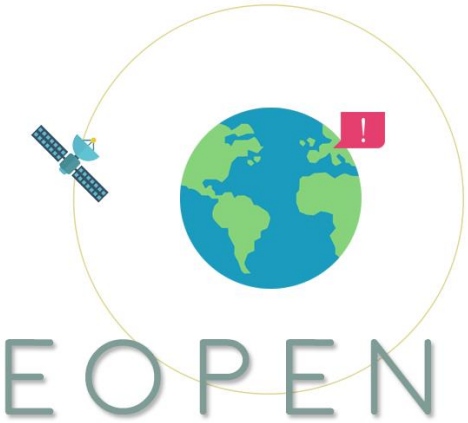

EOPEN

opEn interOperable Platform for unified access and analysis of Earth observatioN data

H2020-776019

D3.3

EOPEN Social Media Crawlers

Dissemination level:	Public
Contractual date of delivery:	Month 32, 30/06/2020
Actual date of delivery:	Month 32, 30/06/2020
Workpackage:	WP3 EO and non-EO data acquisition
Task:	T3.2 Social Media crawling and quality control T3.3 Meteorological and climatological data acquisition
Type:	Other
Approval Status:	Approved
Version:	1.0
Number of pages:	43
Filename:	D3.3-EOPEN Social Media Crawlers_2020-06-30_v1.0.docx

Abstract

This deliverable reports on the current status and latest advances in regards with Work Package 3 “EO and non-EO data acquisition” and specifically tasks T3.2 about social media crawling and quality control and T3.3 about meteorological and climatological data acquisition. The development of the Social Media Crawler that collects and analyses social media data and the progress on the Weather Data Management Module are both reported here. The key contributions of the deliverable are: (1) a complete framework that crawls and analyses social media posts from Twitter, (2) a manually annotated dataset of relevant/irrelevant tweets, (3) a text classification methodology to filter out irrelevant tweets, (4) the developed user interfaces that display the tweets, (5) a large collection of tweets relevant to the PUCs, and (6) the improved Weather Data Management Module including new datasets.

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement 776019

History

Version	Date	Reason	Revised by	Approved By
0.1	26/05/2020	Initial Draft - TOC	Stelios Andreadis (CERTH)	Ilias Gialampoukidis (CERTH)
0.2	15/06/2020	Contributions (Section 2)	Stelios Andreadis (CERTH)	Anastasia Moumtzidou (CERTH)
0.3	17/06/2020	Contributions (Section 3)	Petteri Karsisto (FMI)	Ari Karppinen (FMI)
0.4	23/06/2020	Final contributions (Section 2)	Anastasia Moumtzidou (CERTH)	Ilias Gialampoukidis (CERTH)
0.5	26/06/2020	Internal review	Stefano Casadio (SERCO) Gabriella Scarpino (SERCO)	Gabriella Scarpino (SERCO)
1.0	30/06/2020	Updated document after review, ready for submission	Stelios Andreadis (CERTH) Anastasia Moumtzidou (CERTH) Petteri Karsisto (FMI)	Ilias Gialampoukidis (CERTH) Stefanos Vrochidis (CERTH)

Author list

Organization	Name	Contact Information
CERTH	Stelios Andreadis	andreadisst@iti.gr
CERTH	Anastasia Moumtzidou	moumtzid@iti.gr
CERTH	Ilias Gialampoukidis	heliasgj@iti.gr
CERTH	Stefanos Vrochidis	stefanos@iti.gr
FMI	Petteri Karsisto	petteri.karsisto@fmi.fi
FMI	Ari Karppinen	Ari.Karppinen@fmi.fi

Executive Summary

In this deliverable we report the work that has been done for Work Package 3 about EO and non-EO data acquisition. Even though the deliverable focuses on task T3.2 and the Social Media Crawler, it also includes the progress on task T3.3 and the latest developments regarding the Weather Data Management Module. Progress on the EO data acquisition (T3.1) will be reported in the new version of D3.1.

The key contributions and achievements that are presented in this deliverable are:

1. A complete framework that collects social media from Twitter based on predefined search criteria, analyses them (in order to verify them, localise them, extract concepts from their images and detect nudity in their images) and stores them.
2. The creation of a manually annotated dataset of tweets labelled as relevant or irrelevant to some of the use cases.
3. A text classification methodology to filter out irrelevant tweets and the evaluation of the method.
4. The user interfaces that have been developed for collecting annotation and displaying the collected and analysed tweets.
5. A big collection of millions of tweets in regards with the EOPEN use cases.
6. An improved Weather Data Management Module that includes new datasets.

Abbreviations and Acronyms

API	Application Programming Interface
ARD	Association of public service broadcasters in Germany
AWS	Automatic Weather Station
BMCO	Broadcast Mobile Convergence
DAML	DARPA Agent Markup Language
ECMWF	European Centre for Medium-Range Weather Forecasts
EO	Earth Observation
EPSG	European Petroleum Survey Group
GRIB2	General Regularly-distributed Information in Binary form (file format)
HIRLAM	High Resolution Limited Area Model
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
NCEP	National Centers for Environmental Prediction
PHP	PHP: Hypertext Preprocessor
PUC	Pilot Use Case
RDF	Resource Description Framework
SMOS	Soil Moisture Ocean Salinity Earth Explorer mission
SMOS L3FT	SMOS Level 3 Freeze/Thaw service
SPARQL	Simple Protocol and RDF Query Language
STA/LTA	Short Time Average over Long Time Average
URL	Uniform Resource Locator
WFS	Web Feature Service
WGS	World Geodetic System
WMO	World Meteorological Organization

Table of Contents

1 INTRODUCTION

As social media data have been proven to carry valuable information with regards to crisis events (Xu et al., 2016; Reuter et al., 2018), natural disasters (Kryvasheyeu et al., 2016; Wang et al., 2018), news (Lee & Ma, 2012) and general topics (Aiello et al., 2013), they constitute one of the core sources of non-EO data in the EOPEN framework. Specifically, the EOPEN Social Media Crawler is responsible for integrating Twitter data into the system and is the main subject of this deliverable, covering all Section 2.

In addition, this document includes in Section 3 the progress on Task 3.3 "Meteorological and climatological data acquisition", as the work related to this task was not finished at the time of D3.2 delivery (M26). On the other hand, progress on the EO data acquisition is excluded here and will be reported in the resubmitted D3.1.

In more detail, Section 2 starts with an overview of the Social Media Crawler framework (2.1), describing all the steps from defining the criteria of crawling to real-time collection and from analyzing tweets to their storage and usage. The following subsection (2.2) focuses on how the crawling can be achieved with an API provided by Twitter and what are the search criteria specified by the end users. A description of the analysis stages follows (2.3), including information for the verification, localization, concept extraction, and text classification techniques that are applied. For the latter, the reader is also provided with related work, a presentation of the proposed model, the creation of training datasets, and quantitative experimental evaluation (2.4). Next, the visualization of the collected social media in the EOPEN system is described and illustrated with screenshots (2.5), while the status of the collection after more than the two thirds of the project's lifetime is presented not only with numbers, but with visual analytics, too (2.6). Lastly, the connection of crawled tweets with other EOPEN modules (2.7) and other H2020 EO projects in the frame of synergies (2.8) is reported.

Following, Section 3 presents a short reminder on the Weather Data Management Module in Section 3.1, and recent developments are reported in Section 3.2.

2 SOCIAL MEDIA CRAWLER

2.1 Framework overview

The acquisition of social media data towards a system that involves both EO and non-EO data is achieved through the Social Media Crawler. The module is responsible for making the appropriate queries, collecting data in real time, and analysing them either to improve the quality of incoming information or to obtain additional knowledge.

For this crowdsourcing task we have selected the well-known platform of Twitter. By the end of 2019, Twitter has reached 330 million active users[1], so its high popularity promises rich and up-to-date content. Our choice is further supported by the fact that the platform provides a free API for streaming real-time tweets, i.e. the Twitter Streaming API[2].

The complete workflow of the Social Media Crawler is illustrated in Figure 1 and will be shortly described here, while the details of the various stages will be given in the following sections.

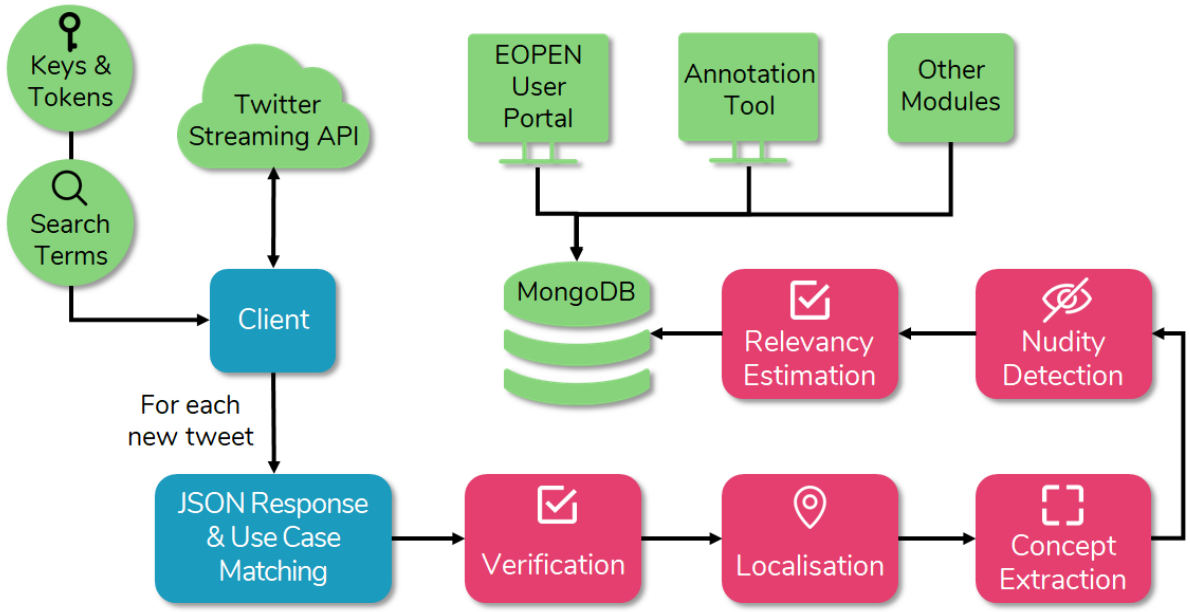

Figure 1: The complete workflow of Social Media Crawler

The core component of the module is the Client, which establishes a single connection to the Twitter Streaming API, using the necessary keys and tokens, and then continuously receives new tweets that satisfy predefined search criteria (more in Section 2.2). Every collected post comes in the form of JSON and a five-step analysis is performed:

1. Verification concerns the estimation of the probability that the tweet carries fake news.
2. Localisation is the detection of locations mentioned in the text and the association to coordinates, in order to geotag the tweet.

[1] https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/

[2] https://developer.twitter.com/en/docs/tweets/filter-realtime/api-reference/post-statuses-filter

3. Concept extraction is the identification of visual concepts that appear in the image of the tweet, if it has one.
4. Nudity detection refers to the assessment of whether the image of the tweet (again, if existing) has inappropriate content.
5. Relevancy estimation involves text classification to mark the tweet as relevant or irrelevant to the examined topics.

The outcomes of the different analysis techniques (described in Sections 2.3 and 2.4) are added as complementary attributes to the original JSON of the tweet, which is finally stored to a secure MongoDB database. External applications, such as the Social Media Dashboard in the EOPEN User Portal (Section 2.5), the Annotation Tool (Section 2.4.3), and other modules that use social media as input (Section 2.7), search directly in the database and fetch the tweets they need.

2.2 **Collection of social media data**

Amongst a multitude of available Twitter API endpoints, the Streaming API can be considered the most suitable for our task, since it allows access to Twitter's global stream of data and retrieves public tweets almost at the instant they are posted. To be able to use the API, first a Twitter account is needed and then an application for a developer account, where a description of the planned usage is mandatory. When the request is approved and the developer account is created, the following credentials are obtained: *Consumer Key*, *Consumer Secret*, *Token*, and *Token Secret*. All four are needed to connect successfully.

