,000 results. To tackle this issue, a recursive algorithm was implemented, which, when a query returns more than 4,000 results, splits the query's date taken interval (initially a 24-hour interval) in two and creates two new queries that are submitted to the API. The process continues until all queries have returned less than 4,000 results, at which point all the results of the initial query have been retrieved. This mechanism offers robustness against data bursts and is particularly useful also in the case of textual Flickr API queries (see Section 2.1.2) which return a very large number of results.

Request frequency increase: Finally, the call frequency was increased from one call every 24 hours to one call every 6 hours (always using a fixed lookback window of 24 hours). Although this change does not affect the total number of images that are collected it increases the number of fresh images (taken within the last 24 hours) that are available to the system at any time point. In addition, using overlapping time windows implies that a large fraction of the images returned with each request, will have already been collected from a previous request. Thus, an efficient way to check for duplicate images was implemented by exploiting MongoDB's indexing and batch querying capabilities.

2.1.2 Collecting Flickr Images with Textual Queries

Given that geographical queries for Europe return only about 5,000 geotagged images per day on average (see Section 2.3 for detailed image collection statistics) and since other social media platforms do not offer free access to their APIs, we explored the potential of utilizing non-geotagged Flickr images after estimating their capture location based on textual metadata such as image tags, title and description. Table 1 shows the total number of images returned by the Flickr API when queries of increasing upload date window size are submitted, as well as the numbers (and percentages) of geotagged images and images geotagged in Europe in the same upload date windows[4]. To obtain the total number of images in a time window, we use the *flickr.photos.search* endpoint and specify the *min/max_upload_date* parameters while leaving all other parameters empty. To obtain the numbers of geotagged images and images geotagged in Europe within the same intervals, the *has_geo and woe_id* parameters are additionally specified[5]. Two important observations can be made based on the results:

- The vast majority of Flickr images are non-geotagged (≈97%). This means that there is a large pool of images that could be utilized for air quality estimation, provided that their location could be accurately estimated.
- A significant percentage of the images in the non-geotagged pool are expected to be from Europe, given the very high representation of Europe (>50%) on geotagged images.

Based on these observations, we expect that even if a small fraction of the non-geolocated images can be accurately geolocated, a significant increase to the total number of Flickr images that can be useful for air quality estimation is possible.

[4] Note that these numbers are significantly higher than the previously reported average number of 5,000 images geotagged in Europe per day. This deviation is mainly due to the fact that the previous number refers to images uploaded within 24 hours after they have been captured while there is no such limitation for the number reported in Table 1.

[5] The max_upload_date timestamp was set to 1496749429 (6/6/2017) in all queries. Therefore, the reported numbers represent only a rough estimate of the actual numbers.

Table 1 – Total, geotagged and geotagged in Europe images returned by the Flickr API for different upload date windows.

Date uploaded window	All	Geotagged	Geotagged in Europe
1	4,989,264	134,122 (2.7%)	74,584 (1.5%)
7	14,348,054	406,017 (2.8%)	220,895 (1.5%)
30	50,084,562	1,610,467 (3.2%)	787,157 (1.6%)

Estimating geographical coordinates (*geotagging*) of multimedia items, such as images and videos, based on massive amounts of geotagged training data is a research topic that has recently attracted significant attention, largely due to the *placing task* (Hauff et al., 2013), (Choi et al., 2014), (Choi et al., 2015), (Choi et al., 2016) of the MediaEval[6] benchmarking initiative for multimedia evaluation. The simplest approach for geotagging is called geoparsing and consists of detecting references to known locations with the help of gazetteers (Amitay et al., 2004). Geoparsing, however, has a few limitations such as the inability to perform inferences from text descriptions that do not explicitly refer to geographic entities and the inability to consider contextual information to deal with ambiguous geographic names (e.g., Athens may refer to the capital of Greece, but also to 23 toponyms in the US). To deal with the limitations of geoparsing, Language Model-based (LM) approaches were proposed (Serdyukov et al., 2009). LM approaches learn a probabilistic textual model using a large set of training items and then use this model to provide estimates about the location that a new piece of text refers to. LM approaches alleviate the disadvantages of geoparsing since they do not operate on an explicit toponym dictionary and take context into account by considering multiple terms to produce their estimates. Indeed, the best performing runs of the last three editions of the MediaEval placing task employ LM-based approaches.

Thus, in the context of location estimation for non-geotagged Flickr images that is in the interest of hackAIR, we evaluate a state-of-the-art LM-based geotagging approach (Kordopatis-Zilos et al., 2016) that has demonstrated excellent results in the latest edition of the MediaEval placing task (2016). According to this approach, the earth surface is divided into (nearly) rectangular cells with sides 0.01° for both latitude and longitude (corresponding to a geodesic length of approximately 1km near the equator), and the term-cell probabilities (Figure 1) are computed based on the user count of each term in each cell, based on a training set comprising of the union of the ≈5M training items provided for the 2016 placing task (Choi et al., 2016) and all geotagged items (≈40M) of the YFCC100M dataset (Thomee et al., 2015). Given a query text, the most likely cell is derived from the summation of the respective term-cell probabilities. On top of this basic idea, the method features several refinements such as text pre-processing, feature selection, feature weighting, use of multiple resolution grids, etc. More details about these refinements can be found in the original paper (Kordopatis-Zilos et al., 2016).

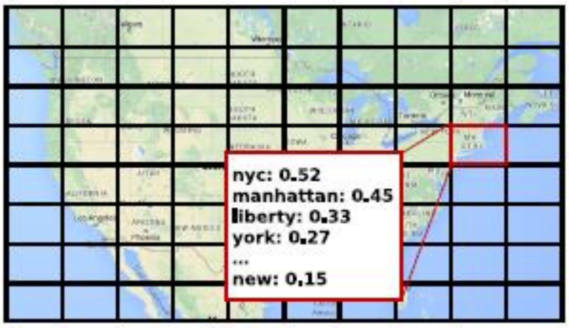


Figure 1 – Illustration of example term-cell probabilities calculated for the grid containing the city of New York.

[6] http://multimediaeval.org/

Here, we use an open-source implementation of the method[7] and evaluate it on the task of geolocating Flickr images based on their textual metadata. Since the area of interest is Europe, the evaluation is carried out on a set of ≈150K images geotagged in Europe that were collected using the basic (geographical) version of the Flickr collector in April 2017. As evaluation measure, we employ the widely used Precision at R ($P@R$) which is defined as:

$$P@R = \frac{|\{i|d\left(G_{pr}(i), G_{ref}(i)\right) < R\}|}{|D_{ts}|},$$

where D_{ts} is a set of image items i, $G_{pr}(i)$ and $G_{ref}(i)$ are the estimated and reference location of i respectively, $d(x, y)$ is the geodesic distance between points x and y and R is a predefined range. In our experiments, we focus on $R =$ 10km and $R =$ 25km as less accurate estimations are not useful in the context of hackAIR.

An advantage of the employed approach is that, in addition to providing an estimate of an image's location, it also calculates a score in $c \in [0,1]$ which expresses the confidence of the estimation. This is important as it allows rejection of low-confidence estimations, in the hope that a better $P@R$ can be achieved for high-confidence estimations. In our experiments, we study the effect of applying different cut-off thresholds $t = \{0.0, 0.1, 0.2, \dots, 0.9\}$ to reject estimations with confidence $c < t$. In addition, we study the effect of using alternative types of metadata, i.e. title-only, tags-only, description-only, and title+tags+description.

Figure 2,3,4, Figure 5 show the P@10Km and P@25Km performance as well as the respective percentage of all images for which estimations are made for different cut-off thresholds, using each type of metadata. As expected, the higher the cut-off threshold, the higher the precision and the lower the percentage of images for which estimations are made[8]. In all cases, P@25Km scores > 0.9 can be achieved with a threshold ≥ 0.7. However, comparing the results obtained with each type of metadata we notice that a significantly better trade-off between precision and recall is achieved when the union of terms in title, tags and description is used. Assuming that a P@25Km performance ≥ 0.9 is sufficiently high, we see that title+tags+description achieves this performance goal while still providing estimations for the 27.3% of all images, compared to 18.7%, 16.0% and 3.5% respectively for title, tags and description. Thus, our analysis suggests that it is clearly advantageous to use all the available textual metadata.

Overall, the obtained results are very encouraging as they show that we can infer the location of a significant percentage of non-geolocated Flickr images with high precision. Hence, we conclude that extending the Flickr collector to include non-geotagged images with inferred location, constitutes a promising strategy of increasing the number of Flickr images that could be useful for air quality estimation.

[7] https://github.com/MKLab-ITI/multimedia-geotagging

[8] Note that the percentage of images for which predictions are made is lower than 100% even with a cut-off threshold equal to 0. This is due to the fact that no estimations can be made for images of which the respective metadata fields are empty (or become empty after pre-processing operations such as stop-word removal).

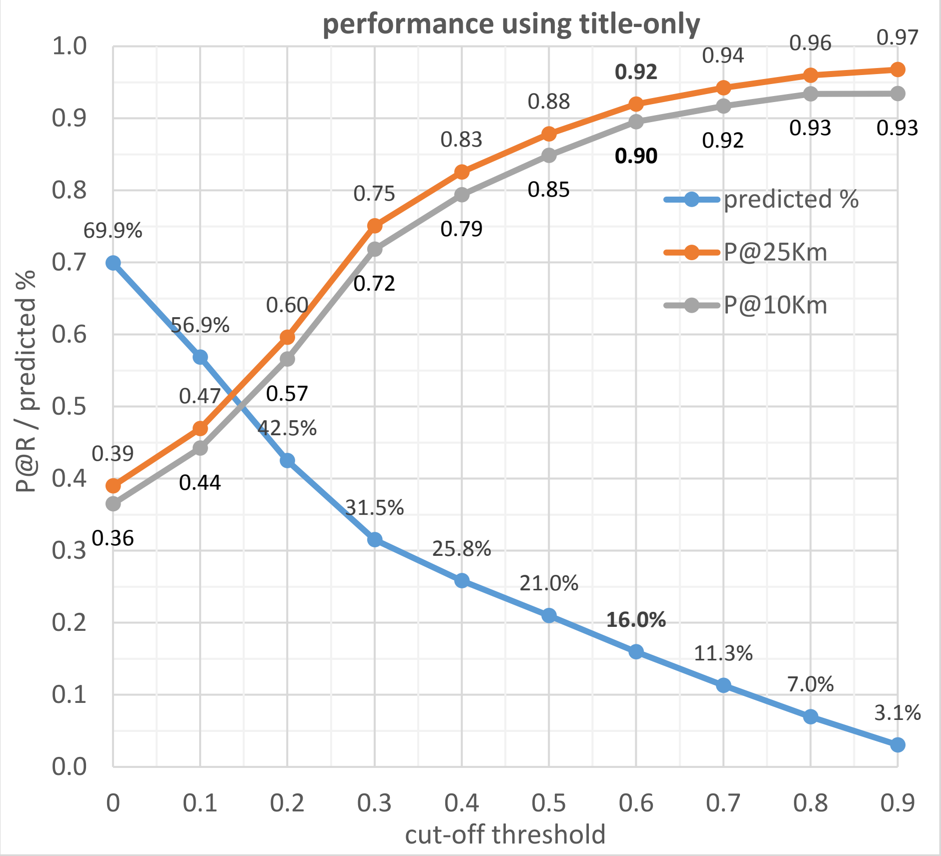


Figure 2 – Location estimation performance with different cut-off thresholds using only terms in title.

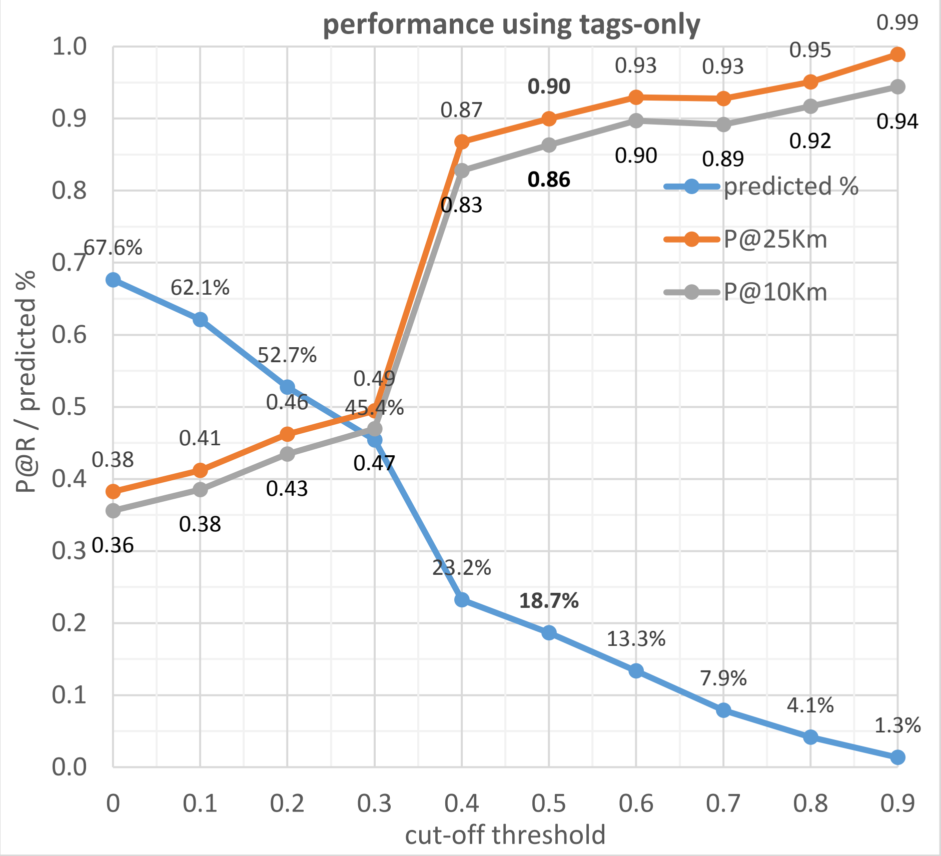


Figure 3 – Location estimation performance with different cut-off thresholds using only terms in tags.

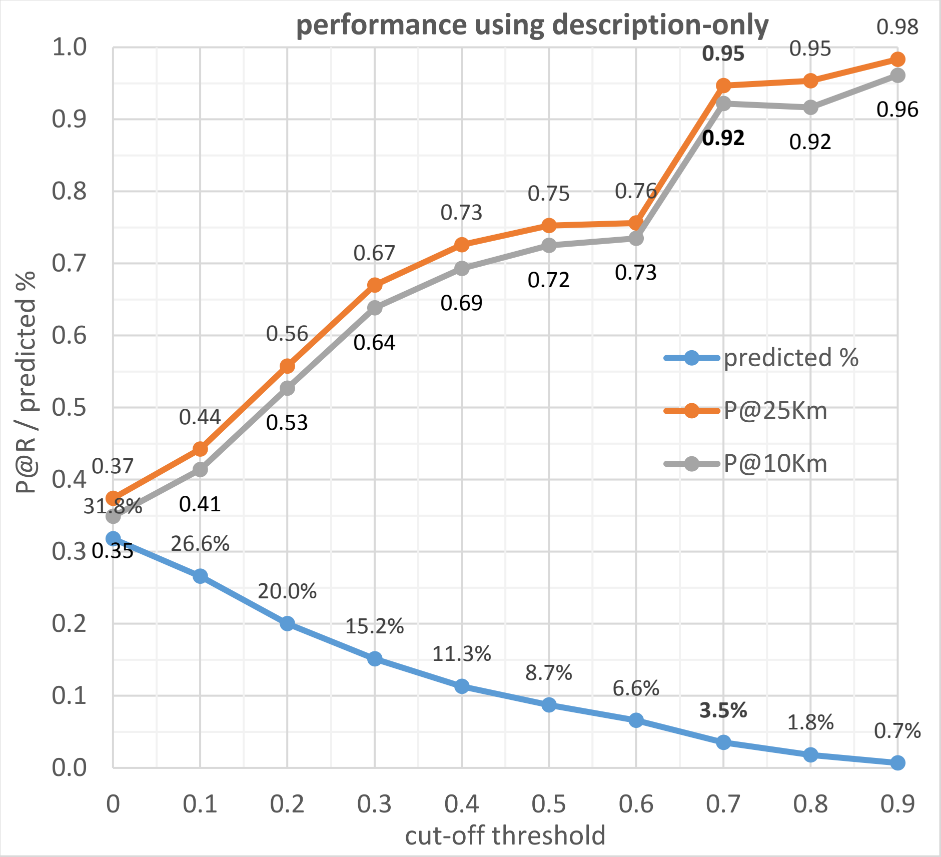


Figure 4 - Location estimation performance with different cut-off thresholds using only terms in description.

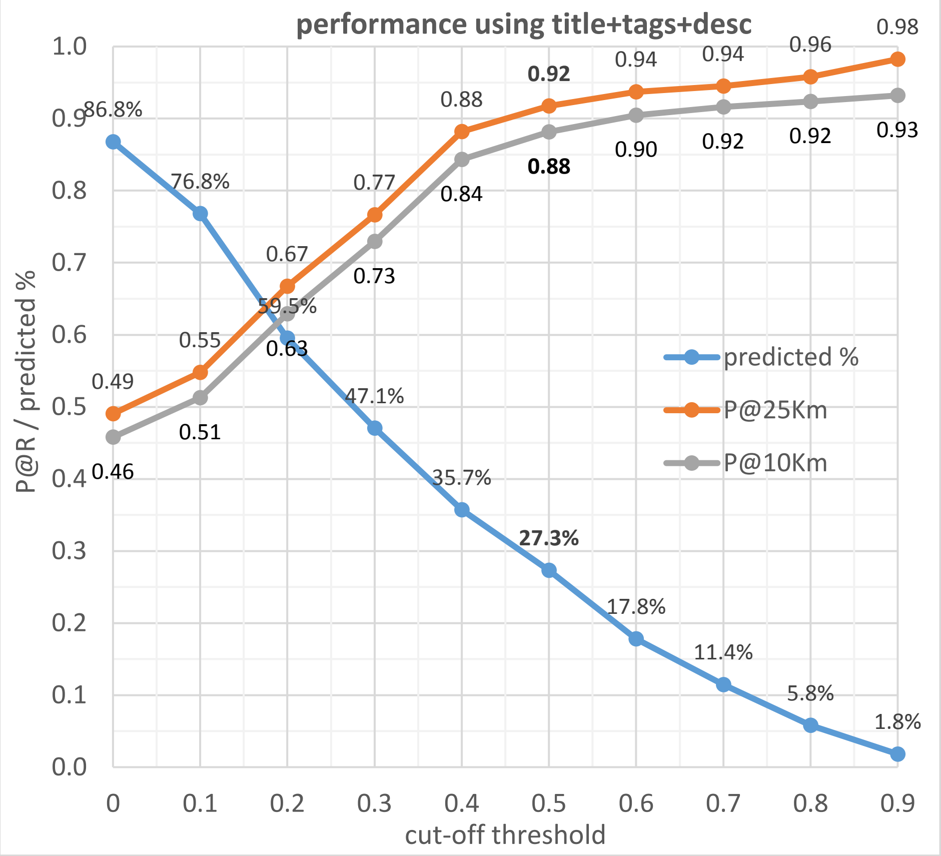


Figure 5 - Location estimation performance with different cut-off thresholds using the union of terms in title, tags and description.

2.2 Webcam Image Collection

This section provides the technical details on the collection of webcam images. As done for Flickr images, the collection does not focus only on webcams from big cities in the countries of the pilots (Germany and Norway) but instead considers webcams located anywhere in Europe. To this end, two large-scale repositories of webcams are used, AMOS[9] and webcams.travel[10]. In the case of AMOS, a web data extraction framework (section 2.2.1) was developed, while in the case of webcams.travel, data is retrieved through a client application for the provided API (section 2.2.2). Combined, these two sources provide data from more than 25,000 webcams in Europe. In a set of exploratory experiments, we found that most of the webcams discovered in a specific location (city/region) using standard search engines (e.g. Google, Bing) or focused crawling approaches (as the one described in D3.1), are already contained in either AMOS or webcams.travel. Based on this, we focused on the integration of these large-scale repositories instead of developing a specialized webcam discovery framework (as suggested in D3.1).

2.2.1 Collecting Images from AMOS Webcams

A detailed description of the AMOS dataset (Jacobs, 2007) was provided in D3.1. Here, we provide the technical details of the data collection framework that we developed in order to retrieve data from the AMOS website.

The first step consists of identifying the ids of all webcams that are located in Europe. This is accomplished by using the advanced filters form of the "Browse Cameras" page[11] that allows searching for webcams with multiple criteria including search for webcams located inside a specific bounding box (latitude/longitude range). Since we are interested in all webcams located in Europe, we define a bounding box that includes the whole European continent, i.e. latitude range: [27.6363 - 81.0088], longitude range: [-31.2660, 39.8693]. When this query[12] is submitted, 4,893 matching webcams are found (note that not all matching webcams are active though) and returned[13] in a results page that shows an image (snapshot) from each webcam as well as its title. The image of each webcam is clickable and links to a page (an example is provided in section 3.5.3.2 of D3.1) that provides all the information that is available for the webcam such as the webcam id, the link to the latest image captured from the webcam, the latest capture date/time and the geolocation of the webcam. Ideally, we would like to visit each webcam page only once to extract its static information (id, geolocation) and then use the script provided in the AMOS website (see section 4.3 of D3.1 for more details) to download the latest image of each webcam. Unfortunately, though, the provided script (and the corresponding REST service that is called by the script) is mainly targeted towards download of historical data and as a result gives the option to download a whole year or month of data from each webcam but not data from a single date or just the most recent snapshot from each webcam. Since it would be very inefficient to download a whole month of images every time a new image needs to be fetched from each webcam, we did not use the provided script and implemented instead a customized web scrapper for the AMOS website.

The web data extraction method that we implemented to retrieve data from the AMOS website works as follows:

- A query is constructed using the advanced filters form to retrieve a list of all webcams located in Europe (as described above).

[9] http://amos.cse.wustl.edu/

[10] https://www.webcams.travel/

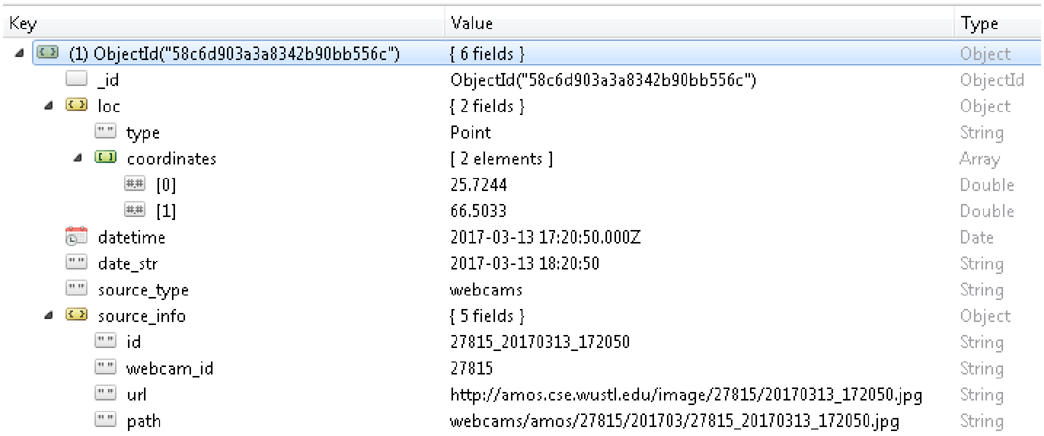
[11] http://amos.cse.wustl.edu/browse_with_filters

[12] http://amos.cse.wustl.edu/browse_with_filters?start=0&step=4893&longitude_1=39.8693&longitude_0=-31.2660&latitude_1=81.0088&latitude_0=27.6363

[13] The results are paginated (25 per page) but we manipulate the start and step URL parameters so that all matching webcams are returned in a single results page.

- The results page is parsed to extract the URLs of the webcam pages.
- Each webcam page is downloaded and parsed to extract the necessary information. In particular, we first check the “Last Captured” date to determine whether a new image is available for this webcam. If the last captured date is older than 24 hours in the past, we know that the webcam is inactive because AMOS normally captures a new image from each webcam every 30 minutes. If the capture date is more recent than 24 hours in the past, we create a new MongoDB record that contains all the necessary information (an example record is shown in Figure 6**Error! Reference source not found.**) and attempt to insert the image in a MongoDB repository where we store all the collected images. Note that in case the same image has already been retrieved, the insert fails because the unique id field of each webcam image is populated using the webcam id and the timestamp of the image.
- Finally, all new webcam images are jointly downloaded using an efficient multi-threaded downloader and stored on the server until image analysis is performed.

The AMOS image collector is executed four times per day (at 7:00, 11:00, 14:00 and 18:00) using a Java-based scheduler. During the period that we collect data from AMOS (6/3/2017-now) we have found that 2,246 of the 4,893 webcams are active. Their geographical distribution is shown in Figure 7. We see that almost all European counties are well represented. Norway is the country with most webcams (430) while a significant number of webcams (134) can also found also in Germany, the other country of the pilots. Of course, not all the discovered webcams depict the sky. However, the effort required to manually check all webcams in order to exclude non-sky-depicting ones would be prohibitive. Therefore, we initially collect images from all the discovered webcams, process them using the image analysis service (see section 3.3) and record the image analysis results. A statistical analysis of these results (presented in section 3.4) can help us determine the fraction of sky-depicting webcam images and facilitates automatic rejection of non-sky-depicting webcams.



Key	Value	Type
(1) ObjectId("58c6d903a3a8342b90bb556c")	{ 6 fields }	Object
_id	ObjectId("58c6d903a3a8342b90bb556c")	ObjectId
loc	{ 2 fields }	Object
type	Point	String
coordinates	[2 elements]	Array
[0]	25.7244	Double
[1]	66.5033	Double
datetime	2017-03-13 17:20:50.000Z	Date
date_str	2017-03-13 18:20:50	String
source_type	webcams	String
source_info	{ 5 fields }	Object
id	27815_20170313_172050	String
webcam_id	27815	String
url	http://amos.cse.wustl.edu/image/27815/20170313_172050.jpg	String
path	webcams/amos/27815/201703/27815_20170313_172050.jpg	String

Figure 6 - The MongoDB record of a webcam image from the AMOS dataset.

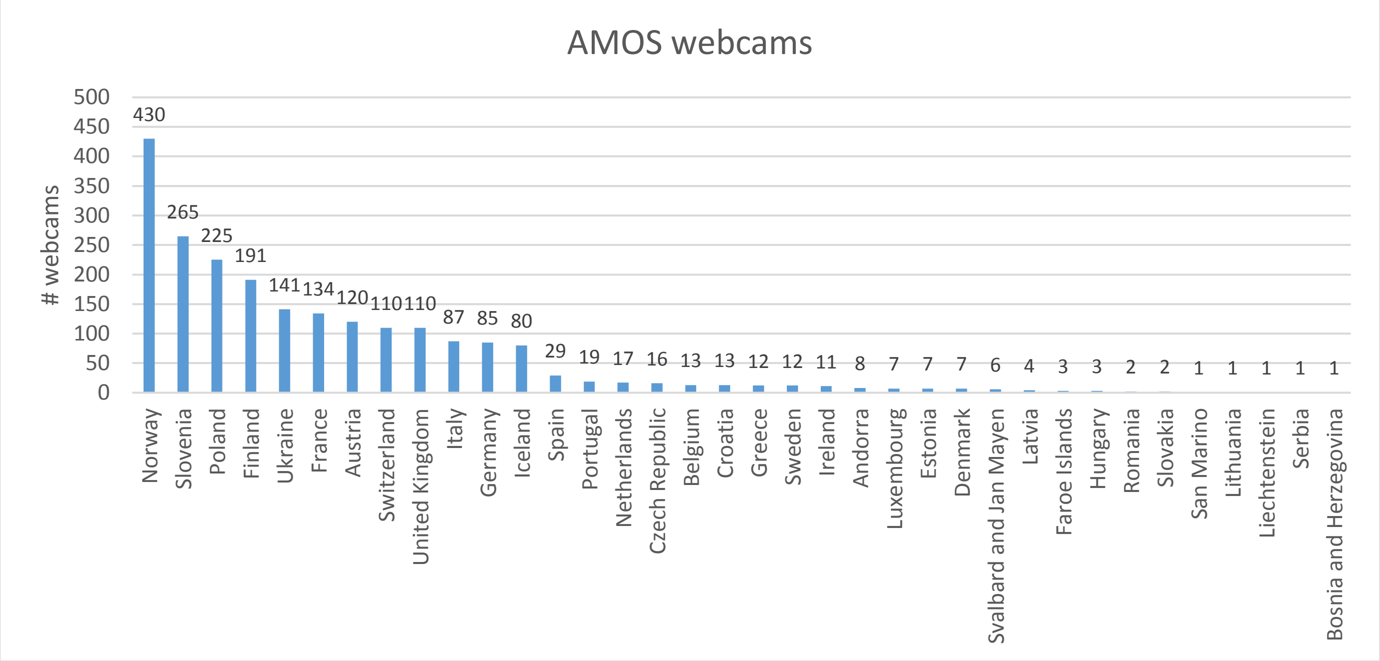


Figure 7 – Geographical distribution of AMOS webcams.

2.2.2 Collecting Images from webcams.travel API Webcams

Webcams.travel is a very large outdoor webcams directory that currently contains 64,475 landscape webcams worldwide. Webcams.travel provides access to webcam data through a comprehensive and well-documented free API[14]. The provided API is RESTful, i.e. the request format is REST[15] and the responses are formatted in JSON (everything is UTF-8 encoded) and is available only via Mashape[16]. For the purposes of hackAIR, we implemented an image collector application that uses the webcams.travel API to collect data from European webcams. The details of the webcams.travel webcam image collector are given below.

To get a list of all webcams located in Europe along with all the required information, queries of the following type are used:

- https://webcamstravel.p.mashape.com/webcams/list/continent=EU/orderby=popular,desc/limit={limit},{offset}?show=webcams:basic,image,location

In this type of queries the */webcams/list/* endpoint is exploited along with the *continent=EU* explicit modifier which narrows down the complete list of webcams to contain only webcams located in Europe. Moreover, two implicit modifiers are used: a) *orderby* and b) *limit*. The *orderby* modifier has the purpose of enforcing an explicit ordering of the returned webcams. This is important because API limitations do not allow us to get data from more than about 1,000 out of the 24,319 European webcams contained in webcams.travel. By enforcing an explicit ordering (in this case webcams are sorted in descending popularity[17] order) we ensure that roughly the same webcams are returned in the top 1,000 results every time new data is pulled from the API. Having regular measurements from the same locations is beneficial for the data fusion module developed in WP4.

[14] https://developers.webcams.travel/

[15] https://en.wikipedia.org/wiki/Representational_state_transfer

[16] https://www.mashape.com/

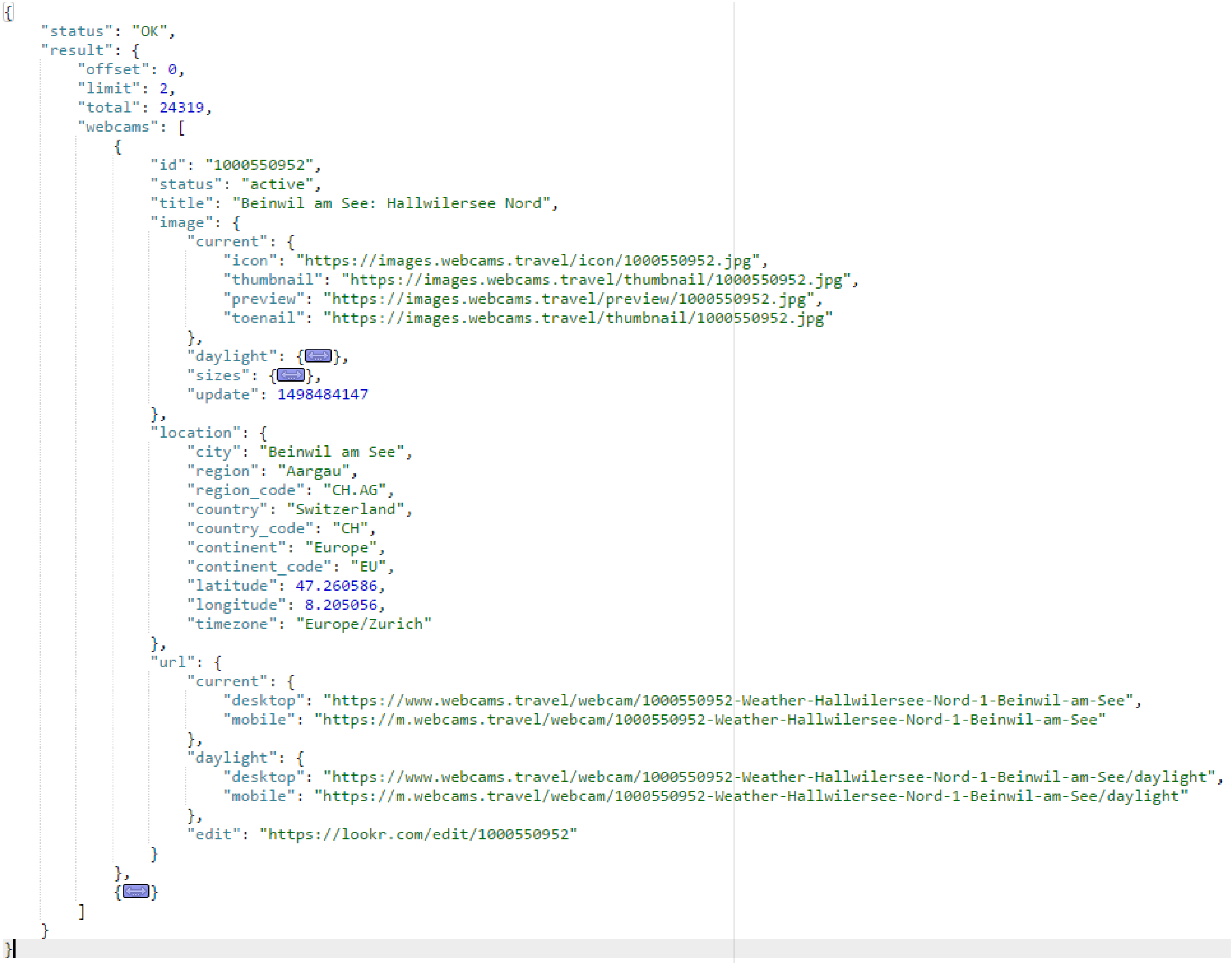
[17] According to webcams.travel API documentation: “popularity reflects which webcams are currently of interest”.

The other implicit modifier (*limit*) is used to slice the list of webcams by *limit* (the number of webcams in the resulting list) and *offset* (the offset from where to start listing the webcam for the resulting list). The use of this modifier is necessary because the maximum number of results that can be returned with a single query is 50 (i.e. the max value of the *limit* parameter is 50) and in our case, we want to pull data from 1,000 webcams. Thus, 20 queries must be performed with appropriate values for the offset parameter. The last part of the query (*show=webcams:basic,image,location*) is used so that the response contains webcam objects that besides the basic information for each webcam (id, status, title) also contain the URL of the latest image captured from the webcam (and its timestamp) and the webcam's exact geographical location.

Figure 8 shows the form of the response returned by webcams.travel to the above query. After all the queries have been completed, all the collected webcam objects are parsed to extract the required information and MongoDB records of a similar form with those created for AMOS webcams are created. Non-active webcams and duplicate images are handled in the same way as described above for the AMOS dataset. We notice that the *webcams.image* object contains pointers to four differently sized images. Among them, we pick the URL pointing to the largest size image which is "preview" and has a size of 400x224. Figure 9 shows an example MongoDB record for an image from webcams.travel.

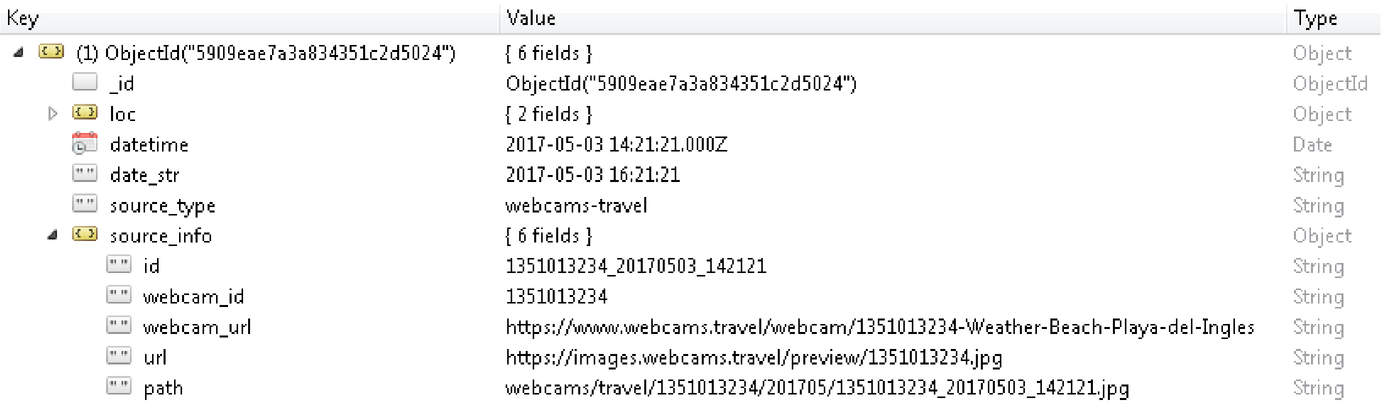
Similarly to the AMOS image collector, the webcams.travel image collector is executed four times per day (at 7:00, 11:00, 14:00 and 18:00). Figure 10 shows the geographical distribution of the 1,000 most popular European webcams from webcams.travel[18]. In this case, Switzerland is the country with the most webcams (283) followed by Italy and Germany with 253 and 177 webcams respectively.

[18] Note that a slightly different set of webcams might be returned each time this query is realized.



```
{
    "status": "OK",
    "result": {
        "offset": 0,
        "limit": 2,
        "total": 24319,
        "webcams": [
            {
                "id": "1000550952",
                "status": "active",
                "title": "Beinwil am See: Hallwilersee Nord",
                "image": {
                    "current": {
                        "icon": "https://images.webcams.travel/icon/1000550952.jpg",
                        "thumbnail": "https://images.webcams.travel/thumbnail/1000550952.jpg",
                        "preview": "https://images.webcams.travel/preview/1000550952.jpg",
                        "toenail": "https://images.webcams.travel/thumbnail/1000550952.jpg"
                    },
                    "daylight": {⇔},
                    "sizes": {⇔},
                    "update": 1498484147
                },
                "location": {
                    "city": "Beinwil am See",
                    "region": "Aargau",
                    "region_code": "CH.AG",
                    "country": "Switzerland",
                    "country_code": "CH",
                    "continent": "Europe",
                    "continent_code": "EU",
                    "latitude": 47.260586,
                    "longitude": 8.205056,
                    "timezone": "Europe/Zurich"
                },
                "url": {
                    "current": {
                        "desktop": "https://www.webcams.travel/webcam/1000550952-Weather-Hallwilersee-Nord-1-Beinwil-am-See",
                        "mobile": "https://m.webcams.travel/webcam/1000550952-Weather-Hallwilersee-Nord-1-Beinwil-am-See"
                    },
                    "daylight": {
                        "desktop": "https://www.webcams.travel/webcam/1000550952-Weather-Hallwilersee-Nord-1-Beinwil-am-See/daylight",
                        "mobile": "https://m.webcams.travel/webcam/1000550952-Weather-Hallwilersee-Nord-1-Beinwil-am-See/daylight"
                    },
                    "edit": "https://lookr.com/edit/1000550952"
                }
            },
            {⇔}
        ]
    }
}
```

Figure 8 - Example response from the webcams.travel API.



Key	Value	Type
◢ (1) ObjectId("5909eae7a3a834351c2d5024")	{ 6 fields }	Object
_id	ObjectId("5909eae7a3a834351c2d5024")	ObjectId
▷ loc	{ 2 fields }	Object
datetime	2017-05-03 14:21:21.000Z	Date
date_str	2017-05-03 16:21:21	String
source_type	webcams-travel	String
◢ source_info	{ 6 fields }	Object
id	1351013234_20170503_142121	String
webcam_id	1351013234	String
webcam_url	https://www.webcams.travel/webcam/1351013234-Weather-Beach-Playa-del-Ingles	String
url	https://images.webcams.travel/preview/1351013234.jpg	String
path	webcams/travel/1351013234/201705/1351013234_20170503_142121.jpg	String

Figure 9 - The MongoDB record of a webcam image from webcams.travel.

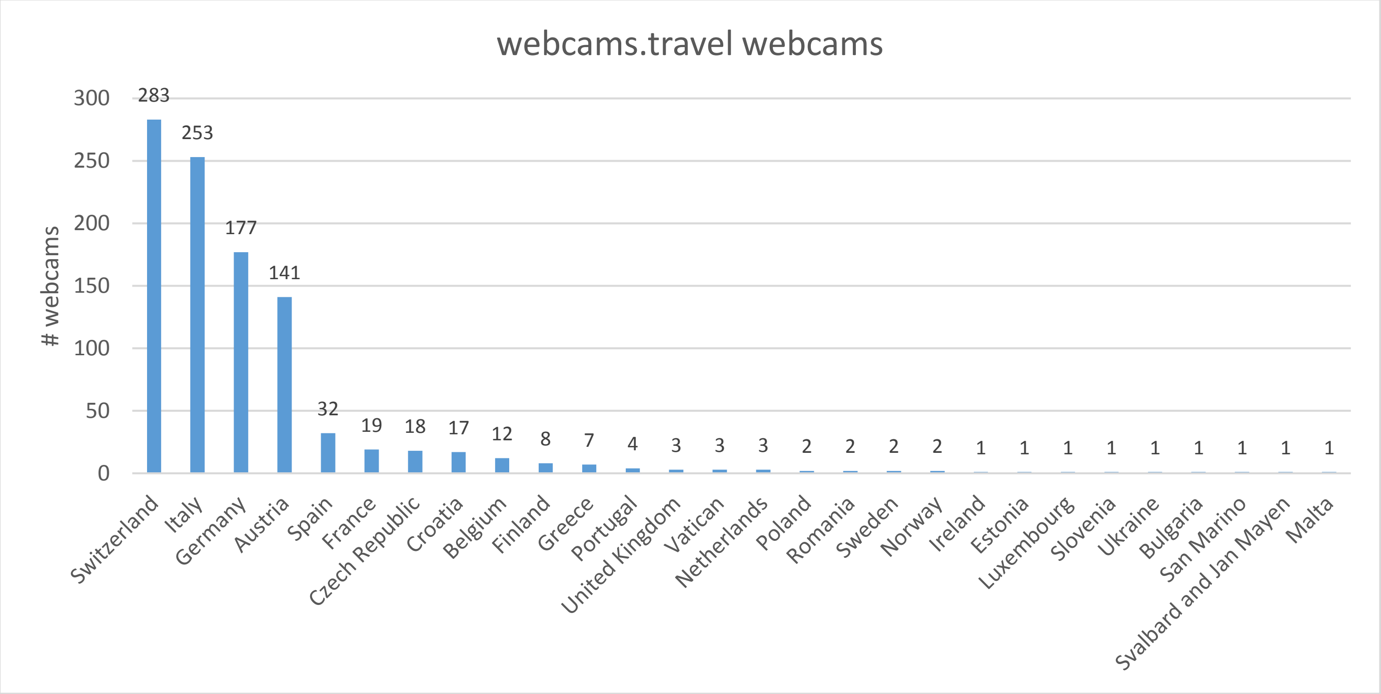


Figure 10 – Geographical distribution of webcams.travel webcams.

2.3 Image Collection Statistics

The three image collectors, i.e. the updated Flickr collector, the AMOS webcams collector and the webcams.travel collector, have been collecting images since 24/2/2017, 6/3/2017 and 2/5/2017, respectively. During this period and until 15/5/2017 (the date when a snapshot of the repository was taken for reporting purposes) 1,019,938 images had been collected in total across the whole Europe from all sources. In the following paragraphs, we present statistics of the image collection.

Figure 11 shows the number of images collected daily from each source. We see that the number of images collected each day by the two webcam image sources is almost stable (apart for few days were the collection of images from AMOS failed due to network connectivity issues with the server that runs the data collector) since an almost fixed number of webcams are visited a fixed number of times each day. In particular, 2,246 webcams from AMOS and 1,000 webcams from webcam.travel are visited exactly four times per day and, as a result, about 9,000 and 4,000 images, respectively, are collected daily from these sources. On the other hand, the number of images collected daily from Flickr exhibits a large variability since it depends on the number of geotagged images (in Europe) that are uploaded daily by Flickr users. As expected, the number of images collected from Flickr increases significantly during Saturday and Sunday, since users tend to capture and upload more images during weekends. On average, about 5,500 images are collected daily from Flickr.

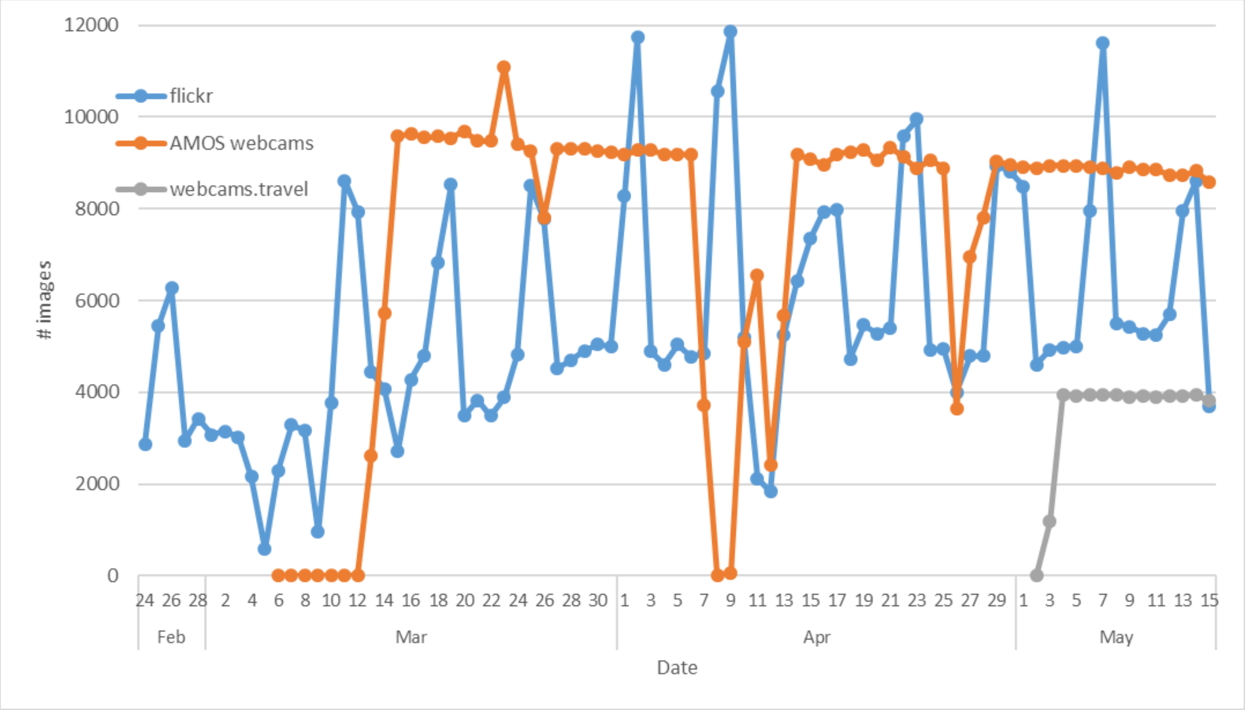


Figure 11 – Number of images collected daily from each source

During the collection period, we retrieved images from almost every country in Europe. Figure 12 shows the percentage of the total number of collected images (≈1M) corresponding to each country, while Figure 13 shows the total numbers of Flickr and webcam images collected from each country (only the top 20 countries are shown in all cases to increase the readability of the figures). We see that most images come from the UK, mainly because most Flickr images are from there, while Norway is second in the rank because it is the country where most webcams are located (given our collection criteria). Germany, another country of interest for hackAIR (since pilots will take place in Germany and Norway) is also very well covered, exhibiting a balanced number of Flickr and webcam images. Figure 14 shows the total number of images collected daily in Germany and Norway. We see that after the full integration of all image sources (in May), more than 1,000 and 2,000 images are collected daily from Germany and Norway, respectively.

As we will see in Section 3.4, these larger numbers of collected images lead to a large number of sky-depicting images that can be used for air quality estimation.

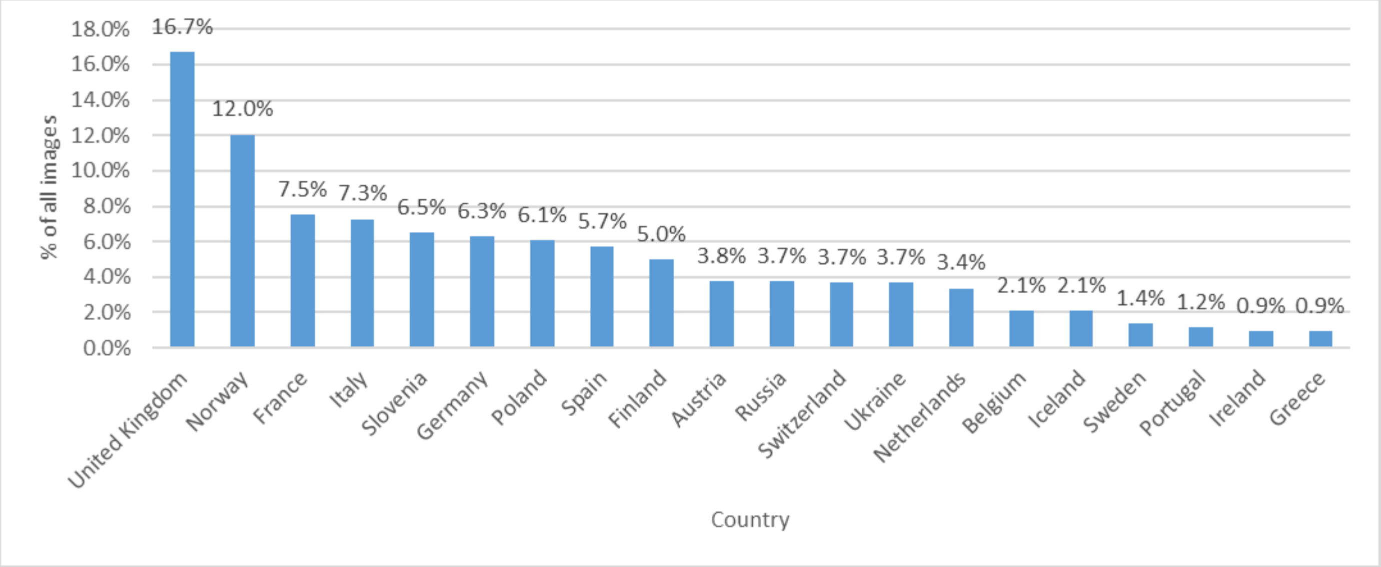


Figure 12 – Percentage of the total number of collected images coming from each European country (top 20 are shown)

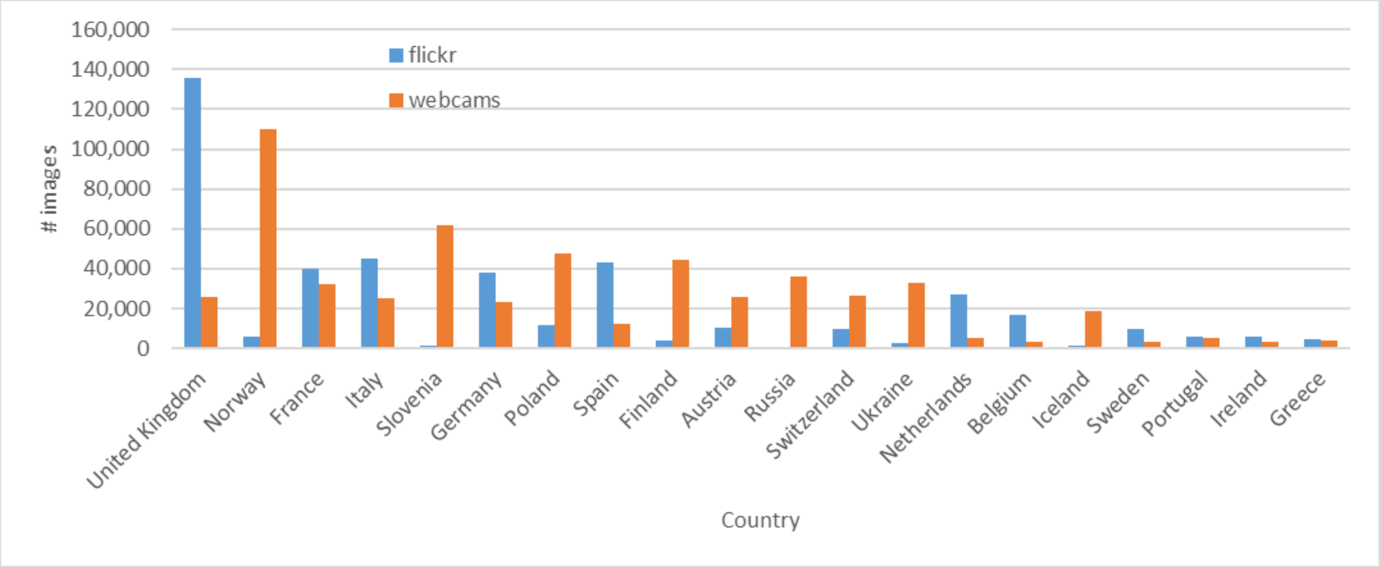


Figure 13 – Total number of Flickr and webcam images collected from each European country (top 20 are shown)

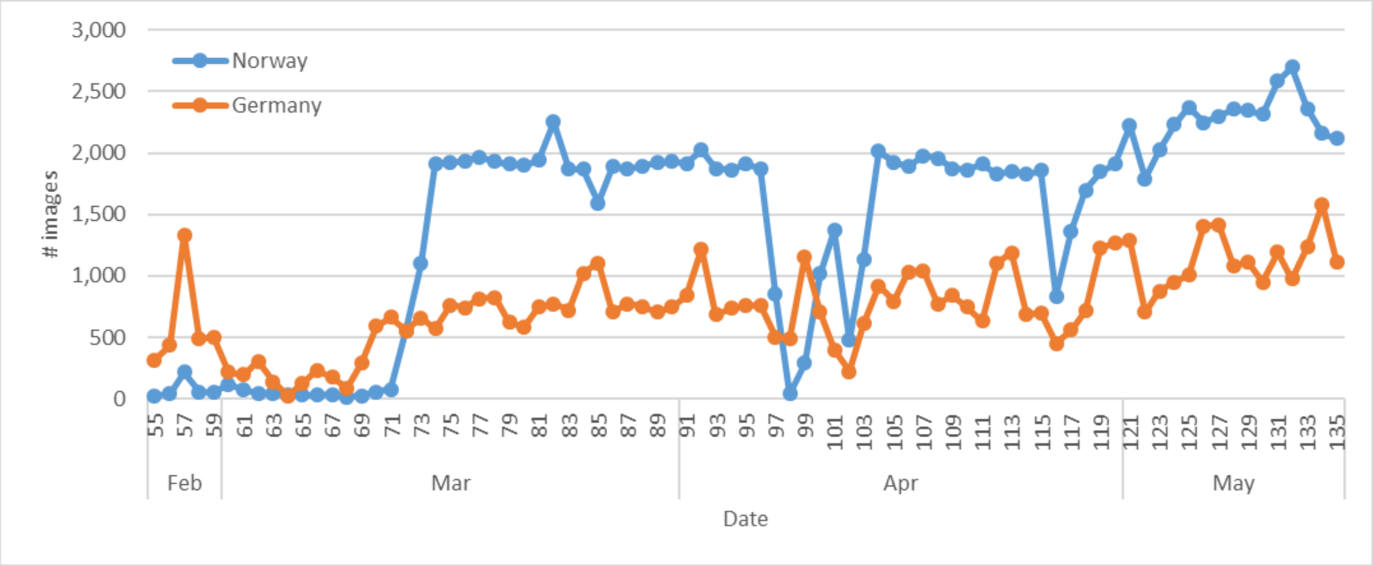


Figure 14 –Total number of images collected daily from Germany and Norway

2.4 Collecting Measurements from Ground Stations

In this section, we present the framework that we developed for collecting air quality measurements (specifically PM10 and PM2.5) from ground stations. The framework is based on the collection of data from OpenAQ[19], an air quality web service that was briefly described in D3.1 (section 3.3.3). OpenAQ is an open data platform that aggregates and shares air quality data from multiple official sources around the world. The data offered by the platform is of high quality as they mainly come from official, usually government-level organizations. The platform offers the data as they are received from their originating sources, without performing any kind of transformations. In particular, the following five main criteria are used for deciding upon the suitability of the data sources that are included in the platform[20]:

1. Data must be of one of these pollutant types: PM10 (of interest to hackAIR), PM2.5 (of interest to hackAIR), sulfur dioxide (SO2), carbon monoxide (CO), nitrogen dioxide (NO2), ozone (O3), or black carbon (BC).
2. Data must be from an official-level stationary, outdoor air quality source, defined as data produced by a government entity or international organizations.
3. Data must be 'raw' and reported in physical concentrations on their originating site.
4. Data must be at the 'station-level,' not aggregated into a higher (e.g. city) level.
5. Data must be from measurements averaged between 10 minutes and 24 hours.

Importantly, the OpenAQ system checks each data source for updates information every 10 minutes. Thus, it is guaranteed that the data will be almost as real-time as they are offered by the original sources. With respect to geographical coverage, the platform collects measurements from 5,629 locations in 48 countries. Since the focus of the hackAIR project is the collection of PM10 and PM2.5 measurements from countries in Europe, Table 2 shows the European countries for which PM10 and/or PM2.5 data is provided, the number of locations that provide data in each country, as well as the corresponding data source. We observe that data for 17 European countries are available. In most cases, the data source is the European Environmental Agency[21] (EEA) but additional official-level data sources are included (e.g. DEFRA[22] in the United Kingdom). We observe that PM10 data are available from more locations in each country compared to PM2.5 data with only three exceptions (United Kingdom, Belgium and Poland) where a similar number of PM10 and PM2.5 locations are available. In total, OpenAQ provides PM10 and PM2.5 data from 1728 and 737 locations in Europe, respectively. We also observe that the countries of the pilots are very well represented, with 434 PM10 and 183 PM2.5 locations in Germany and 48 PM10 and 37 PM2.5 locations in Norway.

Table 2 – European countries for which data is available in the OpenAQ platform along with data source (second column), number of locations with PM 10 data (third column) and number of locations with PM2.5 data (fourth column). The URLs of the data sources are used in some cases because the source name is not provided by OpenAQ.

Country	Data source	# PM 10 locations	# PM 2.5 locations
France	EEA France	372	147
Germany	EEA Germany	434	182
Spain	EEA Spain	243	72
Austria	EEA Austria	199	0
United Kingdom	DEFRA	71	74

[19] https://openaq.org

[20] More details can be found here: https://medium.com/@openaq/where-does-openaq-data-come-from-a5cf9f3a5c85

[21] https://www.eea.europa.eu/

[22] https://uk-air.defra.gov.uk/

Netherlands	http://www.lml.rivm.nl/sos/	94	64
Czech Republic	EEA Czech Republic	83	52
Belgium	EEA Belgium	61	63
Norway	luftkvalitet.info	48	37
Finland	EEA Finland	37	16
Croatia	EEA Croatia	17	9
Hungary	EEA Hungary	24	1
Bosnia and Herzegovina	http://www.fhmzbih.gov.ba	11	5
Poland	http://sojp.wios.warszawa.pl	9	10
FYROM	EEA FYROM	15	0
Sweden	Swedish data from SLB analys	9	4
Ireland	EEA Ireland	1	1
Total		**1728**	**737**

Based on its characteristics, OpenAQ is considered an ideal source of PM10 and PM2.5 ground station measurements for the hackAIR platform. Therefore, a specialized data collection framework was developed to retrieve data from the REST API provided by OpenAQ[23]. The */latest* endpoint[24] of the API is used, which provides the latest value of each available parameter (pollutant) for every location in the system. To avoid retrieving results from non-European countries, we use the optional parameter "country" that is used to limit results by a certain country. In addition, to avoid retrieving results for pollutants other than PM10 and PM2.5, the "parameter" parameter is used. Thus, two queries are performed for each of the 17 European countries, one to retrieve the latest PM10 measurements and one to retrieve the latest PM2.5 measurements. For instance, the following two queries are used to retrieve the latest data PM10 and PM2.5 data for Norway:

- https://api.openaq.org/v1/latest?parameter=pm10&country=NO
- https://api.openaq.org/v1/latest?parameter=pm25&country=NO

Figure 15 shows a part of the response of the OpenAQ API to the first of the above queries. We notice that the response contains all the required information for each measurement, i.e. exact geolocation and time, value and unit.

The air quality data collector queries the OpenAQ API for the latest data once every hour and stores new measurements in the MongoDB-based environmental node repository that was described in D3.1 (section 6.4). Figure 16 shows an example record from this repository. We see that the record contains the following fields:

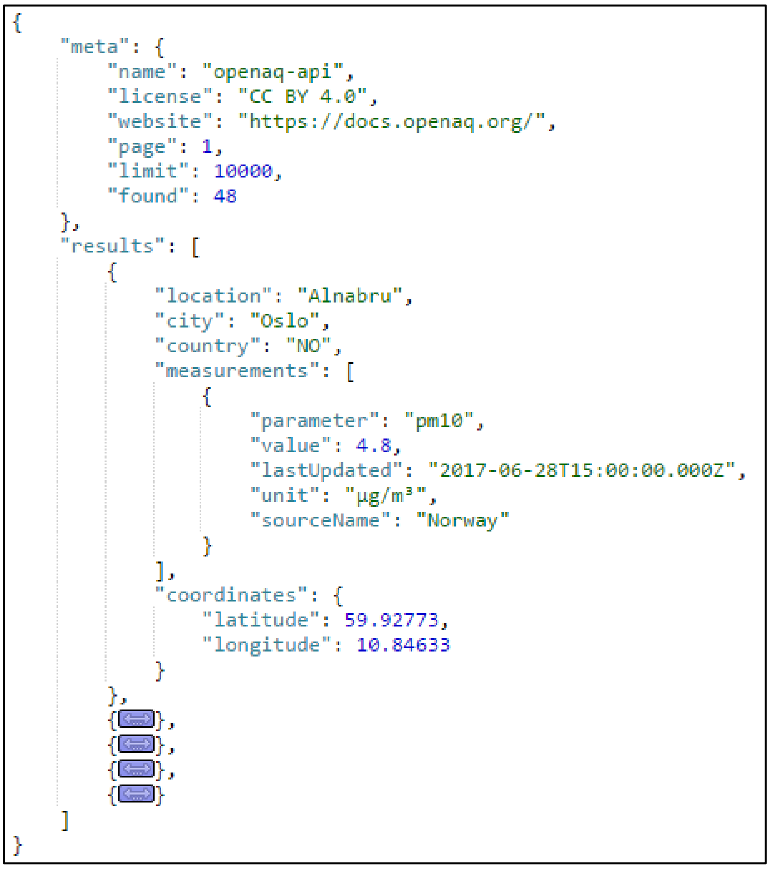
- "_id" is a unique object identifier added automatically by MongoDB
- "datetime" stores the timestamp of the measurement
- "loc" stores the geographical coordinates of the measurement station
- "source_type" stores the type of the data source of the measurement[25]

[23] https://docs.openaq.org/

[24] https://docs.openaq.org/#api-Latest

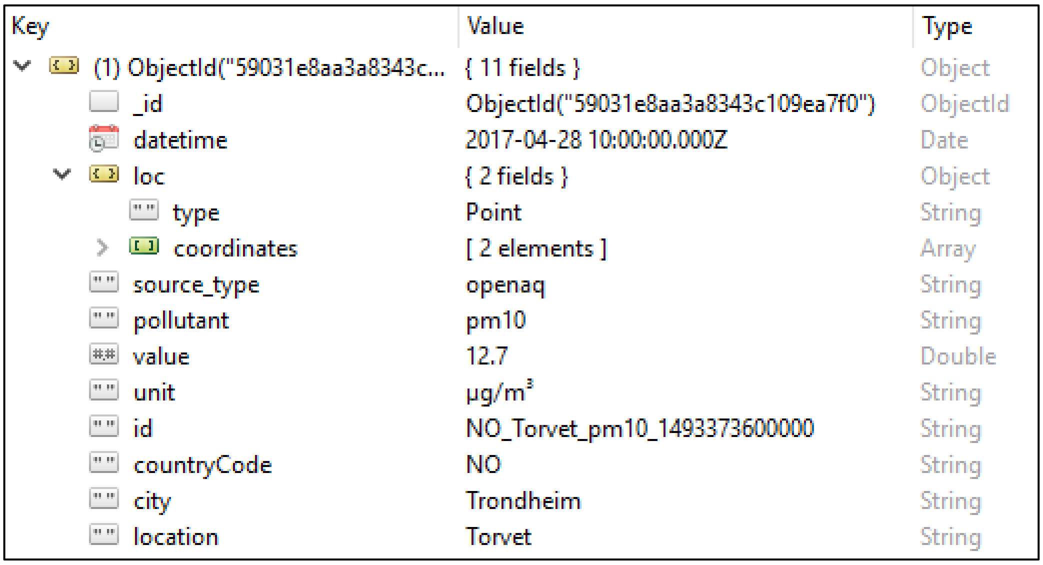
[25] Currently all measurements come from OpenAQ but additional source types can be integrated in the future.