In order to specify what tweets should be retrieved from the whole stream, the API offers three filtering options:

1. Follow: A list of user IDs, indicating the users to return statuses.
2. Track: A list of phrases of keywords to appear in the messages. Spaces between words of a phrase are considered as an AND operator, e.g. "crop losses" means that the status should include both words.
3. Locations: A set of bounding boxes to track. Bounding box is expressed as two pairs of longitude and latitude, one for the southwest corner of the box and one for the northeast.

The combination of multiple filtering options is feasible and the free access level allows up to 400 keywords, 5,000 user ids and 25 location boxes.

The definition of the filters, i.e. the search criteria to collect social media posts, has been accomplished in close collaboration with the PUC leaders, so that they are in accordance with the examined use cases. For PUC1, AAWA suggested Italian keywords relevant to floods in the region of north-eastern Italy and the English word "flooding". For PUC2, KU provided Korean keywords and accounts in the region of South Korea as well as English keywords and international accounts, most of them referring to food security and crop monitoring. Lastly, for PUC3, FMI proposed Finnish words about snow and the English word "snow" in messages from Finland. The detailed filters can be seen in Table 1.

Table 1: The complete filtering options for each PUC and language, grouped by keywords, accounts, and bounding boxes

	PUC1		PUC2		PUC3	
	Italian	English	Korean	English	Finnish	English
Keywords	• alluvione • alluvionevicenza • allagamento • bacchiglione • fiumepiena • allertameteo • sottopassoallagato • allertameteovicenza • esondazione • livellofiume	• flooding	• 식량안보 • 작물생산량 • 곡물생산량 • 생산량 • 과잉생산 • 쌀 • 홍수 • 돌발홍수 • 작물재배 • 농작물재배 • 가뭄 • 농부 • 농민 • 수확 • 추수 • 씨뿌리기 • 파종 • 벌아 • 콩 • 대두	• food security • crop yield • overproduction • crop losses • rice • rice paddy flooding • flash floods • crop disaster • drought • farmers • harvest • seeding • soybean	• lumi • lumihanki • lumet • lunta • luntako • lumeen • lumimyrsky • lumimyräkkä	• snow
Accounts			• @naasstory • @love_rda • @kma_skylove • @KOSTATIN • @mevpr • @mafrakorea	• @FAO • @FSCluster • @WFP • @FAOnews • @FAOEmergencies • @FAOAsiaPacific • @Food_Security		
Bounding Boxes	SW = 45.0, 10.2 NE = 47.3, 14.1 (North-eastern Italy)		SW = 33.85, 125.33 NE = 38.6, 130.0 (South Korea)			SW = 59.45, 19.08 NE = 70.09, 31.58 (Finland)

To consume from the Twitter Streaming API, we adopt the open-source Hosebird Client (hbc)[3], a Java HTTP client that establishes an open connection to the API and continuously listens for new messages. The inputs to the client are the developer credentials and the filtering parameters, while the output is every new tweet as a JSON string. It should be noted here that one developer account can be linked only to one open connection.

The provided JSON format of the received tweets is suitable because (a) it offers flexibility to add more attributes, e.g. outcomes of the analysis, and (b) it is indicated for MongoDB installations. Even though the complete structure is stored, only a subset of the attributes has been proved useful, which can be named "fundamental". The fundamental attributes are:

- *id*: A long number with the unique identifier of the tweet.
- *id_str*: The same identifier as above, but as a string.
- *text*: The message of the tweet.
- *timestamp_ms*: The date and time when the tweet has been published, in Unix time.
- *geo.coordinates*: The location where the tweet has been posted from.
- *lang*: A code that refers to the language of the message.
- *entities.media.media_url_https*: The secure URL of the image of the tweet.
- *retweeted_status*: An object that contains the original tweet being retweeted; based on that we manually add the Boolean attribute *is_retweeted_status* to assist indexing.
- *extended_tweet.full_text*: The complete, untruncated tweet message in case it is longer than 140 characters.
- *extended_tweet.entities.media.media_url_https*: The secure URL of the image of the tweet, in case the message is longer than 140 characters.

Examples of collected tweets are visualised in Figure 2 (regular tweet), Figure 3 (retweet), and Figure 4 (extended tweet), where non-fundamental attributes are omitted.

We note that these attributes are not always available for every tweet. Geographical coordinates need to be enabled by the Twitter user, an image may not necessarily be part of a tweet, etc.

Due to the fact that there is a single connection to the API and the retrieved tweets could satisfy any of the defined filters, a reverse search is performed for each received tweet to detect which use case and language it relates to. In this way, it is possible to store the tweets in different collections of the database, enabling separate access to PUC-related content.

[3] https://github.com/twitter/hbc

Figure 2: Fundamental attributes of a regular tweet

Figure 3: Fundamental attributes of a retweet

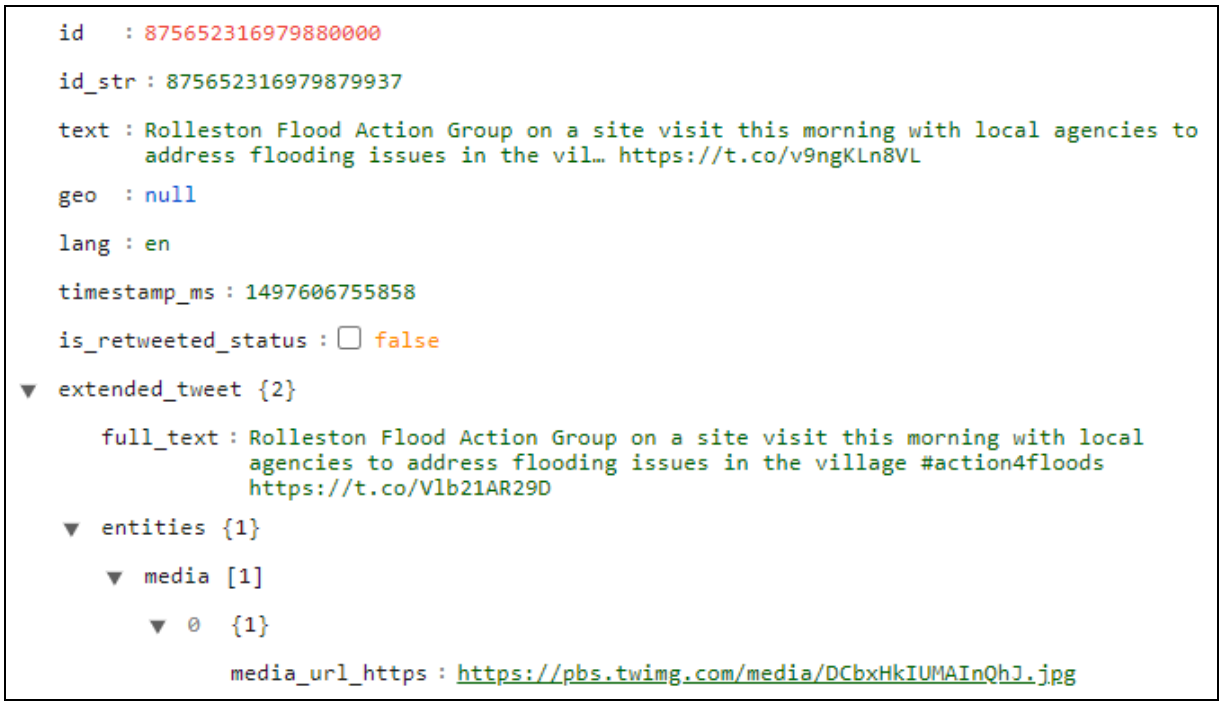

Figure 4: Fundamental attributes of an extended tweet

2.3 **Analysis of social media data**

Before storing the new tweet to its respective collection in the database, a set of analysis techniques is performed in order to (a) estimate the quality of the incoming information, and (b) retrieve further knowledge out of the original post. The results are stored as JSON pairs of attributes-values, enhancing the existing structure provided by Twitter.

2.3.1 **Verification**

In order to handle the problem of fake news and online misinformation, an automatic verification method is applied in the analysis of the collected tweets. Since the task of implementing such a module is outside the scope of the EOPEN project, a mature solution, which has been developed by CERTH, is being adopted and re-used.

This solution relies on two independent classification models built on the training data using two different sets of features: tweet-based and user-based. Tweet-based features can be (a) text-based (e.g., number of uppercase characters), (b) language-specific (e.g., number of detected slang words), (c) twitter-specific (e.g., number of retweets, number of hashtags), and (d) link-based (e.g., existence of external URLs). On the other hand, user-based features can be (a) user-specific (e.g., number of user's followers) and (b) link-based (e.g., existence of a ULR in the Twitter profile description). These types are mentioned here, because we consider it interesting to show some characteristics that are common among fake tweets.

Following the feature extraction, model bagging is used to produce more reliable predictions based on classifiers from each feature set. The classification algorithms are Logistic Regression and Random Forests of 100 trees. In addition, at prediction time, an agreement-based retraining strategy is applied, which combines the outputs of the two bags of models in a semi-supervised learning manner. For more details on the verification technique, the reader is referred to (Boididou et al., 2017) and (Boididou et al., 2018).

The described framework has been also implemented as a standalone service, which receives as an input the original JSON of a tweet and responds with a Boolean label that defines whether the tweet is real or fake and a percentage of confidence for the classification result. The service is called for each crawled tweet and the output is added to the tweet's JSON as in the following example:

```
{
     "verification" :
     {
           "predicted" : false /*Boolean*/,
           "confidence" : 0.56 /*Double*/
     }
}
```

2.3.2 **Localisation**

Twitter posts often lack geolocation information (see also Section 2.6) which makes it hard to associate them to other geo-referenced data. To overcome this limitation, we have developed a localisation module under Work Package 5, which aims to detect any locations mentioned inside the text of the tweet and link them to coordinates in the World Geodetic System (WGS) 84, also known as EPSG 4326.

The solution properly pre-processes the Twitter text and feeds it to a Deep Neural Network, i.e., a Long Short-Term Memory model. Named Entity Recognition labels are assigned to each qualified word of the sentence and the identified locations are used as inputs in queries to OpenStreetMap API[4] that is able to connect them with open geodata and return the exact coordinates. More information and an evaluation of the methodology can be found in deliverable D5.1.

The localisation module has been implemented as a standalone service that receives a string (e.g., a word or a phrase identified as location) and returns the exact coordinates (latitude, longitude) as well as the complete name of the place, as stored in the OpenStreeMap API. An example of how the results are appended to the JSON structure of the tweet can be seen here:

```
{
     "location_in_text" : "Pepper Farm" /*String*/,
     "location_fullname" : "Pepper Farm, Phu Quoc, Phu Quoc District,
Vietnam" /*String*/,
     "geometry" :
     {
           "type" : "Point" /*String*/,
           "coordinates" :
           [
                 104.02020005 /*Double*/,
                 10.24938555 /*Double*/
           ]
     }
}
```

The above structure also complies with the GeoJSON standard format, thus allowing 2dspheres indexing in the MongoDB to support geospatial queries.

[4] https://wiki.openstreetmap.org/wiki/API

2.3.3 **Concept extraction**

Another part of the analysis that concerns knowledge enhancement is the extraction of high-level content, i.e., concepts, from visual low-level information deriving from Twitter images. These concepts can be used as a way to examine whether images are relevant to the topics of interest (e.g., photos of floods or snow) or to retrieve similar content (as, for example, in similarity fusion).

The implementation is based on a framework that uses Caffe (Jia et al., 2014) and involves the use of a fine-tuned 22-layer GoogleNet network on 345 SIN TRECVID concepts (Smeaton et al., 2009). More information about this framework is included in deliverable D4.1.

A standalone service is called for every new tweet that contains an image, with the URL of the image as input and responds with a list of extracted concepts. The concepts are added to the JSON of the tweet as a single string, being separated with spaces. An example follows:

```
{
     "image_concepts" : "Snow Ski Outdoor Skier Trees Mountain Sky"
/*String*/
}
```

2.3.4 **Nudity detection**

The fact that the Twitter platform permits adult material to be posted leads to collecting a lot of inappropriate content. To protect the users of the EOPEN system from viewing pornographic photos, we have utilised a module that automatically estimates whether an image contains nudity or not.

The module is based on a two-step procedure that involves the use of deep neural network and a linear regression model. As far as the deep neural network is concerned, it is used for creating the feature vector representation of the image, while the linear regression model is used for the binary classification of the image to the “nude” class. Specifically, a 22-layer GoogleNet network (Szegedy, 2015) was used that was trained on 5055 ImageNet concepts (Pittaras, 2017), which are a subset of the ImageNet “fall” 2011 dataset[5]. Thus, the output of the trained network that is a fully connected layer had dimension equal to 5055. Then, the linear regression model that is used considers as input the DCNN-based feature vector and classifies each image to one of the two classes (i.e. nudity or non-nudity) and provides a probability for the two classes.

Again, this solution is used as a standalone service that receives the URL of an image and returns the classification decision in a Boolean value (true means it contains nudity). The outcome is appended to the original JSON like this:

```
{
     "nudity" : false /*Boolean*/
}
```

A fifth and final step of the tweet analysis concerns the estimation of its relevance to the use case it has been collected for. Since a lot effort has been made in this subtask, a separate section is dedicated, i.e. the following Section 2.4.

[5] http://academictorrents.com/details/564a77c1e1119da199ff32622a1609431b9f1c47

2.4 **Relevancy estimation with text classification**

Text classification involves the development of text classifiers that assign text to a set of predefined categories. For EOPEN, the categories are drawn directly from the EOPEN use cases, i.e. Italian floods and Finnish snow. Thus, each tweet text collected using the Twitter API (as mentioned in Section 2.2) is examined by taking into account the text information and is either considered related to flood/snow or irrelevant. It should be noted that although all the tweets are retrieved by using specific criteria in the Twitter API, it is common that the use of certain words (e.g. “flood”) can have a different meaning to the desired one (e.g. “my timeline is flooded with photos”), which results in obtaining content from irrelevant tweets. Thus, our aim is to remove tweets that even though they contain use-case-related keywords, are not relevant to it. In this section, we begin with an overview of state-of-the-art methods for text classification, then we present the proposed framework and an evaluation of different methods, and finally we draw some conclusions.

2.4.1 **Related work**

Text classification involves the following steps:

1. Document collection that includes collecting data stored in several formats such as doc, html, or simple text.
2. Preprocessing, which involves several steps including: a) converting the original text data in a data-mining-ready structure; b) tokenization, where each document is partitioned into a list of tokens; c) stop word removal, which involves the removal of frequently occurring words (e.g. “and”, “the”); d) word stemming, which reduces words to their root form.
3. Text representation (Yan, 2009), which models documents and transforms them into numeric vectors. There are several methods for text representation. Among the most common ones is the Vector Space Model (VSM) where documents are represented by vectors of words. Bag of Words model (BOW) is a common VSM that uses all words appeared in the given document as the index of the document vectors. BOW supports different term weighting schemas, including a) the Boolean model, where binary vectors represent documents; b) the Term Frequency model (TF) that uses the frequency of the terms; c) the Term Frequency Inversed Document Frequency (TFIDF) model, which uses real values that capture the term distribution among documents to weight terms in each document vector. However, all the above representations cannot capture polysemy and synonymity as well as the semantics of the documents. A more advanced text representation strategy that was proposed includes the N-gram statistical language models that try to capture the term correlation within document. The main problem of this technique is the exponentially increase of the data dimension which limits its application. The Latent Semantic Indexing (LSI) was proposed to reduce the polysemy and synonym problems. Later, Mikolov et al. (2013) proposed the word2vec approach that involves building novel architectures and models for producing word embeddings (i.e. representation of words from a given vocabulary as vectors in a low-dimensional space) that are based on deep neural networks (NN). Two types of models were proposed, namely the Continuous Bag-of-Words (CBOW) and the Skip-gram models. Both models are trained first on a large corpus and consider the neighboring words in a sentence; however, in the CBOW the NN model tries to predict a word given the context

of the word, while in the Skip-gram the NN model tries to predict the context of a word given the word. The notion behind word2vec can be extended to sentences and documents where the model learns features for representing sentences (SentenceToVec) or documents (Doc2Vec). Another approach similar to word2vec is GloVe (Pennington, et al. 2014). GloVe is also an unsupervised learning algorithm that obtains vector representations for words. In GloVe, training is performed on aggregated global word-word co-occurrence statistics from a corpus. Finally, another more recent approach is the Bidirectional Encoder Representation from Transformers (BERT) algorithm (Devlin, et al. 2018), which involves an attention mechanism that learns contextual relations between words in a text. BERT's goal is to generate a language model, and the used mechanism reads the entire sequence of words at once, contrary to directional models (e.g. n-gram LMs (Rosenfeld, 2020), and neural network LMs (Mikolov et al. 2010; Bengio et al. 2003)) that read the text input sequentially. Therefore, it is considered bidirectional or non-directional. This characteristic allows the model to learn the context of a word based on its surroundings.

4. Feature selection methods (Aggarwal, 2012; Chandrashekar, 2014) that aim at reducing the dimensionality of the dataset by removing features that are considered irrelevant for the classification and thus add noise. There are two main categories of feature selection methods: the filtering and the wrapper methods. Filtering techniques rank the features, keep the highly ranked features and then apply on them the predictor. On the other hand, in wrapper techniques the predictor is wrapped on a search algorithm which will find a subset that gives the best performance. Document Frequency (DF), Information Gain (IG), and Mutual Information (MI) are typical filtering methods, while Sequential Forward Selection (SFS), Sequential Backward Selection (SBS) and Neural Networks are examples of wrapper methods.
5. Classification Algorithms, which are used to model classes and label text. There are several methods used to classify text such as Support Vector Machine, Naive Bayes Classifier, Logistic Regression and Decision Trees.

2.4.2 **The EOPEN methodology**

In order to find the classifier that performs best for two EOPEN use cases, i.e. PUC3 about snow in Finnish and PUC1 about floods in Italian, several text representation and classification algorithms were evaluated. It should be noted that a PUC2 classifier has not been developed, due to the lack of an annotated dataset about food security and due to the significant effort required in order to handle the Korean language. The approach we followed is the following:

1. We collect short text messages from Twitter, as already described in Section 2.2.
2. We preprocess the collected text by removing a) URLs; b) emojis; c) mentions '@'; d) punctuation and all non-characters; and e) stop words. Camel case words are also split since usually they are related to the content of the Twitter text and finally we do word stemming. It should be noted that word stemming is applied only for Italian tweets as for Finnish tweets the stemming realized from the Porter Stemmer does not work satisfactorily and alters the word.
3. Then text representation is applied and the methods evaluated are: BOW using Term Frequency (TF), BOW using TFIDF, word2vec and BERT. Various experiments were realized for different feature length and n-gram values (i.e. n-gram = 1 or 2) for the

BOW representation methods, and different vector dimensions and words window for the word2vec method.

4. Feature selection is not realized.

Finally, we serve the text feature vector as input to a classifier (i.e. SVM, Naïve Bayes, Logistic Regression or Random Forests) which is tuned in order to achieve maximum performance.

2.4.3 **Human annotation**

As it has been mentioned above, supervised classification requires training with annotated data. Since the examined use cases are very specific, i.e. flood monitoring in Italian language and snow coverage in Finnish language, there is lack of annotated datasets. Therefore, manual annotation is required.

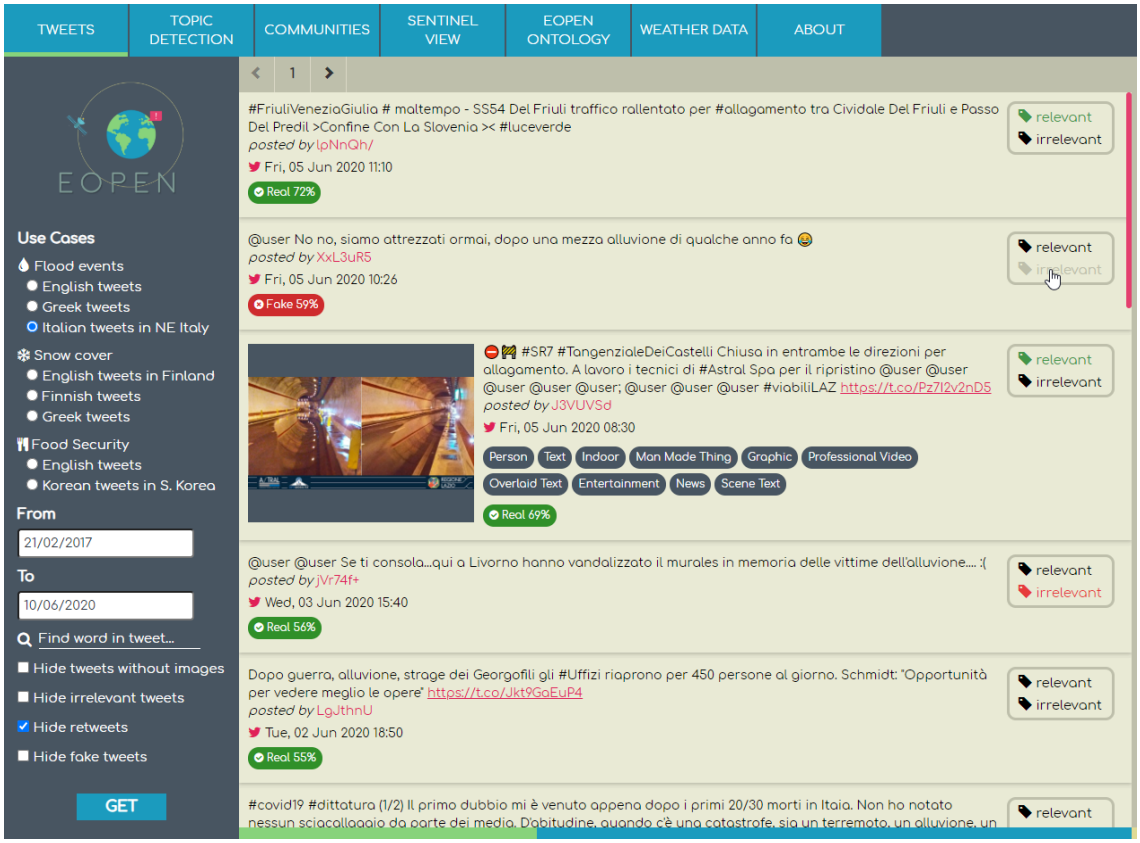

Figure 5: Screenshot of the annotation tool

Manual annotation involves human effort to label a number of tweets as relevant or irrelevant. For this task, we have addressed the PUC1 and PUC3 leaders, i.e. AAWA and FMI respectively, who are not only familiar with the languages but also have a clearer perception of what tweets can be considered relevant to the use cases.

In order to assist AAWA and FMI in manual labelling, an annotation tool (Figure 5) has been developed that allows users to quickly mark tweets as relevant/irrelevant. On the left, there are options to select use case, time period and additional filters (their description is omitted here because they are unrelated to annotation and are later reported in Section 2.5). After clicking the "GET" button, tweets are fetched and displayed in the main view pane of the tool (Twitter details and analysis information are again left for Section 2.5). On the right of

each tweet there is a box where the user can click either “relevant” or “irrelevant”, annotating in this way the post.