- “pollutant” is the type of the pollutant (pm10 or pm25)
- “value” is the value of the pollutant
- “unit” is the measurement unit
- “countryCode” the code of the country as given from OpenAQ
- “location” the name of the location in OpenAQ
- “sourceName” the name of the source in OpenAQ
- “id” is a unique identifier (upon which a unique MongoDB index is built) populated as [“countryCode”_“location”_“pollutant”_timestamp]. It is used to avoid retrieval of measurements that have already been retrieved (i.e. duplicates).



```
{
    "meta": {
        "name": "openaq-api",
        "license": "CC BY 4.0",
        "website": "https://docs.openaq.org/",
        "page": 1,
        "limit": 10000,
        "found": 48
    },
    "results": [
        {
            "location": "Alnabru",
            "city": "Oslo",
            "country": "NO",
            "measurements": [
                {
                    "parameter": "pm10",
                    "value": 4.8,
                    "lastUpdated": "2017-06-28T15:00:00.000Z",
                    "unit": "µg/m³",
                    "sourceName": "Norway"
                }
            ],
            "coordinates": {
                "latitude": 59.92773,
                "longitude": 10.84633
            }
        },
        {↔},
        {↔},
        {↔},
        {↔}
    ]
}
```

Figure 15 – Example response from the latest endpoint of the OpenAQ API.

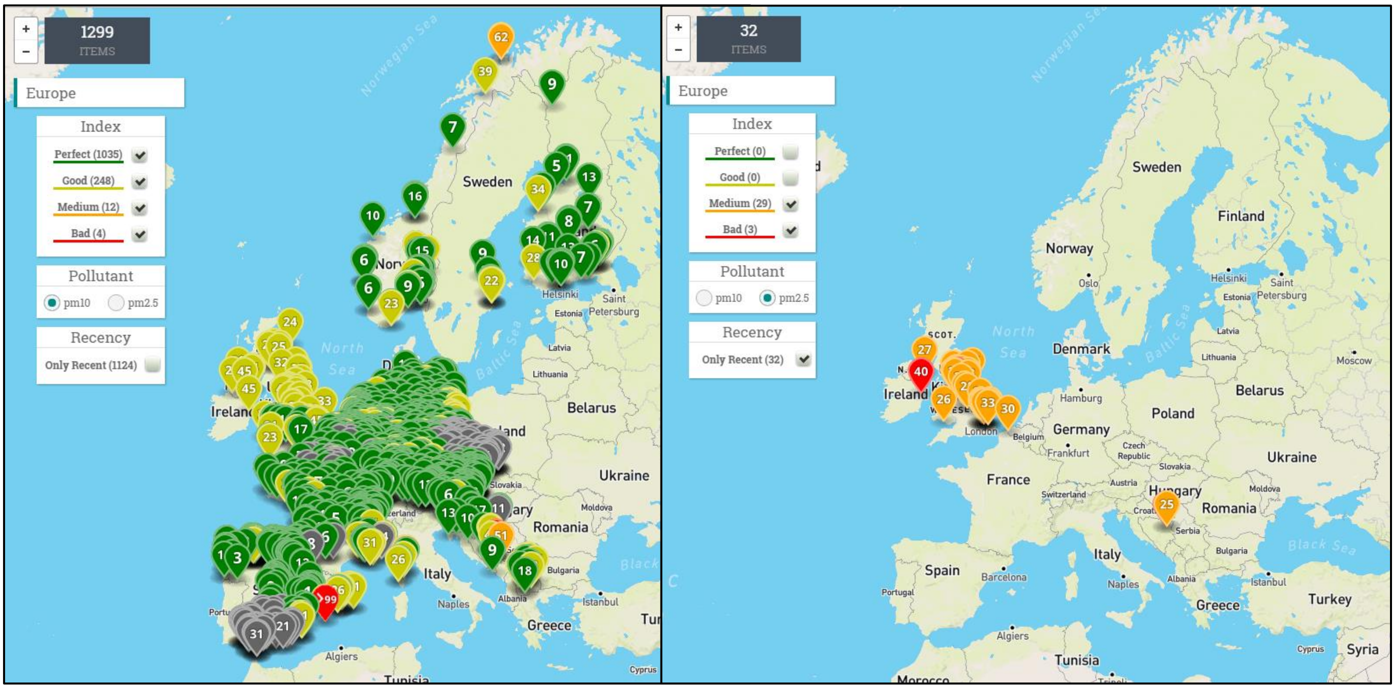


Key	Value	Type
(1) ObjectId("59031e8aa3a8343c...	{ 11 fields }	Object
_id	ObjectId("59031e8aa3a8343c109ea7f0")	ObjectId
datetime	2017-04-28 10:00:00.000Z	Date
loc	{ 2 fields }	Object
type	Point	String
coordinates	[2 elements]	Array
source_type	openaq	String
pollutant	pm10	String
value	12.7	Double
unit	μg/m³	String
id	NO_Torvet_pm10_1493373600000	String
countryCode	NO	String
city	Trondheim	String
location	Torvet	String

Figure 16 – Example MongoDB record of an environmental measurement from the OpenAQ API.

The environmental data retrieval framework was deployed on 21/12/2016 and has been continuously collecting data since then. As a result, the repository currently contains more than 1 million measurements in total. To facilitate an easy inspection of the collected data, their geographical distribution, and the current air quality conditions in Europe (in terms of PM10 and PM2.5) we built a web application[26] that displays the latest PM10 and PM2.5 measurements with appropriate markers on a map. Figure 17 shows two screenshots of the application. We see that the application offers three ways of filtering the results: a) based on country (initially results from all European countries are shown), b) based on pollutant type (PM10/PM2.5) and c) based on the pollution class (index) corresponding to each measurement. The mapping from absolute PM10/PM2.5 values to pollution classes is performed according to Table 3. In addition, there is the option of filtering measurements that are not recent (i.e. older than 24hours). We see that markers contain a number that corresponds to the pollutant value at the specific location and are colored according to the respective pollution class. In case non-recent measurements are not filtered, the corresponding markers have a grey color. When an individual marker is clicked, a pop-up window opens that shows additional details about the measurements such as the time it was last updated and the name of the location.

[26] http://hackair-mklab.iti.gr/sensors/



1299 ITEMS

Europe

Index

Perfect (1035) ✓
Good (248) ✓
Medium (12) ✓
Bad (4) ✓

Pollutant

● pm10 ○ pm2.5

Recency

Only Recent (1124) □

32 ITEMS

Europe

Index

Perfect (0) □
Good (0) □
Medium (29) ✓
Bad (3) ✓

Pollutant

○ pm10 ● pm2.5

Recency

Only Recent (32) ✓

Figure 17 - Screenshots of the ground station data collection visualization web application.

Table 3 - Mapping of absolute PM10/PM2.5 values to pollution classes.

Class name	PM10 scale (µg/m3)	PM2.5 scale (µg/m3)
Very good	>=0 and <=20	>=0 and <=10
Good	>20 and <=50	>10 and <=25
Medium	>50 and <=70	>25 and <=35
Bad	>70	>35

3 Image Analysis Experiments and Statistics

In D3.1, we presented and evaluated two alternative methods for detecting sky regions in sky-depicting images. The first approach is based on state-of-the-art machine learning algorithms and consists of the combination of a visual concept detection framework (see section 5.2.1.1 of D3.1) that is used to detect images that depict sky, and a sky localization framework (see section 5.2.1.2 of D3.1) that is used to specify the sky region of the image. The second approach consists of a set of simple heuristic rules provided by image-based air quality estimation experts (DUTH) and aims at directly detecting sky regions that are suitable for image-based air quality estimation. According to the evaluation reported in D3.1, the machine learning-based approach performed better than the heuristic approach. However, the evaluation was performed on a general-purpose benchmark collection where the regions annotated as sky might not always be suitable for air quality estimation (e.g., because they contain clouds).

In section 3.1, we present a new evaluation of the two methods (and of their combination) on a real-world dataset that was annotated by air quality estimation experts from DUTH based on the suitability of the sky region for performing air quality estimation using the Look Up Table (LUT)-based approach that was presented in D3.3. Then, in section 3.2, we study the robustness of image analysis results with respect to various widely applied image transformation and filters. Section 3.3 describes the final architecture of the image analysis service that we developed and discusses the computational load of its various processing steps. Finally, section 3.4 presents statistics of the image analysis service (collected over a period of more than two months) that allow us to estimate the number of usable images that we can retrieve daily, both across the whole Europe and in specific countries of interest to hackAIR (i.e. Germany and Norway).

3.1 Evaluation of Sky Localization Methods

In this section, we carry out an air quality estimation-oriented evaluation of the two sky localization approaches presented in D3.1: Fully Convolutional Neural Networks (*FCN*) (Long et al., 2015) and the heuristic rule-based approach proposed by DUTH (*heuristic*). In D3.1, both approaches were evaluated on the SUN database[27] (Xiao, 2010), a general-purpose benchmark collection for image annotation and segmentation tasks. The evaluation was carried out on 2,030 images that were annotated with the concept sky and for which the polygons of the sky part of the image were provided. In this evaluation, the *FCN* approach was found to perform significantly better than the *heuristic* approach as it achieved a 0.9177 pixel-wise precision and a 0.9425 pixel-wise recall versus a 0.8245 pixel-wise precision and a 0.5922 pixel-wise recall for the *heuristic* approach. However, a more critical analysis of the results that involved a visual inspection of the ground truth annotations of the collection's images, revealed that the image region that is annotated as "sky" is not always suitable for air quality estimation as in many cases the sky part is not clear (e.g. contains clouds, the sun, small objects, etc.). In addition, hackAIR's image collection framework has been extended to include webcam images which are expected to pose additional challenges to the sky localization methods due to their distinct characteristics (e.g., text overlays).

For these reasons, we designed a new specialized evaluation of the two sky localization methods that focuses explicitly on their ability to correctly identify sky regions that are suitable for air quality estimation using the LUT-based approach. To this end, out of ≈500K images that we collected during the period 24/2/2017-14/3/2017, we filtered out those in which the detection confidence of the sky concept is not very high (< 0.8) to ensure that most of the remaining images will depict sky and then took a random sample of approximately 100 Flickr and 100 Webcam images. For each of these images, we extracted sky masks using: a) the *FCN* approach and b) the *heuristic* approach and with the help of experts from DUTH we answered the following questions for each image:

- Q1-a: Does the image contain a sky region usable for air quality estimation? (Yes/No)

[27] http://groups.csail.mit.edu/vision/SUN/

- Q1-b: Please shortly describe the reason if you answered No to Q1-a.
- Q2-a: Is the sky region selected with the FCN approach usable for air quality estimation? (Yes/No)
- Q3-a: Is the sky region selected with the heuristic approach usable for air quality estimation? (Yes/No)

The first question (Q1-a) aims at helping us identify images with a sky region usable for air quality estimation, so that we can subsequently evaluate the different sky localization methods only on images with a usable sky region. Figure 18 shows the distribution of responses to Q1-a and Q1-b, separately for Flickr images (left) and webcam images (right). We see that in both cases, about 60% of the images contain a sky region that is usable for air quality estimation (Yes to Q1-a). Looking at the distribution of responses to Q1-b, we see that in most cases and for both Flickr and webcam images, it is the presence of clouds or cirrus clouds (a genus of atmospheric cloud generally characterized by thin, wispy strands) or the fact that the image is captured too early in the morning or too late in the evening that render images unusable for air quality estimation, despite the existence of a sky region. Other reasons include humidity, rain/snow, strange images (usually deformed webcam images due to camera movement), artistic images and a very small number of images (5 out of 197) that do not depict sky at all. Figure 19 shows some examples of sky-depicting images that are considered unusable for air quality estimation.

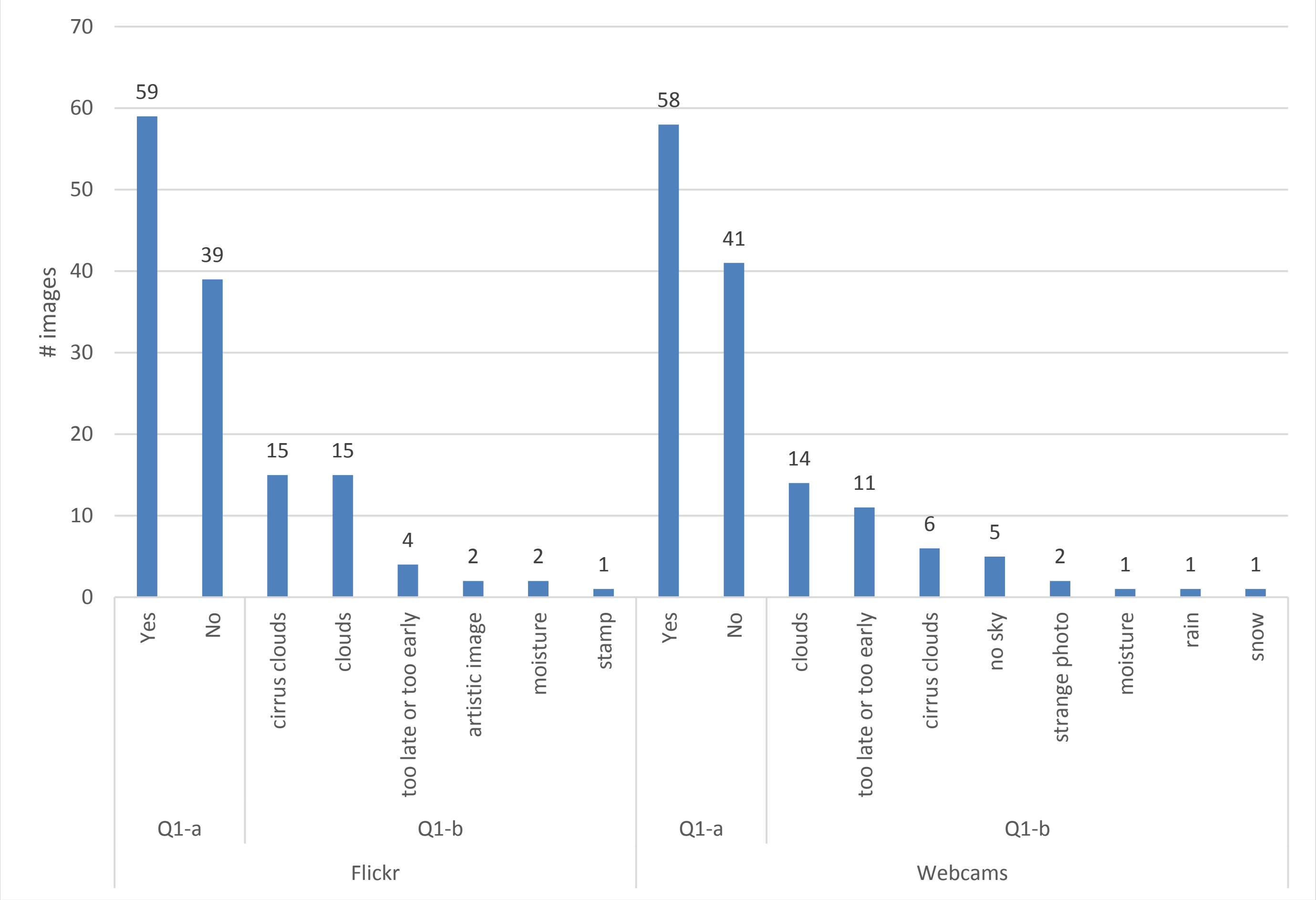


Figure 18 – Reponses to Q1-a and Q1-b for Flickr images (left) and webcam images (right).

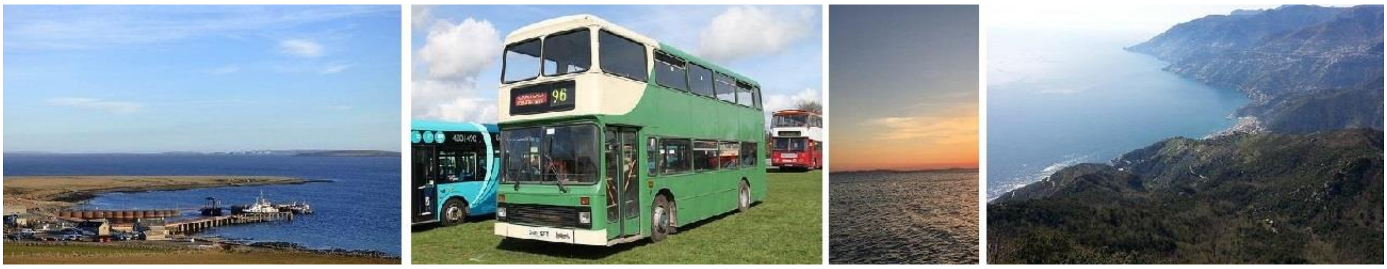


Figure 19 – Examples of sky-depicting images that are considered unusable for air quality estimation due to (left to right): a) cirrus clouds, b) clouds, c) hour of the day (too late or too early), d) humidity

Having identified images with usable sky regions, we now focus our analysis on the ability of each sky localization approach to extract these regions. The results are presented in Figure 20, which shows the percentages of correctly detected image regions using the *FCN* (Q2-a) and the *heuristic* (Q3-a) approach for Flickr and webcam images. At a first glance, the performance of the two methods appears much worse than the performance obtained on the SUN database. Note, however, that the evaluation performed here is much stricter as even if a small percentage of the region recognized as sky includes non-sky elements (e.g., clouds, buildings, text overlays), then the whole region is marked incorrect. An illustrative example is provided in Figure 21, depicting a case where both masks are considered incorrect, even though a sizable percentage of the region recognized as sky is indeed sky (especially in the *FCN* approach). Moreover, we observe that in contrast to the results obtained when the evaluation was performed on the SUN database, the *heuristic* approach performs better than the *FCN* approach as it manages to correctly detect the sky region in 45.76%/50.00% of the Flickr/webcam images versus only 28.81%/20.69% for the FCN approach. As we found by performing a visual inspection of the masks generated by each approach, this difference probably stems from the fact that the *heuristic* approach generates much more detailed sky masks which seems to be advantageous for this type of evaluation.

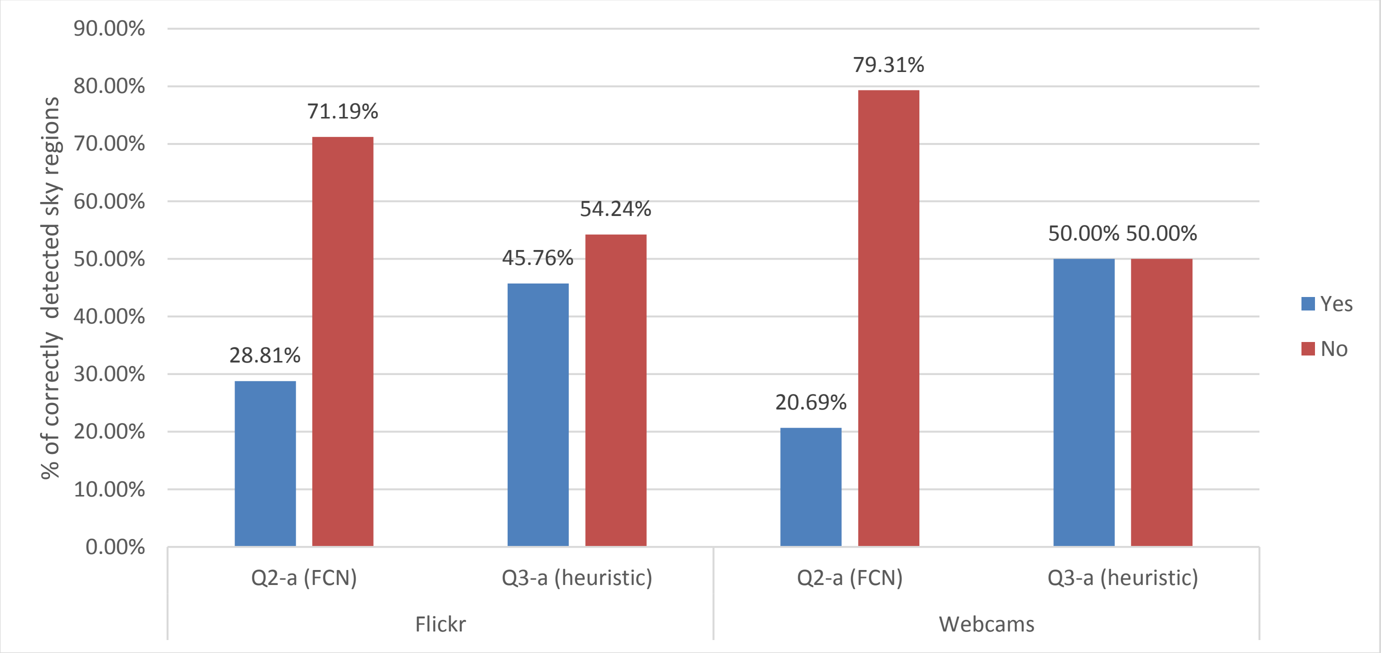


Figure 20 - Percentages of correctly/incorrectly detected sky regions using each sky localization approach for Flickr images (left) and webcam images (right).

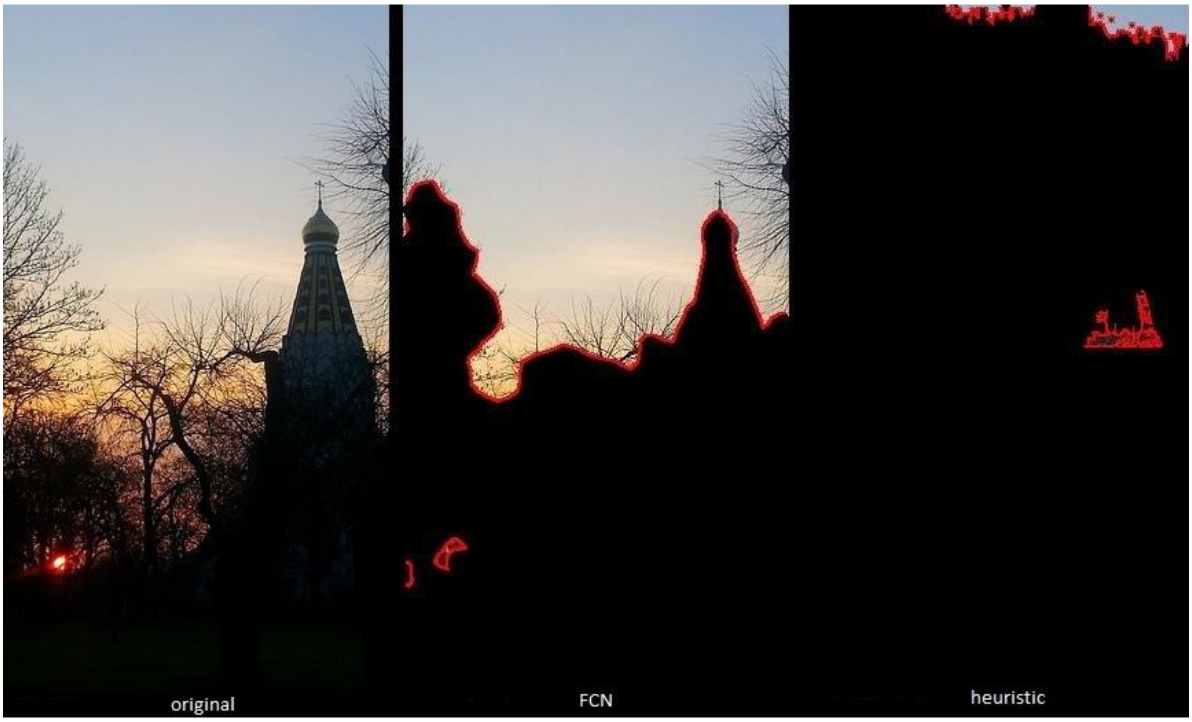


Figure 21 - Inappropriate sky masks extracted using the FCN (middle) and the heuristic (right) approach for a Flickr image (left).

Visual comparison of the generated masks revealed that each approach has its own merits and works better in different situations. Particularly, we noticed that the *FCN* approach is better at avoiding "big" mistakes (e.g. recognizing sea, buildings or windows as sky), while the *heuristic* approach is very good at filtering out small objects (e.g. tree branches) and text overlays that are very common in images from webcams. Some illustrative examples are presented in Figure 22. In the first row, we see that although the *FCN* approach (2nd column) correctly identifies the sky region, it misses the white pillar. The *heuristic* approach (3rd column), on the other hand, correctly filters the pillar but gets confused by the bus windows. Similar is the situation in the second and third row where we see that the *heuristic* approach manages to discard small non-sky elements (clouds in the second row and text overlay in the third row) that are not discarded by the *FCN* approach which, on the other hand, does a much better job at not being confused by water (second row) and part of mountains (third row).

Motivated by the complementarity of the two approaches, we decided to develop a sky localization approach that combines them (*FCN+heuristic*). More specifically, we first calculate a sky mask using the FCN approach and then apply the *heuristic* algorithm described in D3.1, considering only those pixels that have been recognized as sky by the FCN approach. This way, we manage to exploit the effectiveness of the *FCN* approach in roughly recognizing the sky region of the image and then utilize the *heuristic* approach to discard small non-sky elements. The last column of Figure 22 shows the masks extracted by the *FCN+heuristic* approach. We see that in all cases, *FCN+heuristic* correctly identifies the sky region. Besides this visual evaluation, we also performed a quantitative evaluation of the *FCN+heuristic* approach, as we did for *FCN* and *heuristic*, i.e. we counted the number of times a usable sky region was extracted by the *FCN+heuristic* approach by collecting responses to the question: "Q4-a: Is the sky region selected with the *FCN+heuristic* approach usable for air quality estimation? (Yes/No)". The results of this evaluation are presented in Figure 23, which shows the percentages of correctly and incorrectly detected sky regions for each approach, when considering all images (Flickr and webcam images). As expected, there is a very large improvement as **80.34%** of the sky regions are correctly recognized by the *FCN+heuristic* approach, compared to 47.86% for the *heuristic* approach and 24.79% for the FCN approach.

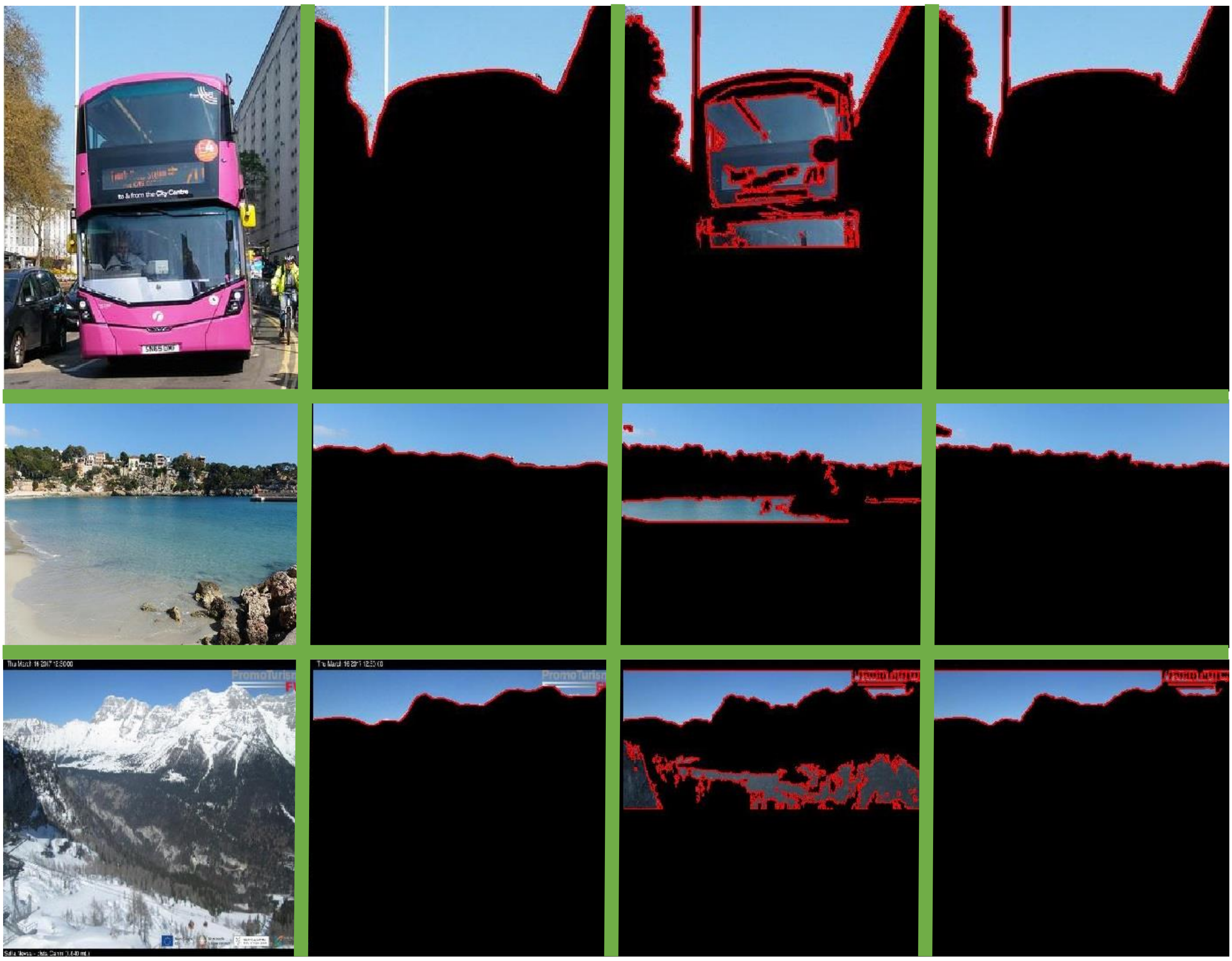


Figure 22 – Comparison of the masks generated by the FCN approach (second column) with the masks generated by the heuristic approach (third column) for the images of the first column. The fourth column shows the masks generated by combining the FCN approach with the heuristic approach.