The assigned label is added as a Boolean attribute to the JSON structure of the tweet, as it is stored in the database, and is considered when creating the training dataset.

```
{
    "relevant" : true /*Boolean*/
}
```

AAWA and FMI were assigned with the task of manual annotation, since both share expertise in the domains of PUC1 and PUC3 respectively. For PUC1, tweets were annotated as relevant when they referred to floods in the area of AAWA competence, i.e. the Eastern Alps partition of North-eastern Italy (Figure 6), or to weather forecasts and recent data/instruments that may be useful to predict rains. On the other hand, tweets were considered irrelevant when they referred to flooding incidents distant from AAWA competence, to historical floods or cases unrelated to floods. For PUC3, the criterion to annotate a tweet as relevant was to refer to snow (snow weather, snow forecast, snowfall) in the area of Finland.

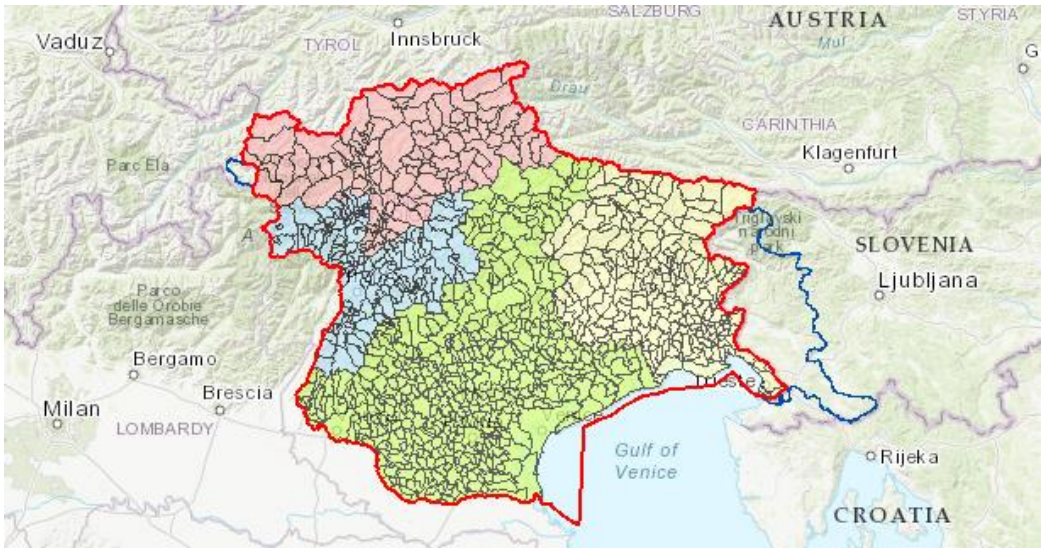

Figure 6: AAWA competence – Eastern Alps partition of North-East Italy

The results after the annotation can be viewed in Table 2 as well as Figure 7 and Figure 8. Apart from the value of this outcome towards having an annotated dataset, it also shows that a large percentage of the collected tweets are not related to the examined use cases, even though they satisfy the search criteria, and confirms the need for including automatic text classification.

Table 2: Annotation results for PUC1 and PUC3

	Flood monitoring in Italian	Snow coverage in Finnish
Relevant	6,584 (17%)	3,833 (50%)
Irrelevant	31,305 (83%)	3,877 (50%)
Total annotated	37,889	7,710

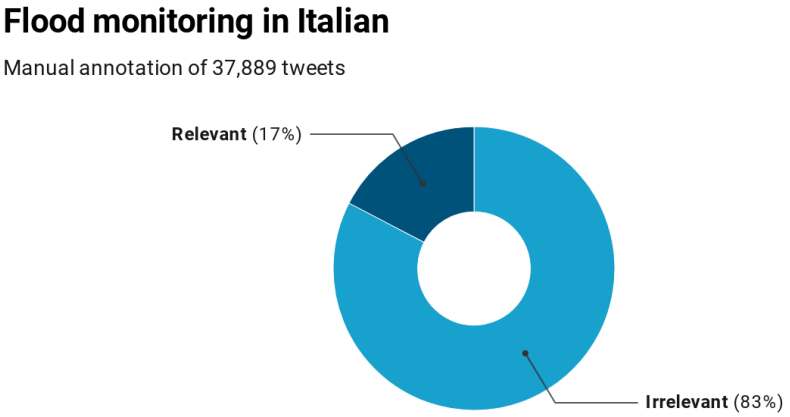

Figure 7: A pie chart with the percentage of relevant versus irrelevant PUC1 tweets

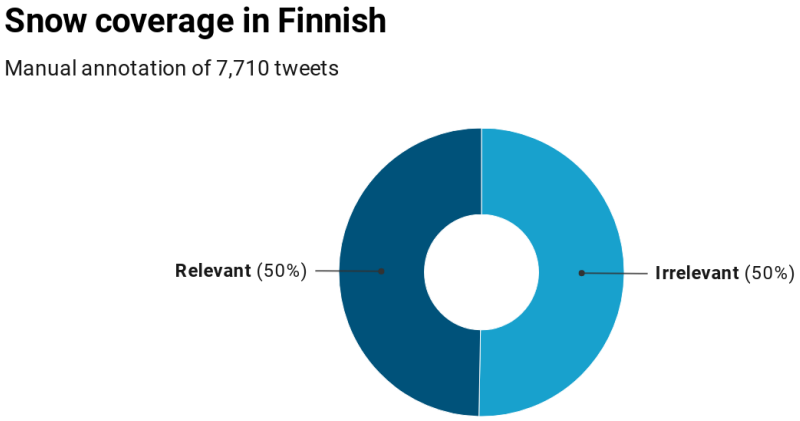

Figure 8: A pie chart with the percentage of relevant versus irrelevant PUC3 tweets

Both annotated datasets were split and two thirds (2/3) of them was used for training and the rest (1/3) for testing the different approaches. However, the dataset needed to be balanced (i.e. have similar number of positive and negative tweets) in order to obtain more accurate results and not to favor the class with higher representability (i.e. the negative tweets). Thus, eventually, the evaluation involved 3,275 tweets for PUC1 and 1,917 tweets for PUC3 testing.

2.4.4 **Evaluation**

In order to evaluate the quality of the classification system, we consider the two afore-mentioned datasets and the following metrics: precision, recall, and F-score that are commonly used in classification problems.

The definition of these measures within the context of a classification problem can be achieved by using the values found in the Confusion Matrix (Figure 9), which is a performance measurement for machine learning classification. Each row of the matrix represents the instances in a predicted class, while each column represents the instances in an actual class. TP or True Positive depicts the number of instances that were considered positive and were actually positive. TN or True Negative depicts the number of

instances that were considered negative and were actually negative. FP of False Positive depicts the number of instances that were considered positive and but were actually negative. Finally, FN or False Negative depicts the number of instances that were considered negative and but were actually positive. Using the above values we can define precision, recall and F-score as follows:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F - score = 2\frac{Precision \cdot Recall}{Precision + Recall}$$

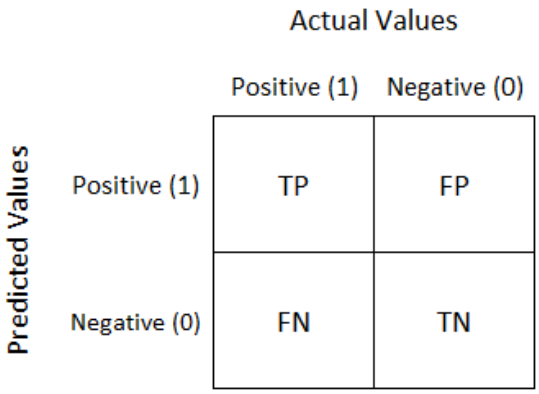

Figure 9: Confusion Matrix

Therefore, precision shows how many out of the classes predicted as positive are truly positive, while recall shows how many out of the truly positive classes have been predicted as positive. F-score (or F-measure) is a combination of precision and recall and is used to facilitate the comparison of models performance that have low precision and high recall or vice versa. These metrics are calculated in every run in order to decide the best performing classification method.

In the BoW and word2vec cases, SVM, Naïve Bayes and Random Forests classifiers are tested for a set of parameters, while for the case of BERT only linear regression was considered. The parameters that were tuned using grid search in order to find the best performing approach can be found in Table 3. For the remaining parameters, default values are used.

Table 3: Classifier parameters

Classifiers	Parameters
SVM	Penalty parameter: 0.01, 0.1, 1.0, 2.0, 3.0, 4.0, 5.0 Kernel type: rbf, poly
Naïve Bayes	Additive smoothing parameter: 0.01, 0.1, 1.0
Random Forests	Number of trees in the forest: 10, 50, 100, 200, 500, 1000 Number of features used for best split: auto, log2, sqrt, None
Linear Regression	Inverse of regularization strength parameter (C): 0.0001 – 100 (step 20)

Table 4 and Table 5 contain the best results of the Italian Floods and Finnish Snow datasets for all the different representation methods. Regarding the BOW representation methods (i.e. TF and TFIDF) different *n-gram* values and *min_df* values are considered during text vectorization. The *min_df* value affects the size of the feature length since it sets the frequency threshold and thus the terms with lower frequency are ignored while building the vocabulary. Specifically, *n-gram* parameter can be either 1 or 2, while *min_df* can be 0.0001, 0.001, 0.002, 0.003, 0.004, 0.005, 0.006, 0.007, 0.008, 0.009, 0.01 or 0.02. The table includes the best performing models for n-gram equal to 1 and 2. The same applies for the TFIDF. Regarding the word2vec methodology, several runs were realized for different vector dimension (i.e. 100, 200, 300, 400, and 500), words window (i.e. 2, 3) and training algorithm (i.e. 0, 1) parameters. The table includes the best performing models for *words_window* =2 and 3. The size of the corpus used is ~118,500 records for the Italian Floods dataset and ~65,500 records for the Finnish Snow dataset and it includes tweets (either relevant or irrelevant) that were collected by the Social Media Crawler when discovering data for the respective use cases. Finally, as far as the BERT representation is concerned in order to capture the text representation of the whole tweet, we used existing pre-trained models. Specifically, for Finnish we used the '*bert-base-finnish-cased-v1*' pre-trained model and for the Italian the '*bert-base-multilingual-cased*' model. The size of the feature vector used is by default 768.

Table 4: Evaluation of different representation and classification methods for the Italian Floods dataset

Parameter	Text input	Classifier	Precision	Recall	F-score
TF representation method					
n-gram = 1	Without stop words	Random Forest	0,6667	0,3618	0,4691
	Without stop words & with stemming	Random Forest	0,7500	0,5234	0,6165
n-gram = 2	Without stop words	Random Forest	0,6656	0,3585	0,4660
	Without stop words & with stemming	Random Forest	0,7605	0,5155	0,6145
TF-IDF representation method					
n-gram = 1	Without stop words	Random Forest	0,7462	0,3325	0,4600
	Without stop words & with stemming	Random Forest	0,7500	0,5198	0,6140
n-gram = 2	Without stop words	Random Forest	0,6760	0,3442	0,4562
	Without stop words & with stemming	Random Forest	0,7628	0,5082	0,6100
word2vec representation method					
Words_window = 2	Without stop words	SVM	0,8889	0,0268	0,0520
	Without stop words & with stemming	SVM	0,5965	0,0413	0,0773
Words_window = 3	Without stop words	SVM	0,9091	0,0335	0,0646
	Without stop words & with stemming	SVM	0,6466	0,0523	0,0967
BERT representation method					
Pre-trained model = bert-base-	Without stop words	Linear Regression	0,64405	0,61817	0,63085

multilingual-cased					
Pre-trained model = bert-base-multilingual-cased	Without stop words & with stemming	Linear Regression	0,66646	0,65961	0,66302

Table 5: Evaluation of different representation and classification methods for the Finnish Snow dataset

Parameter	Text input	Classifier	Precision	Recall	F-score
TF representation method					
n-gram = 1	Without stop words	Random Forest	0,7482	0,7599	0,7540
n-gram = 2	Without stop words	Random Forest	0,7558	0,7526	0,7542
TF-IDF representation method					
n-gram = 1	Without stop words	Random Forest	0,7391	0,7808	0,7594
n-gram = 2	Without stop words	Random Forest	0,7528	0,7756	0,7640
word2vec representation method					
Words_window = 2	Without stop words	SVM	0,7118	0,7296	0,7206
Words_window = 3	Without stop words	SVM	0,7011	0,7004	0,7008
BERT representation method					
Pre-trained model = bert-base-finnish-cased-v1	Without stop words	Linear Regression	0,73278	0,73585	0,73431

The lines highlighted in green are the best performing ones. After a careful observation, we can deduce that for the "Italian Floods" dataset the best performing method is the BERT method. For the "Finnish Snow" the best performing method is the TFIDF method; however the other methods perform satisfactorily enough. Also, it should be noted that for the case of the "Italian Floods" dataset, where the possibility to apply stemming is also checked, the performance when stemming is applied is systematically better compared to when only stop words are removed. Finally, if one compares the performance of BERT method between the two datasets, it is evident that in the case of the language specific model (i.e. bert-base-finnish-cased-v1) the model performs significantly better. This implies that more tests should be realized when a dedicated model for Italian will be developed as it is expected to perform better.