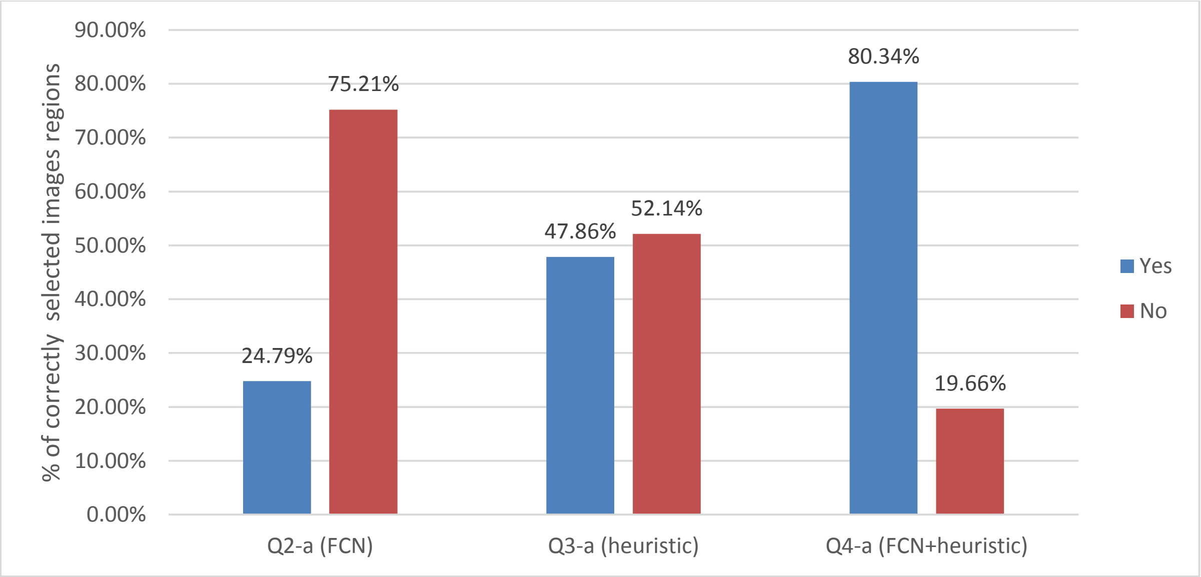


Figure 23 – Comparison of FCN, heuristic and FCN+heuristic sky localization approaches when all images are considered.

3.2 Robustness of Image Analysis against Image Transformations

A positive consequence of relying on images from Flickr and webcams instead of images from Instagram is that: a) the use of image filters that might distort the calculated R/G and G/B ratios is expected to be less frequent on Flickr than it is on Instagram[28], b) webcam images are typically unfiltered. Nevertheless, Flickr introduced a new interface (Figure 24) in its Android/IOS apps in 2014 which allows users to apply Instagram-like filters (14 different filters are provided) and to adjust the overall appearance of the pictures by changing the levels of brightness, contrast, saturation, color balance, etc. Moreover, advanced users sometimes preprocess their images with independent image processing software before uploading them on Flickr. Therefore, it is important to study the impact that several common transformations have on the R/G and G/B ratios that are calculated from the images and used as input by the image-based air quality estimation module.

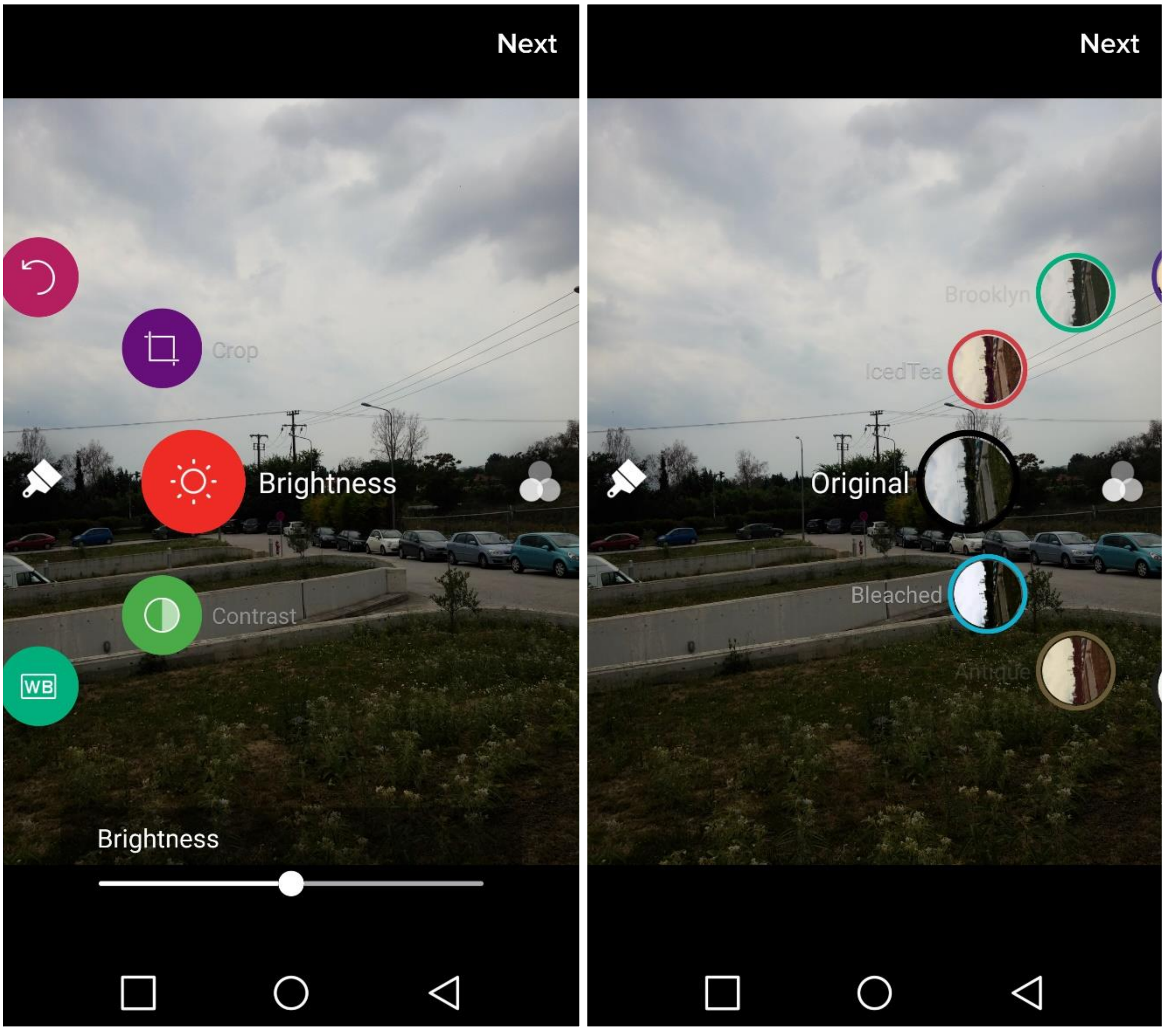


Figure 24 - Image enhancements (left) and filters (right) offered by Flickr's Android/IOS app.

To perform this kind of analysis we had to: a) select a representative set of original images, b) apply a number of widely used image transformations to generate transformed versions of the original images, c) apply image analysis to extract

[28] Although we could not find or collect (filter information is not provided through the API) filter usage statistics, we presume that the practice of using image filters to attract more likes is much more pronounced on Instagram due its app design and mobile-first and “social” nature. In contrast, uploading images on Flickr is typically done through a web browser, while the option to apply filters is given only when images are uploaded through the mobile app.

R/G and G/B ratios from both the original and the transformed images and d) compare the ratios calculated from original versus transformed images.

With respect to the selection of a representative set of original images, we wanted to ensure (as much as possible) that the set will consist of images that: 1) have not already undergone transformations and 2) contain a sky region suitable for air quality estimation. One option would be to manually capture photos of the sky and validate the correctness of the extracted sky regions. However, in order to be able to perform a more large-scale and principled evaluation we followed a different approach. In particular, we used all Flickr images taken in 2012 and geotagged in Thessaloniki, Greece for which the AOD estimations generated by the image-based air quality estimation models (developed within T3.3) agree completely with the ground truth AOD values provided by both MODIS[29] and AERONET[30] (see D3.4 for more details)[31]. Since the images are from 2012 and the filters were introduced in the Flickr app in 2014, we know that no transformations have been applied on the images (at least using the Flickr app). Moreover, the fact that the AOD values estimated from these photos agree with the ground truth AOD values from two independent sources, ensures that the computed R/G and G/B scores are valid (i.e. have been calculated from a valid sky region).

With respect to the application of image transformations to generate transformed versions of the original images, one option would be to manually transform each image using the Flickr app. However, to evaluate the impact of all types of transformation offered by the Flickr app (>20) a prohibitive amount of manual effort would be required (20 transformations x 87 images = 1740 images would have to be manually transformed, uploaded to Flickr and then downloaded). Moreover, since users can preprocess their photos with other software before uploading them to Flickr, we decided to focus our analysis to common image transformations beyond those offered by the Flickr app.

To this end, the Cloudinary[32] platform was used. Cloudinary is a Software-as-a-Service (SaaS) solution for managing media assets of web or mobile applications in the cloud via a set of APIs. What we are interested in here is its comprehensive image manipulation API[33] which offers a wide variety of image transformations such as resizing, cropping, format conversion and, most importantly, image effects and filters. Among the large variety of transformations offered by Cloudinary we focused on the following four popular categories of transformations: a) resizing, b) color level effects, c) automatic image improvement effects and d) artistic filters.by those offered by Cloudinary.

Table 4 presents all the transformations that we applied (grouped by category) along with the effect that each transformation has on an example image and the R/G and G/B scores calculated from that image. From the resizing category, we chose four transformations that resize the original image to progressively smaller sizes. Robustness of the calculated ratios against downsizing is important because a moderate downsizing[34] is applied for efficiency reasons to all the collected images before they are sent for image analysis. The color level effects category includes transformations that are offered by the Flickr app, i.e. change of the image brightness and saturation, and the automatic image improvement effects category includes five transformations that are very commonly offered by image processing software. Finally, the artistic filters category includes eight artistic filters randomly selected by those offered by Cloudinary.

[29] https://modis.gsfc.nasa.gov/

[30] https://aeronet.gsfc.nasa.gov/

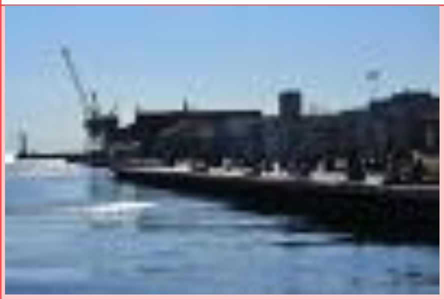
[31] This dataset was originally composed to facilitate an evaluation of the image-based air quality estimation models developed within T3.3.

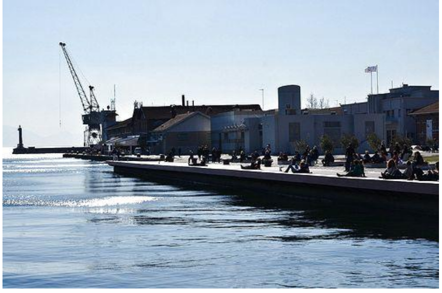
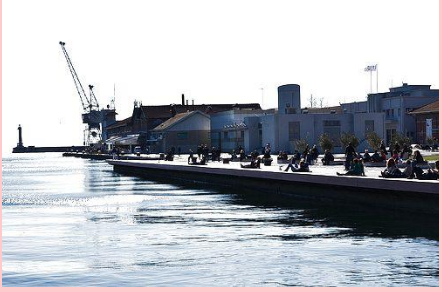
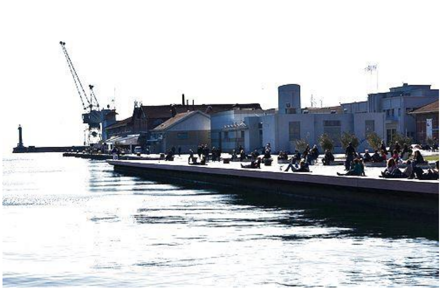
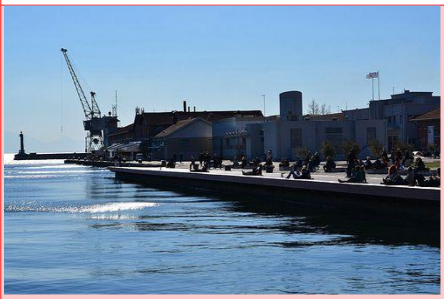
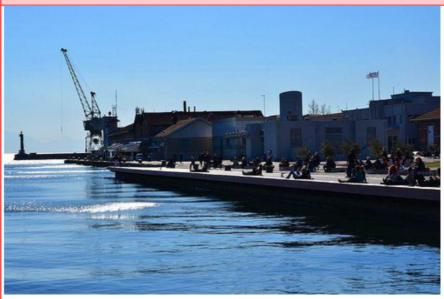
[32] http://cloudinary.com/

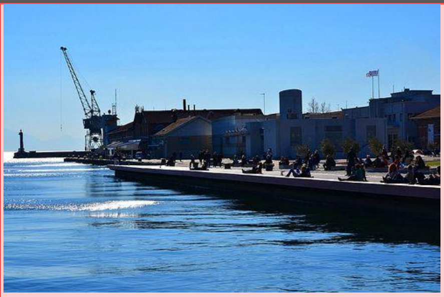
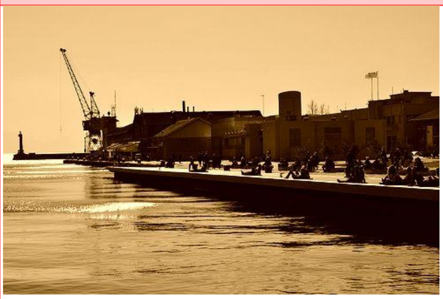
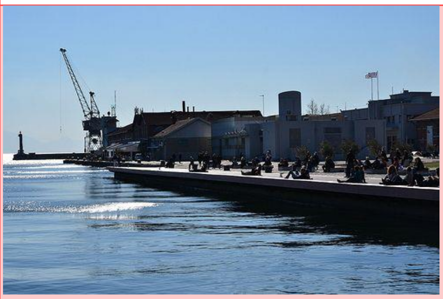
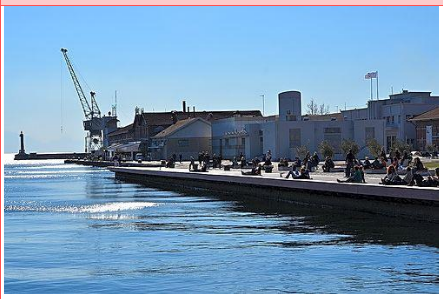
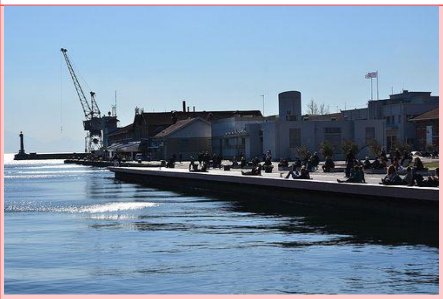
[33] http://cloudinary.com/documentation/image_transformations

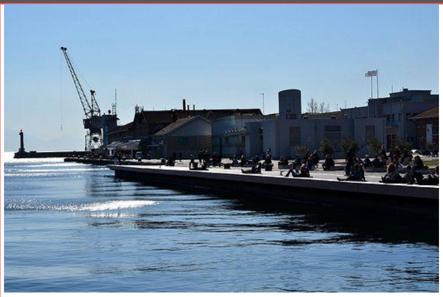
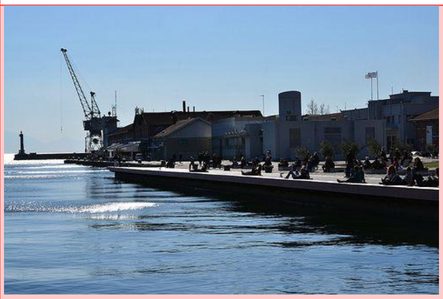
[34]All images are downsized to a maximum size of 250K pixels before image analysis is performed.

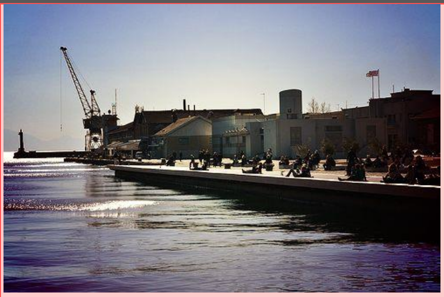
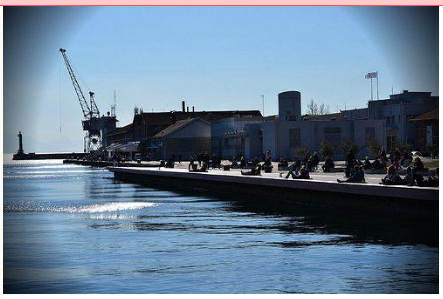
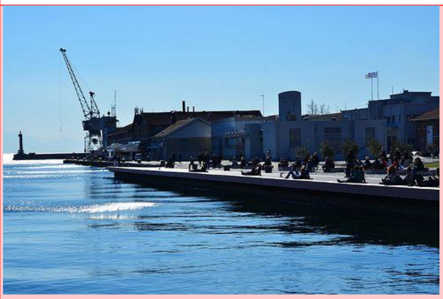
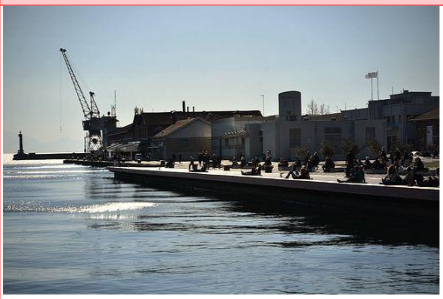
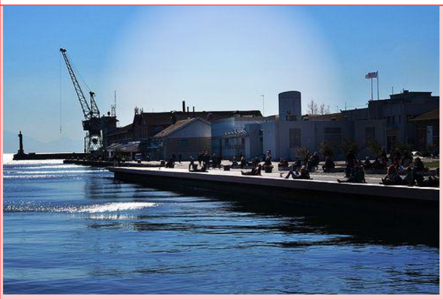
Table 4 – Category (1st column), name (2nd column) and description (3rd column) of the tested image transformations, along with their effects on an example image (4th column) and their impact on the computed R/G and G/B ratios (5th and 6th column). A hyphen (-) in the 5th, 6th column means that no R/G, G/B ratios were computed because no usable sky was detected.

Category	Name	Description	Example	R/G	G/B
-	original	The original image (size 500x331)		0.799	0.8517
Resizing	w:400	Resize width to 400 pixels while maintaining aspect ratio.		0.798	0.8510
	w:300	Resize width to 300 pixels while maintaining aspect ratio.		0.800	0.8516
	w:200	Resize width to 200 pixels while maintaining aspect ratio.		0.800	0.8509
	w:100	Resize width to 100 pixels while maintaining aspect ratio.		0.796	0.8479

Category	Name	Description	Example	R/G	G/B
Color level effects	brightness:20	Increase image brightness by 20% (Range: -99 to 100)		-	-
	brightness:50	Increase image brightness by 50% (Range: -99 to 100)		-	-
	brightness:80	Increase image brightness by 80% (Range: -99 to 100)		-	-
	saturation:20	Increase the image's color saturation by 20% (Range: -100 to 100)		0.766	0.8355
	saturation:50	Increase the image's color saturation by 50% (Range: -100 to 100)		0.718	0.8101

Category	Name	Description	Example	R/G	G/B
	saturation:80	Increase the image's color saturation by 80% (Range: -100 to 100)		0.665	0.7933
	sepia	Change the color scheme of the image to sepia		-	-
Automatic image improvement effects	gamma	Adjust the gamma level		0.799	0.8515
	improve	Automatically adjust image colors, contrast and lightness		0.726	0.8262
	auto_brightness	Automatically adjust brightness.		0.804	0.8545

Category	Name	Description	Example	R/G	G/B
	auto_contrast	Automatically adjust contrast.		0.801	0.8532
	auto_color	Automatically adjust color balance.		0.804	0.8494
Artistic filters	art:al_dente	The art:<filter> effects brighten highlights, intensify shadows, apply sepia-like filters, add vignetting, and more.		0.862	0.8993
	art:eucalyptus	The art:<filter> effects brighten highlights, intensify shadows, apply sepia-like filters, add vignetting, and more.		0.806	0.8990
	art:incognito	The art:<filter> effects brighten highlights, intensify shadows, apply sepia-like filters, add vignetting, and more.		-	-

Category	Name	Description	Example	R/G	G/B
	art:red_rock	The art:<filter> effects brighten highlights, intensify shadows, apply sepia-like filters, add vignetting, and more.		0.870	0.7847
	art:zorro	The art:<filter> effects brighten highlights, intensify shadows, apply sepia-like filters, add vignetting, and more.		0.779	0.8464
	art:primavera	The art:<filter> effects brighten highlights, intensify shadows, apply sepia-like filters, add vignetting, and more.		0.714	0.8375
	art:athena	The art:<filter> effects brighten highlights, intensify shadows, apply sepia-like filters, add vignetting, and more.		0.869	0.9090
	art:aurora	The art:<filter> effects brighten highlights, intensify shadows, apply sepia-like filters, add vignetting, and more.		0.737	0.8128

To automatically generate transformed versions of all the original images, we used the "eager" transformations approach[35] offered through Cloudinary's Java SDK to simultaneously apply all 24 transformations while uploading the images on Cloudinary. Upon completion of the upload and the transformations, Clourinary returns a response that contains the URLs of the transformed versions of the image. The response is parsed to extract the URLs and the transformed images are downloaded and forwarded to the image analysis module which calculates the corresponding R/G and G/B ratios, provided that a usable sky region is detected.

Having generated R/G and G/B ratios for all the original and transformed images, we computed the Pearson correlations between the ratios computed from the original images and those computed for each set of transformed images. The results are shown in Table 5 where we also report the percentage of images for which no ratios were computed after each transformation (due to the fact that no usable sky was detected after the transformation). We see that *both the R/G and the G/B ratios are very robust against image transformations* as a very high Pearson correlation is observed in most cases. In particular, we observe that the correlation is almost perfect for all the resizing transformations, while very high correlations (>0.9) are observed for all automatic image improvement effects. In the case of artistic image filters, again very high correlations are observed except for the *red_rock* filter that leads to low correlation between the R/G scores. Another notable exception is the increase of brightness where we see that a moderate (50%) and especially a large increase (80%) lead to much lower correlations. However, we also notice that when the *brightness:50* and *brightness:80* transformations are applied, a large percentage of the images are rendered unusable for air quality estimation (the fail to pass the tests included in the *heuristic* sky localization approach), thus limiting the negative impact of these transformations on the accuracy of the ratios. Another interesting observation is that two of the transformations (*sepia* and *art:incognito*) lead to 100% of the images rendered unusable for air quality estimation.

To provide a more detailed picture of the relationship between the ratios calculated from the original images and the ratios calculated from each type of transformed images, scatterplots of the original and transformed ratios are created for all 24 transformations and for both R/G and G/B ratio. *Figure 25* shows scatterplots (the least squares regression line is also computed) of the R/G (left) and G/B (right) ratios calculated from original (x axis) and transformed images (y axis), for three transformations (*w:200, brightness:20, improve*). The rest of the scatterplots are shown in Figure 43 of the Appendix.

Table 5 – Pearson correlations between the R/G (2nd column) and G/B (3rd column) ratios of the original images and their transformed versions. The last column reports the percentage of images with no usable sky after each transformation.

Transformation name	R/G Pearson Correlation	G/B Pearson Correlation	% no computation
w:400	0.99954	0.99982	4.6%
w:300	0.99950	0.99981	18.4%
w:200	0.99912	0.99931	33.3%
w:100	0.99867	0.99488	66.7%
brightness:20	0.81156	0.90529	26.4%
brightness:50	0.36323	0.22414	60.9%
brightness:80	0.08060	0.28798	82.8%
saturation:20	0.98422	0.97547	23.0%
saturation:50	0.92476	0.91021	41.4%

[35] http://cloudinary.com/documentation/upload_images#eager_transformations

saturation:80	0.84728	0.72826	59.8%
sepia	-	-	100.0%
gamma	0.99984	0.99991	2.3%
improve	0.90252	0.94631	29.9%
auto_brightness	0.98085	0.99449	1.1%
auto_contrast	0.97171	0.98146	6.9%
auto_color	0.97543	0.99317	14.9%
art:al_dente	0.79885	0.90745	18.4%
art:eucalyptus	0.94662	0.97612	20.7%
art:incognito	-	-	100.0%
art:red_rock	0.15462	0.74757	35.6%
art:zorro	0.99197	0.98892	17.2%
art:primavera	0.91048	0.88283	36.8%
art:athena	0.75699	0.93498	25.3%
art:aurora	0.93227	0.95800	26.4%
Average	0.83321	0.87893	35.5%

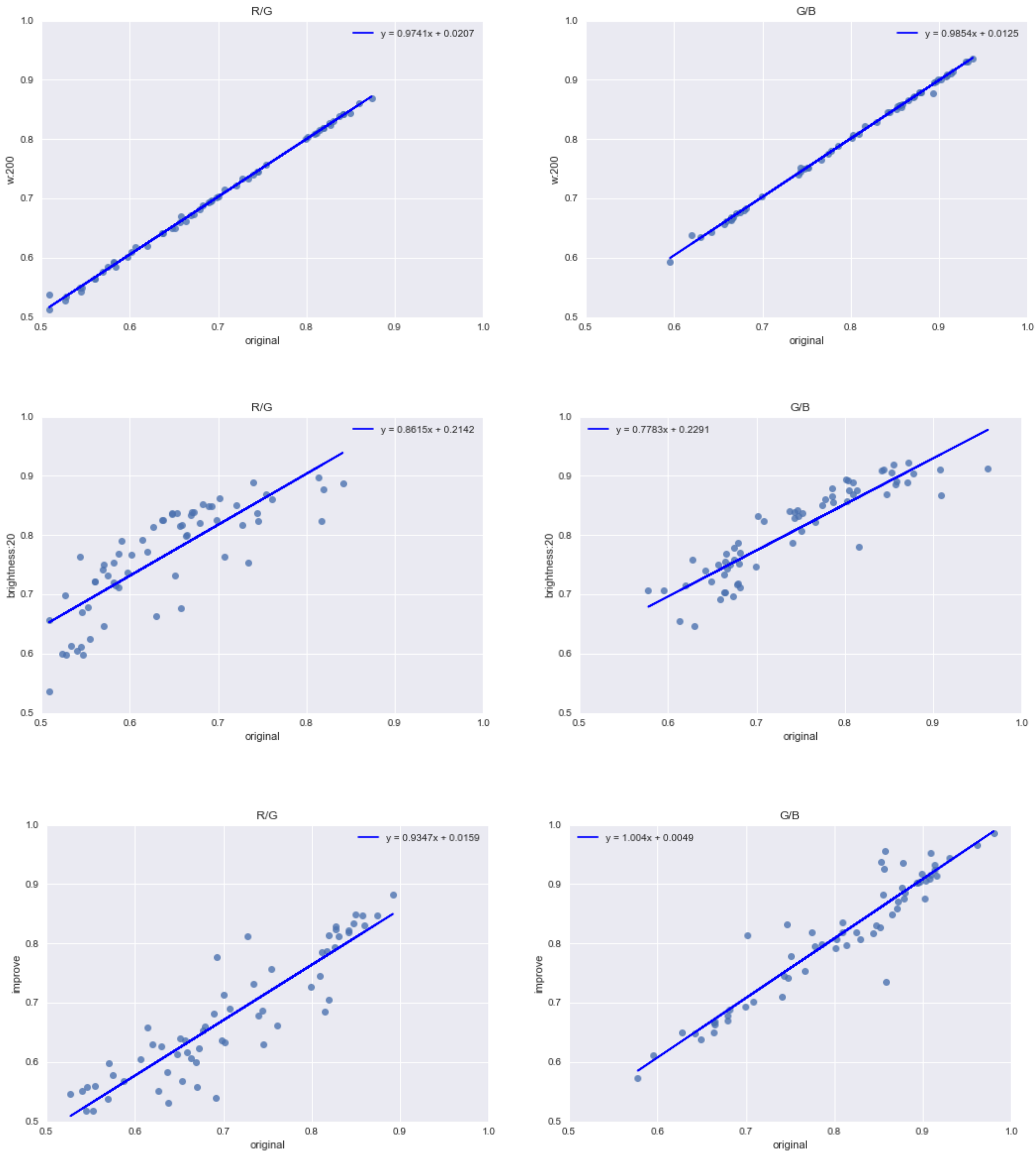


Figure 25 - Scatterplots of R/G (left) and G/B (right) scores calculated from original (x axis) and transformed images (y axis) for the w:200, brightness:20 and improve transformations.

In conclusion, we found that both ratios are generally very robust against most transformations. Nevertheless, we also found that there are few transformations that cause a relatively large distortion on the original ratios. Ideally, we would like to be able to either discard images that have undergone these transformations or to apply filters that revert the impact of these transformations. Unfortunately, however, the Flickr API does not provide information about the transformations applied on an image. Therefore, discarding transformed images or reverting the transformations is a highly non-trivial task that would require the development of specialized transformation detection algorithms. Given, the robustness of the ratios with respect to most transformations and the fact that our image collection framework also includes webcam images that are typically non-transformed, we decided to not pursue this research direction.

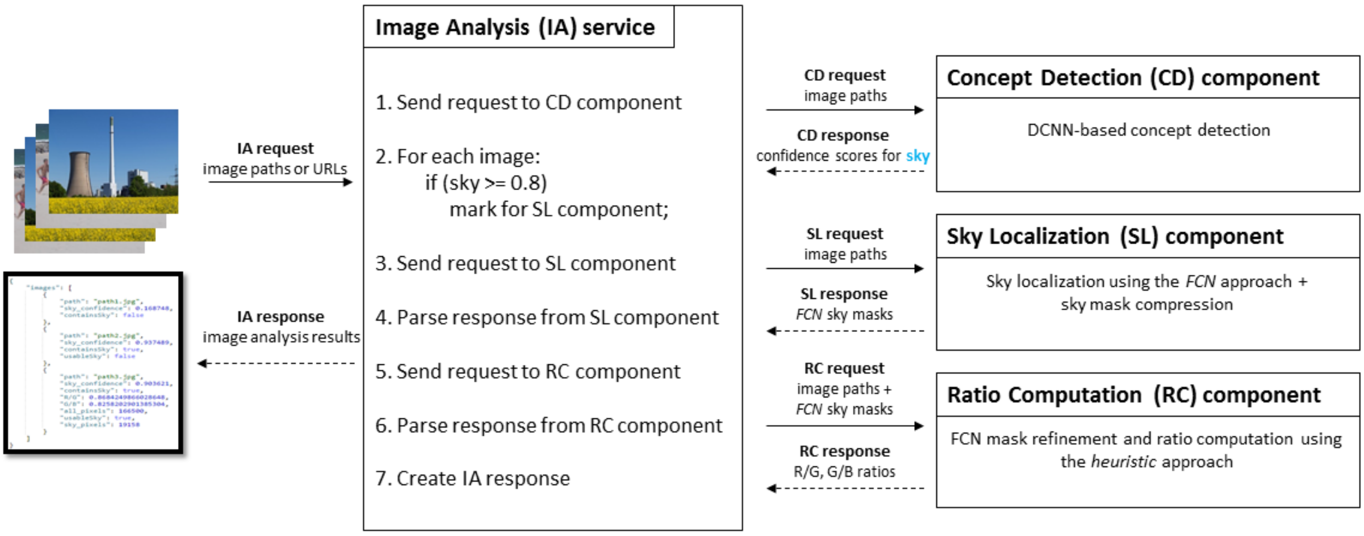
3.3 Image Analysis Service

3.3.1 Service Architecture

During the reporting period (M9-M18), all the image analysis operations required for the extraction of R/G and G/B ratios from sky-depicting images were incorporated into an *image analysis* (IA) Java web service that accepts image analysis requests, carries out all the required processing, and responds with the results of the image analysis. The overall architecture of the service is shown in Figure 26. The service accepts HTTP post requests that specify either a set of local (to the server that runs the service) paths that correspond to images already downloaded on the server through one of the image collectors (Flickr or webcams) or a set of image URLs[36]. In the latter case, all images are first downloaded (using a multi-threaded image download implementation).