2.5 **Visualisation of social media data**

The end users of the EOPEN platform should be able to view what is being collected from Twitter for the use cases of their interest, so as to gain insight into how related topics/incidents are reflected on social media. For this reason, a dedicated dashboard has been implemented and added to the EOPEN User Portal[6], namely the "Social Media" dashboard (Figure 10).

[6] https://proto2.eopen.spaceapplications.com/dashboard/

This dashboard comprises three interconnected components:

1. The “Tweets Filter” component, which provides filtering capabilities for fetching specific tweets.
2. The “Tweets List” component, which displays the results as a scrollable list of tweets.
3. The “Map” component, which displays the results as pins on an interactive map.

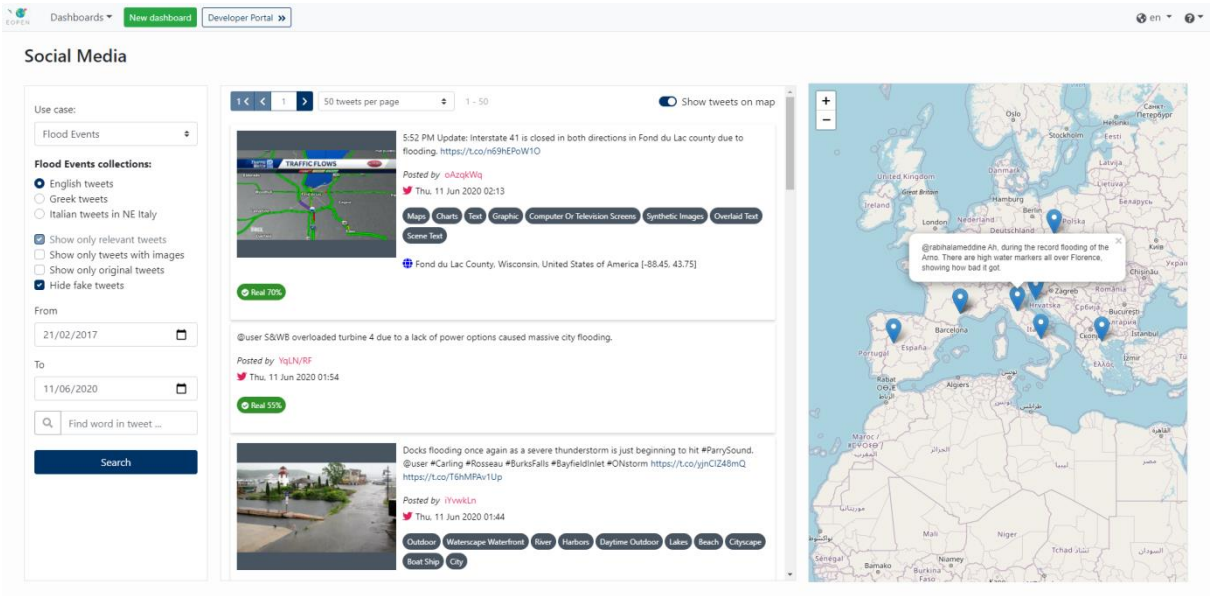

Figure 10: Screenshot of the Social Media Dashboard in the EOPEN User Portal

In the following, the description of the three components and their usage is given in detail, starting from left to right as seen in Figure 10.

The “Tweets Filter” component (Figure 11) begins with a selection box that concerns the Use Case and its values can be “Flood events” for PUC1, “Food Security” for PUC2, and “Snow Cover” for PUC3. According to the user’s selection, the second filter changes to show the available language options. Then, three optional filters follow: (a) to show only tweets that have an image, (b) to show only tweets that are original, i.e. hide retweets, and (c) to hide tweets that are estimated as fake. There is also a preselected filter to show only tweets that are estimated as relevant (by the text classification method presented in 2.4), which cannot be unselected. In addition, two dates can be defined by the user (“From” and “To”) to define the time period during which the tweets were published, while the last filter is a text box where the user can type a word and fetch tweets that contain this word. When all parameters have been set, the user can click on the “Search” button.

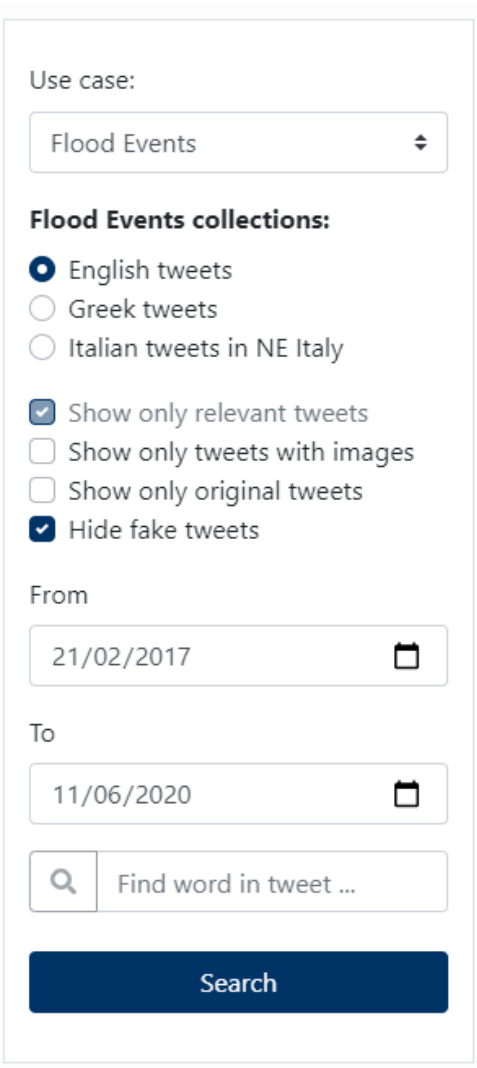

Figure 11: Screenshot of the Tweets Filter component

After the "Search" button is clicked, an API implemented in PHP fetches the tweets that match the user's options and the results are displayed in the "Tweets List" component, in a scrollable list. Since the results might be thousands, they are displayed paginated, while the number of tweets per page can be also defined by the user (50 is the default).

Each tweet in the list is visualised as a box (Figure 12) and contains both original attributes coming from the Twitter API (Section 2.2) and attributes with values deriving from the analysis (Section 2.3). If the tweet has been published along with an image, the attached image is displayed on the left of the box, while the text of the tweet on the right, with any detected external links being linkable. Next follow the name of the user account that posted the tweet, but pseudonymized to further protect the user's anonymity, and the date and time of publication. In case the tweet has an image, the extracted visual concepts are shown as labels. Moreover, if there are locations detected in the text, then the name and the coordinates of the places are displayed. Finally, the box includes the probability in percentage of being a real or fake tweet.

Figure 12: Screenshot of a single tweet in the Tweets List component

Every time the "Tweets List" is updated, the "Map" component (Figure 13) is updated as well. This component offers an alternative visualisation of the results, since they are displayed as pins on an interactive OpenStreetMap map, exploiting the coordinates of the detected locations. By clicking on a pin, a pop-up appears with the complete Twitter text. Back to the "Tweets List" component, there is a switch button to show/hide tweets on the map, while clicking on the name of a detected location forces the map to zoom at the specific coordinates.

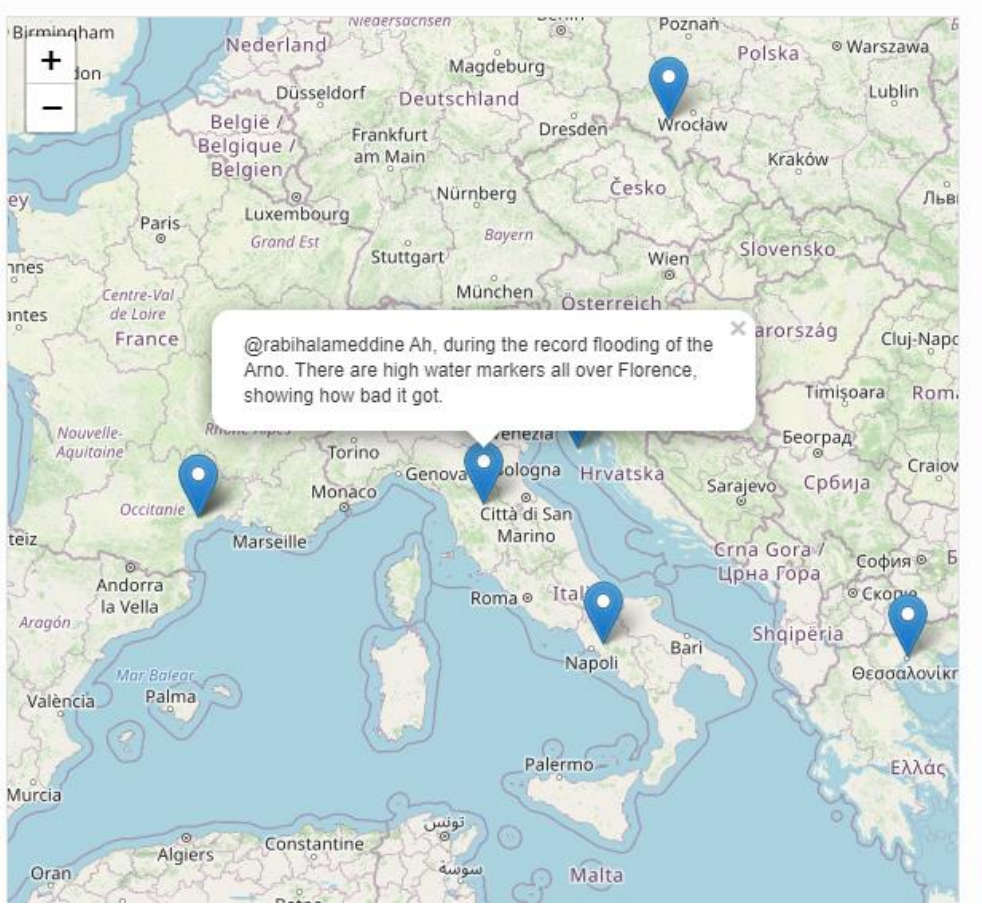

Figure 13: Screenshot of the Map component containing pins of tweets

2.6 **Status of the collection**

After almost three years of the Social Media Crawler running continuously, it is interesting to see how many Twitter posts have been collected for each use case and also to examine some of their characteristics, which could lead to some general conclusions.

The amount of the crawled tweets can be viewed in Table 6. Each row refers to a different collection, where a collection can be defined as a combination of use case and language (first two columns). The “Time period” column contains the month when the collection started and the month when the size has been last measured. For PUC1 we have exploited some Twitter data that has been collected for the H2020 beAWARE project, which shares a common use case about floods in Italy, so the collection has started earlier than for the other PUCs. The “Collected” column refers to the total number of crawled tweets, while the next four columns contain the number of tweets that (a) are retweets, (b) have an attached image, (c) have a location given by Twitter, and (d) have a location detected by the localisation module.

To give a better understanding of the collection status, some visual analytics are provided and commented.

Table 6: Number of tweets that have been collected for each use case and language during the project's lifetime

Use Case	Language	Time period	Collected	Retweets[7]	Containing image[7]	Twitter location[7]	Detected location[7]
Flood monitoring	Italian	March 2017 – May 2020[8]	118,537	68,668 (58%)	13,994 (12%)	260 (0.2%)	N/A[9]
	English	March 2017 – May 2020[8]	9,930,623	6,594,535 (66%)	998,687 (10%)	18,528 (0.2%)	942,842 (9%)
Snow coverage	Finnish	December 2017 – May 2020	65,562	17,144 (26%)	10,916 (17%)	339 (0.5%)	N/A[9]
	English	December 2017 – May 2020	84,019	44,298 (53%)	8,992 (11%)	4,164 (5%)	7,808 (9%)
Food security	Korean	December 2017 – May 2020	6,594	5,880 (89%)	936 (13%)	2 (0%)	N/A[10]
	English	December 2017 – May 2020	965,661	671,563 (70%)	104,840 (11%)	1,414 (0.1%)	165,659 (17%)

[7] Percentages are on the total number of crawled tweets per collection and are rounded to the nearest integer, except values approaching zero.

[8] The collection has started earlier in the frame of the H2020 beAWARE project.

[9] At the time of writing, localisation for the Italian and Finnish languages is still under development.

[10] Localisation will not be implemented for the Korean language.

Figure 14 shows the number of crawled tweets for each collection. It is evident that flood monitoring in English is by far the largest collection. This can be explained by the fact that the respective search criteria is the appearance of the word “flooding”, which can bring many results, since floods are frequent natural disasters worldwide. The next largest collection concerns food security in English, which is quite unexpected, considering that so many Twitter users post about crop production, crop disasters, etc. The rest of collections contain significantly fewer tweets (but still thousands), due to the fact that search is stricter in regards with language and location.

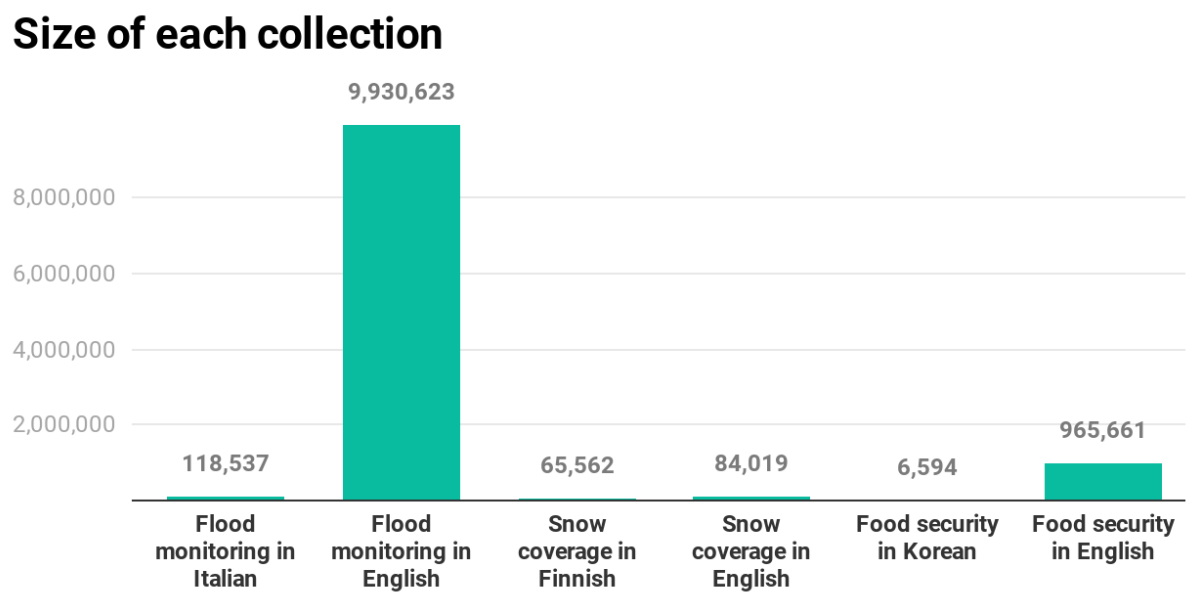

Figure 14: A bar chart with the number of crawled tweets per collection

Figure 15 illustrates the percentage of the tweets that have been posted along with an image for each collection. The values are similar for all cases and one can deduce that one out of ten tweets comes with an image. This is intriguing, given the fact that Twitter is oriented towards posting short text messages.

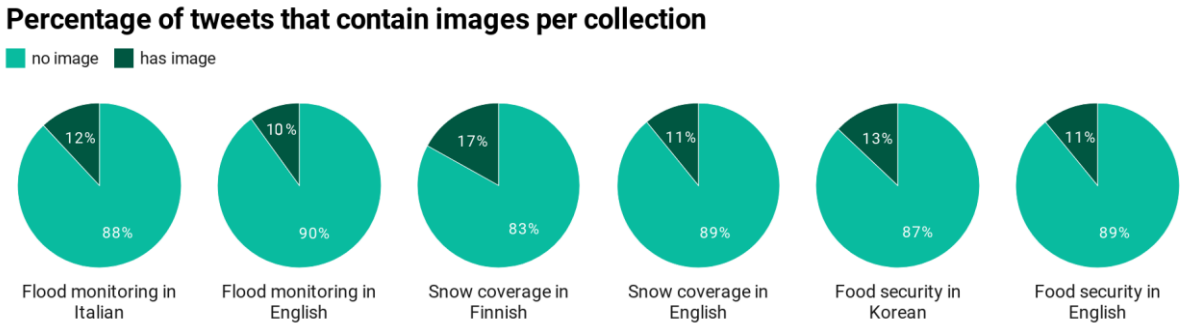

Figure 15: Pie charts with the percentage of tweets per collection that have an attached image

On the other hand, Figure 16 shows the percentage of tweets that are retweets versus the tweets that are original, again per collection. Apart from snow coverage in Finnish, in all other cases the majority of collected tweets are retweets. This is anticipated, because sharing posts of other users is very popular in Twitter, particularly for trending topics and

important events. Thus, they should not be disregarded from the information coming into the EOPEN system.

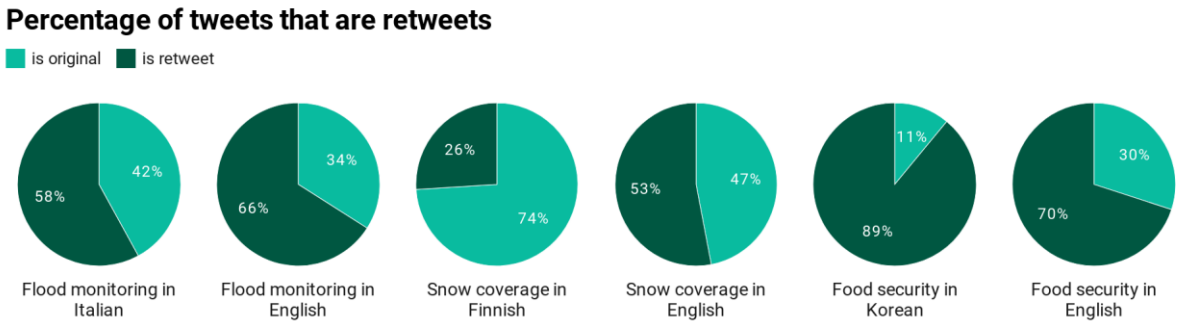

Figure 16: Pie charts with the percentage of tweets per collection that are retweets

Finally, Figure 17 presents a comparison between the percentage of tweets that contain a location originally from Twitter and tweets that have a location detected inside their text by the localisation module. It is apparent from the table that only an extremely low number of tweets have an original location (0-0.5% for most cases), while the figure demonstrates the notable increase of geotagged tweets when the localisation module is used, thus proving its high value.

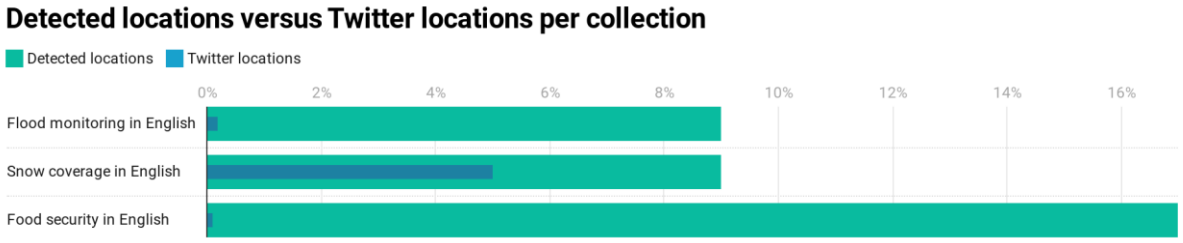

Figure 17: A bullet bar chart that compares tweets that have original location given by Twitter to tweets with a detected location in their text

2.7 **Relation to other EOPEN modules**

The Twitter data that is being collected with the Social Media Crawler serves as an input to multiple other EOPEN modules.

The Event Detection module, which targets to identify potential events based on non-EO data, examines the fluctuation of the number of crawled tweets per day to discover events. When an extensive rise of collected tweets is detected, a notification is produced, containing some insights on the possible event, such as the most frequent location (deriving from the localisation of the posts) and the most-mentioned keywords.

The Similarity Fusion module is able to retrieve the most similar Twitter data according to a query tweet. Retrieval can be based on different modalities, such as the text of the tweet, the visual content or visual concepts of the attached image, and spatiotemporal information of the post (date and time of publication, location), or it can be based on a late fusion of the above. In addition, under the same task, a snow depth estimation method fuses remote sensing data with snow-related tweets.

The Text Clustering module exploits the textual information of collected tweets and groups them by similarity of text, in order to capture trending topics on Twitter, while the Image Clustering module groups the Twitter images by visual similarity.

The Community Detection module aims to identify end-user communities through their relationship. In the EOPEN context, the module focuses on the Twitter accounts, which posted the collected tweets, and their "following" interactions. It can provide the pairs of connected users, the detected communities, and a list of the most influential users.

Finally, every time a location is detected inside the text of a collected tweet, the tweet is forwarded to an API that converts it to its semantic representation, i.e. RDF, and stores it into a Knowledge Base. In this way, geospatial queries made with SPARQL can also return non-EO data (e.g. retrieve tweets that have been posted in the bounding box of a given satellite image).

2.8 **Synergies with other projects**

As advised by the Project Reviewer and the Project Officer to seek collaborations with the other H2020 projects in the EO Big Data cluster (i.e., BETTER, CANDELA, OpenEO, PerceptiveSentinel) and also following the shared Hackathon[11] event in November 2019 in Frascati, Italy, several exchanges have been initiated between EOPEN and the other projects. Regarding the social media task, which is the focus of this deliverable, two synergies are being pursued: (1) one between CERTH, who is responsible for the social media monitoring in EOPEN, and Fraunhofer IAIS from the BETTER project, and (2) another one between CERTH and Deimos, again from BETTER. For both collaborations we have defined a practical exercise, so as to identify what can be used from each side and then combined in a meaningful way. Furthermore, these exercises could be the basis for shared publications in the near future.

Even though this work is still ongoing at the time of writing, a preliminary description of the exercises is presented here.

Exercise 1 (CERTH & Fraunhofer IAIS)

Fraunhofer IAIS explores innovative data analytics that can be executed on top of structured knowledge graphs resulting from semantic transformation on social media data and extracted knowledge from CERTH (e.g. detected locations, concepts and events). The objective is to demonstrate how new value can be generated by semantically processing and analysing data. Vocabularies capturing the required knowledge have been identified for specific use-cases, but are not necessary for some experiments, e.g. machine-learning algorithms.