The IA service uses internally three components, each one implementing a processing step of the image analysis pipeline, i.e. concept detection, sky localization and ratio computation. When an IA request is received, the IA service first sends a request to the *concept detection* (CD) component (step 1) which implements the concept detection framework described in D3.1. The CD component applies concept detection on each image of the request and returns a set of scores that represent the algorithm's confidence that the sky concept appears in each image. When a response is received by the CD component, the IA service parses it to check which images are the most likely to depict sky based on the confidence scores calculated by the CD component (step 2). A relatively high (0.8) threshold is used to lower the probability of sending non-sky-depicting images for subsequent analysis. At step 3, the IA service sends a request to the sky localization (SL) component which implements the FCN-based sky localization framework described in D3.1. This is a computationally heavy processing step that is carried out on the GPU of the IA server. The response of the SL component is the sky mask of each image of the request. To minimize the time required for sending the masks to the IA service, a compression algorithm is first applied to reduce the size of the masks. Then, the IA service receives the response from the SL component (step 4) and sends a request to the ratio computation (RC) component (step 5). The RC component takes the sky masks computed by the *FCN* approach as input, refines them by applying the *heuristic* approach on top of them (see section 3.1) and computes the R/G and G/B ratios of each image (in case all checks of the *heuristic* approach are successfully passed). Finally, the IA service parses the response of the RC component (step 6) and combines the results of all processing steps to synthesize the IA response (step 7).

[36] This option was given so that the IA service can serve requests for images that do not reside on the image analysis server.


IA request
image paths or URLs
IA response
image analysis results
Image Analysis (IA) service
1. Send request to CD component
2. For each image:
if (sky >= 0.8)
mark for SL component;
3. Send request to SL component
4. Parse response from SL component
5. Send request to RC component
6. Parse response from RC component
7. Create IA response
CD request
image paths
CD response
confidence scores for sky
SL request
image paths
SL response
FCN sky masks
RC request
image paths +
FCN sky masks
RC response
R/G, G/B ratios
Concept Detection (CD) component
DCNN-based concept detection
Sky Localization (SL) component
Sky localization using the FCN approach +
sky mask compression
Ratio Computation (RC) component
FCN mask refinement and ratio computation using
the heuristic approach

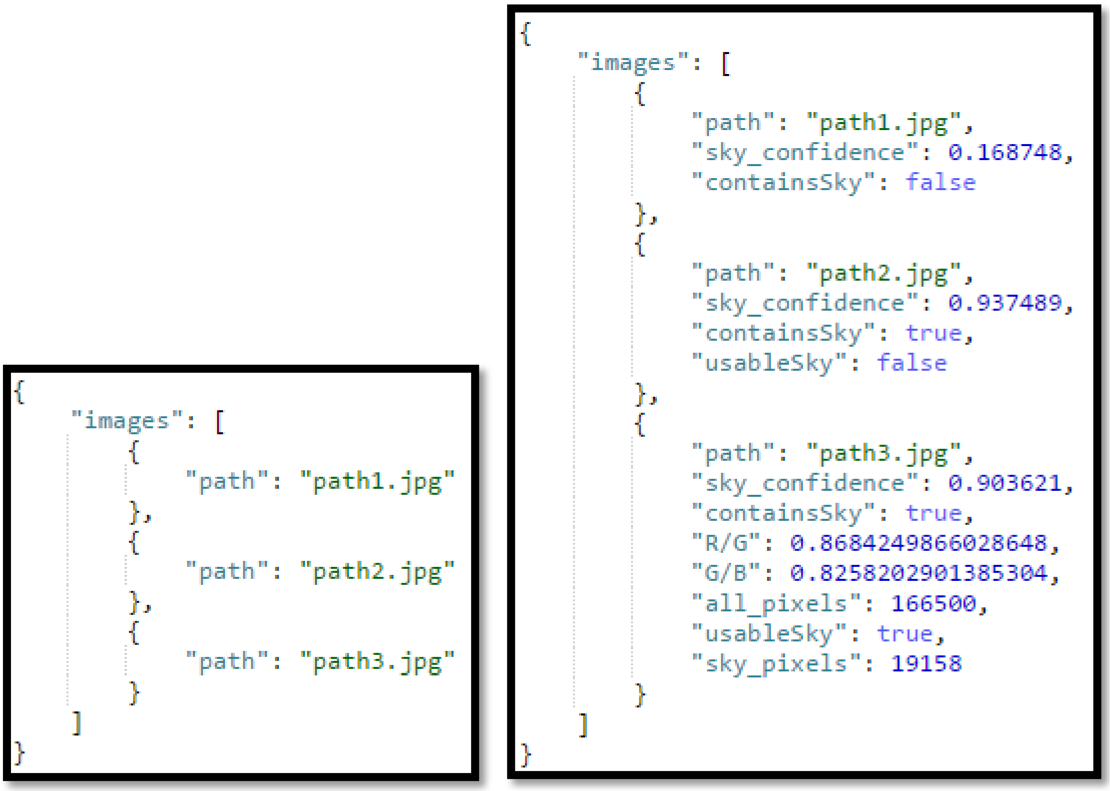
```
{
    "images": [
        {
            "path": "path1.jpg",
            "sky_confidence": 0.168748,
            "containsSky": false
        },
        {
            "path": "path2.jpg",
            "sky_confidence": 0.937489,
            "containsSky": true,
            "usableSky": false
        },
        {
            "path": "path3.jpg",
            "sky_confidence": 0.903621,
            "containsSky": true,
            "R/G": 0.8684249866028648,
            "G/B": 0.8258202901385304,
            "all_pixels": 166500,
            "usableSky": true,
            "sky_pixels": 19158
        }
    ]
}
```

Figure 26 – Architecture of the image analysis service

Figure 27 shows an example (POST) request to the IA service[37] for three images and the corresponding response. We notice that the response contains different information for each image, depending on the results of the analysis. Since all images undergo concept detection, the confidence score for the concept sky is added to the response for all images ("sky_confidence" field) as well as a boolean flag ("containsSky" field) that is true if the confidence score for sky is ≥ 0.8, false otherwise. In case no sky is detected, the response does not contain additional information about the image (this is the case for image "path1.jpg"). Otherwise, an additional boolean flag is added to the image ("usableSky" field) that is true if a usable sky region was detected in the image by the *FCN+heuristic* approach (this is the case for image "path3.jpg"), false otherwise (this is the case for image "path2.jpg"). Finally, in case a usable sky region is detected, the R/G and G/B ratios are added in the response along with the total number of pixels in the image ("all_pixels" field) and the number of pixels corresponding to usable sky ("sky_pixels" field)[38].

[37] http://[SERVICE-HOST]/ImageAnalysisService-v1/post/

[38] The addition of the "all_pixels" and "sky_pixels" fields was requested by DUTH to help with the tuning of the air quality estimation models.



```
{
    "images": [
        {
            "path": "path1.jpg"
        },
        {
            "path": "path2.jpg"
        },
        {
            "path": "path3.jpg"
        }
    ]
}
```

```
{
    "images": [
        {
            "path": "path1.jpg",
            "sky_confidence": 0.168748,
            "containsSky": false
        },
        {
            "path": "path2.jpg",
            "sky_confidence": 0.937489,
            "containsSky": true,
            "usableSky": false
        },
        {
            "path": "path3.jpg",
            "sky_confidence": 0.903621,
            "containsSky": true,
            "R/G": 0.8684249866028648,
            "G/B": 0.8258202901385304,
            "all_pixels": 166500,
            "usableSky": true,
            "sky_pixels": 19158
        }
    ]
}
```

Figure 27 – The body of an example (POST) request to the IA service (left) and the corresponding JSON response (right)r.

3.3.2 Image Analysis Service Efficiency

Besides the accuracy and the robustness of the image analysis service that were studied in sections 3.1 and 3.2, we also wanted to study its time efficiency and try to determine the maximum total number of images that can be processed by the service per hour for a given hardware configuration[39]. To this end, a set of 1,000 randomly selected Flickr images were submitted to the image analysis service and the time taken on each of the three processing steps as well as the total response time were recorded.

Table 6 – Time efficiency characteristics of the IA service presents the results obtained when the default 0.8 confidence threshold is used (2nd column) as well as the results obtained when a 0 threshold is used so that all 1,000 images pass through all the processing steps (3rd column). Looking at the 3rd column, we notice that ratio computation is the most time-consuming step with 0.28 sec per image, followed by the sky localization step with 0.25 sec per images and the concept detection step with 0.15 sec per image. In practice, however, sky is not detected in all images (as shown in the next section, about 40% of the images pass the 0.8 threshold) and, as a result, sky localization and ratio computation are performed only in a subset of the images. This more realistic setting is shown in the 2nd column where we observe that, on average, most time is consumed in concept detection (0.15 sec) followed by ratio computation (0.11 sec) and sky localization (0.10 sec). As shown in the last row of Table 6 – Time efficiency characteristics of the IA service, when the default 0.8 confidence threshold is used the IA service requires only 0.39 sec per image on average. Based on these results, we conclude that with this hardware configuration, around 9,000 images per hour can be processed by the image analysis service. Finally, we should note that use of a latest generation GPU (e.g. NVidia

[39] The image analysis service was set-up on a server running Linux (Ubuntu), equipped with 16Gb of RAM, an Intel i7-3770K processor and an NVidia GeForce GTX 1070 GPU that is used by the sky localization service.

GeForce GTX 1070) is critical for achieving this level of efficiency, as without the use of the GPU the sky localization service takes over 4 sec per image.

Table 6 – Time efficiency characteristics of the IA service.

Processing step	Avg. time per image (sec) Sky ≥ 0.8	Avg. time per image (sec) Sky ≥ 0
CD component	0.15	0.15
SL component	0.10	0.25
RC component	0.11	0.28
IA service	0.39	0.71

3.4 Image Analysis Statistics

The IA service has been running since 1/3/2017, analyzing all the images that we collect with the updated version of the Flickr collector (which was deployed first) and the AMOS and webcams.travel webcam image collectors (which were deployed later). From 1/3/2017 until 15/05/2017 998,966 images have been analyzed from all sources and the results of the analysis have been stored in MongoDB along with basic metadata about each image such as location (latitude, longitude), taken timestamp and source. In this section, we present several statistics that we computed from this data.

Table 7 shows the numbers of sky-depicting images collected from each source (i.e. images for which the concept detector outputs a confidence score >0.8) as well as the number images with usable sky (i.e. the subset of sky-depicting images that meet the criteria defined by the *FCN+heuristic* sky localization approach). We notice that, on average, 18.46% of the collected images are sky-depicting, with the largest percentage observed in images collected from webcams.travel. We also notice that 9.25% of all the collected images contain a sky region that can be used for air quality estimation. Again, the largest percentage is observed in images from webcams.travel.

Table 7 – Numbers and percentages of sky-depicting and usable sky images collected from each source.

Source	Total	Sky	Usable sky
Flickr	424,785	67,642 (15.90%)	43,086 (10.14%)
AMOS webcams	525,998	103,587 (19.69%)	42,318 (08.05%)
webcams.travel	48,183	13,269 (27.53%)	7,010 (14.55%)
Total	**998,966**	**184,498 (18.46%)**	**92,414 (09.25%)**

Figure 28 shows the number of usable sky images collected per day in Europe. We see that after the integration of all image sources (beginning of May), the number of usable sky images that we collect increases to 1800 images per day across the whole Europe. Figure 29 shows the daily number of usable images in the two countries of the pilots, Germany and Norway. We notice that the average daily number of usable sky images in May is ≈106 for Germany and ≈271 for Norway. Thus, in both cases, we significantly surpass the lower limit of 50 measurements per country that are needed by the data fusion module, as described in D4.1.

We also wanted to study the differences between the average daily values of the R/G and G/B ratios. Figure 30 plots the average daily values of the R/G and G/B ratios, while Figure 31 presents a scatterplot of the daily averages. We observe that the two ratios are highly correlated, with G/B being consistently higher than R/G.

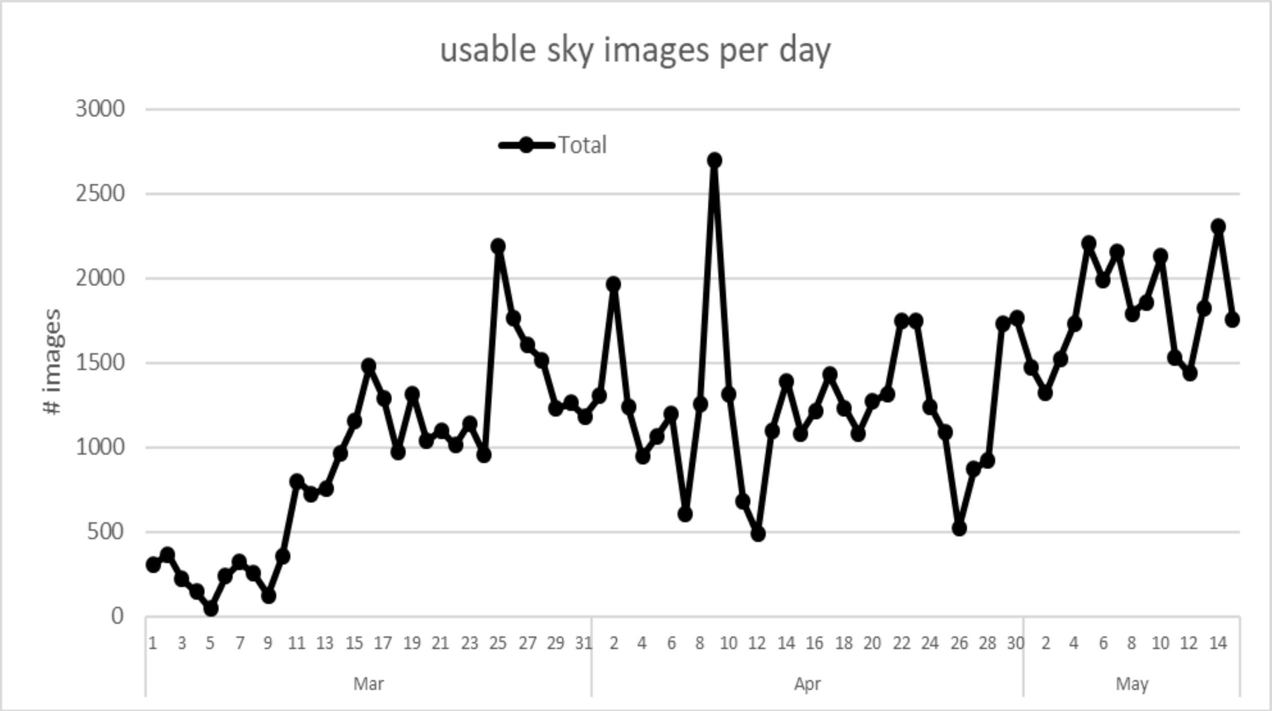


Figure 28 – Daily number of usable sky images in Europe (the increasing trend is due to the addition of webcam images).

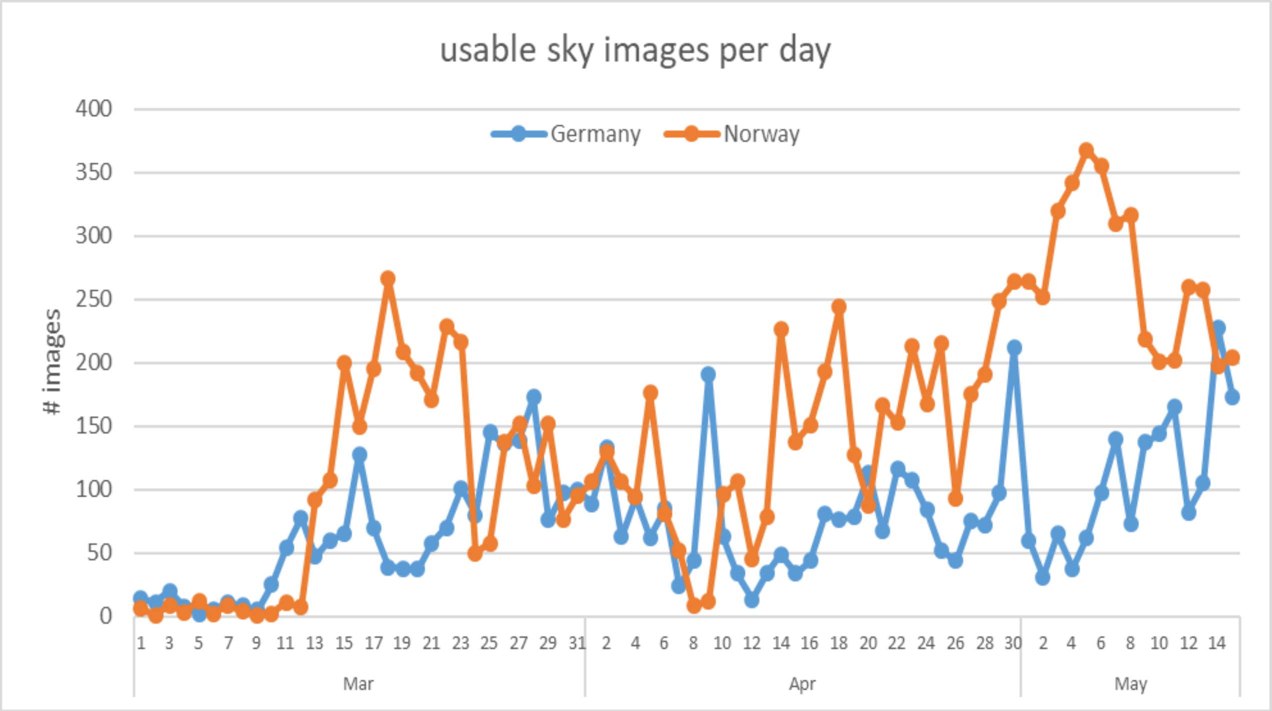


Figure 29 – Daily number of usable sky images in Germany and Norway.

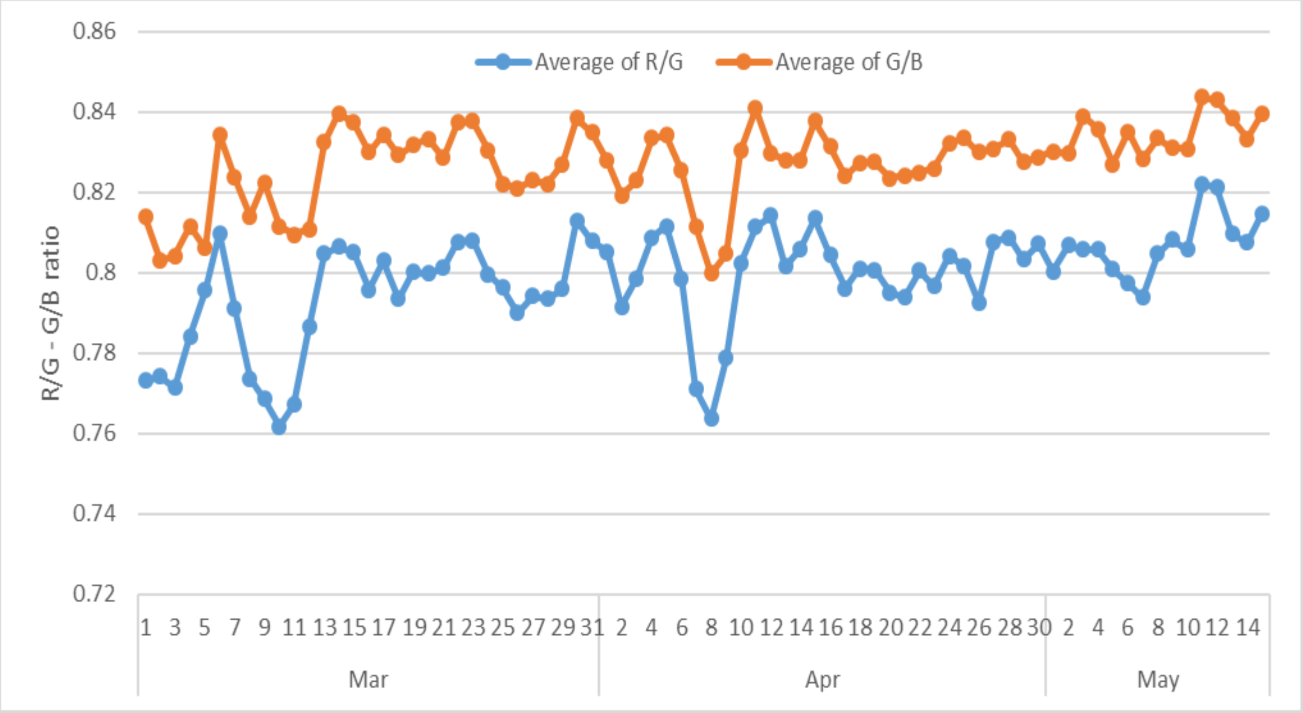


Figure 30 – Daily R/G vs G/B averages.

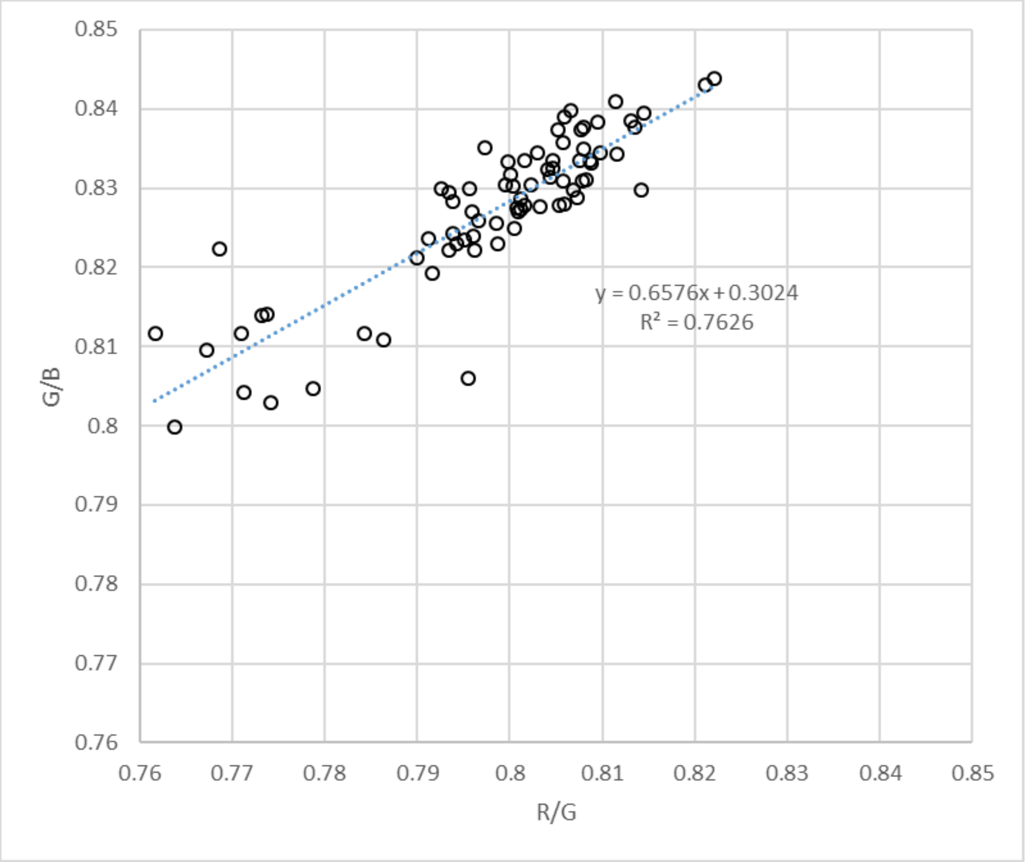


Figure 31 – Scatterplot of average daily R/G and G/B ratios.

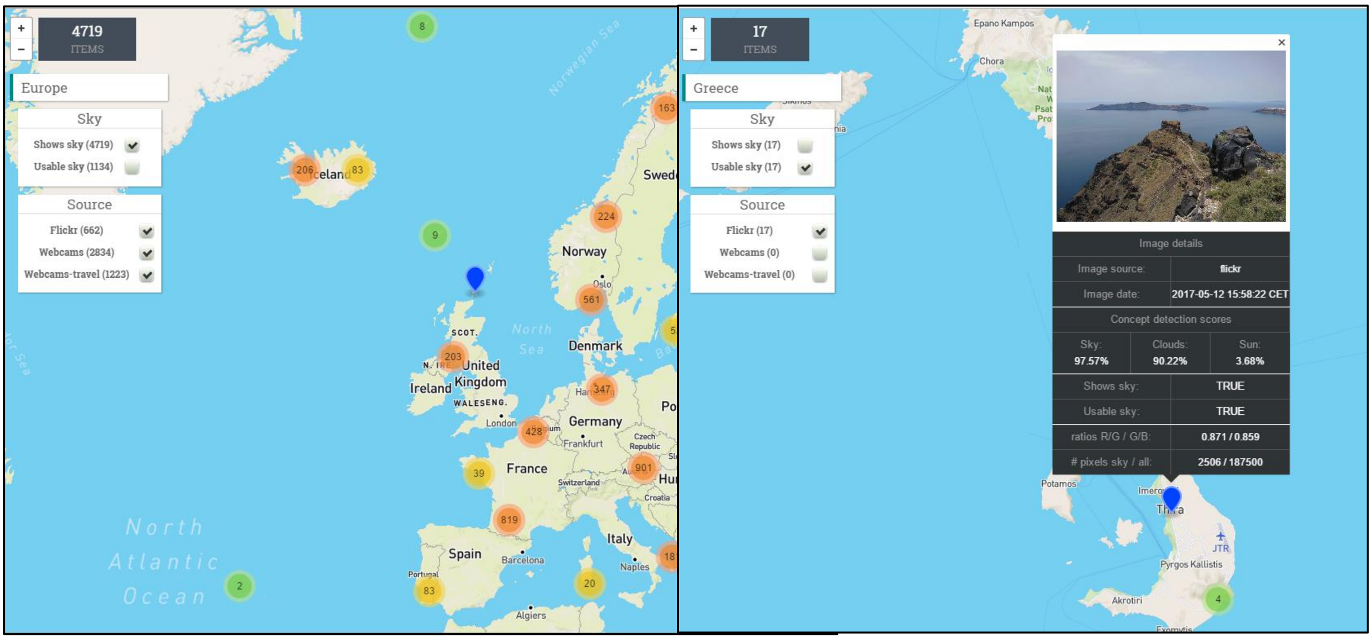
3.5 Visualization of Image Collection & Analysis

To facilitate easy inspection of the collected images, their geographical distribution, and the output of the image analysis module for each image, we built a web application[40] that plots the collected images (within the last 24 hours) as markers on a map, in a similar manner as we do for visualizing air quality measurements from official sources. Figure 32 shows two screenshots of the application. We see that the application offers three ways of filtering the results: a) based on country (initially results from all European countries are shown), b) based on image source (Flickr/Webcams) and c) based on the results of the image analysis, i.e. the user can ask only sky-depicting or only usable sky images to be shown. To reduce the clutter caused by the high number of markers, marker clustering is implemented. When an individual marker is clicked, a pop-up window opens that shows all the details of the image (see left side of Figure 32). On the top of the pop-up window there is thumbnail of the image that links to the original image source. We can also see the source of the image, the exact date and time it was taken, the concept detection scores and whether a usable sky was detected or not. In case a usable sky was detected, the calculated R/G and G/B ratios are also displayed, as well as the number of sky pixels and the number of total pixels in the image.

The front-end of the application is implemented in HTML/CSS and JavaScript, while on the back-end there is REST a web service that accepts requests of the form:

- http://[SERVICE-HOST]/crawl/images?place=greece&hasSky=true&usableSky=true&source=flickr,webcams-travel

and returns a JSON response of the form shown in Figure 33. The web service basically transforms the requests in MongoDB queries that it executes and then formats appropriately as a JSON response. Note that a geospatial MongoDB index is used to efficiently perform geographical queries (when results from a specific country are requested). In particular, a bounding box is defined for each country and all images whose coordinates lie inside the corresponding bounding box are returned.



4719 ITEMS

Europe

Sky
- Shows sky (4719) ☑
- Usable sky (1134) □

Source
- Flickr (662) ☑
- Webcams (2834) ☑
- Webcams-travel (1223) ☑

17 ITEMS

Greece

Sky
- Shows sky (17) □
- Usable sky (17) ☑

Source
- Flickr (17) ☑
- Webcams (0) □
- Webcams-travel (0) □

Image details

Image source:	flickr
Image date:	2017-05-12 15:58:22 CET

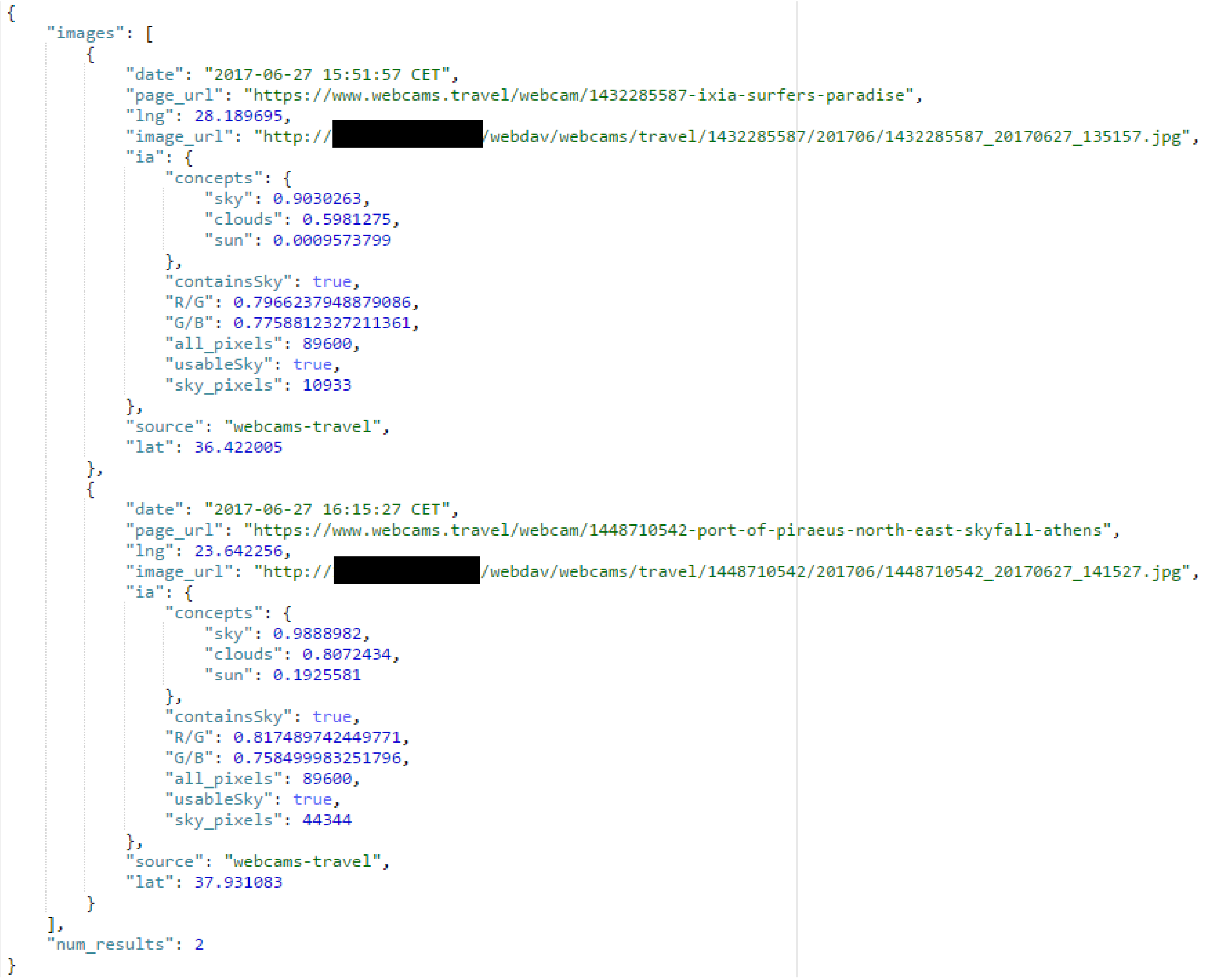
Concept detection scores

Sky:	Clouds:	Sun:
97.57%	90.22%	3.68%

Shows sky:	TRUE
Usable sky:	TRUE
ratios R/G / G/B:	0.871 / 0.859
# pixels sky / all:	2506 / 187500

Figure 32 – Screenshots of the image collection and analysis visualization web application.

[40] http://hackair-mklab.iti.gr/map



```
{
    "images": [
        {
            "date": "2017-06-27 15:51:57 CET",
            "page_url": "https://www.webcams.travel/webcam/1432285587-ixia-surfers-paradise",
            "lng": 28.189695,
            "image_url": "http://[redacted]/webdav/webcams/travel/1432285587/201706/1432285587_20170627_135157.jpg",
            "ia": {
                "concepts": {
                    "sky": 0.9030263,
                    "clouds": 0.5981275,
                    "sun": 0.0009573799
                },
                "containsSky": true,
                "R/G": 0.7966237948879086,
                "G/B": 0.7758812327211361,
                "all_pixels": 89600,
                "usableSky": true,
                "sky_pixels": 10933
            },
            "source": "webcams-travel",
            "lat": 36.422005
        },
        {
            "date": "2017-06-27 16:15:27 CET",
            "page_url": "https://www.webcams.travel/webcam/1448710542-port-of-piraeus-north-east-skyfall-athens",
            "lng": 23.642256,
            "image_url": "http://[redacted]/webdav/webcams/travel/1448710542/201706/1448710542_20170627_141527.jpg",
            "ia": {
                "concepts": {
                    "sky": 0.9888982,
                    "clouds": 0.8072434,
                    "sun": 0.1925581
                },
                "containsSky": true,
                "R/G": 0.817489742449771,
                "G/B": 0.758499983251796,
                "all_pixels": 89600,
                "usableSky": true,
                "sky_pixels": 44344
            },
            "source": "webcams-travel",
            "lat": 37.931083
        }
    ],
    "num_results": 2
}
```

Figure 33 – Example response of the REST web service that supports the image collection and visualization application.

4 Twitter-based air quality estimation

This section presents an experimental line of work that investigates the possibility of making air quality estimations based on the analysis of content posted on Twitter. Although generating estimations of this type was not foreseen in the initial planning of the project, we were motivated to explore air quality estimation approaches that are complementary to the image-based approach, which is considered as the primary approach based on opportunistic social media and web content sensing. This was due to the fact that a) there are several cases where images cannot lead to any estimation (e.g. cloudy conditions during the whole day around a location), b) having a second option for performing air quality estimations could potentially benefit the overall reliability of the platform outputs. Some preliminary evidence on the feasibility of estimating air quality based on analysis of microblogging content has been provided by a few recent works, e.g. (Mei et al., 2014), (Jiang et al., 2015), (Wang et al., 2015), (Tao et al., 2016), that studied the correlation of air quality in China with the content posted in Sina Weibo[41] (often called the Chinese Twitter). Here, we report some promising results that we obtained in initial experiments that we conducted on this topic. However, we highlight that image-based estimations remain the primary type of air-quality estimations within the hackAIR platform and Twitter-based estimations might be used as a complementary and experimental approach that needs considerable further investigation and testing before being deployed in the platform.

4.1 Methodology

Our efforts are grounded on the assumption that the tweets posted at a specific location can provide information about (exhibit statistical correlations with) the current air quality levels at this location. Based on this assumption, we investigated the possibility of using machine learning methods to build coarse reliable city-level air quality estimation models based on Twitter posts. In particular, we focus on estimating current PM10 and PM2.5 levels, for cities without official air quality measurement stations. To accomplish this task, we develop a transfer learning framework where prediction models are trained using data from nearby cities (for which both Twitter data and official air quality measurements are available) and then transferred to the cities of interest (for which only Twitter data is supposedly available).

More formally, let $D = \{D_{c_1}, \dots, D_{c_M}\}$, denote a set of M city-specific datasets $D_{c_j} = \{(\boldsymbol{x}_{c_j}^1, y_{c_j}^1), \dots, (\boldsymbol{x}_{c_j}^N, y_{c_j}^N)\}$ where $\boldsymbol{x}_{c_j}^i$ is a vector of predictive variables (or features), each one providing a potentially useful (in the sense that it is correlated with the dependent variable) summary statistic of the Tweets posted around city c_j during a fixed length time period (more details about the features used are provided in Section 4.3), and $y_{c_j}^i$ be the average value of the pollutant at the same time period. Our goal is to use one or more of the datasets in D in order to learn a function $h: \boldsymbol{X} \rightarrow Y$, that given a feature vector $\boldsymbol{x}_{c_q}$ calculated from a city c_q without measurements stations will be able to accurately estimate the unknown pollutant value y_{c_q}.

4.2 Data Collection

To carry out experiments according to the previously defined prediction task, we need to collect Twitter data (to facilitate the creation of feature vectors $\boldsymbol{x}$) and the corresponding pollutant values (the dependent variable y) for a set of (relatively) closely located cities. To simplify the task, we focus only on the English language and select five cities in the UK (London, Liverpool, Manchester, Birmingham and Leeds). To simulate the previously described transfer learning scenario, each city is in turn considered as test city (hypothetically without official air quality measurements) and all the remaining cities are used for training. In this set of preliminary experiments, we collected data for a time

[41] https://en.wikipedia.org/wiki/Sina_Weibo

period of approximately 3.5 months[42] (1/2/2017 until 16/5/2017). The details of data collection are provided in Sections 4.2.1 (Twitter data collection) and 4.2.2 (ground station measurements collection).

4.2.1 Twitter Data Collection

Twitter data is collected using the Twitter Streaming API[43]. The Streaming API offers real-time access to a sample of the public data that are posted on Twitter. Once applications establish a connection to the streaming endpoint, they are granted access to a feed of tweets. Twitter streaming volume is not constant. Throughout the course of a 24-hour period, there is a natural ebb and flow to the number of tweets delivered per second. Twitter documentation does not mention the fraction of the total number of tweets that are retrieved in every time slot.

There are two main methods to query the Streaming API. The first method (*location-based*[44]) allows the retrieval of geotagged tweets around an area of interest by specifying bounding box coordinates. The second method (*keyword-based*[45]) retrieves both geotagged and non-geotagged tweets based on a specified list of keywords. Ideally, we would like to combine the two methods in order to retrieve tweets that contain air quality-related keywords and, at the same time, are located in the areas (cities) of interest. Unfortunately, the API does not allow the combination of the two methods. Therefore, the are two alternatives: a) retrieve tweets using the location-based method and then perform keyword-based filtering, b) retrieve tweets using the keyword-based method and then perform location-based filtering.

Experimenting with the first approach, we noticed that the number of tweets that we retrieve for the locations of interest is very low (approximately 150 tweets per day). This is justified by the fact that only a small fraction of the total number of tweets (<5%) are geolocated[46]. Thus, we focused on the second approach. In particular, we use the keyword-based approach to track a list[47] of 120 English keywords related to air pollution (composed collaboratively with other project partners) and store all the returned tweets[48].

Keyword-based retrieval implies that most of the returned tweets will not be geolocated. Thus, to take advantage of the much higher volume of non-geolocated tweets, we took the assumption that, in most cases, the tweet location will match the location that the user has specified in his/her Twitter account. To this end, the LM-based multimedia geotagging method presented in Section 2.1.2 was employed to assign exact geographical coordinates to the textual description of the user's location. Using this approach, we can recover the city corresponding to the Twitter location of significantly more users than we would be able to recover with a trivial text matching approach because location descriptions referring to well-known districts or boroughs can also be assigned an exact location.

To evaluate the accuracy of the inferred locations, we exploited the *place* field[49]. This field is assigned to tweets whenever users want to incorporate non-precise location information in their tweets. In that case, Twitter provides recommendations about nearby locations using the device's IP or exact geographical coordinates (if available) and users can choose their preferred location. Although this location might not always correspond to the actual tweet location (as users can freely choose any location), we expect that it will be very strongly correlated with the actual

[42] Note that data collection will be continued with the aim to expand the dataset to a period that spans at least one year in order to avoid seasonal effects and achieve better time invariance.

[43] https://dev.twitter.com/streaming/overview

[44] https://dev.twitter.com/streaming/overview/request-parameters#locations

[45] https://dev.twitter.com/streaming/overview/request-parameters#track

[46]http://firstmonday.org/article/view/4366/3654

[47] https://docs.google.com/spreadsheets/d/1FIumUHdfHUZbArJ-CnpuwPu7-riMeMijYsHJZtzTQ-Q/edit#gid=0

[48] We also use the *language* parameter of the API to ensure that only tweets written in English will be returned.

[49] https://dev.twitter.com/overview/api/places

tweet location in most cases. Moreover, the place field is available for a much larger number of tweets compared to those with precise location and was therefore considered more suitable for evaluating the accuracy of the inferred locations. As we can see in Figure 34, the top 10 locations (according to the place field) of the tweets of which the inferred location is London, are indeed referring to London. Although a small number of tweets are assigned a wrong location using the above approach, we found that this is preferable compared to basing our estimations on a much smaller number of accurately geolocated tweets.

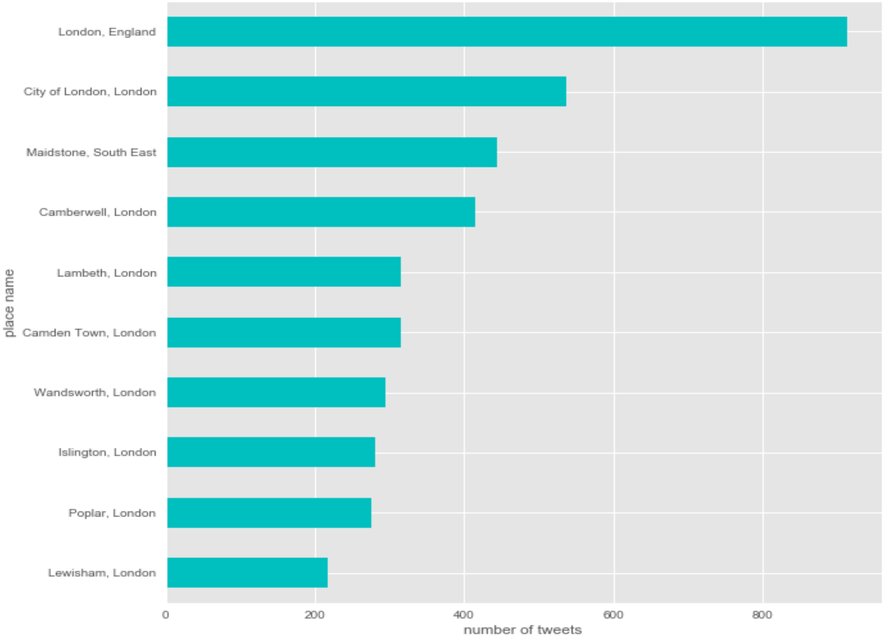


Figure 34– Top 10 place fields from tweets assigned to London

Figure 35 shows the number of tweets that are retrieved daily from each city using the previously described approach for a period of 3.5 months, while Table 8 reports the average daily number of tweets per city. As expected, there is a strong correlation between city populations and numbers of retrieved tweets.

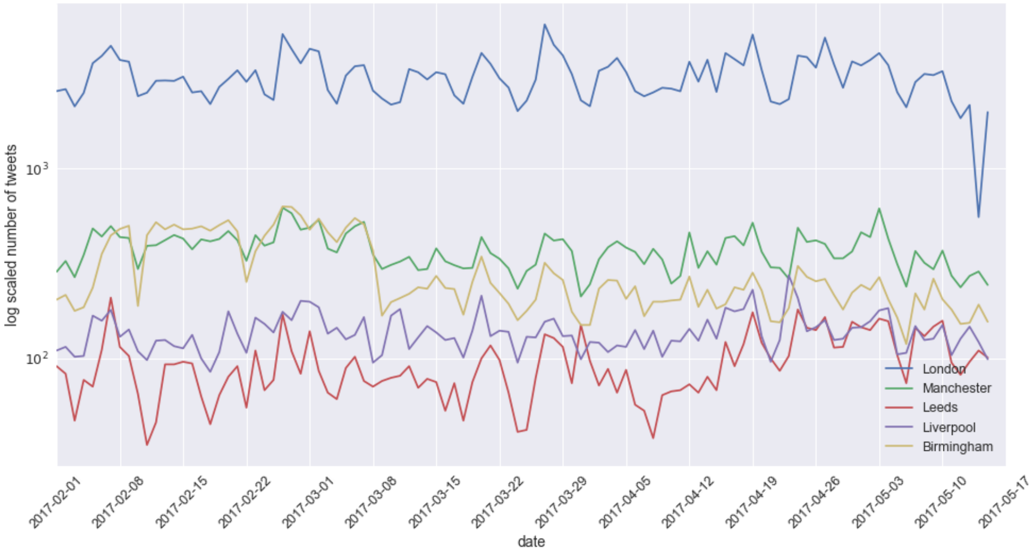


Figure 35 – Number of tweets retrieved daily for each city.

Table 8 – Daily average number of tweets retrieved per city.

City	Avg. number of daily tweets retrieved
London	3,240
Manchester	385
Liverpool	140
Birmingham	303
Leeds	105

4.2.2 Collection of Ground Station Measurements

In order to collect ground truth PM10 and PM2.5 measurements for the cities of interest we use the OpenAQ API (already described in Section 2.3). In particular, we use the https://api.openaq.org/v1/measurements endpoint[50] to retrieve hourly historical measurements from all stations located in each city of interest by appropriately setting the *city*, *date_from* and *date_to* parameters. Table 9 shows the number of stations from which measurements of each pollutant are available for each city. To calculate a single hourly measurement for each city, we average the measurements of the respective stations, without considering outlier values (values that deviate more than three standard deviations from the corresponding city mean[51]) which are likely a result of sensor or API failures. Figure 36 shows the daily means of PM10 and PM2.5 in the city of London[52]. We observe that the values of the two pollutants exhibit very similar trends (the Pearson correlation is 0.96) and the situation is similar in all cities. In the following sections, we report results only for PM10 since very similar results are obtained for PM2.5.

Table 9 - Numbers of air pollution stations per city

Location	Ground stations with PM10	Ground stations with PM2.5
London	10	9
Manchester	2	3
Liverpool	1	2
Birmingham	2	2
Leeds	2	2

In addition to the correlations between PM10 and PM2.5, we also measured the correlations between daily PM10 values in different cities. Figure 37 shows the daily PM10 means of London, Manchester and Liverpool. As expected, similar air pollution trends are observed (especially for the cities of Manchester and Liverpool which lie closer to each other), suggesting that it is reasonable to attempt model transfer between nearby cities. This can be seen more clearly in Figure 38 which shows the scatterplot of the distances and corresponding PM10 correlations of all distinct city pairs. Clearly, the smaller the distance between two cities, the higher the correlation between their PM10 values.

[50] https://docs.openaq.org/#api-Measurements

[51] Indicatively, in a period of 3.5 months 33 measurements were rejected in London.

[52] The discontinuities in the plot are due to the fact that the OpenAQ API does not provide measurements for all time periods.

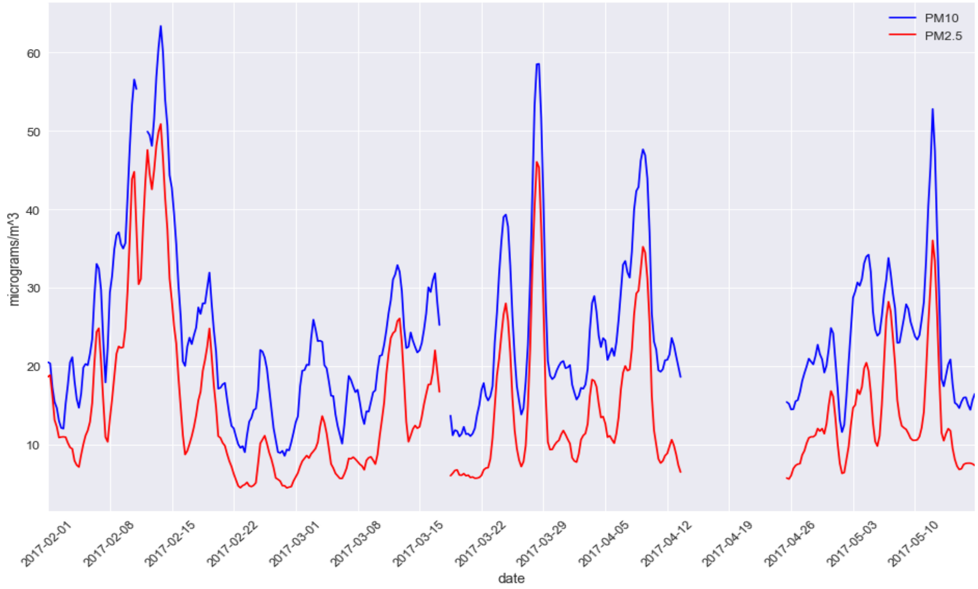


Figure 36 – Daily PM10 and PM2.5 means in London.

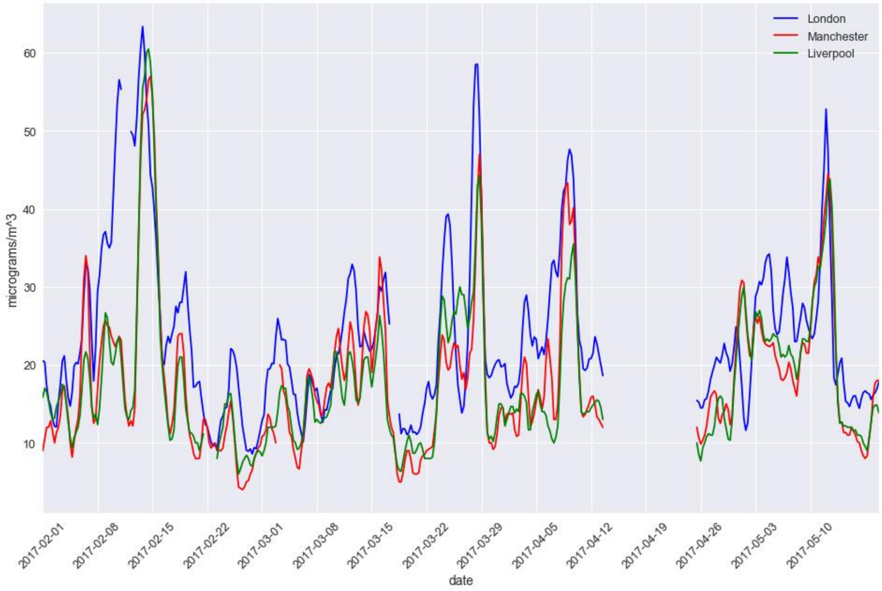


Figure 37 - Daily PM10 means in London, Manchester and Liverpool.

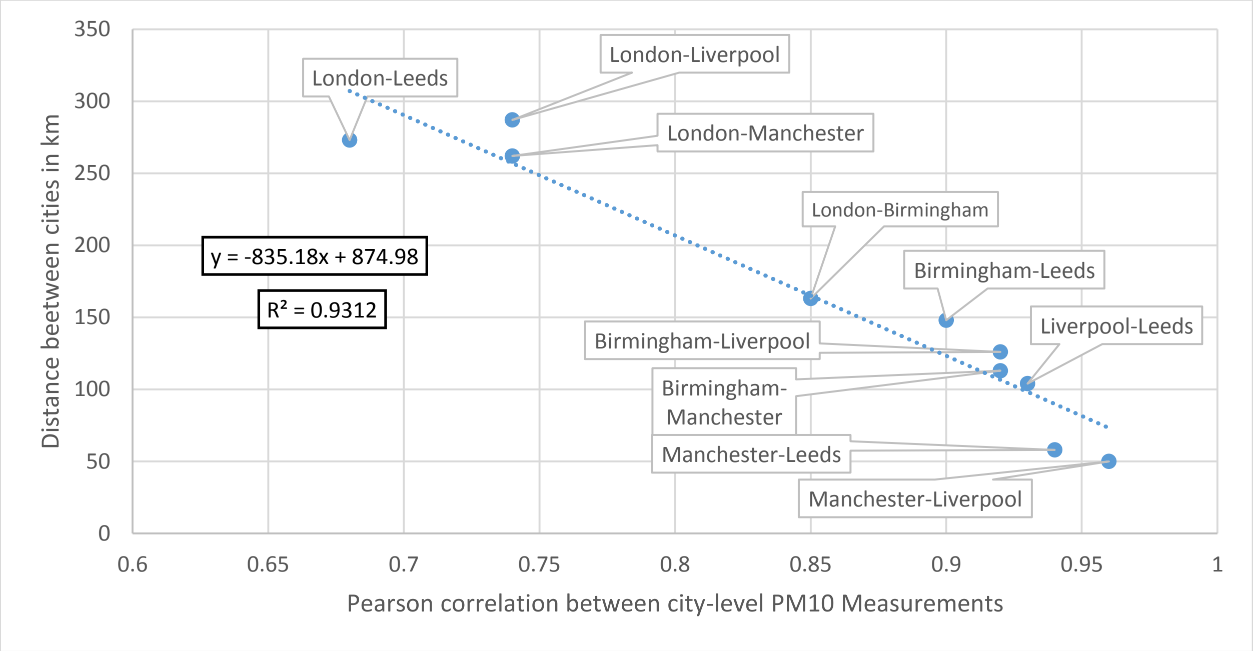


Figure 38 – Scatterplot of distances and PM10 correlations between cities in the UK.

In most of our experiments, we adopt a classification instead of a regression formulation of the air quality prediction task, i.e. we predict pollution classes instead of exact PM10/PM2.5 values, because in preliminary experiments we found that such a formulation leads to better performance. As a result, the continuous dependent variable y (corresponding to exact PM10/PM2.5 values) is transformed to a discrete variable (where each level corresponds to a different pollution class) according to Table 3.

4.3 Predictive Feature Generation

In this section, we describe the various predictive features that we generated from the collected Twitter data. Our goal is to generate features that provide a descriptive summary of the tweets posted around a specific city during a fixed period of time (timestep), and at the same time exhibit strong correlations with the target variable (PM10/PM2.5). For our experiments, each day is divided into four 6-hour timesteps and features are computed over all tweets posted during the respective timestep to generate the feature vector [53]. In total, we generated 54 different features which are summarized in Table 10.

Table 10 - Predictive features.

No	Feature name	Description
1	smog	Number of tweets that contain the word "smog"
2	emissions	Number of tweets that contain the word "emissions"
3	airpollution	Number of tweets that contain the phrase "air pollution"
4	pollutionalert	Number of tweets that contain the phrase "pollution alert"
5	tweetsNo	Total number of tweets
6	AQ_D	Estimated number of tweets related to air quality (AQ_Discussion)

[53] The corresponding target variable y is computed by taking the mean of the hourly pollutant values.

7	AQ_S	Estimated number of tweets with information on current air quality (AQ_Sense)
8	low	Estimated number of AQ_S tweets referring to low pollution (AQ_Severity)
9	high	Estimated number of AQ_S tweets referring to high pollution (AQ_Severity)
10	active_day	1 if timestep is in an active day (Mon, Tue, Wed, Thur), 0 otherwise
11	active_hour	1 if timestep contains active hours (9:00 - 24:00), 0 otherwise
13	rolling_mean_[FEATURE]_[N]	The mean value of feature [FEATURE] for the [N] previous timesteps
14	[FEATURE]_[N]	The value of feature [FEATURE] [N] timesteps in the past
15	[FEATURE]_future_[N]	The value of feature [FEATURE] [N] timesteps in the future
12	mean_[PM10/PM2.5]_nearby	The distance-weighted PM10/PM2.5 mean from nearby cities

Features #1-4 correspond to the numbers of tweets in each timestep that contain specific words or phrases associated with bad air quality conditions (i.e. “smog”, ‘’emissions”, “air pollution”, “pollution alert”). Since tweets are collected using air quality related keywords, we expect that the total number of tweets in each timestep (feature #5) might provide useful information with respect to air quality in the respective timestep (i.e. we expect more air quality related tweets to be posted when the air quality is bad).

In addition to these simple features, we also used four advanced features (#6-9) which are generated by utilizing the outputs of different tweet classifiers to obtain estimates of the numbers of tweets (in each timestep) that: a) discuss about air quality[54] (feature #6), b) provide information about current air quality (feature #7) c) refer to low air pollution (feature #8), d) refer to high air pollution (feature #9). Some details about the classifiers used to generate the above features are provided in Table 11. The first classifier (*AQ_Discussion*) is built using a training set of 600 tweets, manually labelled as relevant/irrelevant with air quality. The second classifier (*AQ_Sense*) is built using a training set of 600 tweets, manually labelled with respect to whether they provide information about current air quality levels (relevant) or not (irrelevant). The third classifier (*AP_Severity*) is built using a training set of 200 tweets, manually labelled with respect to whether they refer to high (relevant) or low (irrelevant) current air pollution levels. In all cases, tweets were preprocessed by applying stemming and stop-word removal and represented using a tf-idf bag-of-words representation. As classification algorithm, we used L2-regularized L2-loss Support Vector Machine (the LibLinear[55] implementation) with default parameters.

Table 11 – AQ_General, AQ_Sense and AP_Severity classifier details.

Classifier	# examples	# relevant	# irrelevant	Precision	Recall
AQ_Discussion	600	350	250	90.6%	91.2%
AQ_Sense	600	200	400	88.4%	89.9%
AP_Severity	200	100	100	80.4%	81.7%

In Table 11, we can also see the classification performance of the three classifiers in terms of precision and recall (measured by applying 10-fold cross-validation on the respective training set). We notice that, in all cases, both precision and recall are higher than 80%. This suggests that the estimates generated using the outputs of these classifiers (features #6-9) will be quite representative of the actual numbers. Table 21 and Table 22 of the Appendix, show examples of tweets classified as relevant by the *AQ_Discussion* and the *AQ_Sense* classifier respectively, while

[54] Although tweets are collected with air quality related keywords, still many tweets irrelevant with air quality are retrieved.

[55] https://www.csie.ntu.edu.tw/~cjlin/liblinear/

Table 23 and Table 24 of the Appendix, show examples of tweets classified by the *AP_Severity* classifier as High and Low respectively.

In *Figure 39* we plot the daily numbers of London tweets classified as relevant/irrelevant by the *AQ_Discussion* classifier (AQ_D/Irrelevant), classified as relevant by the AQ_Sense classifier (AQ_S) and classified as relevant/irrelevant by the AP_Severity classifier (High/Low). We notice that a considerable number of tweets irrelevant with air quality are retrieved. Moreover, we see that, as expected, the number of AQ_S tweets is always smaller than the number of AQ_D tweets and that the majority of AQ_S tweets are classified as related to high air pollution. This can be explained by the fact that the list of keywords used for querying the Twitter API is primarily oriented towards bad air quality conditions and by the fact that people tend to tweet more about bad air quality.

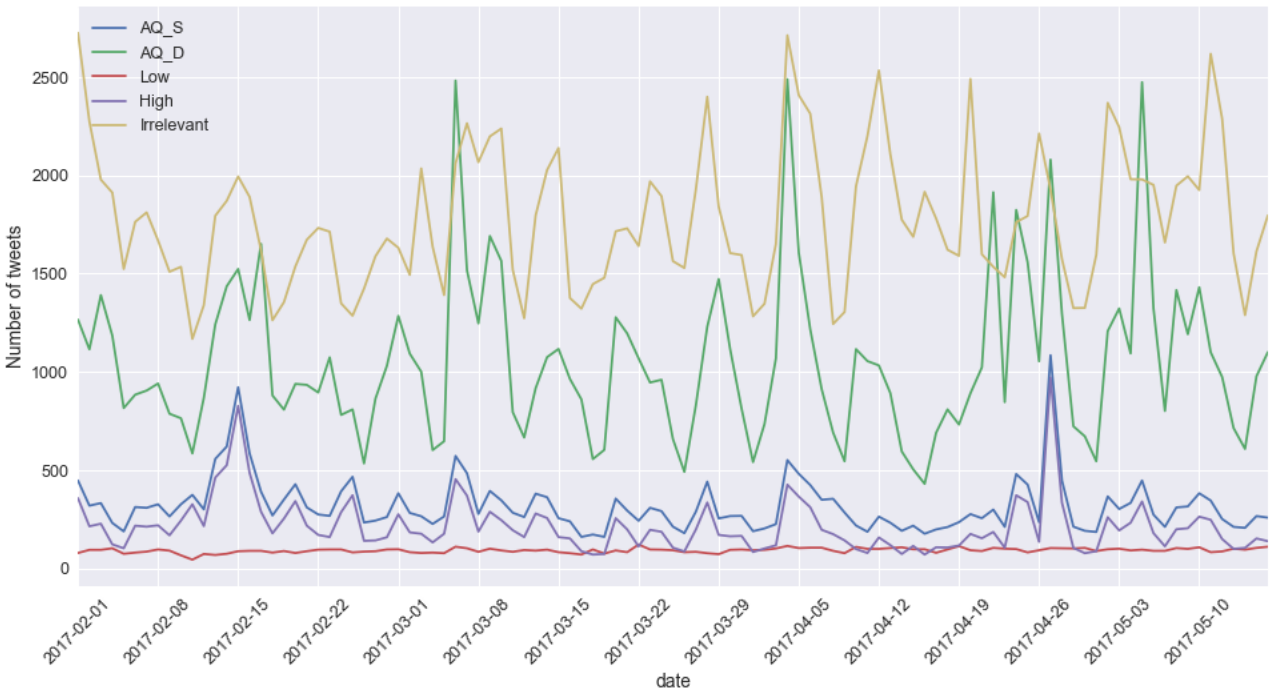


Figure 39 – Daily numbers of AQ_D, AQ_S, High, Low and Irrelevant tweets in London. Irrelevant refers to tweets classified as irrelevant by the AQ_Discussion classifier.

Examining the daily and hourly distributions of the collected tweets (*Figure 40*) we notice that, in all cities, significantly less tweets are posted towards weekends and during night hours. Thus, to make the models take this information into account, two additional features were created (features #10,11). Feature #10 is set to 1 if the timestep belongs to an active day (Monday, Tuesday, Wednesday or Thursday) and is set to 0 otherwise. Feature #11 is set to 1 if the timestep contains active hours (9:00 - 24:00) and is set to 0 otherwise.

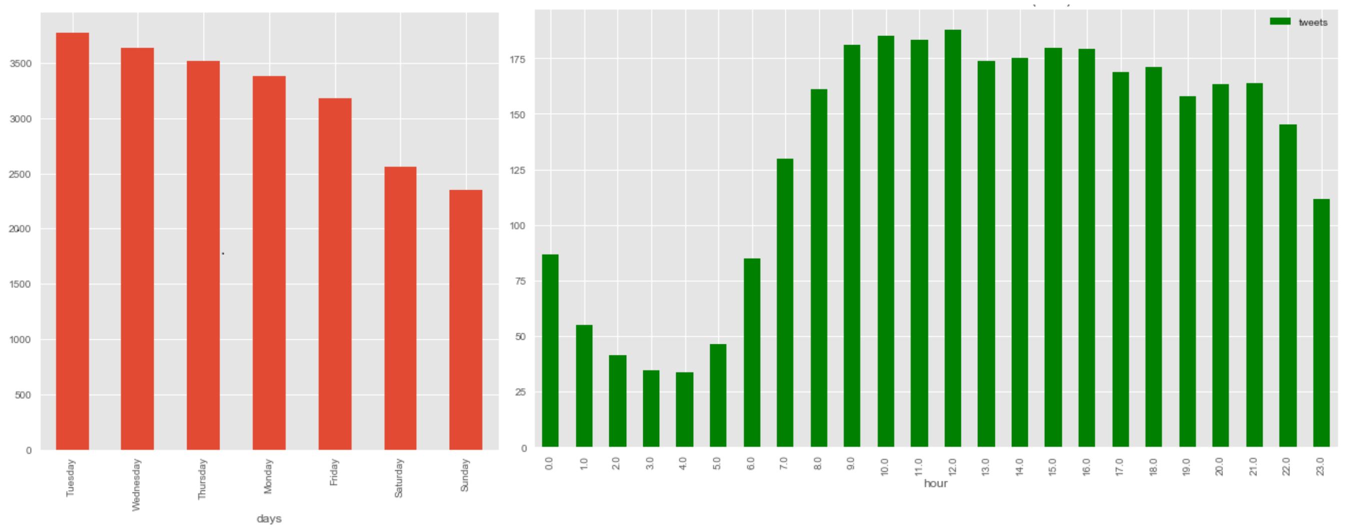


Figure 40 - Daily and hourly tweet distributions.