The BETTER-EOPEN use case involves geo-located data derived from social media –English tweets about floods, collected by CERTH's Social Media Crawler and geo-tagged by CERTH's localisation module– and considers attribute-based grouping over multiple observations (machine learning, thus without assuming any other domain knowledge or information a priori). The use of purposely created libraries for the Semantic Analytics Stack (SANSA Framework) allows both geo-clustering and mapping, based on the attributes (coordinates).

[11] https://ec.europa.eu/info/events/h2020-eo-big-data-hackathon-2019-nov-07_en

In the future, the combination of social media data plus other EO data can also be jointly analysed to determine whether new patterns can be identified or discovered based on training data with additional prior assumptions.

Exercise 2 (CERTH & Deimos)

In the second exercise, CERTH's Social Media Crawler is adapted to a field of application that is not one of the existing EOPEN use cases. The scope is to detect earthquake incidents in Japan, so the Crawler is utilised simply by changing the search parameters to the keywords "earthquake" and "Japan", while the EOPEN localisation module is used as-is for the English language, in order to geotag the collected tweets. In addition, a dedicated API has been developed by CERTH, allowing the Deimos team to fetch the crawled tweets. The format of the API query and the structure of the API response are given below.

```
http://160.40.49.181:4000/tweet_provider_api?productType=<CollectionName
>&fromDate=<YYYY-MM-DDTHH:MM:SSZ>&toDate=<YYYY-MM-DDTHH:MM:SSZ>

E.g.,
http://160.40.49.181:4000/tweet_provider_api?productType=japanEarthquake
Tweets&fromDate=2020-05-18T10:00:00Z&toDate=2020-05-18T23:59:59Z
```


```
{
    "total_results": 3,
    "results":
    [
        {
            "timestamp": "Wed Apr 12 04:53:25 +0000 2017",
            "coordinates":  "35.6528 139.8394",
            "id": "1258762082683478016"
        },
        {
            "timestamp": "Wed Apr 12 05:03:25 +0000 2017",
            "coordinates":  "37.0504 140.8876",
            "id": "1258762082683478017"
        },
        {
            "timestamp": "Wed Apr 12 05:03:26 +0000 2017",
            "coordinates":  "35.6528 139.8394",
            "id": "1258762082683478018"
        }
    ]
}
```


After gaining access to the collection of tweets by using the API, Deimos is able to run a short-term-average over long-term-average (STA/LTA) algorithm (Earle et al., 2012), common in seismology when discovering seismic phases, to detect earthquake events based on the rapid increase in the frequency of tweets.

Short note: It should be mentioned here that CERTH has also shared some analysed Twitter data with the CANDELA project (but not raw Twitter data so as to comply with the Twitter data policy). Despite the fact that a concrete collaboration has not yet been defined at the time that this deliverable is produced, CERTH is in contact with many representatives of EO Big Data projects, through the Programme Committee of MULTISAT2021 https://mklab.iti.gr/multisat2021/organisation/.

3 METEOROLOGICAL DATA WRAPPER

This section presents an update to the task T3.3 Meteorological and climatological data acquisition. When deliverable D3.2 (M26) was first delivered, the work of the task T3.3 was still in progress. The purpose of this section is thus to present the work done afterwards. First, we shortly recap the D3.2 (Section 3.1) and then focus on the new developments (Section 3.2).

3.1 Wrapper overview

The Weather Data Management Module (WDMM) is designed and implemented under task T3.3. It is a collection of individual processes that can be used in workflows. Additionally, we've implemented some standalone workflows, which harvest the meteorological and climatological data from data providers' services and store the data in the EOPEN database. The EOPEN users and developers can then obtain the harvested data directly from the EOPEN database without the need to use the processes in their workflows. As of submission of D3.2, the processes were written in Python 2. However, as presented in the following section, the process implementation is updated to run under Python 3.

3.2 New developments

3.2.1 Processes and Workflows

As Python 2 reached its end-of-life in January 2020, WDMM development has changed to using Python 3. The EOPEN platform porting to Python 3 was finished in May 2020. Also, it was not required to recreate the previously created processes in Python 3 as they could still use Python 2 interpreter.

Table 7, which is an updated version of Table 3 presented in D3.2, shows the current status of the process implementation. Changed values are marked with green font color. Note that Table 7 has a new column, which shows the Python version used by the process. Also, the version numbering now reflects the version numbering used in the current iteration of EOPEN platform.

The previous HIRLAM process had a bug that caused WDMM to download an older forecast file. This was fixed in March 2020. Notably, the fix was implemented in Python 2 as the platform didn't support Python 3 then. However, we used Python's built-in features to ensure Python 3 compliancy, so that the HIRLAM process can be maintained and used in Python 3 environment.

The KMA process was extended and refactored to improve scheduled workflow executions support. As this modification work would have required rewriting large parts of the process, recreating the process in Python 3 was a logical choice.

The Copernicus Climate Data Store (CDS) and FMI Open Data WFS connectors are still under development, however they will be finalised before end of July (M33). For now, CDS is accessed through API and requires the end user to supply their own apikey. FMI Open Data WFS connector does not require any apikeys, as FMI Open Data service has discontinued their use.

Table 7: Updated status of WDMM connector processes with changes highlighted

Connector	Version	Status period	Python	Related datasets
Copernicus CDS	N/A	In progress	3.6+	PUC3_DC4
FMI Open Data WFS Connector	N/A	In progress	3.6+	PUC3_DC5, PUC3_DC7, PUC3_DC8
GlobSnow harvester	2	Public	2.7	PUC3_DC1
HIRLAM	**v1**	**Public**	**2.7 (3.6+)**	PUC1_DA13.1, PUC3_DC9
KMA	**v1**	**Public**	**3.6+**	PUC2_DB2_a
NASA POWER	3	Public	2.7	PUC2_DB2_a
SMOS L3FT harvester	2	Public	2.7	PUC3_DC2.1

3.2.2 **Use Case data need developments**

As the Use Cases have been progressing since D3.2, their data needs have also been refined during that time.

For PUC1, the HIRLAM precipitation accumulation forecasts are of great importance. To ingest the forecasts into AMICO model, the forecast files are first converted from the GRIB2 files into Ascii ArcInfo Grid files using wgrib2[12] software. Wgrib2 is developed by the NCEP Central Operations (NCO). However, the wgrib2 software uses NCEP notation, which is slightly different from ECMWF's notation, which in turn is used in the HIRLAM data files provided by FMI Open Data service. Due to this, we have investigated the HIRLAM dataset metadata in more detail and created a table that shows the corresponding parameters in NCEP's notation. This table will be discussed in Section 3.2.3.

For PUC2, we found out that the daily automatic weather station observations from Korea Meteorological Administration are not useful for Use Case's aims. As reported in D1.4 (footnote in Table 7)[13], the meteorological data was deemed spatially too sparse to be utilised in machine learning algorithms. Regardless, the previously fetched data will not be deleted from the EOPEN database, and the KMA connector will remain for future users.

For PUC3, FMI had a demonstration webinar session with stakeholders on the 3rd of June 2020. After seeing the EOPEN platform and currently available tools and datasets, the stakeholders had suggestions for additional datasets. These additional datasets would enable the stakeholders to answer more questions in their line of work; however, some of the suggestions were rather specific (e.g. data that shows ice layers within the snow cover).

[12] https://www.cpc.ncep.noaa.gov/products/wesley/wgrib2/

[13] "…Since, ground truth data, such as fertilization usage, cultivating practices, h**igh resolution meteorological or soil data, are not freely available** we will only make use of Sentinel data..." (emphasis added)

Currently, FMI is collecting these suggestions via a feedback questionnaire and looking into whether the suggested data could be integrated into EOPEN.

Finally, we have investigated two additional data sources, based on feedback from the Project Officer. First recommendation was to include ECMWF forecast datasets[14]. Out of the datasets, only WMO Essential data is freely available, including commercial reuse. This dataset includes five basic meteorological parameters: mean sea level pressure at surface level, geopotential height at 500 hPa level, air temperature at 850 hPa level, and both wind components at 850 hPa level. While this data could technology-wise be offered in EOPEN, we have ultimately decided against it. The data is of very limited use for the PUCs, and the data requires meteorological expertise to properly utilise.

Second recommendation was to include High Resolution Snow and Ice Monitoring products from Copernicus Land Services[15]. This data looks very promising; however, it is not yet available (as of June 2020). Due to this unavailability, we have chosen to not implement this data in EOPEN during the remainder of the project.

3.2.3 **Dataset progress**

Here, we list the progress for each dataset since the D3.2, with two exceptions: FMI ClimGrid and NASA POWER datasets. There are no changes for them, as the WDMM processes for these datasets were finished prior submitting D3.2.

Previously, KMA AWS data was provided for 2018. However, the data was deemed unsuitable to PUC2's purposes. This data will still be available from EOPEN database, and the KMA process will remain available for future use.

The ERA5 process is under development. The process is similar to HIRLAM downloader process, as it downloads a binary file that contains the requested data. However, as ERA5 is a historical reconstruction, the data is ingested on-demand instead of scheduled workflow executions. The target time of release for this process is in early July.

The remaining FMI Open Data datasets (AWS observations, AWS climatological values, and Climate Change Projections) utilise a common process – the FMI Open Data WFS connector. As with ERA5 process, this connector process is under development, with target time of release in early July. Recently, we have learned that the Climate Change Scenarios are out-dated, and according to FMI's experts, that data should not be used anymore as newer (updated) scenario data exists. As of June 2020, the updated scenario data has not been made available in FMI Open Data service. We do not know when this data will be included to the service. Meanwhile, the updated data can be visualised and explored at Ilmasto-opas.fi[16], a service that disseminates information on Climate Change to the general public. However, the service does not offer the data for downloading or integrating directly to other services such as EOPEN.

[14] https://www.ecmwf.int/en/forecasts/datasets

[15] https://land.copernicus.eu/pan-european/biophysical-parameters/high-resolution-snow-and-ice-monitoring

[16] https://ilmasto-opas.fi/en/

The GlobSnow and SMOS L3FT data are ingested to EOPEN database on a daily basis, however, it turns out that OpenSphere is not compatible with the files' structure and/or projection, and cannot display the data automatically. Thus, we are implementing a reprojection process to convert the data into a more suitable form. The process is currently work in progress.

Lastly, HIRLAM data has not changed. However, we examined the metadata in more detail due to apparent discrepancies between HIRLAM metadata notation and NCEP GRIB2 Code Tables, and due to potential confusion about FMI Open Data WFS query parameters correspondence to HIRLAM data file parameters.

First, we examined how the GRIB file metadata is reported by Wgrib2. This gave us a list of NCEP code names for the variables. These are listed in the leftmost column of Table 8. We obtained the variable description and measure unit (Table 8) from NCEP GRIB2 Code table 4.2, specifically utilising the tables listed under table "Product Discipline 0".[17]

Additionally, six of the HIRLAM parameters did not have a corresponding NCEP code name, and for these parameters Wgrib2 reported the internal parameter identification attributes. These are listed in the leftmost column of Table 9. These parameters have an attribute "Discipline=192", which is noted as "reserved for local use" in both NCEP Code tables and ECMWF Code tables. However, these "discipline-parmcat-parm" identification combinations exist in ECMWF parameter definitions, and can be found in ECMWF parameter database[18]. Since these parameters have no corresponding NCEP code name, we cannot report their NCEP description or NCEP measure units.

To match the HIRLAM file metadata to NCEP parameters, we looked into the internal parameter identification attributes found in HIRLAM data. As these attributes match the NCEP Code table structure, the process was straightforward. The HIRLAM metadata is included in Table 8 and Table 9, with column names corresponding to the metadata attribute names. In addition, there are two extra columns: "Accumulation period" and "Notes". These columns contain additional information about the parameters.