Besides the basic versions of features #1-9 which correspond to descriptive statistics of the tweets posted in the current timestep, we also generated *lagged* and *rolling mean* versions of the features. Such features are commonly used in timeseries prediction problems to account for temporal dependencies between the dependent variable and the independent variables that go beyond the current timestep. A lagged version of a feature consists of the value of that feature n timesteps in the past while a rolling mean version of a feature is the average value of that feature during the previous n timesteps. Since we are interested in making predictions not only for the current timestep but also for few timesteps in the past, we also generate *future* versions of the features, which correspond to the values of those feature n timesteps in the future. Features of this type are expected to be quite important since a lag is expected between the time that someone experiences bad air quality conditions until he/she tweets about it. Up to four timesteps in the past are considered for *rolling mean* and *lagged* features and up to two timesteps in the future are considered for *future* features. Although potentially helpful, considering more than two timesteps ahead would imply that estimations would have to be provided with a delay larger than 12 hours.

Finally, to let models take into account the air pollution levels in nearby cities (for which we assume that official air pollution measurements are available) we use a non-Twitter feature (#12) that corresponds to the distance-weighted average pollution levels (PM10/PM2.5) in nearby cities[56]. For a city c_j, this distance-weighted average is calculated as:

$$\sum_{i=1,i\neq j}^{m} \frac{\frac{1}{d(c_j,c_i)}}{\sum_{i=1,i\neq j}^{m} \frac{1}{d(c_j,c_i)}} p_{c_i}$$

where p_{c_i} is the pollutant value in the nearby city c_i and $d(c_j, c_i)$ is the distance between the two cities, i.e. the weight of each nearby city c_i is set equal to the inverse of its distance from c_j, divided by the sum of the inverse distances of all nearby cities (to make weights sum up to 1). This feature is similar to deterministic spatial interpolation techniques for PM10 prediction (Taheri Shahraiyni & Sodoudi, 2016).

Figure 41 shows the observed correlations between PM10 and each of the basic features (#1-9,12) when all cities are jointly considered (feature-feature correlations are also presented). We observe that in both cities, the distance-

[56] Lagged versions of this feature are also considered.

weighted average PM10 in nearby cities exhibits a very high positive correlation with the target, followed by *smog*, *high* and *AQ_S* which also exhibit relatively high positive correlations with the target. On the other hand, features such as *low* and *emissions* exhibit a negative correlation.

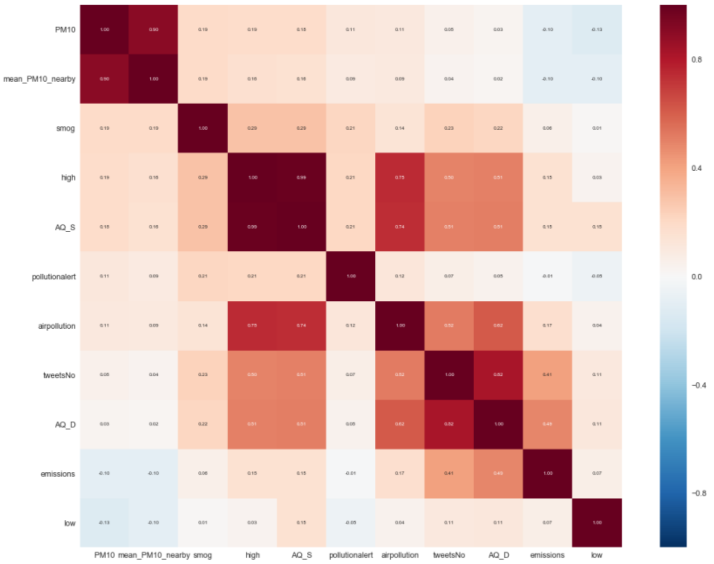


Figure 41 - Feature correlations

4.4 Air Quality Estimation Experiments

In this section, we present our experimental results. We build and evaluate models that predict the PM10[57] pollution class (see Table 3) and use *Accuracy* (the percentage of correct classifications) and *Area Under ROC* (AUC) as classification performance metrics. Both Accuracy and AUC are standard classification metrics, with AUC being more suitable for imbalanced classification problems. As formulated here, air quality prediction is an imbalanced classification problem since the *very good* and *good* pollution classes are observed much more frequently than the *medium* and *bad* pollution classes in the studied cities. As classification algorithm, we use Stochastic Gradient Boosting (Friedman, 2001) (an implementation from scikit-learn[58] in all our experiments[59]. As a baseline, we use a classifier that always predicts the most frequent class (*majority-always* classifier). Finally, to facilitate model transfer across cities, all variables (both features and target) are standardized before model training.

4.4.1 One-to-one vs Many-to-one Models

In the first experiment, we compare the performance of models built using a single city-specific dataset for training with the performance of a model that is trained on a dataset that combines all city-specific datasets (excluding of course the dataset of the city for which predictions are made). The results of this experiment are presented in Table 12 (Accuracy) and Table 13 (AUC). We observe that, for most cities, the best performance is obtained using the many-to-one approach (except for Birmingham when considering Accuracy using only Manchester for training leads to better performance). Therefore, we use the many-to-one approach for the rest of the experiments. Comparing the performance across cities, we see that London is predicted with significantly lower accuracy. This is probably related to the fact that London is far from all other cities that we consider and is therefore difficult to predict its air pollution

[57] Identical experiments have been conducted with PM2.5-based pollution classes, leading to similar results.

[58] http://scikit-learn.org

[59] Random forest (Breiman, 2001) was also tested but exhibited slightly inferior performance.

based on data from the other cities. We also notice that the best performance obtained on each city is significantly better than the performance of the majority-always classifier.

Table 12 - One-to-one vs many-to-one (Accuracy).

Accuracy	London	Manchester	Leeds	Liverpool	Birmingham	Average
London	-	0.636	0.667	0.626	0.607	-
Manchester	0.584	-	0.815	0.832	**0.822**	
Leeds	0.624	0.808	-	0.802	0.781	-
Liverpool	0.590	0.844	0.798	-	0.760	-
Birmingham	0.539	0.826	0.798	0.805	-	-
Best one-to-one	0.624	0.844	0.815	0.832	**0.822**	0.787
Many-to-one	**0.628**	**0.848**	**0.828**	**0.844**	0.804	**0.790**
Majority-always	0.533	0.664	0.610	0.680	0.574	0.612

Table 13 - One-to-one vs many-to-one (AUC).

AUC	London	Manchester	Leeds	Liverpool	Birmingham	Average
London	-	0.824	0.870	0.893	0.804	-
Manchester	0.633	-	0.865	0.952	0.869	-
Leeds	0.667	0.925	-	0.930	0.914	-
Liverpool	0.646	0.948	0.850	-	0.870	-
Birmingham	0.621	0.910	0.824	0.899	-	-
Best one-to-one	0.667	0.948	0.870	0.952	0.914	0.870
Many-to-one	**0.691**	**0.952**	**0.936**	**0.953**	**0.932**	**0.893**
Majority-always	0.500	0.500	0.500	0.500	0.500	0.500

4.4.2 Feature Selection Experiments

Classification algorithms tend to induce models that focus on features that exhibit high correlations with the target variable. While this characteristic is appropriate for a typical classification problem setup, it is not very suitable under dataset shift, i.e. classification problems where the distribution of the test data is different from the distribution of the training data. This is the case with the problem that we deal with here, since models are tested on different cities than the cities they are trained on. To deal with this problem, we apply feature selection to keep features that are both strongly but also consistently correlated with the target across different cities. To this end, we propose the following feature selection metric that simultaneously considers average correlation and stability of the correlation across cities (*S&C*):

$$\max_{f_j \in F} \frac{\frac{1}{N}\sum_{c_i=1}^{N} |cor_{c_i}(f_j)|}{var(cor_{c_i}(f_j))}$$

where $corr_{c_i}(f_j)$ denotes the correlation (Pearson correlation) of feature f_j with the target for the city c_i and $var(cor_{c_i}(f_j))$ is the variance of the correlation of feature f_j across different cities. In addition to the proposed *S&C* metric we also experiment with the following feature selection metrics:

- $\min_{f_j \in F} var(cor_{c_i}(f_j))$, stability only (***Sonly***)
- $\max_{f_j \in F} \frac{1}{N} \sum_{c_i=1}^{N} |cor_{c_i}(f_j)|$, correlation only (***Conly***)
- $\max_{f_j \in F} |cor_{c_i}(f_j)|$, correlation only, city agnostic (***Conly_a***)

Sonly prefers features that exhibit similar correlations across all cities, without taking the strength of the correlation into account. ***Conly***, on the other hand, prefers features that are highly correlated with the target in all cities (on average), without considering stability. Similarly, ***Conly_a*** considers only the strength of the correlation but in a city agnostic manner as it ignores the fact that examples come from different cities.

To evaluate the impact of feature selection on model accuracy, we apply each feature selection metric to select 20 features[60] *and features*[60] *and compare the obtained performance with no feature selection. The results are presented in Table 14 and Table 15 for Accuracy and AUC respectively. We see that in most cities, feature selection leads to better results than no selection. Comparing the feature selection metrics with each other, we notice that S&C outperforms the rest of the metrics in three out of five cities for both Accuracy and AUC and it also obtains the best average performance that is better than the average performance of no selection. To give a better idea of the type of errors made by the S&C many-to-one model (the best performing model),*

Table 16 and Table 17 show the confusion matrices that we obtain for London and Manchester.

Table 14 – Comparison of feature selection metrics (Accuracy)

Accuracy	London	Manchester	Leeds	Liverpool	Birmingham	Average
S&C	0.625	0.851	**0.832**	**0.855**	**0.835**	**0.800**
Sonly	0.609	**0.862**	0.826	0.841	0.802	0.788
Conly	0.619	0.839	0.814	0.834	0.820	0.785
Conly_a	0.614	0.839	0.824	0.844	0.815	0.787
No selection	**0.628**	0.848	0.828	0.844	0.804	0.790
Majority	0.533	0.664	0.610	0.680	0.574	0.612

Table 15 - Comparison of feature selection metrics (AUC)

AUC	London	Manchester	Leeds	Liverpool	Birmingham	Average
S&C	**0.709**	**0.952**	**0.940**	0.953	0.924	**0.896**
Sonly	0.673	0.950	0.913	0.948	0.916	0.880
Conly	0.703	0.951	0.931	0.942	**0.934**	0.893
Conly_a	0.698	0.942	0.932	**0.954**	**0.934**	0.892
No selection	0.691	**0.952**	0.936	0.953	0.932	0.893

[60] The top-20 features selected with each metric are shown in Table 25 of the Appendix.

Majority	0.500	0.500	0.500	0.500	0.500	0.500

Table 16 - Confusion matrix for London

London					
		Predicted			
		Very good	Good	Medium	Bad
Actual	Very good	133	10	0	0
	Good	103	93	2	0
	Medium	2	14	6	0
	Bad	0	5	3	0

Table 17 - Confusion matrix for Manchester

Manchester					
		Predicted			
		Very good	Good	Medium	Bad
Actual	Very good	193	33	0	0
	Good	16	87	3	0
	Medium	0	0	6	0
	Bad	0	0	0	0

4.4.3 Contribution of Twitter-based Features

Among the features that we generated, the distance-weighted average pollutant values from nearby cities (mean_PM10_nearby) exhibit very high correlations with the target variable while Twitter-based features have relatively weaker correlation. To study the impact that each type of features has on final performance, in this paragraph we build models that take only one type of features into account (i.e. only mean_PM10_nearby features or only Twitter-based features) and compare their performance to the performance of the S&C many-to-one model**.** The results are presented in Table 18 (Accuracy) and Table 19 (AUC). We observe that models based only on mean_PM10_nearby features perform significantly better than models based only Twitter-based features. When both types of features are jointly considered, however, the performance improves further, especially in terms of Accuracy.

Table 18 - Accuracy obtained with Twitter-based vs mean_PM10_nearby vs all features

Accuracy	London	Manchester	Leeds	Liverpool	Birmingham	Average
Twitter-based	0.430	0.603	0.505	0.639	0.621	0.560
mean_PM10_nearby	0.569	0.841	0.814	0.824	0.823	0.774
All features	**0.625**	**0.851**	**0.832**	**0.855**	**0.835**	**0.800**

Table 19 - AUC obtained with Twitter-based vs mean_PM10_nearby vs all features

AUC	London	Manchester	Leeds	Liverpool	Birmingham	Average
Twitter-based	0.603	0.635	0.596	0.676	0.750	0.625
mean_PM10_nearby	0.691	**0.952**	0.936	**0.953**	**0.932**	0.893
All features	**0.709**	**0.952**	**0.940**	**0.953**	0.924	**0.896**

Finally, the performance of the models is compared in a regression setup[61], i.e. when PM10 values are directly predicted instead of pollution classes, using Mean Squared Error (MSE) as the performance metric. As shown in Table 20 and in accordance with the previous results, we observe than models based on mean_PM10_nearby features make significantly smaller errors than models based on Twitter-based features, while the best results are obtained when all features are jointly considered. To better illustrate the improvements incurred by the combination of Twitter-based and mean_PM10_nearby features over using mean_PM10_nearby features alone, Figure 42 plots the daily PM10 values for the city of Manchester (where the largest gains are obtained when all features are combined) against the estimated values when either all or only the mean_PM10_nearby features are used.

Table 20 - MSE obtained with Twitter-based vs mean_PM10_nearby vs all features

MSE	London	Manchester	Leeds	Liverpool	Birmingham	Average
Twitter-based	0.65	1.00	0.98	1.05	0.96	0.92
mean_PM10_nearby	0.40	0.20	**0.19**	0.21	0.20	0.24
All features	**0.39**	**0.14**	**0.19**	**0.16**	**0.18**	**0.21**

[61] Here we use a Stochastic Gradient Boosting regressor.

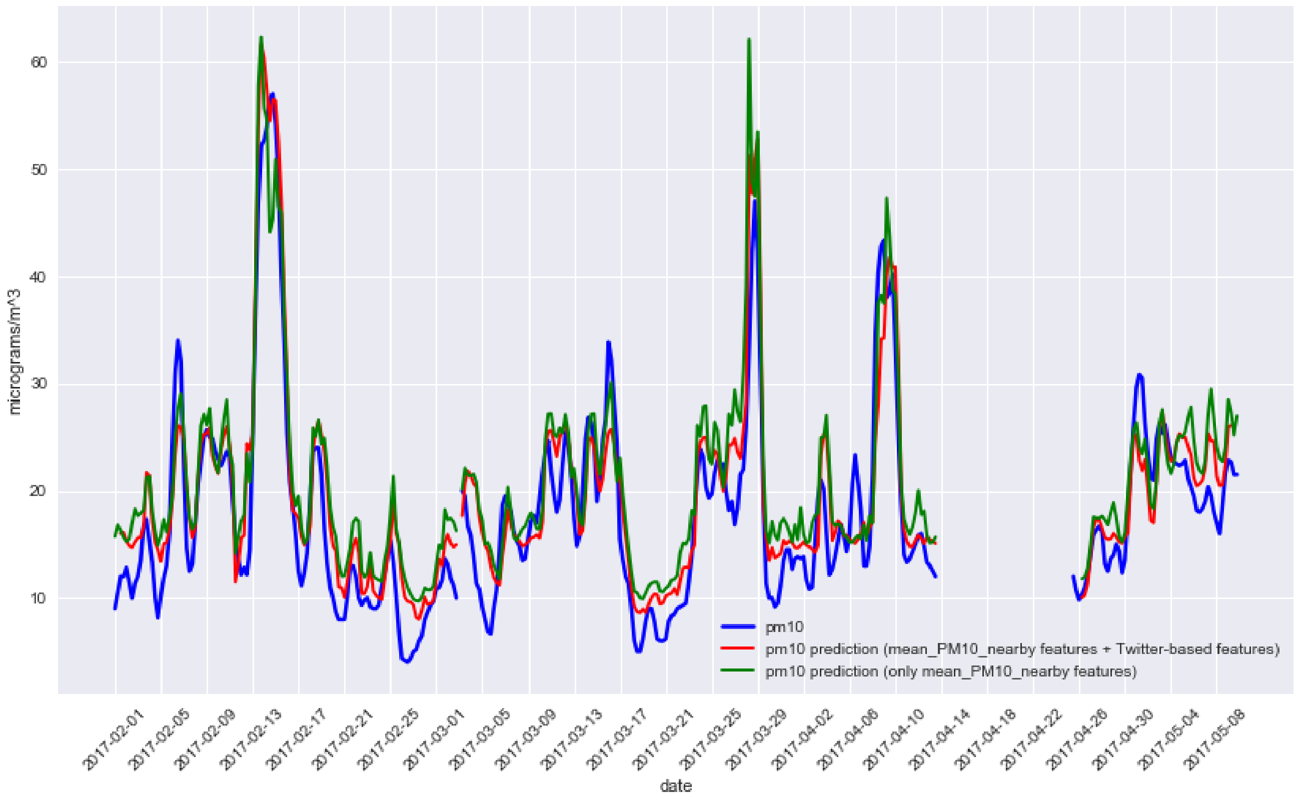


Figure 42 - Regression with and without Twitter-based features in Manchester

5 Conclusions

This deliverable concluded the research and development activities that concern the collection and indexing of environmental nodes (T3.1) as well as the acquisition and processing of user-generated images for air quality prediction (T3.2), having led to the delivery of a variety of data collection modules (Flickr images, webcam images, air quality measurements from ground stations) and a comprehensive image analysis service that conducts the necessary processing steps in order to deliver the inputs required by the air quality estimation models developed within T3.3.

As far as image collection is concerned, an important development compared to D3.1 was the expansion of its geographical scope to the whole European continent instead of only specific cities in Germany and Norway (the countries of the hackAIR pilots). The decision of the expansion was taken jointly with all project partners and aims at increasing the number and the geographical spread of image-based estimations that are given as input to the data fusion module, thus improving the quality and breadth of its outputs. To this end, an improved version of the Flickr image collector was developed which retrieves all Flickr images that are geotagged in Europe by appropriately querying the Flickr API. Besides the expanded geographical scope, the new version of the Flickr collector includes additional improvements such as the rejection of images with uncertain date taken timestamps and a better mechanism for handling queries that return a very large number of results. As a result, approximately 5,000 Flickr images are collected daily.

In addition to these improvements, a number of ways to further increase the number of images that we collect where investigated. Firstly, two very large repositories of webcam images were analyzed, AMOS and webcams.travel. Both repositories were found to contain a significant number of webcams located in Europe and, at the same time, offer a relatively simple way of retrieving images and other required information (location and time) from them. Consequently, two specialized collectors were implemented, facilitating the collection of images from approximately 3,500 different locations at regular time intervals. Moreover, a set of experiments were conducted to investigate the feasibility of exploiting non-geotagged Flickr images, which represent the vast majority of images on Flickr (and other social image sharing platforms). More specifically, a recent state-of-the-art location estimation approach was evaluated in terms of its ability to accurately estimate the location of non-geotagged Flickr images based on textual metadata such as title, description and tags. Experiments conducted on a large-scale collection of approximately 200,000 images, showed that the location of a large percentage of non-geotagged images can be accurately estimated. Thus, a large additional number of Flickr images[62] could be included in the hackAIR system, provided that some small uncertainty with respect to their capture location is permissible.

Besides the collection of images, the collection of air quality measurements was also expanded to the whole Europe. To this end, an environmental data collection module was implemented that retrieves data from the OpenAQ platform which provides PM10 and PM2.5 measurements from a large number of environmental stations in Europe. The collected data cover the majority of European countries, including the countries were the hackAIR pilots will take place (Germany and Norway), are very frequently updated, and come from official, government-level sources such as EEA and DEFRA. In addition, a web-based visualization interface of the data was implemented which facilitates a quick overview of the current air quality conditions (in terms of PM10 and PM2.5).

As far as the techniques for processing user-generated images is concerned, this deliverable includes a comprehensive performance evaluation of the techniques developed during the first year of the project, which are documented in D3.1. In particular, we performed an air quality estimation-oriented evaluation of the accuracy of the two alternative sky localization approaches (*FCN* and *heuristic*) using the help of image-based air quality estimation experts. This evaluation highlighted the complementarity of the two approaches and led to the development of a new approach

[62] Up to about 40,000 images based on rough estimations regarding the number of non-geotagged images that are actually captured in Europe (160,000) and considering the fact that the location of about 25% of them can be accurately estimated.

(*FCN+heuristic*) that exhibits significantly better performance. Moreover, we studied the impact of a variety of common image transformations and filters on the outputs of the image processing techniques, concluding that they are highly robust against most transformations. Based on the results of these evaluations, the final version of the image analysis module was designed and implemented as a web service, which was deployed in 1/3/2017, analyzing all the collected images. The deliverable also reports important statistics with respect to the results of the image analysis service over a period of about 2.5 months such as the percentage of usable sky images among the images that are collected daily in the countries of the hackAIR pilots. We find that that in both Germany and Norway a sufficient number of usable sky images are collected. In addition, the computational performance of the service was optimized and computationally intensive operations were transferred to the GPU.

Finally, we presented an experimental line of work exploring the feasibility of performing air-quality estimations based on Twitter posts. More specifically, we collected Twitter data and ground truth air pollution measurements from five cities in the UK for a period of 3.5 months and developed air quality estimation models using text-mining and machine learning techniques. Despite, the encouraging results that we obtained we highlight that the approach is still under development and additional experimentation is required to conclude about the possibility of using it as an additional air quality estimation source. In particular, we found that a large amount of the predictive ability of the models is due to the use of actual air quality measurements from nearby cities while the contribution of Twitter features in accuracy is relatively small. Further experiments that include additional countries and smaller cities have to be conducted in order to validate the significance of the improvements offered by the inclusion of Twitter-based features to the accuracy of the models.

Concluding, although we believe that to the objectives of tasks T3.1 and T3.2 have been successfully completed with the work reported in this deliverable and in D3.1, there are a number of extensions and improvements that are foreseen within Task 5.2 “Component Development and Integration” and Task 7.3 “hackAIR support services and methodology update” of WP7. More specifically, additional sources of air quality measurements could be integrated to the hackAIR platform, such as data from official organizations in countries that are not currently covered by OpenAQ or data from low-cost sensors offered by initiatives such as http://luftdaten.info/. What is more, the evaluation presented in section 3.1, highlighted that there is still room for improvement with respect to the accuracy of the sky detection and localization methods. In particular, the analysis showed that the presence of cirrus clouds is in many cases the reason why an image is considered unsuitable for air quality estimation. Even though in many cases it is difficult to decide whether an image is unsuitable for air quality estimation due to the presence of cirrus clouds even with a naked eye (see e.g. Figure 19), a possible direction for future work would be the development of a specialized concept detector that would automatically recognize and filter sky-depicting images where sky is covered by this type of clouds. Moreover, given the significant increase in the number of images that will need to be processed in case non-geotagged Flickr images are included, mechanisms to increase the scalability of the image analysis module will have to be investigated. Finally, the possibility of deploying the Twitter-based air-quality prediction in a practical setting needs to be further investigated. Apart from those extensions, we also expect a number of incremental refinements and improvements to be required based on pilot feedback.

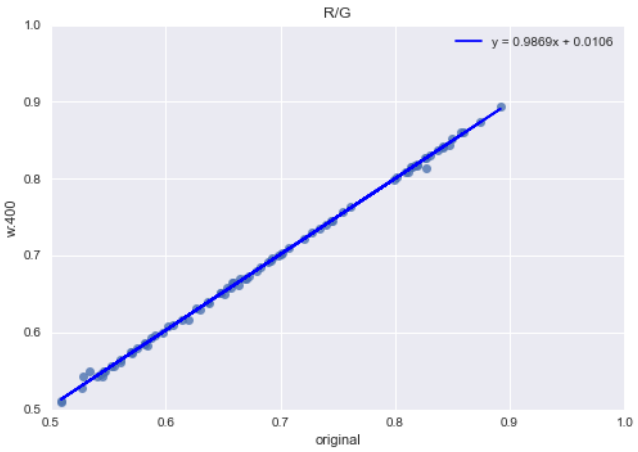
6 References

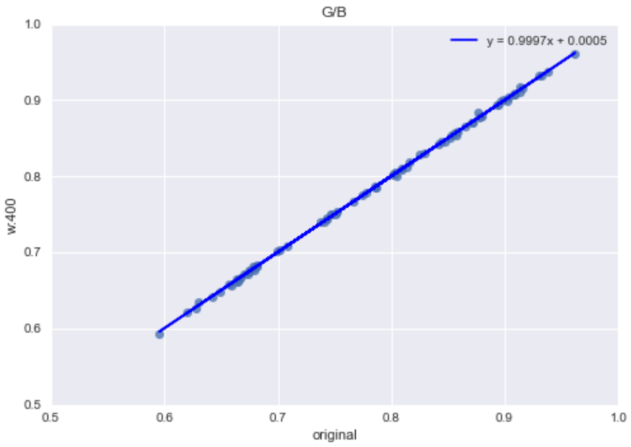
- Amitay, E., Har'El, N., Sivan, R., & Soffer, A. (2004, July). Web-a-where: geotagging web content. In Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 273-280). ACM.
- Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.
- Choi, J., Thomee, B., Friedland, G., Cao, L., Ni, K., Borth, D., ... & Poland, D. (2014, November). The placing task: A large-scale geo-estimation challenge for social-media videos and images. In Proceedings of the 3rd ACM Multimedia Workshop on Geotagging and Its Applications in Multimedia (pp. 27-31). ACM.
- Choi, J., Hauff, C., Van Laere, O., & Thomee, B. (2015). The placing task at mediaeval 2015. In MediaEval 2015, Wurzen, Germany, 14-15 September 2015; Ceur Workshop Proceedings 1436, 2015. CEUR.
- Choi, J., Hau, C., Van Laere, O., Thomee, B. (2016). The placing task at mediaeval 2016. In MediaEval.
- Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. Annals of statistics, 1189-1232.
- Hauff, C., Thomee, B., & Trevisiol, M. (2013). Working Notes for the Placing Task at MediaEval 2013. In MediaEval.
- Jacobs, N., Roman, N., Pless, R. (2007). Consistent Temporal Variations in Many Outdoor Scenes. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
- Jiang, W., Wang, Y., Tsou, M. H., & Fu, X. (2015). Using social media to detect outdoor air pollution and monitor air quality index (AQI): a geo-targeted spatiotemporal analysis framework with Sina Weibo (Chinese Twitter). PloS one, 10(10), e0141185.
- Kordopatis-Zilos, G., Popescu, A., Papadopoulos, S., & Kompatsiaris, Y. (2016). Placing Images with Refined Language Models and Similarity Search with PCA-reduced VGG Features.
- 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.
- Mei, S., Li, H., Fan, J., Zhu, X., & Dyer, C. R. (2014, August). Inferring air pollution by sniffing social media. In Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on (pp. 534-539). IEEE.
- Serdyukov, P., Murdock, V., & Van Zwol, R. (2009, July). Placing flickr photos on a map. In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval (pp. 484-491). ACM.
- Tao, Z., Kokas, A., Zhang, R., Cohan, D. S., & Wallach, D. (2016). Inferring Atmospheric Particulate Matter Concentrations from Chinese Social Media Data. PloS one, 11(9), e0161389.
- Taheri Shahraiyni, H., & Sodoudi, S. (2016). Statistical modeling approaches for pm10 prediction in urban areas; a review of 21st-century studies. Atmosphere, 7(2), 15.
- Thomee, B., Shamma, D. A., Friedland, G., Elizalde, B., Ni, K., Poland, D., ... & Li, L. J. (2015). The new data and new challenges in multimedia research. arXiv preprint arXiv:1503.01817, 1(8).
- Wang, S., Paul, M. J., & Dredze, M. (2015). Social media as a sensor of air quality and public response in China. Journal of medical Internet research, 17(3), e22.

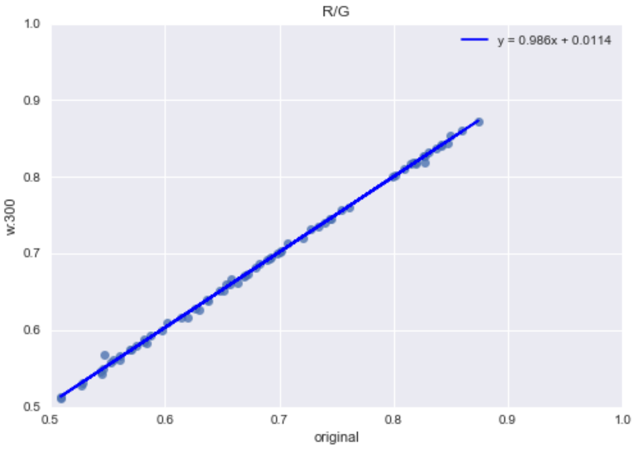
- Xiao, J., Hays, J., Ehinger, K., Oliva, A., and Torralba, A. (2010). SUN Database: Large-scale Scene Recognition from Abbey to Zoo. IEEE Conference on Computer Vision and Pattern Recognition.
- Zhijie, Z., Qian, W., Huadong, S., Xuesong, J., Qin, T., & Xiaoying, S. (2015). A Novel Sky Region Detection Algorithm Based On Border Points. International Journal of Signal Processing, Image Processing and Pattern Recognition, 8(3), 281-290.

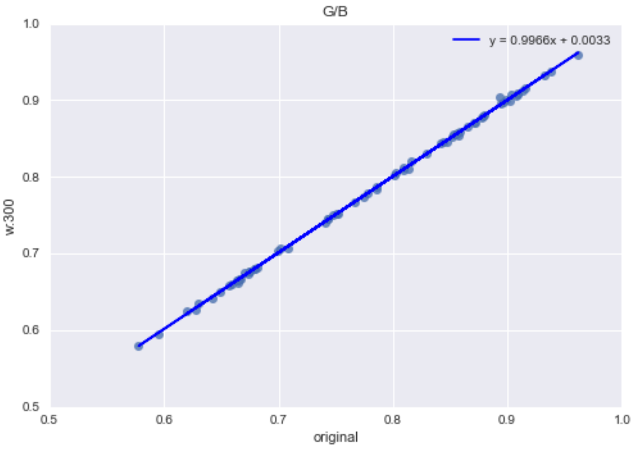
7 Appendix

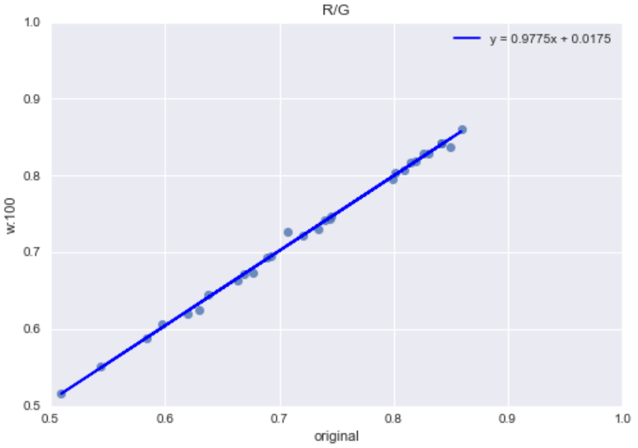
7.1 Additional Results on Robustness against Image Transformations

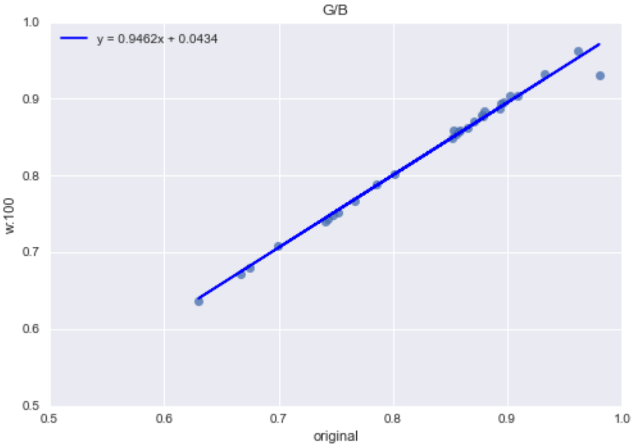


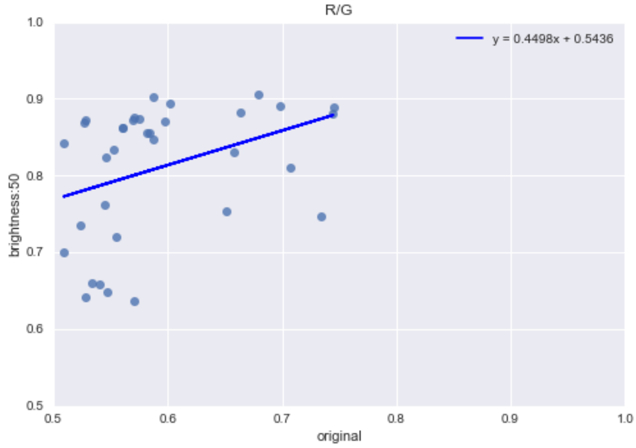


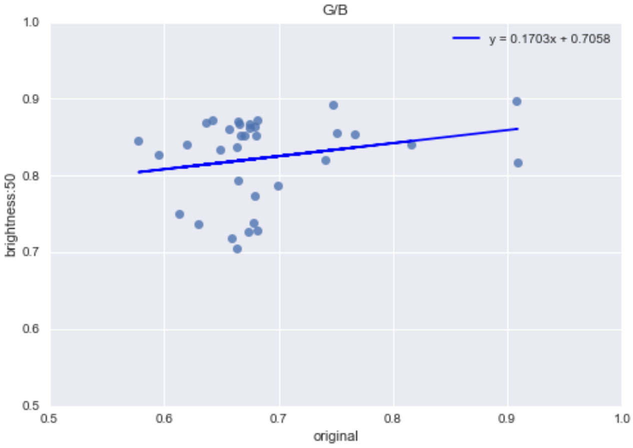


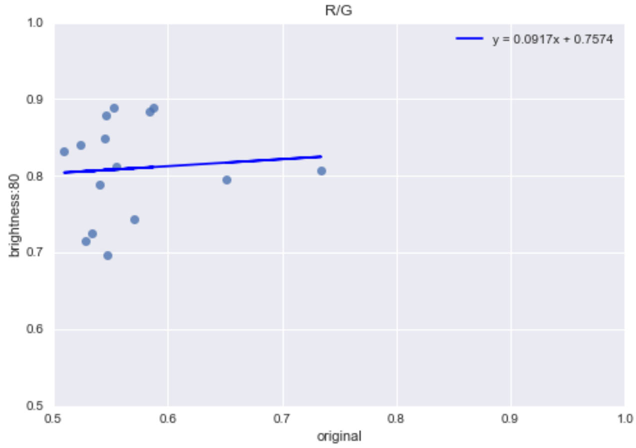


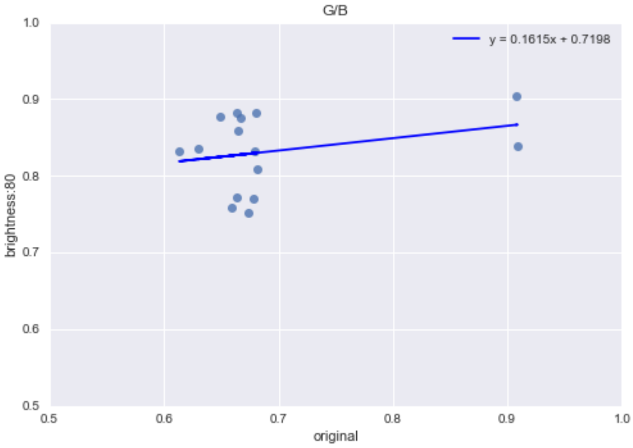


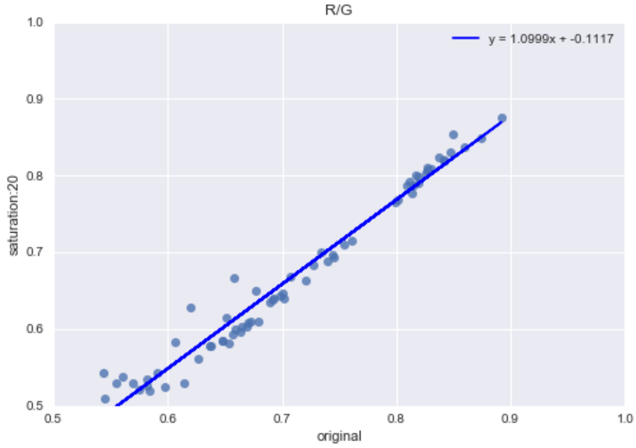


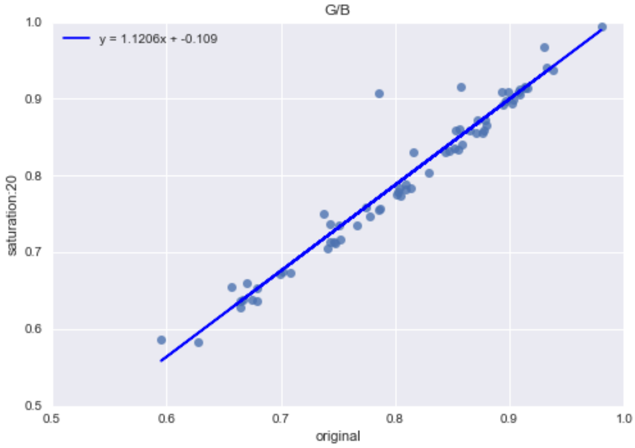

R/G
y = 0.4498x + 0.5436
brightness:50
original
1.0
0.9
0.8
0.7
0.6
0.5
0.5
0.6
0.7
0.8
0.9
1.0

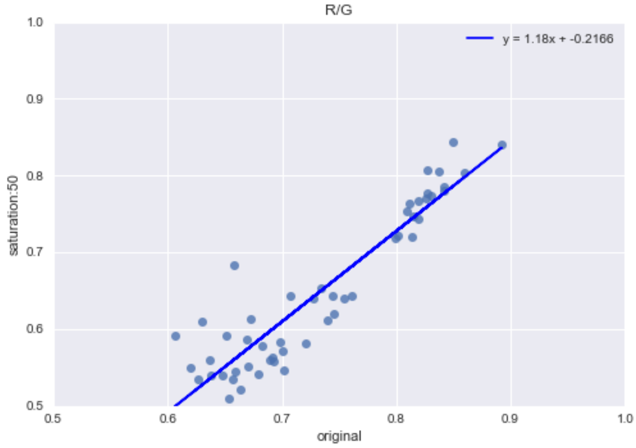

G/B
y = 0.1703x + 0.7058
brightness:50
original
1.0
0.9
0.8
0.7
0.6
0.5
0.5
0.6
0.7
0.8
0.9
1.0

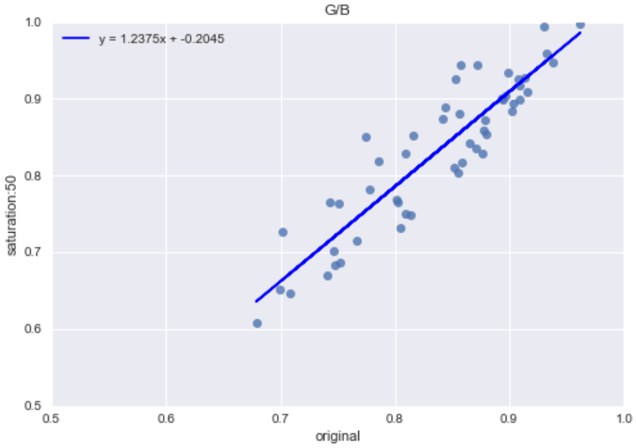

R/G
y = 0.0917x + 0.7574
brightness:80
original
1.0
0.9
0.8
0.7
0.6
0.5
0.5
0.6
0.7
0.8
0.9
1.0

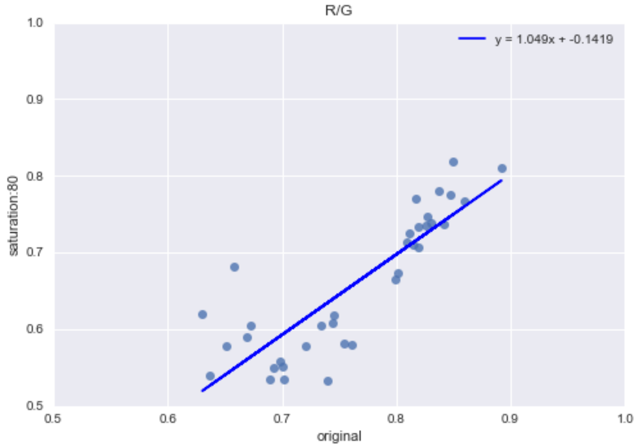

G/B
y = 0.1615x + 0.7198
brightness:80
original
1.0
0.9
0.8
0.7
0.6
0.5
0.5
0.6
0.7
0.8
0.9
1.0

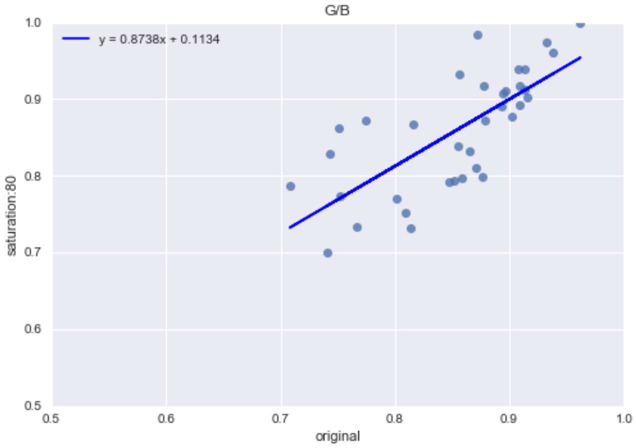

R/G
y = 1.0999x + -0.1117
saturation:20
original
1.0
0.9
0.8
0.7
0.6
0.5
0.5
0.6
0.7
0.8
0.9
1.0

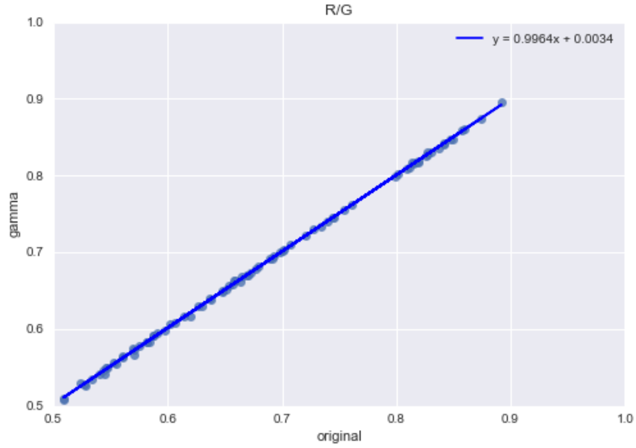

G/B
y = 1.1206x + -0.109
saturation:20
original
1.0
0.9
0.8
0.7
0.6
0.5
0.5
0.6
0.7
0.8
0.9
1.0

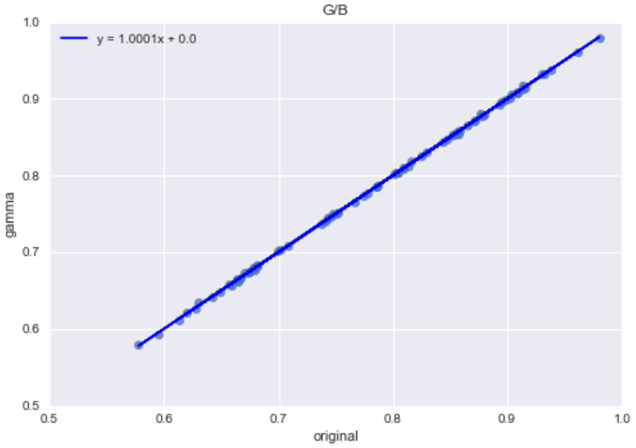

R/G
y = 1.18x + -0.2166
saturation:50
original
0.5
0.6
0.7
0.8
0.9
1.0

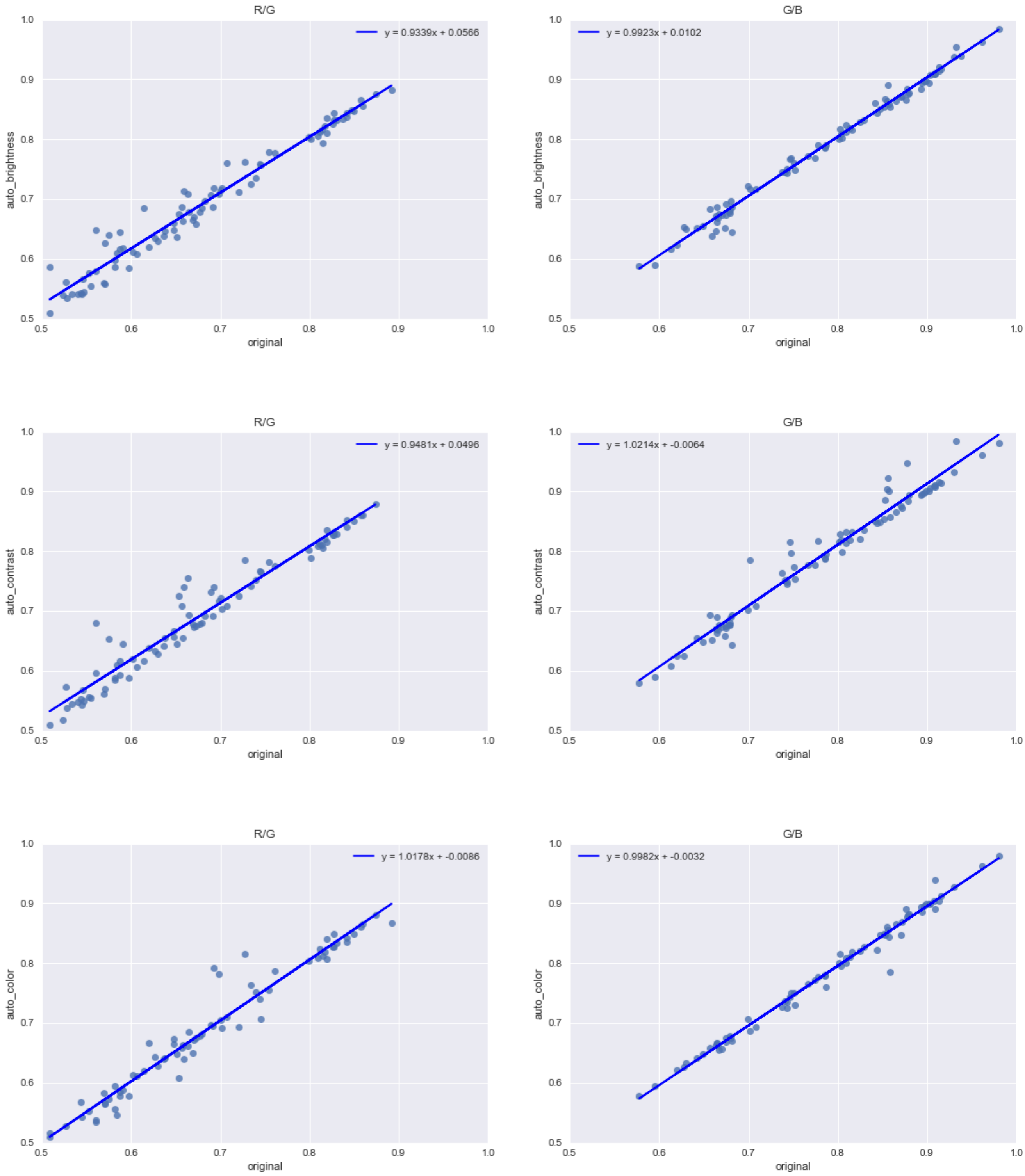

G/B
y = 1.2375x + -0.2045
saturation:50
original
0.5
0.6
0.7
0.8
0.9
1.0

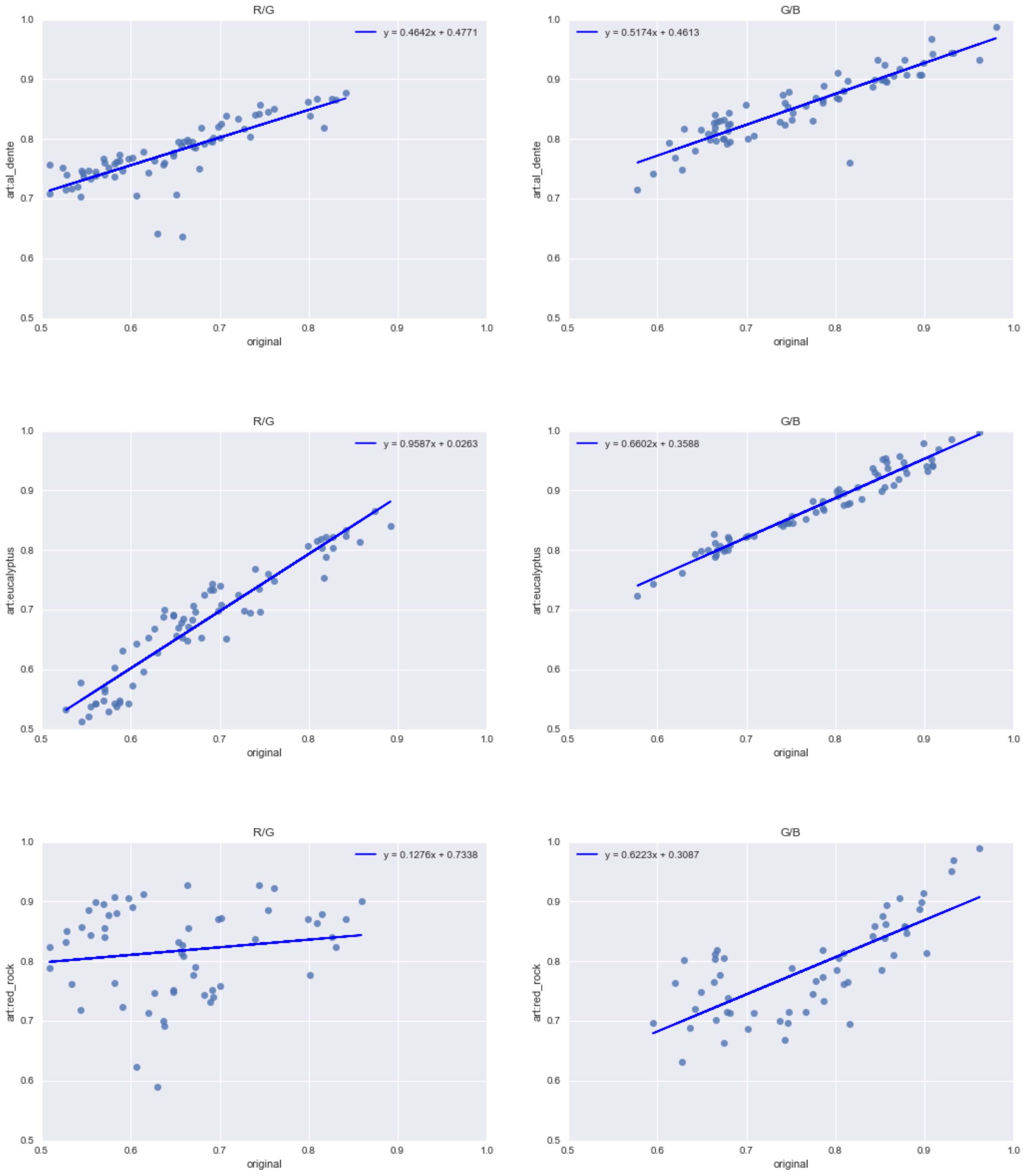

R/G
y = 1.049x + -0.1419
saturation:80
original
0.5
0.6
0.7
0.8
0.9
1.0

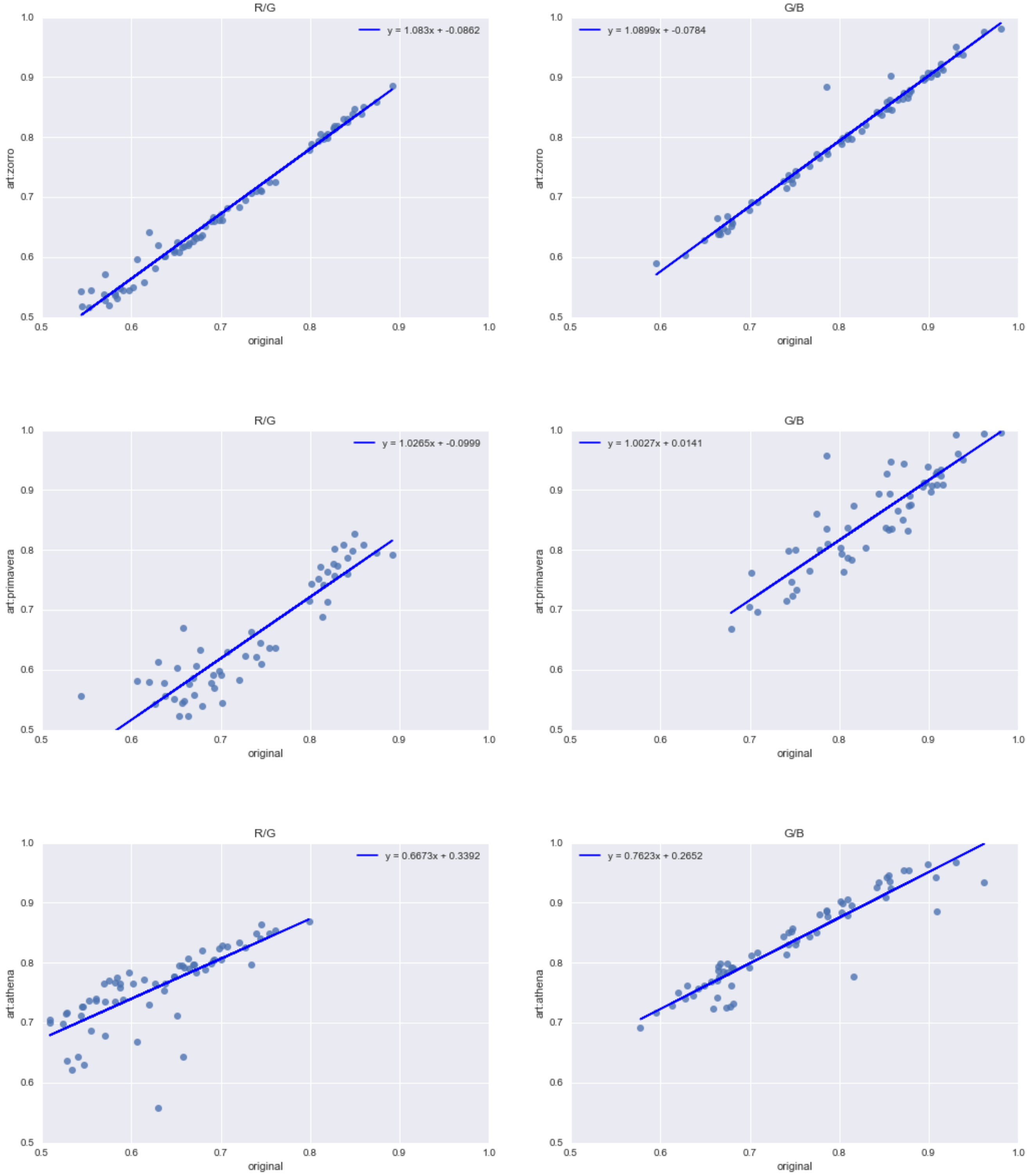

G/B
y = 0.8738x + 0.1134
saturation:80
original
0.5
0.6
0.7
0.8
0.9
1.0


R/G
y = 0.9964x + 0.0034
gamma
original
0.5
0.6
0.7
0.8
0.9
1.0


G/B
y = 1.0001x + 0.0
gamma
original
0.5
0.6
0.7
0.8
0.9
1.0


R/G
y = 0.9339x + 0.0566
auto_brightness
original
G/B
y = 0.9923x + 0.0102
auto_brightness
original
R/G
y = 0.9481x + 0.0496
auto_contrast
original
G/B
y = 1.0214x + -0.0064
auto_contrast
original
R/G
y = 1.0178x + -0.0086
auto_color
original
G/B
y = 0.9982x + -0.0032
auto_color
original
0.5
0.6
0.7
0.8
0.9
1.0


R/G
y = 0.4642x + 0.4771
art:al_dente
original
G/B
y = 0.5174x + 0.4613
art:al_dente
original
R/G
y = 0.9587x + 0.0263
art:eucalyptus
original
G/B
y = 0.6602x + 0.3588
art:eucalyptus
original
R/G
y = 0.1276x + 0.7338
art:red_rock
original
G/B
y = 0.6223x + 0.3087
art:red_rock
original
0.5
0.6
0.7
0.8
0.9
1.0


R/G
y = 1.083x + -0.0862
art:zorro
original
G/B
y = 1.0899x + -0.0784
art:zorro
original
R/G
y = 1.0265x + -0.0999
art:primavera
original
G/B
y = 1.0027x + 0.0141
art:primavera
original
R/G
y = 0.6673x + 0.3392
art:athena
original
G/B
y = 0.7623x + 0.2652
art:athena
original
0.5
0.6
0.7
0.8
0.9
1.0

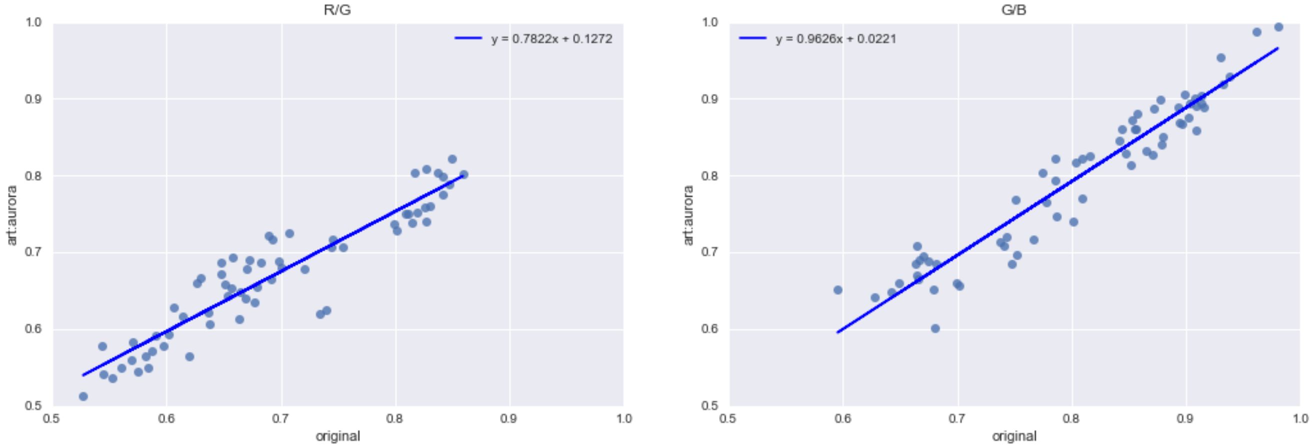


Figure 43 - Scatterplots of R/G (left) and G/B (right) scores calculated from original (x axis) and transformed images (y axis).

7.2 Additional Results on Twitter-based Air Quality Estimation

Table 21 – Examples of London tweets classified as relevant by the AQ_Discussion classifier

AQ_Discussion tweets
RT @Jackie_News: Wandsworth, Kingston and Hounslow are some of the areas under a toxic "red alert" today http://t.co/o1pkytqhJM
RT @ClientEarth: #Knightbridge has become the third monitor breach legal #airpollution limits. Support our fight for #clearnair…
RT @LondonAir: High air pollution forecast valid from Thursday 19 January to end of Thursday 19 January https://t.co/W3D5gD9fxO #airpollution
RT @MayorofLondon: London's dirty air is a public health crisis. I'm committed to tackling this. Read more about my plans here: https://t.c...
@tony_olmstead @SenSanders no a hypocrite if no alternative. Do people campaigning against air pollution have to stop breathing?
RT @ LondonAir: This pic shows PM2.5 particulate #airpollution building through this cold snap since Sat. Widespread Moderate, like…

Table 22 – Examples of London tweets classified as relevant by the AQ_Sense classifier

AQ_Sense tweets
RT @RSLenvironment: AirPollution is a crisis that "plagues" the UK, particularly children, according to UN human rights expert http://t.c...
Greenhouse Morning News is out! Top stories: EU 'ready to fight' for # climate & UK #airpollution crises…
RT @GasNaturally: Another proof that #NatGas reduces #PowerGen CO2 #emissions when used instead of #Coal

Our #future. EVs will help “goal to reduce greenhouse gas #emissions by 80-95% by 2050” https://t.co/L46Xm7VI29
This ‘smog-eating’ city sculpture can combat London’s toxic pollution as effectively as 275 trees https://t.co/99FY7EyiSW
RT @CleanSpaceLDN: Bad night's sleep? #Airpollution could be to blame, study finds https://t.co/MjYQiaVDB7 via @guardian

Table 23 - Examples of London tweets classified as High by the AP_Severity classifier

AP_Severity High
RT @PlumeInLondon: High pollution (50) at 10PM. High for #London. Avoid physical activities if sensitive https://t.co/3LVRgps965
London's air pollution is killing me. Coughs now sound like squeaky chew toy. #sendhelp #sendventolin
RT @cargill_taxi: And the mayor of London tries to blame poor air quality on toxic air from German factories. You need to look a bit…
@claireL23 The traffic, poor air quality, the light pollution, the lack of green space, the concrete jungle, the bu… https://t.co/JkXYWA16GM
RT @SkyNews: THE GUARDIAN FRONT PAGE: "Toxic air risk to one in four London schools" #skypapers https://t.co/2c6ANlujep
RT @MayorofLondon: London’s toxic air is a public health emergency. Here is what I’m doing about it https://t.co/YHw2CVepPI

Table 24 - Examples of London tweets classified as Low by the AP_Severity classifier

AP_Severity Low
RT @DefraUKAir: Latest Wed 9am: Low air pollution measured across all regions of the UK. https://t.co/dDB6mTQz37 #ukair
Always, if I were able to lie down, I would be laying on the grass right now looking at the beautiful sky whilst li… https://t.co/qAbMx6kgG1
Air is OK near Sevenoaks - Greatness Park (Pollution Low : 1)
The pollution is evil, but the London sky was oh so beautiful https://t.co/kWWPVAx7Io
Feeling minor dirt in the air near Enfield - Bush Hill Park (Pollution Low : 2)
LOW air pollution forecast for Central London on Fri 9th May. Health advice: https://t.co/fYVbZm2jd8

Table 25 – Top-20 features selected using each feature selection metric.

S&C	Sonly	Conly	Conly_a

mean_PM10_nearby_1	active_hour	mean_PM10_nearby_1	mean_PM10_nearby_1
mean_PM10_nearby_4	mean_PM10_nearby_4	mean_PM10_nearby_2	mean_PM10_nearby_2
mean_PM10_nearby_2	emissions	mean_PM10_nearby_3	mean_PM10_nearby_3
mean_PM10_nearby_3	active_day	mean_PM10_nearby_4	mean_PM10_nearby_4
emissions	mean_PM10_nearby_3	high_future_2	high_future_2
active_hour	mean_PM10_nearby_1	aqs_future_2	aqs_future_2
active_day	mean_PM10_nearby_2	high_future_1	high_future_1
airpollution	aqd_future_2	aqs_future_1	rolling_mean_high_2
aqs_future_2	aqd	rolling_mean_high_2	rolling_mean_aqs_2
high_future_2	aqd_1	rolling_mean_aqs_2	aqs_future_1
high_future_1	aqd_2	rolling_mean_high_1	rolling_mean_high_3
aqs_future_1	aqd_3	rolling_mean_high_3	rolling_mean_aqs_3
high	airpollution	rolling_mean_aqs_3	rolling_mean_high_1
aqs	rolling_mean_aqd_1	rolling_mean_aqs_1	rolling_mean_aqs_1
rolling_mean_high_1	tweetsno	smog	rolling_mean_high_4
smog	aqd_4	rolling_mean_low_3	rolling_mean_aqs_4
high_1	rolling_mean_aqd_2	rolling_mean_low_2	smog
rolling_mean_aqs_1	aqs_4	rolling_mean_low_4	rolling_mean_low_3
low_future_1	high_4	high	high
low_1	rolling_mean_aqd_3	rolling_mean_high_4	aqs