Notably, there are some inconsistencies between NCEP Code table information and HIRLAM metadata information. We have highlighted the differing values in Table 8. Parameters listed in Table 9 cannot be compared this way, as they do not exist in the NCEP code tables in the first place. The first difference is found in geopotential height. NCEP gives the values in gpm (geopotential metres), whereas HIRLAM uses metres as units. According to American Meteorological Society, the two units are interchangeable for the most meteorological applications[19]. A rough approximation is 1 gpm = 9.8 metres. The second difference is that the radiative parameters are considered fluxes in NCEP notation, implying an instantaneous value, while they are accumulated values in HIRLAM. This causes an apparent discrepancy between the units. However, the accumulation is simply a sum of the fluxes in the time period, so the conversion from accumulation (flux) to flux (accumulation) is to divide (multiply) the former with the accumulation time.

[17] https://www.nco.ncep.noaa.gov/pmb/docs/grib2/grib2_doc/grib2_table4-2.shtml

[18] https://apps.ecmwf.int/codes/grib/param-db

[19] http://glossary.ametsoc.org/wiki/Geopotential_height

The second problem was to figure out if the precipitation variables are usable for PUC1 purposes. PUC1 has requested both hourly accumulated precipitation and the total precipitation up to 48 hours. The FMI Open Data WFS query has two suitable parameters, “Precipitation1h” and “PrecipitationAmount”, which correspond to hourly and total precipitation respectively. However, in the HIRLAM data file we find parameters ”Precipitation rate” and ”surface precipitation amount, rain, convective” (see column ”name” in Tables Table 8 and Table 9). Additionally, the “Precipitation rate” has units of kg/m^2s, which is correct for instantaneous precipitation measurement, but not for accumulation. Regardless, we were able to confirm that these parameters correspond to the WFS query parameters. According to the NWP model experts in FMI, HIRLAM does not provide precipitation rate at all, and the Precipitation1h WFS query parameter is an accumulated value. This leads us to conclude that the metadata in HIRLAM file is misleading due to an improperly chosen ECMWF parameter. Specifically, the hourly accumulation should be represented with a parameter that has the “units” attribute of kg/m^2. We have emphasised this issue with red colour in Table 8.

Table 8: HIRLAM file parameters and their correspondence to NCEP code tables. See text for explanation of colours.

Parameter code (NCEP)	Description (NCEP)	Measure unit (NCEP)	short Name	name	parameterName	units	paramId	Accumulation period	Notes
PRES	Pressure	Pa	msl	Mean sea level pressure	Pressure	Pa	151		
PRATE	Precipitation rate	kg/m^2s	prate	Precipitation rate	Precipitation rate	kg/m^2s	3059	1 hour	Metadata is misleading, this is hourly accumulated precipitation. Correct units are kg/m^2. (Precipitation1h in WFS query)
LAND	Land cover	proportion	lsm	Land-sea mask	Land cover (0 = sea, 1 = land)	(0–1)	172		Land=1, sea=0
HGT	Geopotential height	gpm	orog	Orography	Geopotential height	m	228002		
GUST	Wind speed (gust)	m/s	10fg	10 metre wind gust since previous post-processing	Wind speed (gust)	m/s	49	1 hour	Maximum
TMP	Temperature	K	2t	2 metre Temperature	Temperature	K	167		
UGRD	wind (u comp.)	m/s	10u	10 metre U wind component	u-component of wind	m/s	165		
TCDC	Total cloud cover	%	tcc	Total Cloud Cover	Total cloud cover	%	228164		
DLWRF	downward long wave radiation flux	W/m^2	strd	Surface thermal radiation downwards	Downward long-wave radiation flux	J/m^2	175	1 hour	Accumulation

Parameter code (NCEP)	Description (NCEP)	Measure unit (NCEP)	short Name	name	parameterName	units	paramId	Accumulation period	Notes
DPT	dewpoint	K	2d	2 metre dewpoint temperature	Dew point temperature	K	168		
RH	relative humidity	%	2r	2 metre relative humidity	Relative humidity	%	260242		
NSWRF	Net Short-Wave radiation flux	W/m^2	ssr	Surface net solar radiation	Upward short-wave radiation flux	J/m^2	176	1 hour	Accumulation
DSWRF	Downward short-wave radiation flux	W/m^2	ssrd	Surface solar radiation downwards	Downward short-wave radiation flux	J/m^2	169	1 hour	Accumulation
NLWRF	Net Long-Wave radiation flux	W/m^2	str	Surface net thermal radiation	Net long wave radiation flux	J/m^2	177	1 hour	Accumulation
VGRD	wind (v comp.)	m/s	10v	10 metre V wind component	V-component of wind	m/s	166		
WIND	wind speed	m/s	10si	10 metre wind speed	Wind speed	m/s	207		

Table 9: HIRLAM file parameters that are unlisted in NCEP code tables. See text for explanation.

Parameter (as shown by wgrib2 software)	shortName	name	parameterName	units	paramId	Accumulation period	Notes
var discipline=192 center=98 local_table=0 parmcat=140 parm=242	mdvi	Mean wind direction	242	degrees	140242		
var discipline=192 center=98 local_table=0 parmcat=201 parm=113	rain_con	surface precipitation amount, rain, convective	113	kg/m^2	201113	From the beginning of the forecast	PrecipitationAmount in WFS query
var discipline=192 center=98 local_table=0 parmcat=128 parm=186	lcc	Low cloud cover	186	(0–1)	186		
var discipline=192 center=98 local_table=0 parmcat=128 parm=187	mcc	Medium cloud cover	187	(0–1)	187		
var discipline=192 center=98 local_table=0 parmcat=128 parm=188	hcc	High cloud cover	188	(0–1)	188		
var discipline=192 center=98 local_table=0 parmcat=201 parm=187	vmax_10m	Maximum wind velocity	187	m/s	201187		

4 CONCLUSIONS

This deliverable focused on the Social Media Crawler, the module that is responsible for collecting and analysing social media data in the EOPEN system and serves as one of the main sources of non-EO information.

Chapter 2 began with the reasons for selecting Twitter as the platform of interest and an overview of the social media crawling framework, together with a descriptive figure of the framework's steps. The next subsection presented the Twitter API that is used for crawling, its available filtering options and its limitations. The search criteria, which have been defined by the end users for collecting tweets per each PUC and constitute the input to the Twitter API, were gathered in a respective table, while the JSON format of the API's output was also described.

Next, the chapter continued with the various analyses that are performed on the collected data. Namely, a verification technique to estimate whether a tweet is real or fake, a localisation methodology to detect locations mentioned in the text of tweets, a concept extraction approach to retrieve visual concepts from the Twitter images, and a nudity detection method to filter inappropriate photos.

Another step of the analysis, i.e. a text classification model to estimate whether a tweet is relevant or not to the examined use cases, was presented separately, since there has been a lot of effort for this subtask, it falls under WP3, and it is not described in any other deliverables. After a description of the related work, the proposed model was introduced, along with some details on the creation of the training data set and an extensive evaluation with experiments. The results showed that the BERT method achieves the best performance for the Italian tweets about floods, while TFIDF is the most suitable method for Finnish tweets about snow coverage.

Having reported the crawling and analysis stages, the following subsections concerned the implementation of a dashboard to display the collected data, including screenshots of the EOPEN User Portal, as well as the status of the collections at the time of writing. Apart from a table that contained the exact numbers of crawled tweets, some visual analytics were also given to support certain conclusions on the nature of tweets.

Chapter 2 concluded with how social media data are exploited in other EOPEN modules, such as the event detection, the similarity fusion, the text and image clustering, the community detection, and the Knowledge Base, and how they can be the basis in synergies with other EO-clustered projects, e.g. H2020 BETTER, by describing two collaborative exercises.

Furthermore, Chapter 3 of the deliverable presented the progress on Task 3.3 "Meteorological and climatological data acquisition", subsequent to the submission of D3.2. The progress included new PUC developments, exploration of potential new datasets, process implementation, and HIRLAM metadata investigation. Each of the Use Cases has been utilising the meteorological data, and the dataset implementation has been progressing in the parallel.

5 REFERENCES

Aiello, L.M., Petkos, G., Martin, C., Corney, D., Papadopoulos, S., Skraba, R., Göker, A., Kompatsiaris, I. and Jaimes, A., 2013. “Sensing trending topics in Twitter”, IEEE Transactions on Multimedia, 15(6), pp.1268-1282.

Aggarwal, C.C. and Zhai, C. eds., 2012. “Mining text data”, Springer Science & Business Media.

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. 2003. “A neural probabilistic language model”, Journal of machine learning research, 3(Feb), 1137-1155.

Boididou, C., Papadopoulos, S., Apostolidis, L. and Kompatsiaris, Y., 2017, June. “Learning to detect misleading content on twitter”, In Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval (pp. 278-286).

Boididou, C., Papadopoulos, S., Zampoglou, M., Apostolidis, L., Papadopoulou, O. and Kompatsiaris, Y., 2018. “Detection and visualization of misleading content on Twitter”, International Journal of Multimedia Information Retrieval, 7(1), pp.71-86.

Chandrashekar, G., Sahin, F., 2014. “A survey on feature selection methods”, Computers & Electrical Engineering, 40 (1), 16-28.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. 2018. “Bert: Pre-training of deep bidirectional transformers for language understanding”. arXiv preprint arXiv:1810.04805.

Earle, P.S., Bowden, D.C. and Guy, M., 2012. Twitter earthquake detection: earthquake monitoring in a social world. Annals of Geophysics, 54(6).

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S. and Darrell, T., 2014. “Caffe: Convolutional architecture for fast feature embedding”, In Proceedings of the 22nd ACM international conference on Multimedia (pp. 675-678). ACM.

Kryvasheyeu, Y., Chen, H., Obradovich, N., Moro, E., Van Hentenryck, P., Fowler, J. and Cebrian, M., 2016. “Rapid assessment of disaster damage using social media activity”, Science advances, 2(3), p.e1500779.

Lee, C.S. and Ma, L., 2012. “News sharing in social media: The effect of gratifications and prior experience”, Computers in human behavior, 28(2), pp.331-339.

Mikolov, Tomas, et al. 2013. "Distributed representations of words and phrases and their compositionality", Advances in neural information processing systems.

Pennington, J., Socher, R., & Manning, C. D. 2014. “Glove: Global vectors for word representation”, In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).

Pittaras, P., Markatopoulou, F., Mezaris, V., Patras, I. 2017. “Comparison of Fine-tuning and Extension Strategies for Deep Convolutional Neural Networks”, Proc. 23rd Int. Conf. on MultiMedia Modeling (MMM'17), Reykjavik, Iceland, Springer LNCS vol. 10132, pp. 102-114.

Reuter, C., Hughes, A.L. and Kaufhold, M.A., 2018. “Social media in crisis management: An evaluation and analysis of crisis informatics research”, International Journal of Human–Computer Interaction, 34(4), pp.280-294.

Rosenfeld, R. 2000. “Two decades of statistical language modeling: Where do we go from here?”, Proceedings of the IEEE, 88(8), 1270-1278.

Smeaton, A.F., Over, P. and Kraaij, W., 2009. “High-level feature detection from video in TRECVid: a 5-year retrospective of achievements”, In Multimedia content analysis (pp. 1-24). Springer, Boston, MA.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A., 2015. “Going deeper with convolutions”, In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).

Wang, Z. and Ye, X., 2018. “Social media analytics for natural disaster management”, International Journal of Geographical Information Science, 32(1), pp.49-72.

Xu, Z., Liu, Y., Yen, N., Mei, L., Luo, X., Wei, X. and Hu, C., 2016. “Crowdsourcing based description of urban emergency events using social media big data”, IEEE Transactions on Cloud Computing.

Yan, J., 2009. “Text Representation”, Encyclopedia of Database Systems, 3069-3072.